

Oracle[®] COREid Federation

COREid Federation Guide

**10g Release 2 (10.1.2)
Part No. B19016-01**

May 2005

ORACLE[®]

Copyright © 1996-2005 by Oracle. All rights reserved. US Patent Numbers 6,539,379; 6,675,261; 6,782,379; 6,816,871.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>). Copyright © 1999-2003, The Apache Software Foundation. All rights reserved. Copyright © 2003 The Apache Software Foundation.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Oracle COREid Access and Identity products includes RSA BSAFE™ cryptographic or security protocol software from RSA Security. Copyright © 2003 RSA Security Inc. All rights reserved.

This program contains third-party code from Apache. Under the terms of the Apache Software License, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the Oracle program, including the Apache software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the Apache software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or Apache.

* The Apache Software License, Version 1.1

*

* Copyright (c) 2000 The Apache Software Foundation. All rights reserved.

*

* Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

*

* 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

*

* 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

*

* 3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment:

* "This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>)."

* Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.

*

- * 4. The names "Apache" and "Apache Software Foundation" must
- * not be used to endorse or promote products derived from this
- * software without prior written permission. For written
- * permission, please contact apache@apache.org.
- *
- * 5. Products derived from this software may not be called "Apache",
- * nor may "Apache" appear in their name, without prior written
- * permission of the Apache Software Foundation.
- *
- * THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESSED OR IMPLIED
- * WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
- * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
- * DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR
- * ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
- * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
- * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF
- * USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
- * ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
- * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
- * OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
- * SUCH DAMAGE.
- * =====
- *
- * This software consists of voluntary contributions made by many
- * individuals on behalf of the Apache Software Foundation. For more
- * information on the Apache Software Foundation, please see
- * <http://www.apache.org/>.
- *
- * Portions of this software are based upon public domain software
- * originally written at the National Center for Supercomputing Applications,
- * University of Illinois, Urbana-Champaign.
- * /

Bouncy Castle License

Copyright (c) 2000 The Legion Of The Bouncy Castle (<http://www.bouncycastle.org>)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

This program contains third-party code from Jason Hunter & Brett McLaughlin for JDOM. Under the terms of the JDOM license, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the Oracle program, including the JDOM software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the JDOM software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle, Jason Hunter, or Brett McLaughlin."

JDOM License

Copyright (C) 2000-2003 Jason Hunter & Brett McLaughlin.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions, and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the disclaimer that follows these conditions in the documentation and/or other materials provided with the distribution.
3. The name "JDOM" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact <license AT jdom DOT org>.
4. Products derived from this software may not be called "JDOM", nor may "JDOM" appear in their name, without prior written permission from the JDOM Project Management <pm AT jdom DOT org>.

In addition, we request (but do not require) that you include in the end-user documentation provided with the redistribution and/or in the software itself an acknowledgement equivalent to the following:

"This product includes software developed by the JDOM Project (<http://www.jdom.org/>)."

Alternatively, the acknowledgment may be graphical using the logos available at <http://www.jdom.org/images/logos/>.

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE JDOM AUTHORS OR THE PROJECT

CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software consists of voluntary contributions made by many individuals on behalf of the JDOM Project and was originally created by Jason Hunter <jhunter AT jdom DOT org> and Brett McLaughlin <brett AT jdom DOT org>. For more information on the JDOM Project, please see <<http://www.jdom.org/>>.

This program contains third-party code from the OpenSSL Project. Under the terms of the OpenSSL Project license, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the Oracle program, including the OpenSSL software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the OpenSSL software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or the OpenSSL Project"

OpenSSL License

```
/* =====  
* Copyright (c) 1998-2003 The OpenSSL Project. All rights reserved.  
*  
* Redistribution and use in source and binary forms, with or without  
* modification, are permitted provided that the following conditions  
* are met:  
*  
* 1. Redistributions of source code must retain the above copyright  
* notice, this list of conditions and the following disclaimer.  
*  
* 2. Redistributions in binary form must reproduce the above copyright  
* notice, this list of conditions and the following disclaimer in  
* the documentation and/or other materials provided with the  
* distribution.  
*  
* 3. All advertising materials mentioning features or use of this  
* software must display the following acknowledgment:  
* "This product includes software developed by the OpenSSL Project  
* for use in the OpenSSL Toolkit. (http://www.openssl.org/)"  
*  
* 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to  
* endorse or promote products derived from this software without  
* prior written permission. For written permission, please contact  
* openssl-core@openssl.org.  
*
```

* 5. Products derived from this software may not be called "OpenSSL"
* nor may "OpenSSL" appear in their names without prior written
* permission of the OpenSSL Project.
*
* 6. Redistributions of any form whatsoever must retain the following
* acknowledgment:
* "This product includes software developed by the OpenSSL Project
* for use in the OpenSSL Toolkit (<http://www.openssl.org/>)"
*
* THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS" AND ANY
* EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR
* ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
* OF THE POSSIBILITY OF SUCH DAMAGE.
* =====
*
* This product includes cryptographic software written by Eric Young
* (eay@cryptsoft.com). This product includes software written by Tim
* Hudson (tjh@cryptsoft.com).
*
*/

Contents

Preface	13
Intended Audience	13
Documentation	14
Typographical Conventions	15
Contact Information	15
Corporate Headquarters	15
Before Contacting Customer Care	16
Accessing the Customer Care Knowledge Base	16
Chapter 1 Introduction	17
About COREid Federation	18
COREid Federation Features	19
How COREid Federation Works	20
About COREid Federation Support for SAML	23
Source and Destination Domains	23
About Assertions	24
How Sources and Destinations Use Assertions	25
Assertion Contents	28
COREid Federation SAML Profiles and Services	32
Web Browser Profiles for COREid Federation SAML	32
SAML Transport Protocol	33
COREid Federation Web Services and Components	33
COREid Federation Single Sign-On Process	35
Single Sign-On Process Using the POST Profile	35
Single Sign-On Process Using the Artifact Profile	37
Single Sign-On Process Using the X.509 Attribute Sharing Profile	38
Chapter 2 Planning a COREid Federation Installation	45
About COREid Federation Installation	46
Supported Platforms	47
Planning for Installation	48
Choosing a COREid Federation System Configuration	48

Hub and Spoke Networks	48
Avoid Installing Multiple Instances in the Same DNS Domain	50
About Putting the COREid Federation Server in the DMZ	51
Choosing an IdMBridge	52
Planning for an LDAP IdMBridge (Source Domain Only)	52
Planning for a COREid IdMBridge (Source or Destination Domains)	56
Planning for an RDBMS IdMBridge (Source Domain Only)	60
Planning for a SiteMinder IdMBridge (Source or Destination Domain)	61
Planning COREid Federation Source Authentication	61
About User Login and Single Sign-On	61
Planning Assertion Profiles to use with COREid Federation	64
Creating an Assertion Profile	64
About SAML Assertion Elements and Attributes	65
About Assertion Mappings	66
Optional Assertion Attribute Properties	69
Policies Created in COREid	69
Choosing Artifact or POST Profiles to use with SAML	70
Advantages and Disadvantages of Artifact and POST	70
COREid Federation Security Requirements for the POST Profile	71
Security Requirements for the Artifact Profile	71
About Authentication Options for the Artifact Profile	72
POST Profile Without a Proxy	73
POST Profile Using a Proxy in the Destination DMZ	74
Artifact Profile Using a Proxy in the Source and Destination DMZ	75
Planning Key and Certificate Security Configuration	77
Exchanging Information Between Domains.....	77
Additional Information Provided by Source Domains to a Destination	78
Chapter 3	
Installing COREid Federation	79
About COREid Federation Installation.....	80
Before Installing COREid Federation	80
Installing COREid Federation and the COREid Federation Proxy Server.....	81
Installing COREid Federation	81
Installing the COREid Federation Proxy Server	85
Starting Up and Logging In to COREid Federation	87
Configuring COREid Federation	89
Deploying COREid Federation	91

	Uninstalling COREid Federation	92
Chapter 4	Configuring Source Domains	95
	Using the COREid Federation Administration Console	96
	About Source Domain Configuration	98
	Methods Available for Configuration	99
	Using the Setup Wizard	99
	Source Domain Administration from the COREid Federation Console.....	112
	Configuring the IdMBridge	113
	Setting the Active IdMBridge	114
	Configuring a COREid IdMBridge	115
	Configuring an LDAP IdMBridge	121
	Configuring an RDBMS IdMBridge	124
	Configuring a SiteMinder IdMBridge	129
	Configuring a Login Method.....	133
	Configuring Passwords and the Certificate Store	136
	Changing the Session Password Encryption Key	136
	Configuring the Certificate Store	137
	Configuring Assertion Profiles.....	139
	Configuring Domains	147
	To view, modify, or delete a domain	147
	Configuring MyDomain	148
	Configuring a Destination Domain for a Source Domain	155
Chapter 5	Configuring Destination Domains.....	169
	Using the COREid Federation Administration Console	170
	About Destination Domain Configuration	172
	Configuration Prerequisites	173
	Methods Available for Configuration	174
	Using the Setup Wizard	175
	Destination Domain Administration from the COREid Federation Console	177
	Configuring an IdMBridge	178
	Configuring Assertion Mappings	179
	Configuring MyDomain	184
	Configuring a Source for a Destination Domain.....	189
	Configuring Password and Certificate Store Encryption	196

Chapter 6	Advanced Configuration	197
	About Advanced Configuration	198
	Redirecting Users to a Login Form	198
	About SmartMarks Authentication Schemes	199
	SmartMarks Login Process	201
	SmartMarks Login for Source Domain Users	202
	SmartMarks Login for Destination Domain Users	202
	Configuring SmartMarks as a Source Domain	202
	Configuring SmartMarks as a Destination Domain	203
	Implementation Notes for SmartMarks	204
	Restricting Users to Their Local Domain Using SmartWalls	204
	How Destination Domains Implement SmartWalls	205
	About the Issuer Statement and the Domain Attribute	205
	Attribute Mappings for SmartWalls	206
	Automatically Mapping User Identities Using SmartMaps	208
	Customizing the Login Form	209
	Error Reporting	210
	Error Redirection for Destination Domains	210
	Error Redirection for Source Domains	211
	The ObSAMLError Servlet and Template	211
	Configuring Error Redirection as the Destination	212
	Error Redirection Scenario	213
	Options for Error Message Customization	214
	Setting up COREid Federation Load Balancing and Failover	215
	Setting up a Common Assertion Store Database	216
	Configuring COREid Federation to Use X.509 Attribute Sharing Profiles	221
	Identity Provider (Source or Home) Domain Configuration	221
	Service Provider/Destination Configuration	227
Chapter 7	Configuring Keys and Certificates	243
	About Configuring Secure Connections	244
	About the Certificate Stores	244
	Default Versus Third-Party Certificates	247
	Locations for Certificates and the Keytool Command	248
	Certificate Configuration Overview for Artifact and POST	249
	Maintaining Separate Signing and CA Keys	251
	Guidelines for Certificate Configuration	253
	Configuring Certificates for the POST Profile on the COREid Federation Server	254

	Configuring a Test Environment for POST on the COREid Federation Server	254
	Configuring a Production Environment for POST Operations	257
	Artifact Profile Using Basic Authentication on the COREid Federation Server.....	260
	Testing Basic Over SSL with the Artifact Profile	260
	Artifact Using Basic Over SSL Authentication	264
	Artifact Using Client Certificate Authentication on the COREid Federation Server ..	264
	Configuring Client Certificate Authentication in a Test Environment	265
	Configuring Client Certificate Authentication in a Production Environment	270
	Configuring Certificates for the Proxy	270
	Generating Keys and Certificates for the Proxy	271
	Configuring Certificates for the POST Profile	272
	Configuring Client Certificate Authentication for the Artifact Profile	272
	Verifying SSL and Client Certificate Authentication on the Proxy	273
	Changing the Certificate Store Location and Passwords.....	273
	Configuring Certificates for the LDAP Data Source	274
Chapter 8	COREid Federation Auditing and Logging.....	277
	About COREid Federation Audit and System Logs	277
	Audit Logging	277
	System Activity Logging	278
	279
Appendix A	COREid Federation Security.....	281
	About COREid Federation Configuration for SSL.....	282
	Public and Private Key Pairs	282
	Certificates and Certificate Authorities	282
	Digital Signatures	283
	About Keystores for SSL Using the POST profile	285
	About Keystores for the Artifact Profile	286
Appendix B	SiteMinder System Configuration for COREid Federation	291
	SiteMinder Requirements	292
	SiteMinder System Configuration for COREid Federation.....	293
	SiteMinder Configuration and Setup for COREid Federation	301
	Install the SiteMinder SDK	302
	Copy Jar Files and Configure Your Environment for the SiteMinder SDK	302
	Create a SiteMinder WebAgent Identity	305
	Create and Configure a SiteMinder User Directory	306

	Create a SiteMinder Domain	307
	Changing Agent Cookie Settings (COREid Federation Destination)	309
	Configure and Enable SiteMinder Logs	310
	Configuring SiteMinder for Operation as a COREid Federation Domain	312
	SiteMinder Verification Steps Following COREid Federation Configuration ...	313
Appendix C	Troubleshooting and Optimizing COREid Federation Performance	327
	Troubleshooting COREid Federation	328
	Possible Issues or Problems	328
	Optimizing COREid Federation Performance.....	334
	Tunable Parameters	335
	References	336
Index		337

Preface

The *COREid Federation 2.5 Guide* provides information for people who install and administer the Oracle COREid Federation product. This guide describes COREid Federation concepts, pre-installation considerations, product installation and configuration.

Note: Oracle *COREid Federation* was previously known as Oblix *SHAREid*. All legacy references to Oblix and SHAREid should be understood to refer to Oracle and COREid Federation, respectively.

This Preface covers the following topics:

- “Intended Audience” on page 13
- “Documentation” on page 14
- “Typographical Conventions” on page 15
- “Contact Information” on page 15

Intended Audience

This guide is intended for administrators who are responsible for installing and configuring COREid Federation 2.5. This document assumes that you are familiar with your network architecture and concepts related to storing user and identity information in an LDAP directory or relational database (RDBMS). It also assumes, if you are responsible for the installation and configuration of a destination domain (site that is serving up content), that you are familiar with Identity Management products such as Oracle COREid.

Documentation

The product previously marketed as Oblix *SHAREid* is now named Oracle COREid Federation. Any legacy references to Oblix or SHAREid in this documentation or the associated software interface (including illustrations and screen shots) should be understood to refer to Oracle and COREid Federation, respectively. The manuals that are available for this release include:

Release Notes—Provides up-to-the minute details about the latest release.

COREid Federation 2.5 Guide—Provides COREid Federation concepts and considerations, product installation and configuration, to help you create a Federated identity management system that provides cross-domain SSO to connect your partners and customers to your systems while reducing compliance risks.

COREid Federation 2.5 Upgrade Guide—Explains about preparing your COREid Federation 2.0 environment (previously known as SHAREid) and upgrading it to COREid Federation 2.5.

COREid Federation 1.0 to COREid Federation 2.0 Guide—Provides information about migrating COREid SAML Services (COREid Federation 1.0) to COREid Federation 2.0.

Documentation is also available for Oracle COREid (also previously known as NetPoint), including:

COREid Installation Guide—Explains how to install and configure the COREid components.

COREid Upgrade Guide—Explains how to upgrade earlier versions of COREid to the latest version.

COREid Administration Guide—Explains how to configure COREid applications to display information stored in the directory, how to assign view and modify permissions for data displayed on the COREid applications, and how to assign access controls to users.

COREid Deployment Guide—Provides information for people who plan and manage the environment in which COREid will run. This guide covers capacity planning, system tuning, failover, load balancing, caching, and migration planning.

COREid Customization Guide—Explains how to change the appearance of COREid applications and how to control COREid by making changes to operating systems, Web servers, directory servers, directory content, or by connecting CGI files or JavaScripts to COREid screens. This guide also describes the Access Server API and the Authorization and Authentication Plug-in APIs.

COREid Developer Guide—Explains how to create AccessGates and how to develop plug-ins. This guide also provides information to be aware of when creating CGI files or JavaScripts for COREid.

COREid Integration Guide—Explains how to set up COREid to run with third-party products such as BEA WebLogic, the Plumtree portal, and IBM Websphere.

COREid Schema Description—Provides details about the COREid schema.

Typographical Conventions

COREid Federation manuals use the following typographical conventions:

- When you are instructed to select elements sequentially, the actions are separated with angle brackets, as shown below:

Click System Admin > System Configuration > View Server Settings

- Paths to a file are shown using syntax for either the Unix or Windows platform:

```
/SHAREid_install_dir/identity/oblix/logs/debugfile.lst  
\SHAREid_install_dir\identity\oblix\logs\debugfile.lst
```

where *SHAREid_install_dir* refers to the directory where the component, in this case, the COREid Federation Server, is installed.

Contact Information

For a list of contacts including corporate offices world wide, sales, and other details, visit the Oracle Web site at:

<http://www.oracle.com>

You can contact Oracle with questions or comments as follows:

Customer Care—<http://www.oracle.com/support/contact.html>

Corporate Headquarters

Oracle maintains offices world wide. Oracle corporate headquarters is located at:

500 Oracle Parkway
Redwood Shores, CA 94065
Phone: (650) 506-7000

Before Contacting Customer Care

Before contacting Customer Care, please have available the following:

- Oracle product name and version number
- Type of computer and operating system you are using

Accessing the Customer Care Knowledge Base

For more information about using COREid Federation, see the Oracle Customer Care Knowledge Base. To access the Knowledge Base, you need a login name and password, which you can obtain from your Oracle sales representative.

To access the Knowledge Base:

1. Enter the following URL in your browser and press Return.
`http://www.oracle.com/support/contact.html`
2. Click the phrase, Login to the Oracle PremiumCare Online Portal.
3. Enter your user name and password in the box that appears, then click Login.
4. Under Oracle Support Tools, click Case Manager.
5. In the next screen, click Find Answers to gain access to the Knowledge Base.

1 Introduction

In a collaborative business environment, users need a simple way to move back and forth between content provided on different corporate Web sites. These sites also need a common way to provide authentication and authorization of users to access protected content. The Oracle COREid Federation product enables single sign-on, eliminating the need for users to re-login and automatically authorizing users to access protected content across different domains.

This chapter addresses the following topics:

- “About COREid Federation” on page 18
- “About COREid Federation Support for SAML” on page 23
- “Source and Destination Domains” on page 23
- “About Assertions” on page 24
- “COREid Federation SAML Profiles and Services” on page 32
- “COREid Federation Single Sign-On Process” on page 35

About COREid Federation

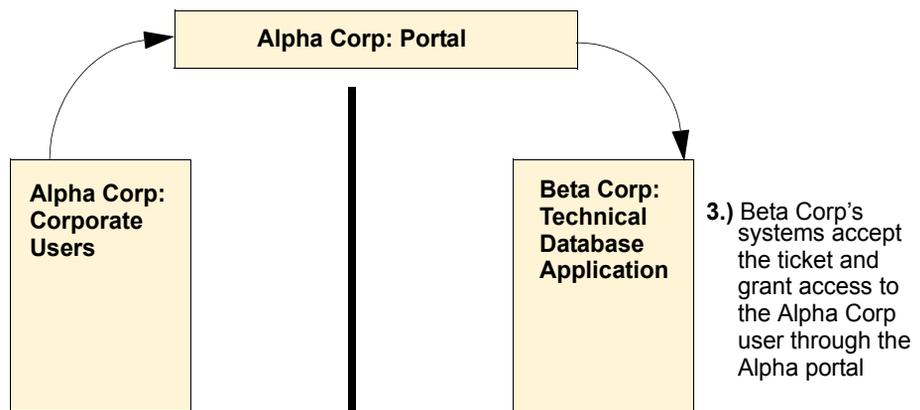
COREid Federation enables single sign-on across diverse organizations and security systems. When a user tries to access a protected resource on a remote Web site, COREid Federation at the user's site transfers information about the user to the remote site for use in authorizing the user's request. For example:

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

Figure 1 illustrates how COREid Federation establishes trust:

Figure 1 Cross-domain trust using COREid Federation

- 1.) Users from Alpha Corp authenticate in Alpha Corp's portal.
- 2.) Alpha Corp uses COREid Federation to send an identity "trusted ticket" to Beta Corp's application.



In this example, users might click on a link on their own company domain's web site requesting access to content on a partner's domain web site. The first time users request access to this content, they are authenticated on their own site with user profile information stored in their own user data repository. The user's home or source domain forwards the user's access request to the associated destination site along with the necessary credentials the destination site can then use to authorize the user's access to content on the destination domain.

COREid Federation Features

COREid Federation provides all the tools and functionality to implement single sign-on and user authentication for both source and destination domains in a multi-domain trusted identity network. COREid Federation features include:

- the ability to implement cross-site security.
- the ability to configure, enable, and disable external sites.
- the ability to enable users to access applications at destination sites using only one ID and password.
- support for SAML 1.0 and 1.1, and future support for other protocols such as WS-Federation and SAML 2.0. Also allows use of SAML Attribute sharing profiles to support distributed authorization using X.509v3-based authentication. (OASIS SAML Draft “X.509 Authentication-based Attribute Sharing Profile.”)
- close integration with COREid Identity Management System components, including the COREid Server, WebPass, Access Server, and AccessGate and associated functions, including:
 - user self-service for registration and updating of identity information.
 - identity-based workflows for functions such as user creation and updating user identity data.
 - the ability to integrate workflows with back-end functions, for example, provisioning user email accounts.
 - support for various user data repositories, including LDAP and relational databases, and Identity Management solutions such as COREid and SiteMinder.
 - distributed transactions (attribute sharing) that allow COREid to retrieve user attributes for authorization from a partner COREid Federation Server.
- assertion load balancing and failover for both source and destination domains.

COREid Federation also provides the following additional federation features to support single sign-on across security domains:

- **SmartMarks**—Redirects a user to the correct domain for authentication when a user tries to access a bookmarked page on the destination (or directly types in the URL) without a session cookie or with a session cookie that has expired. See “Redirecting Users to a Login Form” on page 198 in Chapter 6. "Advanced Configuration."
- **SmartWalls**—Ensures that security information about a user comes only from the user’s domain. The assertion is rejected if the domain of the originator does not match the domain of the assertion subject. See “Restricting Users to Their Local Domain Using SmartWalls” on page 204 in Chapter 6. "Advanced Configuration."
- **SmartMaps**—Enables COREid Federation to authenticate users who do not yet have an identity at the destination site. The user is automatically mapped to a local identity, or a new identity is created for the user. See “Automatically Mapping User Identities Using SmartMaps” on page 208 in Chapter 6. "Advanced Configuration."

How COREid Federation Works

COREid Federation takes information about users from a user data repository and uses this information as the basis for communication across corporate Web sites. The data repository is an application that can manage and authenticate the user’s identity.

For originating identity or source domain sites, from which users request data or resources from other service provider sites, COREid Federation works with one of the following data repositories that provide user profile information for the individual requesting resource access on another destination domain:

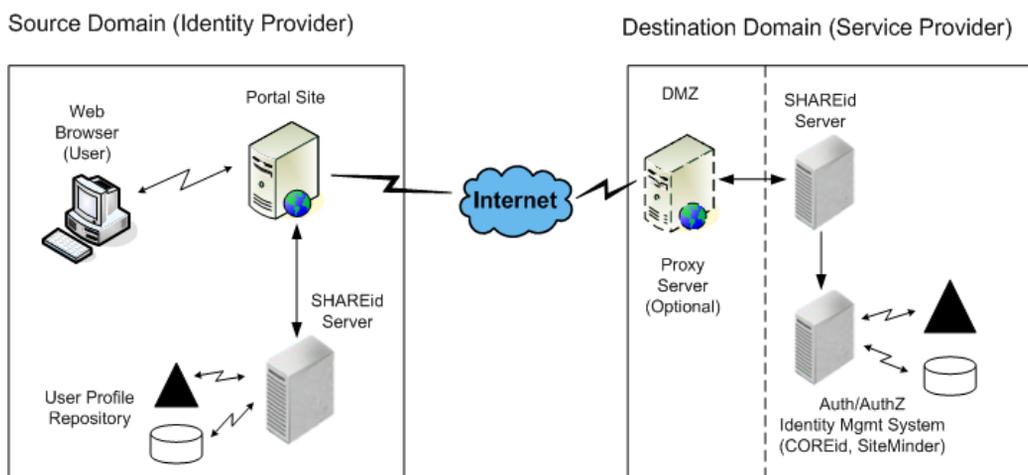
- An LDAP directory; COREid Federation provides support for Microsoft Active Directory, ADAM, Siemens DirX 2.0 Extranet Edition, and SunONE iPlanet Directory Server.
- A relational database or RDBMS; COREid Federation provides support for databases running on Microsoft SQL Server or Oracle 9i.
- The Oracle COREid product (including the COREid Server, COREid Access Server, AccessGate, and the LDAP directory that communicates with COREid)
- SiteMinder 5.5 (including Oracle COREid Federation plug-ins that allow SiteMinder to authenticate and authorize users).

On the Destination domain or service provider side, a COREid Federation Server can be configured to process user resource requests (using standard protocols such as SAML), for authenticating and authorizing users across different separate and autonomous domains. In addition, COREid Federation provides interoperability, so COREid Federation-configured source domains can talk to with other SAML (protocol) compliant destinations. and vice-versa.

Note: A single COREid Federation Server can be configured as both a source and destination domain.

The following illustration provides a typical configuration for COREid Federation installed at both source (identity provider) and destination (service provider) domains.

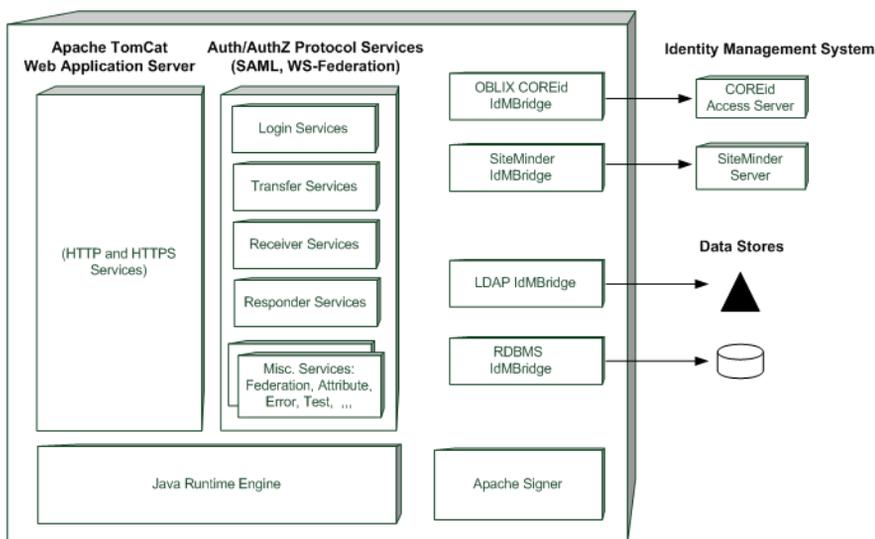
Figure 2 Typical COREid Federation Source and Destination Domain Configurations



For destination sites containing the requested resources, COREid Federation works with the Oracle COREid Identity Management product as well as Netegrity SiteMinder.

Using COREid Federation, users authenticate at their local or source domain site. COREid Federation uses the information in the user data repository to construct claims (called assertions) about the user. When the user attempts to access resources on a destination, the source domain passes the necessary user information or credentials on to the destination, which allow it to authenticate and authorize the user's access to those resources.

Figure 3 COREid Federation System Architecture and Components



COREid Federation includes an integrated Apache Tomcat Web application server, Java runtime environment, and a number of servlet/jar file-based services to support authentication and authorization using SAML and other security transport protocols. In addition, COREid Federation provides interfaces to support access to user profile, authentication, and authorization information stored in an LDAP directory, a relational database, or Identity Management Systems such as COREid and SiteMinder.

About COREid Federation Support for SAML

COREid Federation is designed to take advantage of various protocols that provide methods for authenticating and authorize users across different separate and autonomous domains. The most common protocol in use today is SAML (Security Assertion Markup Language) and COREid Federation currently provides support for SAML 1.0 and 1.1. The SAML standard provides document formats and protocols to enable single sign-on across diverse sites using different security systems. SAML is a standard developed by the Organization for the Advancement of Structured Information Standards (OASIS) and other standards bodies such as the International Telecommunications Union (ITU). SAML provides a framework based on the Extensible Markup Language (XML) document standard.

A SAML installation responds to requests from other SAML sites for information about users. The SAML installation generates information about users and sends the information to other sites. The information is used to authenticate users and to grant or deny access to resources. Users sign in to a site that is protected by a SAML-compliant product and access other SAML-enabled Web sites without re-authentication. In addition, COREid Federation provides support of SAML Attribute sharing profiles to support distributed authorization using X.509v3-based authentication. (Oasis SAML Draft “X.509 Authentication-based Attribute Sharing Profile.”)

COREid Federation sites can communicate with any SAML-compliant site. See “About COREid Federation Support for SAML” on page 23 for details.

Source and Destination Domains

A *domain* is a Web site and applications that enable users to access resources. A SAML installation acts as a *source* domain or a *destination* domain, or both:

- **Source**—A source domain, also known as an *identity provider* in SAML terminology and WS-Federation terminology, is the point of origin for a request. Users from source domains request permission to access resources on destination domain Web sites.
- **Destination**—A destination domain is known as a *resource provider* in WS-Federation terminology and as a *service provider* in SAML terminology. Destination domains contain the Web site resource that users from source domains want to access.

A COREid Federation installation can be a source domain, a destination domain, or both.

About Assertions

When a user from a source domain tries to access a resource on a destination domain's Web site, the destination queries the source about the user. The source generates an *assertion* about the user. The assertion attests to a user's identity. The destination interprets the assertion.

In reality, assertions can be sent back and forth, but for ease of discussion assertions can be thought of as created by a source domain and used by a destination.

Assertions are SAML XML elements that contain security information about a user in a nested structure.

Information in an assertion:

- **Issuer**—The issuer element identifies the domain that issued the assertion.
- **Time range**—The time range indicates the validity period for the assertion.
- **Signature (optional)**—This is a digital signature that allows the recipient to verify that the assertion was generated by the issuer and has not been altered.
COREid Federation provides other ways of verifying the issuer and validating assertion contents, so the assertion signature is not required. But if an assertion is passed to another party, that party may want to validate the assertion, in which case the signature is useful.
- **One or more statements**—There are three types of statements:
 - **Authentication**—Assert that the subject has logged in with a given identity.
 - **Attribute**—Assert subject properties, such as the person's role or credit status.
 - **Authorization decision**—Assert that the subject is authorized to access a particular resource.

How Sources and Destinations Use Assertions

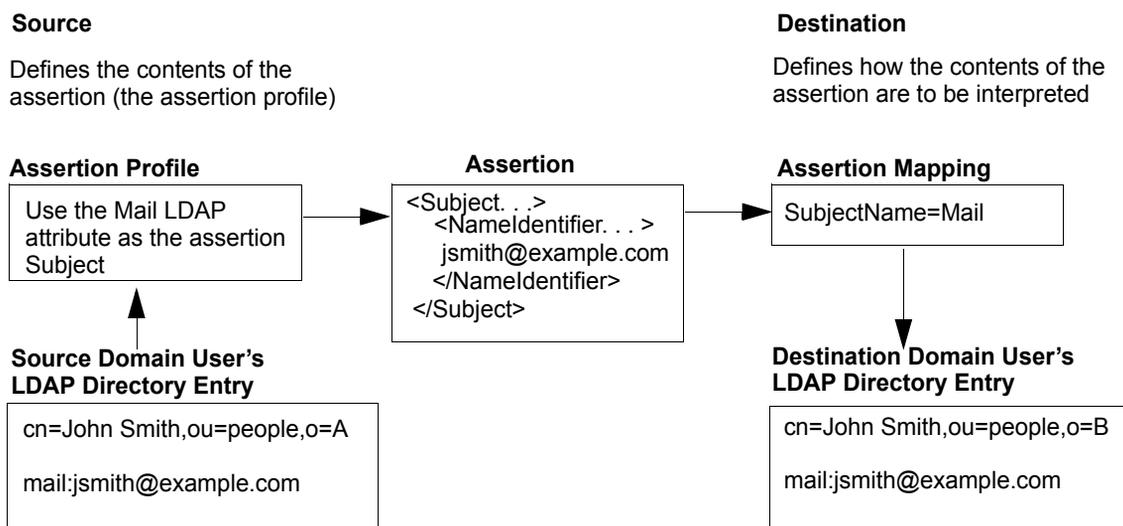
Source and the destination administrators must agree on the contents of an assertion. Using COREid Federation, the source domain administrator defines a template for an assertion, called a *profile*. The profile is a template that defines a correspondence between an user data store attribute used at the source domain, an attribute that is used in an assertion, and the data store attribute used at the destination domain, as follows:

Process overview: How sources and destinations use assertions

1. A user from a source domain requests a resource.
2. The source domain sends an assertion on behalf of the user.
3. The assertion contains a subject element and zero or more assertion attributes that correspond to data store attributes for this user.
4. The destination domain matches the assertion subject, and optionally one or more assertion attributes to a data store user attribute on the destination domain.
5. The destination domain finds an entry in the directory that matches the data store attribute or attributes that were obtained from the assertion.
6. The user is given permission to access the requested resource, and the user's browser is served the resource.

Figure 4 illustrates how domains use assertions.

Figure 4 How sources and destinations use assertions



In the illustration above, the value of the LDAP Mail attribute is `jsmith@example.com`. The source domain creates an assertion profile that uses the Mail attribute as the subject of the assertion. See “Authentication Statements” on page 29 for a complete example of the Subject assertion element. The attribute value is included in the assertion as follows:

```
<saml:Subject>  
  <saml:NameIdentifier. . .>  
    jsmith@example.com  
  </saml:NameIdentifier>  
</saml:Subject>
```

The destination site extracts the value of the assertion subject and maps that value to a value in the directory. In the example above, `SubjectName` is one of a few attributes that the destination can deduce from an assertion even if an attribute called “`SubjectName`” does not appear in the assertion. The `SubjectName` attribute identifies the assertion subject. The destination site creates a mapping between `SubjectName` and the LDAP Mail attribute on the destination site. This enables the destination site to search for an LDAP entry that contains the attribute Mail with a value of `jsmith@example.com`.

Source domains can create assertion profiles that contain arbitrarily named assertion attributes. See “Attribute Statements” on page 30 for an example of how assertion attributes appear in an actual assertion. The destination domain maps the arbitrarily named attribute to LDAP attributes at the destination, as illustrated in Figure 5.

Figure 5 An assertion with arbitrarily named attributes

Source

Defines the template for the assertion (the assertion profile)

Destination

Defines how the contents of the assertion are to be interpreted

Assertion Profile

Assertion attribute Foo=
LDAP attribute Mail

Assertion

```
<AttributeStatement. . . >
<AttributeName="Foo" >
jsmith@example.com
</AttributeName >
</AttributeStatement >
```

Assertion Mapping

Foo=Mail

Source Domain User's Directory Entry

cn=John Smith,ou=people,o=A
mail:jsmith@example.com

Destination Domain User's Directory Entry

cn=John Smith,ou=people,o=B
mail:jsmith@example.com

Assertion Contents

Table 1 describes the contents of an assertion.

Table 1 Assertion statements and elements

Authentication statement	Asserts that the subject of the assertion has logged in with a given identity. For example, a subject may be a person, identified by an email address in a particular internet DNS domain.
Authorization statement	Asserts that a security service has allowed or denied the subject permission to perform one or more actions.
Attribute statement	Asserts that the subject has certain attributes defined in a data repository.
Issuer attribute	Identifies the SAML authority who issued the assertion. Every assertion must include an Issuer attribute.
Conditions element (optional)	Specifies limitations on use of the assertion, for example, the time range that the assertion is valid.
Other required information	An assertion contains the assertion's namespace. For example: <code>saml:Assertion</code> <code>xmlns:saml="urn:oasis:names:tc:SAML:1.1:assertion"</code> It also contains the SAML version number and the assertion ID.

Authentication Statements

An authentication statement verifies a user's identity, for example, "This is John Smith."

Figure 6 illustrates an assertion that contains an authentication statement. The user is specified on the NameIdentifier sub-element of the Subject element. In the following example, the subject is `mplanck@hiphy.com`. The method used to authenticate the user is specified on the AuthenticationMethod keyword.

Figure 6 Assertion containing an authentication statement

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:1.1:assertion"
  MajorVersion="1" MinorVersion="1" AssertionID="zny9MpmEC33F6ECvkPpUIQ99"
  Issuer="http://saml.oracle.com" IssueInstance="2004-05-28T01:40:30Z">
  <saml:Conditions
    NotBefore="2004-05-28T01:38:30Z" NotOnOrAfter="2004-05-28T02:40:30Z"/>
  <saml:AuthenticationStatement
    AuthenticationMethod="urn:oasis:names:tc:SAML:1.1:am:password"
    AuthenticationInstant="2004-05-28T01:40:29Z">
    <saml:Subject>
      <saml:NameIdentifier NameQualifier="tuxedopark.com"
        Format="urn:oasis:names:tc:SAML:1.1:assertion#emailAddress">
        mplanck@hiphy.com
      </saml:NameIdentifier>
    </saml:Subject>
    <saml:SubjectLocality IPAddress="216.200.159.58"/>
  </saml:AuthenticationStatement>
</saml:Assertion>
```

Attribute Statements

An attribute statement passes attributes such as the user's title, role, or credit status, for example, "This user has Platinum status."

Figure 7 illustrates an attribute statement. In this illustration, the attribute is indicated by the `AttributeName` keyword. The attribute name is `Position`, and the value of `Position` is `President`.

Figure 7 Assertion containing an attribute statement

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:1.1:assertion"
  MajorVersion="1" MinorVersion="0" AssertionID="dwYwpmg0b8wLuRrGA+1bg=="
  Issuer="saml.oracle.com" IssueInstance="2004-05-28T01:51:43Z">
  <saml:Conditions
    NotBefore="2004-05-28T01:49:43Z" NotOnOrAfter="2004-05-28T02:51:43Z"/>
  <saml:AttributeStatement>
    <saml:Subject>
      <saml:NameIdentifier NameQualifier="company.com"
        Format="urn:oasis:names:tc:SAML:1.1:assertion#emailAddress">
        fdyson@inadstd.com
      </saml:NameIdentifier>
    </saml:Subject>
    <saml:Attribute AttributeName="Position"
      AttributeNamespace="http://princ.com">
      <saml:AttributeValue>President</saml:AttributeValue>
    </saml:Attribute>
  </saml:AttributeStatement>
</saml:Assertion>
```

Authorization Statements

An assertion that contains an authorization statement indicates if a user may perform an operation on a resource, for example, “This user is allowed to GET a specific URL.” In the following assertion, containing an authorization statement, the URL of the resource is `http://www.oblix.com:80/webgatetest/math.jpg`. The decision is Allow.

Figure 8 Assertion containing an authorization statement

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:1.1:assertion"
  MajorVersion="1" MinorVersion="0" AssertionID="ONfbqDgbEQLvEaLV2E/gcA=="
  Issuer="saml.oblix.com" IssueInstance="2004-06-29T01:04:16Z">
  <saml:Conditions
    NotBefore="2004-06-29T00:59:16Z" NotOnOrAfter="2004-06-29T01:14:16Z"/>
  <saml:AuthorizationDecisionStatement
    Resource="http://www.oblix.com:80/mywebgate/math.jpg"
    Decision="Permit">
    <saml:Subject>
      <saml:NameIdentifier NameQualifier="purists.com"
        Format="urn:oasis:names:tc:SAML:1.1:assertion#emailAddress">
        hardy@purists.com
      </saml:NameIdentifier>
    </saml:Subject>
    <saml:Action
      Namespace="urn:oasis:names:tc:SAML:1.1:action:ghpp">GET</saml:Action>
    <saml:Action
      Namespace="urn:oasis:names:tc:SAML:1.1:action:ghpp">POST</
      saml:Action>
    </saml:AuthorizationDecisionStatement>
  </saml:Assertion>
```

Note: COREid Federation can produce authorization statements for other applications, but it does not use them when it processes assertions.

COREid Federation SAML Profiles and Services

Assertions are transmitted between source and destination domains using profiles and services that are defined in the SAML protocol. These profiles and services enable:

- Establishing secure connections
- Sending authentication information about users across that connection
- Receiving and interpreting queries and assertions from other SAML domains.

Web Browser Profiles for COREid Federation SAML

The *Artifact* and the *POST* Web browser profiles determine how source and destination domains send and receive assertions. These profiles enable secure exchange of assertions using a browser, as follows:

- **POST profile**—The POST profile sends a full assertion from a source to a destination. COREid Federation sends the assertion to the user’s browser as a hidden variable in an HTML form. The user’s browser posts the assertion to the destination site.
- **Artifact profile**—Some browsers handle only a limited number of URL characters. Instead of the full assertion, the Artifact profile transmits data using a compact reference to an assertion called an artifact. The artifact is placed in in a URL that redirects the user’s browser to the destination site. To complete the transfer, the destination site contacts the source site for the full assertion.
- **X.509 Authentication-based Attribute Sharing Profile** —COREid Federation also supports use of a special “SAML Attribute sharing” profile that uses X.509v3 certificate based authentication based on the Oasis SAML Draft specification “X.509 Authentication-based Attribute Sharing Profile.” For more information on COREid Federation operation using the X.509 certificate based attribute sharing profile, refer to the section “Single Sign-On Process Using the X.509 Attribute Sharing Profile” on page 38.

Regarding POST and Artifact, there are advantages and disadvantages to using either type of assertion profiles. COREid Federation supports both methods of sending and receiving assertions, so you can choose the solution that fits your particular environment and security needs best. (Refer to “Advantages and Disadvantages of Artifact and POST” on page 70.

SAML Transport Protocol

To transmit data, SAML uses *SOAP binding* which sends and receives SAML messages using the Simple Object Access Protocol (SOAP) over HTTPS. SOAP provides a way to exchange structured information between Web-based systems or applications so, in this case, it used to send and receive SAML protocol messages wrapped in SOAP envelopes and bodies.

- A SAML *request* is an XML query. Requests are made for an assertion that matches specific criteria, such as an assertion ID and subject.
- A SAML *response* returns a status of success or failure, error information, and matching assertions, if any exist.

COREid Federation web receiver and responder service (described in the following section) receive and interpret SAML requests and similarly construct and send SOAP messages that contain SAML responses to previous requests.

COREid Federation Web Services and Components

The transmission of requests and responses for SAML (and future supported protocols such as WS-Federation) is made possible through several Web services and components described in Table 2:

Table 2 COREid Federation SAML Web services and components

Service or Component	Description
ObSAML Login and ObSAMLLoginService	Process credentials from a login form and perform HTTP basic or form login.
Transfer service (Artifact and POST profiles)	The Transfer service enables a user who is logged in to the source domain to access a destination. The Transfer service constructs an assertion about the user, and directs the user's browser to the destination domain's Receiver service, along with: <ul style="list-style-type: none">• POST profile—An HTML form containing the assertion.• Artifact profile—An artifact.
Requester component (Artifact profile only)	A Requester component on the destination domain sends requests to and receives assertions from a Responder on the source domain. The Requester: <ul style="list-style-type: none">• Opens HTTPS connections to the Responder• Sends requests and receives responses.
Responder service (Artifact only)	When a source domain receives a request, the Responder service on the source domain: <ul style="list-style-type: none">• Constructs or retrieves assertions.• Returns a response to the destination.

Table 2 COREid Federation SAML Web services and components

Service or Component	Description
Receiver service (Artifact and POST)	<p>The Receiver service at the destination domain:</p> <ul style="list-style-type: none"> • POST profile—Extracts assertions from posted data. • Artifact profile—Retrieves full assertions from the source domain using the SOAP over HTTP binding of the SAML protocol. <p>The Receiver service uses information in the assertion to create a local secure session, and it redirects the user's browser to the requested resource.</p>
Attribute Sharing Service	Used with SAML Attribute sharing profiles and X.509-based authentication. Maps subject name to the home domain for the subject and use SAML Attribute queries to determine whether subject has a specified attribute value and obtain attribute values for the subject.
WS Federation	Prototype implementation of the WS-Federation protocol.

Note: A SAML response is an XML document. The Responder service is a Web service that produces a response to a SAML request using SOAP over HTTP. The POST profile uses the response format without using the Responder service.

Transfer and Error Redirection Services

Two optional SAML services are the *Transfer Form Service* and the *Error Redirection Service*.

Table 3 describes the COREid Federation components that implement these services:

Table 3 Transfer Form and Error Redirection Services

Service	Description
Transfer Form	<p>If a user requests a resource from a destination site without authenticating at the source site, this service redirects the user's browser to the source for user authentication.</p> <p>For example, if the user selects a bookmark for the resource before authenticating to the source site, this service ensures that the user is authenticated.</p>
ObSAMLError	Allows a source site to display customized error pages to users or redirect users to different target error pages.

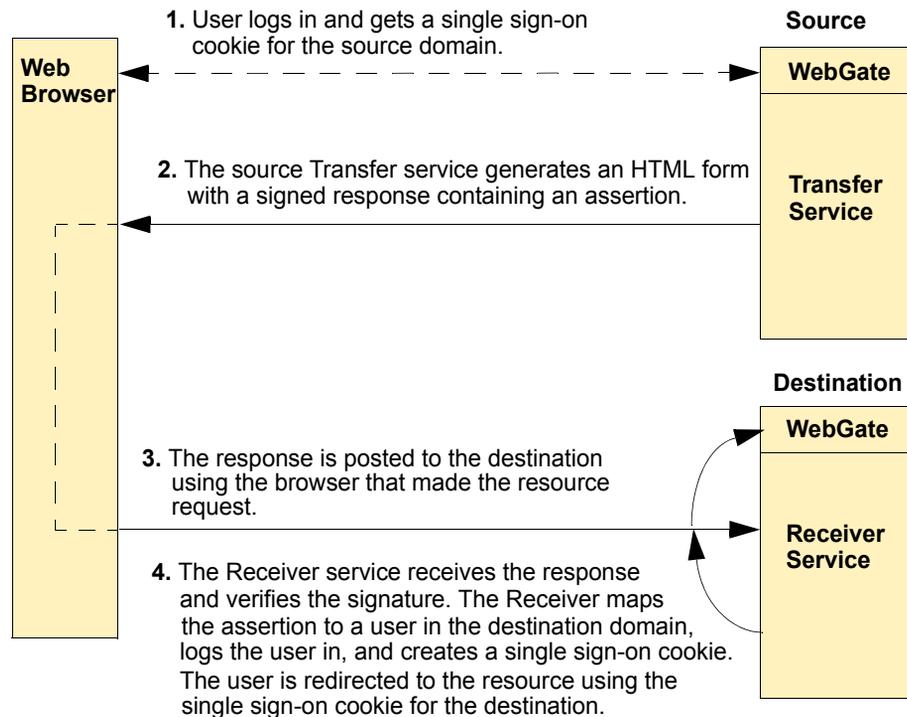
COREid Federation Single Sign-On Process

The following sections illustrate how users in source domains log in and access resources on destination domains using COREid Federation.

Single Sign-On Process Using the POST Profile

The following illustrates single sign-on using the POST profile. In this example, the COREid product is the user data repository:

Figure 9 Single sign-on using the POST profile



Process overview: Single sign-on using the POST profile

1. The user logs in.

In this example, the user logs in to a COREid component called WebGate. The WebGate sets a single sign-on session cookie, and the user is automatically authenticated to COREid Federation on the source domain.

The cookie is ObSSOCookie if the source uses COREid. The cookie is ObSHAREidSourceSession if the source uses LDAP or RDBMS.

Note that the user may log in to COREid Federation directly.

2. The user accesses a link to a protected resource on the destination domain, and the source's Transfer service generates an assertion for the user.

The assertion contents are based on user attributes that reside in an LDAP directory. If the COREid product is installed, these attributes are LDAP attributes managed by COREid. If COREid is not installed, these may be any LDAP attributes.

The source sends a response that contains the assertion and the destination domain as hidden variables. The response is signed using a signing key.

3. The user's browser posts the form to the destination Receiver service.

The Receiver service decodes and verifies the response to ensure that the assertion matches the original assertion and was issued by the source.

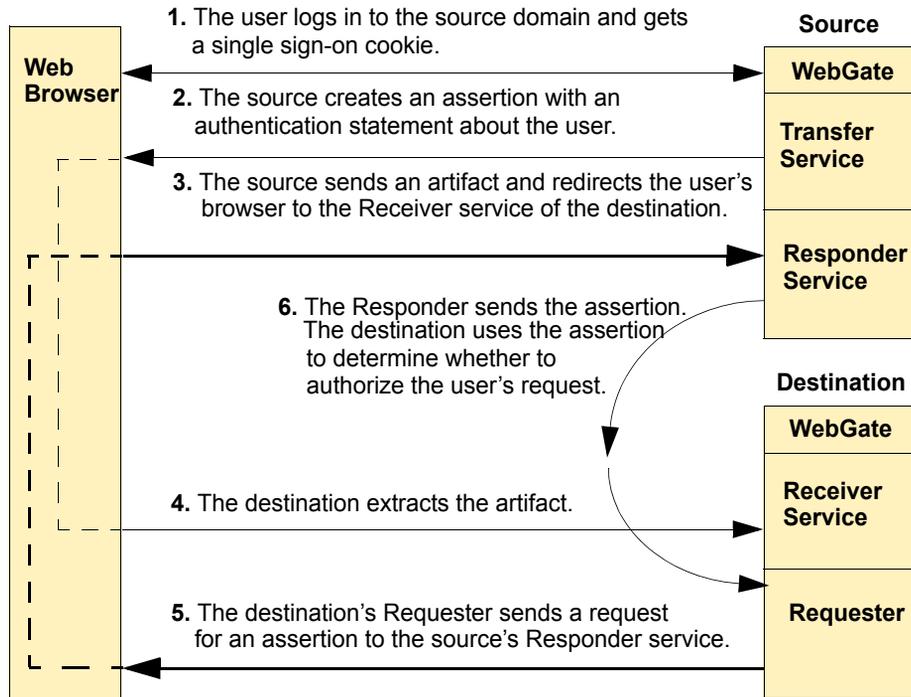
The Receiver service maps the assertion to a user in the destination domain. The destination domain user is logged in and assigned a single sign-on cookie.

4. If the destination's WebGate grants the user access to the requested resource, the destination serves the resource.

Single Sign-On Process Using the Artifact Profile

The following is a process flow diagram for the Artifact profile:

Figure 10 Single sign-on using the Artifact profile



Process overview: Single sign-on using the Artifact profile

1. In this example, the user logs in to a COREid component called WebGate. The WebGate sets a single sign-on cookie. The user is automatically authenticated to COREid Federation on the source domain. The cookie is ObSSOCookie if the source uses COREid. The cookie is ObSHAREidSourceSession if the source uses LDAP or RDBMS. Note that the user may log in to COREid Federation directly.
2. The user requests a resource in the destination domain. The source Transfer Service initiates single sign-on to the destination and generates an artifact. The artifact contains the source ID and a handle to the full assertion.
3. The source redirects the user's browser to the destination Receiver service. The query string of the URL contains the artifact.

4. The destination's Receiver service extracts the artifact typecode, source ID, and assertion handle.
5. The destination Requester service sends a request to the source for the full assertion.
6. The source Responder service processes the request and sends the assertion to the destination, where the destination's Receiver service:
 - Maps the assertion to a user in the destination and the login for that user.
 - Creates a single sign-on cookie containing this information.
 - Determines if the user is allowed access to the requested resource, and if so, redirects the user's browser to the resource.

If access is denied, the destination sends an error to the browser.

Single Sign-On Process Using the X.509 Attribute Sharing Profile

COREid Federation also supports single sign-on using a variation of the SAML attribute query/response protocol (sstc-saml-x509-authn-attribute-protocol-profile) in which an X.509 certificate installed in a browser is used to authenticate a user trying to access a protected resource. An identity provider or source domain is also used to return assertion attribute values to the service provider or destination domain, which the service provider may then use to authorize access to protected resources.

Using this type of SAML attribute sharing profile, a browser provides a client certificate. This certificate is used to authenticate the browser user to a service provider when the user requests a protected resource from the service provider.

Note: For more information on configuring COREid Federation Servers to use the X.509 certificate-based attribute sharing profile, see “Configuring COREid Federation to Use X.509 Attribute Sharing Profiles” on page 221.

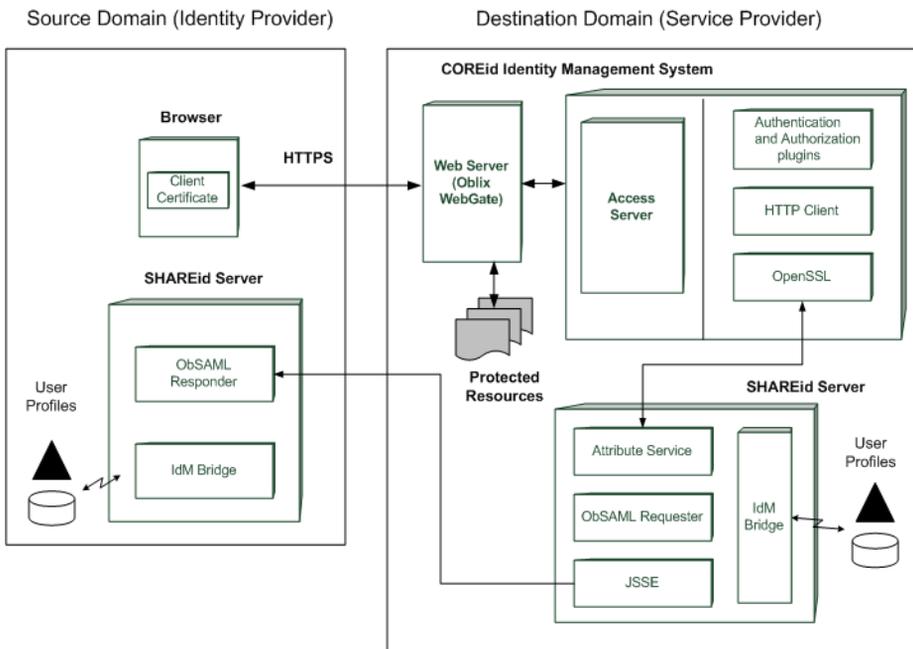
In addition to requiring that browser clients obtain a certificate, supporting this SAML user profile requires that a COREid System (WebGate, Access Server, etc.) is configured at destination (service provider) sites from which users will request protected resources.

Using this profile, the COREid WebGate component configured on the service provider's web server intercepts the resource request from the browser requesting protected resources, and then performs SSL client certificate authentication, followed by authorization of the request.

The Access Server for the service provider processes the authentication and authorization requests from the WebGate, using the customized plug-ins automatically installed with COREid Access Server (requires version 7.0.3 or later). No modification is required to the Access Server itself, although you need to configure authentication and authorization in the COREid Access System Console.

The following diagram provides more detail on the operations of specific COREid Federation and COREid components.

Figure 11 Attribute Sharing Operation between Domains



In this diagram, the roles or operations performed by specific COREid Federation or COREid Server components are described in the following two sections for source and destination domains.

Destination (Service Provider) Domain Components

WebGate—intercepts the resource request from the browser and performs SSL client certificate authentication, followed by authorization of the request.

Note: Some web servers (particularly IIS and Apache) do not allow the WebGate to dynamically initiate the SSL client authentication handshake. These web servers require that the SSL client authentication be configured statically for the protected resources. If requestor sites (source or identity provider domains) do not allow use of SSO Cookies, so ObSSOCookies are not sent in subsequent browser user requests (after initial authentication), the WebGate will perform SSL client authentication on each request. After the initial SSL handshake, the SSL client layer caches the certificate so subsequent requests for the certificate do not incur the handshake overhead. To suppress sending the ObSSOCookie to source domain sites, existing WebGates can be configured with a non-existent cookie domain.

Access Server—processes the authentication and authorization requests from the WebGate, using the COREid SubjectDN and Attribute Authorization plug-ins described below.

SubjectDN Authentication Plug-In—extracts the SubjectDN field from the user's certificate and returns it as a header action that is passed to the Attribute Authorization Plug-In. This plug-in also sets the authenticated user DN to a fixed value (e.g., an Anonymous user defined in COREid) to satisfy the Access Server's requirement that the user DN is a directory entry.

Attribute Authorization Plug-In—sends a request to the Attribute Sharing Service in the service provider's COREid Federation installation to obtain user attributes from the identity provider. This request is a simple HTTP GET with the SubjectDN, the attribute name and expected value included in the query string. The Attribute Sharing Service sends back an HTTP response indicating whether the user has the attribute value (returning authorization success or failure status accordingly).

COREid Federation Attribute Sharing Service—handles the SAML requests sent to the Identity Provider or Source Domain, including message signing as required by the X.509 Authentication-based Attribute Sharing profile. It also processes the HTTP GET request from the Attribute Authorization Plug-In. Tasks that the Attribute Sharing Service performs are the following:

- Authenticate the requester.
- Optionally, verify that the request is from one of a list of configured Access Server hosts.
- Determine which domain to send the request to, based on the user's SubjectDN.

- Construct a SAML AttributeQuery and sends to appropriate identity provider or source domain. That domain will return the attribute assertion response containing the value of the requested attribute. The result also includes an expiration time for caching the information, derived from the Attribute Sharing configuration or from the assertion expiration time.

ObSAML Requester—opens a HTTPS connection to the Identity Provider or Source Domain's SAML Responder. In addition, the Requester constructs and sends a SOAP message containing a SAML Request to the Responder, and receives and parses the SAML Response. The Requester also provides HTTP basic or SSL client certificate authentication to the Responder and can sign requests and verify signed responses.

JSSE provides the HTTPS connection to the identity provider's SAML Responder. This standard Java package is bundled with COREid Federation.

Source (Identity Provider) Domain Components

COREid Federation Server (Source or Identity Provider Domain)—processes the SAML AttributeQuery from the service provider. Note that this communication follows the standard SAML protocol, so the identity provider can utilize any SAML compliant product that process the requests.

For COREid Federation-configured source domains:

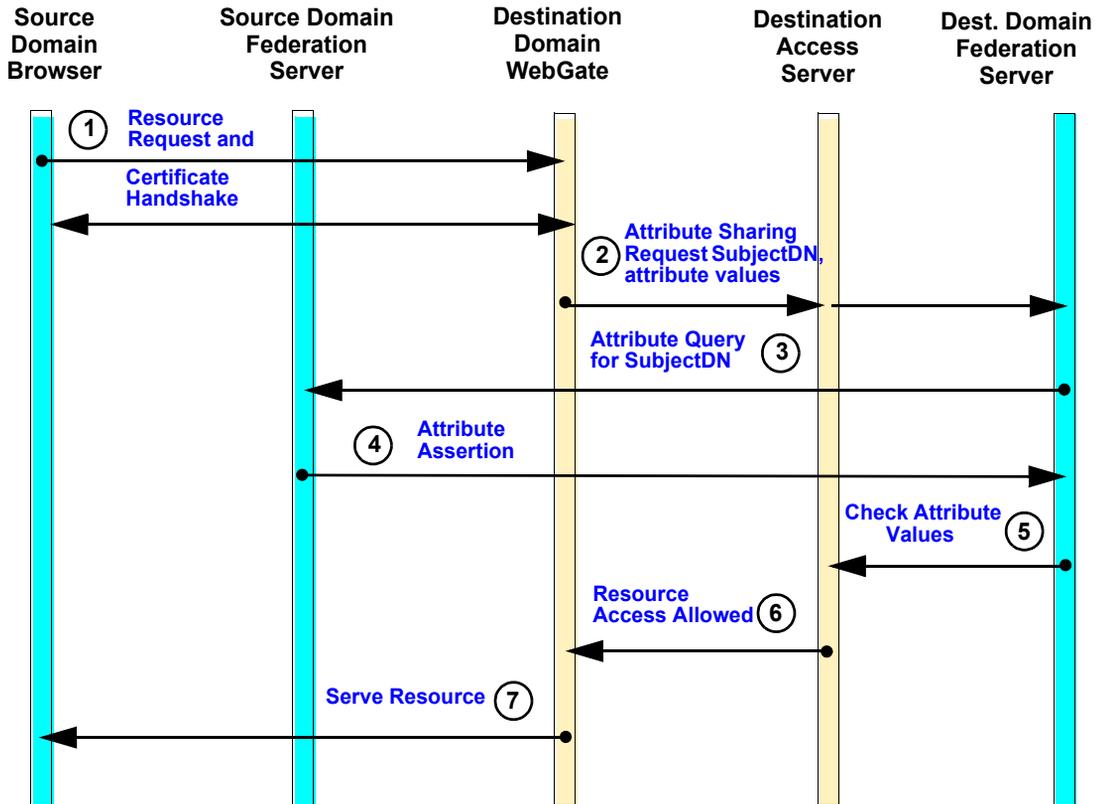
- **ObSAML Responder Service**—receives the SOAP message request, including authentication of the requester (using HTTP basic or SSL client certificate authentication), and sends back a SOAP message with the SAML Response.
- **ObSAML Responder**—processes the request, which includes parsing the request, verifying a signature on the request, mapping the request's SubjectDN to a local user, retrieving attributes for the user, and constructing and optionally signing the response.

The AttributeQuery Handler returns attribute values based on the AssertionProfile configured for the requester's domain. It returns an indeterminate result if the requester is not allowed access to the attribute or value.

- **IdMBridge**—provides the interface between the Responder class and the Identity Provider or Source Domain's Identity Management System or user repository, retrieving attribute values for users mapped from the Subject of the queries. Searches are performed on fields in the user repository matching the SubjectDN from the user's certificate.

The following illustration provides a more detailed process flow diagram for the SAML X.509 Authentication-based Attribute Sharing profile followed by a description of the communication and interaction between COREid Federation Servers and other components involved in processing attribute transactions:

Figure 12 Single sign-on using an X.509 Attribute Sharing Profile



The following section describes, in detail, the interaction between a user and the Service Provider domain's COREid and COREid Federation Servers and the interaction between COREid Federation Servers at both the source and destination domain.

1. The user issues a request for a Destination Domain (Service Provider) resource through his or her browser. To do that, the browser sends an HTTP GET request for the resource to the Destination domain's web server over an SSL connection. The WebGate protecting the Destination Domain's web server then sends a request to the user to obtain their certificate.
2. To authorize the user's access of a requested resource, the COREid Access Server contacts the designated COREid Federation Server (on the destination domain) to determine if the user has a specific attribute value.

3. Based on the SubjectDN of the user requesting a resource, the COREid Federation Server determines the domain that acts as the user's attribute authority. The COREid Federation Server now sends a SAML Attribute query to the user's home domain (attribute authority) to obtain the attribute value for the user.
4. The user's home domain sends back an attribute assertion with the values of the requested attribute.
5. The COREid Federation Server at the destination domain evaluates the response and determines if the user has the expected value, and then sends a response back to the Access Server with the information.
6. Based on the response from the COREid Federation Server, the Access Server determines if the user is now authorized and communicates this decision to the WebGate.
7. The WebGate serves up the content to the user.

For more information on configuring COREid Federation Servers to use the X.509 certificate-based attribute sharing profile, see “Configuring COREid Federation to Use X.509 Attribute Sharing Profiles” on page 221.

2

Planning a COREid Federation Installation

COREid Federation can be installed in a number of different source and destination domain configurations within a multi-domain trusted identity network to implement single sign-on, user authentication and resource authorization.

This chapter addresses the following topics of COREid Federation configuration planning:

- “About COREid Federation Installation” on page 46
 - “Supported Platforms” on page 47
 - “Planning for Installation” on page 48
- “Choosing a COREid Federation System Configuration” on page 48
- “Choosing an IdMBridge” on page 52
- “Planning COREid Federation Source Authentication” on page 61
- “Planning Assertion Profiles to use with COREid Federation” on page 64
- “Planning Key and Certificate Security Configuration” on page 77
- “Exchanging Information Between Domains” on page 77

About COREid Federation Installation

You can install COREid Federation as a source domain, a destination domain, or both:

- **Source**—A source domain, also known as an *identity provider* in WS-Federation and other federation protocol terminology, is the point of origin for users requesting access of resources from a service provider or destination domain. Source domains also provide authentication for users when the destination domain must determine whether to grant access permission to access protected resources.

COREid Federation source domains can use an LDAP directory server, a relational database, the COREid product (and its associated components, including the COREid Access Server and AccessGate), or SiteMinder as a data repository for user identities.

The identity management component for COREid Federation is known as an *IdMBridge*. The IdMBridge binds and provides user attributes for assertions and is responsible for communication with different authoritative sources of user data.

- **Destination**—A destination domain is known as a *resource* or *application provider* in WS-Federation terminology and as a *service provider* in SAML terminology. This is the domain that contains the target resources that users from source domains want to access.

A COREid Federation destination domain must use an Identity Management System such as COREid or SiteMinder to provide authorization of user requests to access protected content. Using COREid, for example, the COREid Federation IdMBridge communicates with COREid components, including the Access Server and AccessGate, to determine if the user requesting a target resource is authorized.

For testing purposes, you can install two instances of COREid Federation on the same machine, using different ports. The source must use an LDAP or RDBMS IdMBridge.

Note: In a COREid Federation network, two companies' domains can serve as both a source and a destination for each other.

Supported Platforms

COREid Federation can be installed on the following platforms:

- Microsoft Windows 2000 Advanced Server with Service Pack 4
- Microsoft Windows 2003
- Red Hat Linux Advanced Server 3
- Sun Solaris 8 or 9

COREid Federation can run with the following versions of COREid and its related components:

- COREid 7.0.3 or higher
- Netegrity SiteMinder Policy Server, Version 5.5 with SiteMinder Optional Pack on Solaris and Windows platforms

COREid Federation LDAP IdMBridges can be configured to run with the following directory servers:

- Microsoft Active Directory (Windows 2000, Windows 2003). When using COREid with Windows 2000 Active Directory, also see “Using Dynamic or Static Auxiliary Classes” on page 59.
- Microsoft ADAM (Windows 2003)
- Oracle Internet Directory (OID) 10.1.2.0.0 on all platforms where OID is supported. (See the Oracle web site for more information.)
- Siemens DirX 2.0 Extranet Edition (Windows 2000, Windows 2003, Linux, Solaris)
- SunONE iPlanet Directory Server 5.2 (Linux, Solaris)

COREid Federation RDBMS IdMBridges can run with the following RDBMS products providing a repository to store user data:

- Microsoft SQL Server 2000 (Windows 2000)
- Oracle 9i (Solaris 9)

COREid Federation Assertion Store load balancing and failover can be configured to run on the following RDBMS:

- MySQL 4.1

COREid Federation is compatible with the following browsers:

- Mozilla on Windows 2000, Windows 2003, Red Hat Advanced Server 3, and Solaris
- IE 6 SP1 on Windows 2000, Windows 2003

Planning for Installation

Before installing COREid Federation, you should understand the COREid Federation architecture and the role you will play in a federation exchange network. In addition, you need to make decisions about:

- whether you want to install COREid Federation as a source domain, a destination domain, or both.
- for source domains, which IdMBridge to use and what type of authoritative repositories you want to install. For destination domains, what Identity Management System you want to use to authorize access to protected resources.
- whether you want to install a proxy server with COREid Federation and where the COREid Federation and proxy servers will reside, for example, in the DMZ or behind a firewall.
- what type of SAML profiles and transport security protocols and certificates you want to use.
- whether you want to set up a common assertion store database to make assertion data available to more than one COREid Federation server in a load balancing and failover configuration.

Choosing a COREid Federation System Configuration

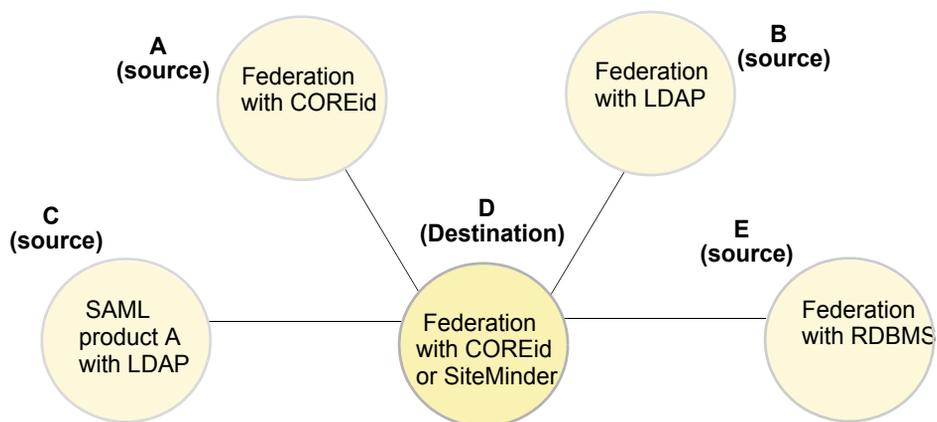
Hub and Spoke Networks

You have the option of installing COREid Federation in one of two network configurations:

- A simple hub-and-spoke network where multiple sources communicate with one destination
- A network where a site may be a source domain for some purposes and a destination domain for others.

Figure 1 illustrates a COREid Federation network where one hub serves as the focal point for multiple spokes:

Figure 1 COREid Federation deployment in a simple hub-and-spoke network



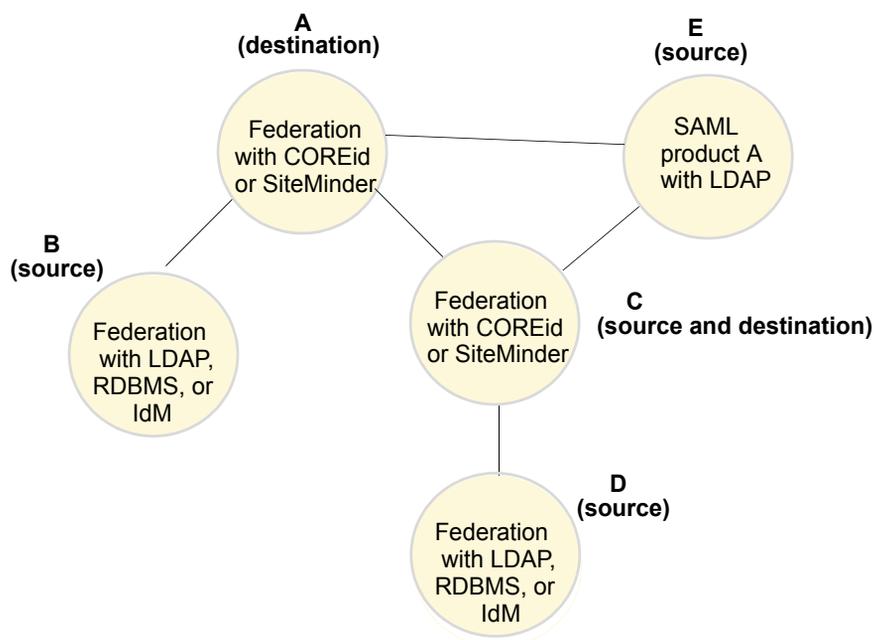
In Figure 1, Company D is a destination for people in Companies A, B, C, and E. That is, Company D has resources that people in the other companies want to access.

In this configuration, Companies A, B, C, and E are source sites. Note that Company A can also serve as a destination for other companies because COREid is installed.

Note: COREid 7.0.3 required for configurations using the X.509-based Attribute Sharing profile. (For more information, see “Single Sign-On Process Using the X.509 Attribute Sharing Profile” on page 38 and “Configuring COREid Federation to Use X.509 Attribute Sharing Profiles” on page 221.)

Figure 2 illustrates the roles played by COREid Federation domains in a network where multiple domains serve as hubs:

Figure 2 COREid Federation deployment with multiple hubs



In Figure 2, Company A is a destination for Companies B, C, and E. Company C is a destination for Companies D and E, but could also be a destination for Company A.

Avoid Installing Multiple Instances in the Same DNS Domain

Avoid installing two COREid Federation instances that use a COREid IdMBridge in the same Domain Name Services (DNS) domain, for example, domain.xyz.com. In this scenario, the COREid Federation Transfer Service may report an error indicating that the user's session has logged out or expired. If COREid is installed, this error occurs because two installations are on the same domain and share the same ObSSOCookie. When a transfer occurs from domain A to domain B, domain A's cookies are invalidated and are made valid for domain B.

If you use an LDAP directory or RDBMS as the user data repository, a similar problem can occur if you install two COREid Federation instances in the same DNS domain.

Two COREid Federation instances can reside on the same domain if one uses an LDAP IdMBridge and the other uses a COREid IdMBridge because the LDAP instance uses an ObSHAREidDomain cookie and the COREid instance uses ObSSOCookie.

About Putting the COREid Federation Server in the DMZ

Before installing COREid Federation, you must decide what components to put in the DMZ. Oracle also provides a COREid Federation Proxy Server, so you can choose to install the COREid Federation Server or the COREid Federation Proxy Server in the DMZ.

If you choose to install the COREid Federation Proxy Server, you'll need to perform the following Proxy Server configuration steps:

- Change all the urls in the domain entries (at both source and destination) to use the proxy server.
- If you are adding a Proxy Server to a source domain, import the CA of the source Proxy Server into the destination COREid Federation Server.
- If you are using artifact x.509 client cert mode, import the CA of your destination COREid Federation Server into the source Proxy Server (setting up the ca-bundle.crt file).

If you put the COREid Federation Server behind the DMZ (e.g., behind a firewall), the COREid Federation Proxy Server forwards SAML protocol requests and responses to the COREid Federation Server. The proxy is transparent to COREid Federation and provides a standard method of providing Internet access to internal resources.

Note: To make third-party proxy servers work with COREid Federation, the proxy server needs to pass the client certificate as a header variable (HTTP_SSL_CLIENT_CERT) to the COREid Federation server. Also, the certificates need to use digital signatures and key enciphers. In addition, when configuring COREid Federation, you need to change COREid Federation source and destination configurations respectively to point to the proxy server.

Choosing an IdMBridge

The IdMBridge component of COREid Federation communicates with a user data repository or Identity Management System, based on whether COREid Federation is configured as a source or destination domain. For source domains, COREid Federation uses information in the data repository to verify user identities and build protocol assertions. At destination domains, COREid Federation accesses the Identity Management System to map information in received assertions to user identities at the destination and subsequently authorize users to access protected resources. COREid Federation supports several different types of data repositories and Identity Management Systems for use as a COREid Federation IdMBridge:

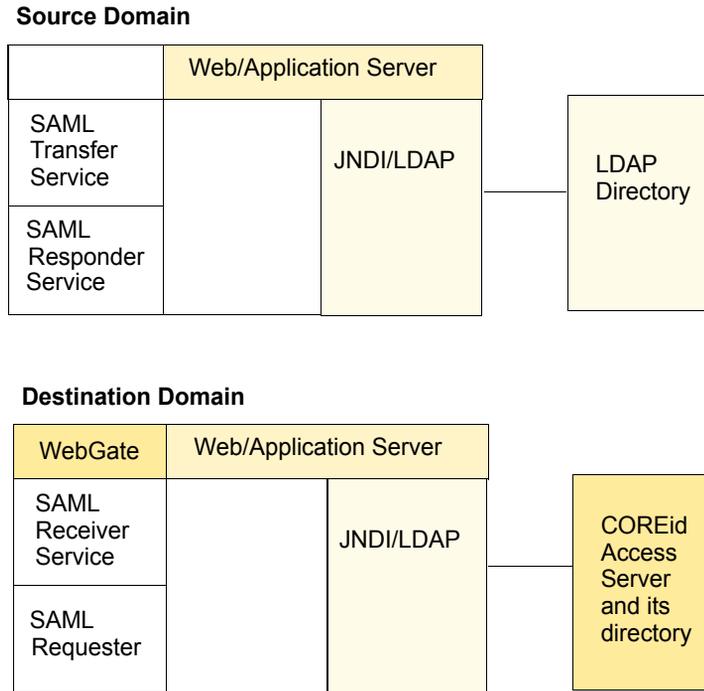
- Source domains can configure IdMBridges to use an LDAP directory, an RDBMS database, COREid, or SiteMinder.
- Destination domains can configure IdMBridges to use COREid or SiteMinder.

Note: Set up the data repository, COREid, or SiteMinder installation you plan to use with an IdMBridge, before installing and configuring COREid Federation.

Planning for an LDAP IdMBridge (Source Domain Only)

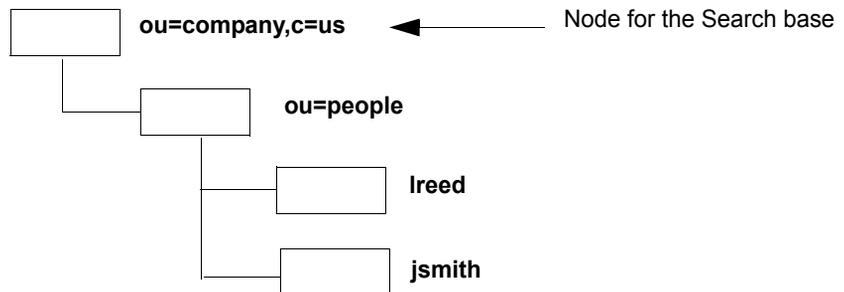
For a COREid Federation source domain, you can configure an LDAP IdMBridge that uses Microsoft Active Directory, Microsoft ADAM, or SunONE iPlanet Directory as the repository for user data. Figure 3 shows an example COREid Federation installation with an LDAP directory configured at the source domain and COREid configured for the destination domain. In this scenario, the source domain creates and sends assertions to the destination domain.

Figure 3 A Federation installation with an LDAP directory at the source



When using an LDAP directory as the user data repository for a source domain, make sure there is an entry in the directory to which COREid Federation can bind. COREid Federation must bind to the directory at a node higher than all the other nodes that will be searched. Also, COREid Federation must have read access to the directory entry and nodes where information about your local users is stored. Figure 4 shows an example scenario of directory entry binding for COREid Federation.

Figure 4 The directory entry to which COREid Federation binds



COREid Federation can perform a comprehensive directory search, or it can apply a filter to limit the search. A filter may be useful if a comprehensive search would

adversely affect performance. See “Using LDAP Filters” on page 54 for details.

When configuring an LDAP IdM Bridge, you need to decide whether communication with the directory will be via ldap or ldaps (secure mode). There are several factors in this decision:

- If the directory only provides an open port, COREid Federation must use ldap.
- If the directory only provides an SSL port, COREid Federation must use ldaps.
- If both ports are available, the choice depends on your organization’s policies.
If the user data used by COREid Federation must be protected, COREid Federation must use ldaps.
- If you are free to choose which port to use, Oracle recommends ldaps for increased security. However, you may also want to take into consideration the extra overhead imposed by SSL.

Using LDAP Filters

When using an LDAP IdM Bridge, COREid Federation requires you to specify a search filter, for example:

```
(&(objectclass=inetorgperson)(uid=%userid%))
```

The syntax for a typical filter is as follows:

```
(&(objectclass=objectclass_name)(LDAP_attr1=%cred1%  
LDAP_attr2=%cred2%...))
```

where

- & creates an "and" relationship between filter entries in parentheses.
- *objectclass_name* is the name of an LDAP object class
- *LDAP_attr1* and *LDAP_attr2* represent the names of one or more LDAP attributes
- *%cred1%* and *%cred2%* represent one or more credentials that the user supplies when logging in to COREid Federation.

The name of the credential is taken from the authentication method selected when you configure the Login page using the COREid Federation Administration Console:

- For basic login, the credential is always userid.
- Form login depends on the names of the variables used in the login form.

The sample login form provided with COREid Federation uses userid and password, so the same filter works for basic login and the default form.

- For external authentication, the credentials are the header variables as listed in the External Credentials field.

Usually all of these header variables would be used in the LDAP filter.

This filter is used as follows, assuming only one attribute-credential pair is used in the filter. When the user logs in to COREid Federation, the value of the user's login credential is used in the search filter. A search is conducted of the directory. The search is applied to entries that contain the object class and the LDAP attribute named in the filter.

A search is successful when an entry is found where the attribute name in the entry matches the attribute name in the filter, and the value of the attribute in the directory entry matches the value of the login credential used in the filter. The search needs to find exactly one user entry for the login to succeed. In most cases, you will only use one attribute-credential pair when writing an LDAP filter.

Pre-Installation Checklist for LDAP IdMBridge Configuration

If you plan to use an LDAP directory, have the following information available prior to installing and configuring COREid Federation:

- The connection URLs to your host and port, for example:

ldap://directoryhost.company.com:port

Or for secure communications:

ldaps://directoryhost.company.com:port

- The DN that COREid Federation can use to bind to the directory, for example:

cn=shareid,dc=mycompany,dc=com

- The directory bind password

- An LDAP filter to search for users in the directory using credentials such as a user ID taken from a login to the source domain, for example:

(&(objectclass=inetorgperson)(uid=%userid%))

- The top directory node for searching for users, for example:

dc=mycompany,dc=com

Planning for a COREid IdMBridge (Source or Destination Domains)

A source domain can use a COREid identity management system, which includes the COREid Server, WebPass, the Access Server, and an AccessGate, as its user data repository. If COREid Federation is to work with COREid, you need to coordinate with your COREid administrator, or see the following documents for information about installation and setup of COREid components:

- *NetPoint Installation Guide*
- *NetPoint Administration Guide*

Be sure the following items are complete before installing COREid Federation:

- Add an AccessGate in the COREid Access System Console.
The AccessGate port can be left blank.
Be sure that the Access Management Service is turned on for the AccessGate.
- Associate an Access Server with this AccessGate.
Be sure that the Access Management Service is turned on for the Access Server.
- If you plan to install a COREid Federation proxy for a destination domain, set the IP Validation flag to false in the COREid Access System.

Source Domain Considerations

Source domains must:

- Source domains must choose the user attributes to use in assertion profiles.
See “About Assertions” on page 24 and “Planning Assertion Profiles to use with COREid Federation” on page 64 for details.

- For any domain using COREid, you need to set up a Basic Over LDAP authentication scheme from the COREid Access System Console.

Usually, this means that you automatically created the rules to protect NetPoint during Access Manager configuration. This creates the NetPointBasic Over LDAP authentication scheme. COREid Federation checks that this scheme exists. If it does not, a Master Access Administrator must create the scheme and the COREid Federation administrator must restart COREid Federation.

To create an authentication scheme, click Access System Configuration > Authentication Management from the Access System Console, then click Add on the Authentication Management: List All Authentication Schemes page. In the Challenge Method field, click the radio button for the authentication scheme challenge method you want to use, in this case, Basic. The following display shows the Define a new authentication scheme page of the General tab page appears, as it appears in the COREid Access Console:

Logged in user: Master Admin

System Configuration | NetPoint System Management | Access System Configuration

General | Plugins | Steps | Authentication Flow

Define a new authentication scheme

Name:

Description:

Level:

Challenge Method: None Basic X509Cert Form Ext

Challenge Parameter:

SSL Required: No Yes

Challenge Redirect:

Enabled: No Yes

Update Cache

See the *NetPoint Administration Guide* for details on setting up authentication schemes for COREid to use.

- You can configure multiple COREid and Access Servers for failover.

This is recommended to ensure high availability of your COREid installation. See the *NetPoint Administration Guide* for details about failover.

Destination Considerations

Destination domains must:

- Choose the resources to protect.
- Choose the LDAP attributes to be mapped to assertion attributes. See “About Assertion Mappings” on page 66 for details.
- Create a host identifier named SHAREid with the hostname and port or ports of the COREid Federation Server. The name SHAREid is case-sensitive.
- If a COREid Federation proxy is to be installed in front of the COREid Federation server, you may want to turn off the IPvalidation flag for the COREid Federation destination server's WebGate.
- Set cookie domains for all the WebGates and AccessGates.

If you are using the COREid Federation Proxy in front of the COREid Federation Server on the destination site and you are using Host Identifiers, add the proxy's hostname and ports to the COREid Federation Host ID.

For example, if a COREid Federation Server and COREid Federation Proxy have the following configuration:

COREid Federation Server: example.oblix.net; open port 8101, SSL port 8113

COREid Federation Proxy: example2.oblix.net: open port 7101, SSL port 7113

The COREid Federation Host ID hostname variations would be:

example.oblix.net:8113

example.oblix.net:8101

example2.oblix.net:7101

example2.oblix.net:7113

If WebGate is installed on the proxy server for the destination COREid Federation installation, the /shareid/saml resource that is created during installation of COREid Federation should be unprotected.

- If you are not using host identifiers, unprotect /shareid/saml by using a “None” authentication scheme.
- If you are using host identifiers, you can unprotect /shareid/saml using the None authentication scheme, or by adding the host and port of the COREid Federation server (not the proxy server) to the COREid Federation host identifier.

About Transport Security and the COREid IdMBridge

You can configure SSL to be used between COREid Federation and COREid to encrypt data to be sent between these applications. When configuring transport security for the COREid IdMBridge, note the following:

- **Open mode**—There is no authentication or encryption between the COREid Federation AccessGate and Access Server.
- **Simple mode**—The COREid Federation AccessGate and Access Server authenticate to each other using default (internal COREid) x.509 certificates.
- **Cert mode**—The COREid Federation AccessGate and Access Server authenticate using x.509 certificates obtained from a Certificate Authority.

Using Dynamic or Static Auxiliary Classes

For both source and destination domains, if you are using COREid with a Windows 2000 Active Directory, note that you need to manually specify that static auxiliary object classes are to be used. Be sure the parameter `dynamicAuxiliary` is set to `false` in the following file:

```
Access_Server_Install_Dir/access/oblix/data/common/ldapconfigdbparams.lst
```

where *Access_Server_Install_Dir* is the directory where the Access Server is installed. This is required for Windows 2000 Active Directory, which only uses static auxiliary object classes. It is also required for Windows 2003 Active Directory, if the dynamic option is not enabled.

Pre-Installation Checklist for COREid IdMBridge Configuration

Obtain the following information before installing COREid Federation:

- **AccessGate name**—A unique, descriptive name for this AccessGate that was provided during configuration of the AccessGate.
- **AccessGate password**.
- **Access Server host name**—The name of the machine where the Access Server is installed.
- **Port**—Port number that the Access Server listens to.

- **Master Access Administrator login name**—These administrators have the right to perform any task in the Access System except the right to create other Master Access Administrators.

During COREid Federation setup, you need to provide the login name of one Master Access Administrator.

- **Master Access Administrator password.**
- **Security mode**—The type of communication (secure or not secure) used between Access System components and COREid Federation.

Planning for an RDBMS IdMBridge (Source Domain Only)

If you are planning to use a Relational database (RDBMS) to provide a user data repository for COREid Federation, you will need to make sure that you have set up your database correctly, before adding or configuring the RDBMS IdMBridge for a source domain from the COREid Federation Administration Console.

COREid Federation allows you to configure an RDBMS IdMBridge using connections to a Microsoft SQL Server or Oracle 9i database. Within the database, you'll need to define columns in a table (or view) that can store all the user identity or credential attributes you want to include in an assertion profile. The table columns you need to define are:

- **Login ID column**—contains user name or login ID of user that is making request.
- **Password column**—stores password for user name or login ID of user that is making request. Not required if external authentication is used for login.
- **Other user attribute columns**—stores other information and optional data you may want to include in an assertion profile.

Note: Database columns may be defined using any of the column types available in the underlying database (e.g., char, varchar, integer, long), however, Login ID and Password columns must be based on a char or varchar database column type.

If user attribute or credential columns exist in multiple tables, you will need to create a view and specify the view name in the COREid Federation Administration Console when configuring the RDBMS IdMBridge for use in your source domain.

Planning for a SiteMinder IdMBridge (Source or Destination Domain)

If you plan to configure a SiteMinder IdMBridge for either a source domain, destination domain, or both, you will need to coordinate with your SiteMinder administrator to make configuration changes to your SiteMinder installation to work with COREid Federation. For more information on SiteMinder requirements and configuration steps, see Appendix B. “SiteMinder System Configuration for COREid Federation” on page 291. For information on configuring a SiteMinder IdMBridge, see “Configuring a SiteMinder IdMBridge” on page 129.

Planning COREid Federation Source Authentication

About User Login and Single Sign-On

Users can log in directly to COREid Federation, or COREid Federation can validate user identities based on header variables or ObSSOCookie cookie set by an external application rather than prompting the user to log in. COREid Federation typically would allow users to login via the following methods:

- **Direct login**—Users point their browsers to a COREid Federation login page. This is either a standard browser pop-up or a configurable HTML form. See “Configuring a Login Method” on page 133 for details.
- **Single sign-on**—Users log in to an application, such as a portal, which sets a header variable.

See “Single Sign-On Based on a Header Variable” below for details.

Single sign-on can also be based on a cookie, if you have the COREid product installed and a WebGate sets the ObSSOCookie cookie. See “Single Sign-On Based on a COREid Cookie” on page 63 for details.

- **Redirection**—A user who has not yet authenticated to COREid Federation, or whose session has expired, accesses a link to a resource on the destination domain and is redirected back to the source domain for authentication. See “Redirecting Users to a Login Form” on page 198 for details.

Single Sign-On Based on a Header Variable

COREid Federation supports single sign-on based on a header variable. This is a process in which:

- The user logs in to an application other than COREid Federation.

- The application sets a header variable.
- The user points their browser at a COREid Federation-enabled link that points to a destination resource, but in fact directs the user to the source domain's Transfer service.
- COREid Federation uses the header variable instead of prompting the user to log in, and directs the user to the resource.

For example, suppose an employee logs in to an HR Web site using a portal. The portal sets a header variable for the user during login. If COREid Federation has been configured to permit login based on a header variable, COREid Federation gets the user identity from the header variable set during the user's login and maps the header variable to a user in the directory. The employee goes through the portal to access a link to a 401(k) service provider on the company's HR Web site. This link points to the source domain's COREid Federation Transfer service. This service requests a federation transfer. Using SAML, the link would be similar to the following:

```
https://shareid.company.com:8113/shareid/saml/ObSAMLTransferService
?DOMAIN=My401K&METHOD=post
&TARGET=https://workplaceservices411.My401K.com/NBHome.html
```

The user can access the resource without logging in again to COREid Federation.

To implement single sign-on using a header variable, using the URL above as an example:

- Set the header variable.
- The Web designer for the HR portal would need to know the syntax for constructing a URL to the destination resource that uses the SAML transfer service.
- The Web designer would get the Transfer service URL (ObSAMLTransferService), the DOMAIN and the METHOD from the COREid Federation administrator.
- The Web designer would get the TARGET URL from an administrator at My401K.com.

For this type of single sign-on to work, the portal must pass the header variable or variables on the user's first access through the portal to the Transfer Service. COREid Federation will use the header variable or variables to construct the ObSSOCookie for the COREid IdMBrige, ObSHAREidSourceSession for the LDAP or RDBMS IdMBrige, or SMSESSION for a SiteMinder IdMBrige. Refer to your portal documentation for information on how to set header variables.

Single Sign-On Based on a COREid Cookie

When the user logs in to any COREid-protected resource, a single sign-on cookie known as ObSSOCookie is set. If a user points to a COREid Federation-enabled link and COREid Federation detects a COREid single sign-on cookie, the user is not prompted to log in.

The ObSSOCookie cookie can be used by any AccessGate, so in addition to being enabled for single sign-on to COREid Federation, the user has single sign-on to all COREid-protected resources.

When the user's session ends, the ObSSOCookie cookie is deleted.

When you install and configure COREid Federation using a COREid IdMBridge, the policies needed to set the ObSSOCookie cookie are configured automatically. Note, however, that a Master Access Administrator still must configure the AccessGate used by the COREid IdMBridge to enable single sign-on with COREid.

Login Based on User Redirection from a Destination

In this method of login, the user has not yet authenticated to COREid Federation or the user's session has expired. The user clicks a link to a resource on the destination domain, and a special-purpose authentication scheme configured in the destination's COREid Access System redirects the user back to a login form. The user authenticates and is served the resource.

This scenario uses the SmartMarks function. See "Redirecting Users to a Login Form" on page 198 for details.

Customizing the Login Form

COREid Federation provides an option to configure form-based login, and provides a default HTML login form that you can customize. See "Configuring a Login Method" on page 133 and "Customizing the Login Form" on page 209 for details.

You can change the default login form if you want to use forms other than the one provided by COREid Federation. You can also change the login form location, for example, if you need to point to a full URL similar to the following:

```
https://host.company.com:8113/shareid/login.jsp
```

This might be necessary if you have a Web or application server on the same host as COREid Federation. In this situation, the COREid Federation Tomcat servlet may require the full URL to be able to find the login form.

To illustrate when you may want to specify the full URL to the login form, suppose you have a login form at the following URL:

`http://host.company.com/TransferFormDestination.html`

On the destination site, this URL is protected by the COREid Federation SmartMarks authentication scheme. If you do not specify the full URL, COREid Federation will try to redirect you to:

`http://host.company.com/shareid/login.jsp`

Planning Assertion Profiles to use with COREid Federation

When a user requests a resource in a destination, COREid Federation constructs an assertion that attests to the user's identity. An assertion profile defines the information that is put in an assertion. You must configure at least one assertion profile.

To create an assertion profile, you have the option to configure one or more assertion attribute names and assign to each assertion attribute the value of a local user attribute. If you are using a COREid IdMBridge, you specify LDAP attributes that COREid manages. See "How Sources and Destinations Use Assertions" on page 25 for an illustration.

As a source domain administrator, you and the destination domain must agree on any attributes that you configure in an assertion profile. The destination domain usually determines the names of the assertion attributes because the destination maps the user who is identified by the assertion to an identity at the destination. A non-COREid Federation destination can rely on the information in the assertion to make authorization decisions. Destinations can also use assertion attributes for tracking purposes and authorization, for example, the user's title.

If you do not configure any assertion attributes in an assertion profile, the destination uses the assertion Subject to identify the user. During assertion profile configuration, you are prompted to supply a directory attribute that will be used in the Subject element. If you do not supply an attribute for the Subject, the COREid Federation default is to supply the user's DN in the Subject element of the assertion.

Creating an Assertion Profile

1. Work with the destination to determine the user attribute to be used when specifying the assertion subject.

If you do not specify a user attribute to use for the assertion subject, the user's DN is used in the assertion for LDAP, COREid, or SiteMinder using LDAP.

2. Work with the destination domain to select assertion attribute names, if any are to be used.
3. Determine which of your user attributes are to correspond to each assertion attribute.
4. Use the COREid Federation Administration Console's Setup Wizard to create the first assertion profile.
5. Use the COREid Federation Administration Console's Source Assertions menu option to create additional assertion profiles.

About SAML Assertion Elements and Attributes

When COREid Federation constructs a SAML assertion, the assertion contains sets of elements with associated attributes. One element in an assertion is an authentication statement. The authentication statement contains a NameIdentifier attribute with a value indicating the user who is the subject of the assertion, similar to the following:

```
<saml:Subject>
  <saml:NameIdentifier NameQualifier="tuxedopark.com"
    Format="urn:oasis:names:tc:SAML:1.1:assertion#emailAddress">
    jsmith@hiphy.com
  </saml:NameIdentifier>
</saml:Subject>
```

An assertion may also contain an attribute statement. The attribute statement contains one or more user attributes, similar to the following:

```
</saml:Attribute AttributeName="Position"
  AttributeNamespace="http://companyA.com">
  <saml:AttributeValue>
    President
  </saml:AttributeValue>
</saml:Attribute>
```

You use these attributes to map users from source domains to local destination identities. You also can use attributes in the attribute statement for record keeping and for user authorization.

Prior to running the Setup Wizard, the two partners must agree on the attributes to be used in assertions. The destination domain usually determines what attribute names are placed in the assertion, since the destination must extract and use the attributes. The source domain then builds the assertion with the appropriate user profile attributes.

About Assertion Mappings

When a user from a source domain requests a resource in a destination domain, the SAML-protocol application on the source domain constructs an assertion that attests to the user's identity. To actually access a resource on your Web site, the source user's identity must be mapped to an identity of a local user who is authorized to access the resource.

As a destination domain administrator, you map the identity of the source user to the identity of a destination user. To do this, you identify one or more attributes in an assertion received from a source domain and map this attribute or attributes to your local user's attribute or attributes.

When your destination domain (using COREid) receives an assertion, the COREid Access Server searches the directory for the user entry, based on the LDAP search parameters in a COREid Access System authentication scheme. COREid Federation created the mapping authentication scheme when an administrator adds the assertion mapping.

User entries in the directory have probably been created using the COREid User Manager. However, COREid Federation can retrieve directory entries that were created using other applications.

Process overview: How assertion mappings work

1. If the assertion subject will be used to map source users to destination identities, the source and destination domain administrators agree on what value is used in the assertion subject.

If the source does not configure a specific user attribute for the assertion subject, the user's DN is used in the assertion Subject element. Note that the source and destination do not have to disclose what attributes are being configured on the source or mapped on the destination. The domains do have to agree on whether the assertion subject is relevant and the type of data being used as the value for the subject.

Note that the source and destination do not have to agree on the subject if they are using assertion attributes as the significant element of the assertion.

2. The source and destination domain agree on what assertion attributes, if any, are to be configured in the source domain's assertion profile.

The destination may extract attributes from assertions and use this information to identify a destination user identity that matches the source user.

3. The destination domain administrator creates assertion mappings in COREid Federation that will be used to authorize the same user.

4. COREid Federation's IdMBridge creates policies and authentication schemes in the COREid Access System.

These policies and authentication schemes correspond to the attributes in the assertion mapping.

5. When a user from a source domain selects a COREid Federation-enabled link, COREid Federation on the destination domain uses the attributes in the assertion sent by the source domain to map the user to the identity of a destination user.
6. The policies and authentication schemes that the COREid Access Administrator creates are applied to the mapped destination users.

Task overview: Configuring an assertion mapping in COREid Federation

1. Determine the attributes to put in the assertion.

Usually, destination domains will choose the names of the attributes that source domains need to include in assertions.

2. Determine the attribute that will identify the user (the subject of the assertion).

If no attribute is configured, COREid Federation uses the source domain user's DN as the assertion subject.

3. Ensure that you create appropriate destination identities for source domain users.

For example, source and destination domains may want to agree on a process for adding and deleting user identities at the destination domain. Or you may need to decide if there will be a one-to-one correspondence between source domain identities and local destination identities, or if you will categorize source domain identities by role.

Note that if you are manually entering users in your COREid system or SiteMinder at a destination domain, this step occurs outside of the COREid Federation Administration Console. You must create one or more COREid user entries to which source domain users can be mapped.

A destination domain can map source domain users to identities at the destination by defining classes of users. For example, suppose a source domain provides an attribute named Role in an assertion. The destination administrator can map the assertion attribute called Role to the destination attribute uid.

Optionally, the local COREid Federation at a destination domain can trigger a COREid workflow or other process to create a local identity for a source domain user who does not yet have a local identity. See "Automatically Mapping User Identities Using SmartMaps" on page 208 for details.

4. Determine the local user attributes managed by COREid or SiteMinder at a destination domain to which you will map each assertion attribute.
COREid Federation at the destination searches the COREid user entries for users whose attribute values match the values of the mapped assertion attributes.
5. Use the COREid Federation Setup Wizard to create the first assertion mapping. Use the COREid Federation Administration Console's Destination Mapping option to create additional assertion mappings.

About the SubjectName Attribute

For a source domain, configuring an assertion profile in the COREid Federation Administration Console, the field User attribute for Subject identifies an attribute in the source domain's user profile or user data repository whose value will be used in the Subject of the assertion, or more accurately, the NameIdentifier sub-element of the Subject element.

For a destination domain administrator who is mapping assertion attributes, you can use the attribute SubjectName if you want to use the assertion subject as the information to be mapped to the destination user identity. If you use SubjectName, be sure that User Attribute for Subject is configured in the source domain's assertion profile, and ask the source domain administrator for the format of the User attribute for Subject in the user data repository.

Commonly Used Attributes

COREid Federation recognizes certain assertion attributes even if they do not actually appear in an assertion. Commonly used assertion attributes are:

- **SubjectName**—Identifies the user who is being authenticated.
COREid Federation recognizes the SubjectName attribute even though it does not actually appear in the assertion. Destinations can use the SubjectName attribute even if the source has not configured this attribute name in the assertion profile that it uses when constructing assertions to send to your domain. The SubjectName attribute corresponds to the User attribute for Subject field on the Source Assertions page in the COREid Federation Administration Console.
- **Issuer**—Identifies the source domain, including the domain name and other data about the source. Assertions always contain an Issuer element.
- **Arbitrarily named attributes**—In an assertion profile, the source domain administrator can configure arbitrarily named attributes. For instance, you can ask the source domain administrator to create a profile that associates an assertion attribute named Email with the Mail attribute stored in the source domain's user data repository. You can then create an assertion mapping

between the Email assertion attribute and the user data repository Mail attribute.

Note: The Domain attribute does not appear in an assertion, however you can use this attribute when configuring an assertion mapping. Like SubjectName, SAML makes use of a built-in assertion attribute called Domain that uses the domain name of the Issuer. The Domain attribute may be more stable than the Issuer attribute, which may contain information in addition to the domain name.

Optional Assertion Attribute Properties

When configuring an assertion profile, you can optionally specify attribute types and namespaces. The namespace qualifies an attribute and eliminates ambiguity about attributes. The type provides information about the interpretation of an attribute value.

COREid Federation puts the namespace and type in the assertion. However, COREid Federation does not use the attribute namespaces and types when mapping assertion attributes to users at the destination domain. Other SAML-protocol products or applications might use the assertion namespaces or types, so these fields provide a way to put them in an assertion.

Policies Created in COREid

When you create assertion mappings, COREid Federation automatically creates corresponding policies in the COREid Access System, if you are using a COREid IdMBridge. The default authentication scheme for the policy domain is the NetPoint Basic Over LDAP scheme. The policies include the following:

- **Policy Domain**—COREid Federation.
- **Policy**—User Attributes: returns user attributes in authorization success actions.
- **Policy**—Transfer Service: protects the Transfer Service URL.
- **Policies**—One COREid Federation mapping for each named assertion mapping, and each policy uses an authentication scheme to map assertions to users.
- **Authentication Schemes**—One COREid Federation mapping for each named assertion mapping. Each mapping uses the credential-mapping plug-in to map assertion data (credentials) to users.

Note: If you uninstall COREid Federation, you must manually delete the COREid Federation information in COREid. Delete the COREid Federation policy domain first, then each COREid Federation mapping authentication scheme.

Choosing Artifact or POST Profiles to use with SAML

From the COREid Federation Administration Console, you can specify whether source and destination domains exchange assertions using the Artifact or POST profile. If support is enabled for both profiles on both the source domain and the destination, COREid Federation determines what profile to use from the METHOD query string parameter of the Transfer service URL (originating from the source domain), for example:

```
http://<src-host>:<port>/shareid/saml/ObSAMLTransferService?DOMAIN=  
dest-domain-name&METHOD=post&TARGET= target url
```

where *dest-domain-name* is the name of the destination domain and *target url* is the URL of the resource on the destination domain. COREid Federation provides configuration pages that allow you to specify the source and the destination domain's default Transfer Service URLs. You should include a METHOD parameter in your Transfer Service URL. If the METHOD is not present, the default profile selected is Artifact.

Note: If a source domain tries to use POST and POST is not enabled for the source domain configured for a destination, the transfer will not be accepted.

Advantages and Disadvantages of Artifact and POST

As described earlier (see “Web Browser Profiles for COREid Federation SAML” on page 32), COREid Federation supports both the *Artifact* and *POST* Web browser profiles, which provide different methods for source and destination domains to send and receive assertions. These profiles enable secure exchange of assertions using a browser.

There are advantages and disadvantages to using either type of assertion profiles. COREid Federation supports both methods of sending and receiving assertions, so you can choose the solution that fits your particular environment and security needs best.

Advantages and disadvantages of the Artifact profile are:

- The Artifact profile is less resource-intensive than the POST profile because the POST profile uses XML signatures.
- A disadvantage is that you must have same consumer on the destination and SAML provider on the source domain must reside in the DMZ.

Advantages and disadvantages of the POST profile are:

- For a source domain, the POST profile does not require putting your company's SAML components in a DMZ.

The SAML components can be placed behind a firewall.

- The POST profile uses XML signatures, and signing and verifying the signature is resource-intensive.

If you plan to send or receive many requests and responses, the POST profile can affect performance.

COREid Federation Security Requirements for the POST Profile

Security Requirements for the POST Profile are:

- The SAML specification requires HTTP over SSL (HTTPS) for communication from the browser of a requesting user to the destination.
- The source domain signs responses that it sends to destinations using an XML signature.
- The destination domain verifies the response using the SSL protocol.

Oracle recommends implementing HTTPS between the source COREid Federation and the user's browser. COREid Federation provides an HTTPS port for this purpose.

Security Requirements for the Artifact Profile

When you use the Artifact profile, the SAML specification requires that the source and destination domains establish a secure connection, as follows:

- For communications from the source to the destination:
 - SSL is required.
 - The destination domain must have a certificate for its Requester service.
- For redirection from the source domain to the user's browser, and the browser to the destination domain, HTTPS must be used.
- The source domain must have a CA certificate for its Responder service.

Note: For information about the Requester and Responder services, see "COREid Federation Web Services and Components" on page 33.

About Authentication Options for the Artifact Profile

If you use the Artifact profile, the options for securing communication between a source and destination domain are basic or X.509 authentication.

- **Basic authentication**—You can configure Basic authentication over open ports for testing purposes. For production, the destination needs to import your CA certificate and SSL must be used.
- **X.509 certificate authentication**—For client certificate authentication, the source and destination administrators must have exchanged and installed each other's CA certificates.

The following section, “About Basic Authentication” on page 72 provides more information about configuring Basic authentication. See “Configuring Keys and Certificates” on page 243 for details on certificate configuration.

About Basic Authentication

Basic authentication relies on verification of a user name and password. Basic authentication leaves communication exposed to intruders.

How basic authentication works

1. The destination domains sends an HTTPS request to the source domain.
 - A userid and password are sent in the authorization variable of the HTTPS header.

Other Web applications can access the header variable and obtain the user's password.
 - The password is encoded using the base64 encoding algorithm, not an encryption algorithm.

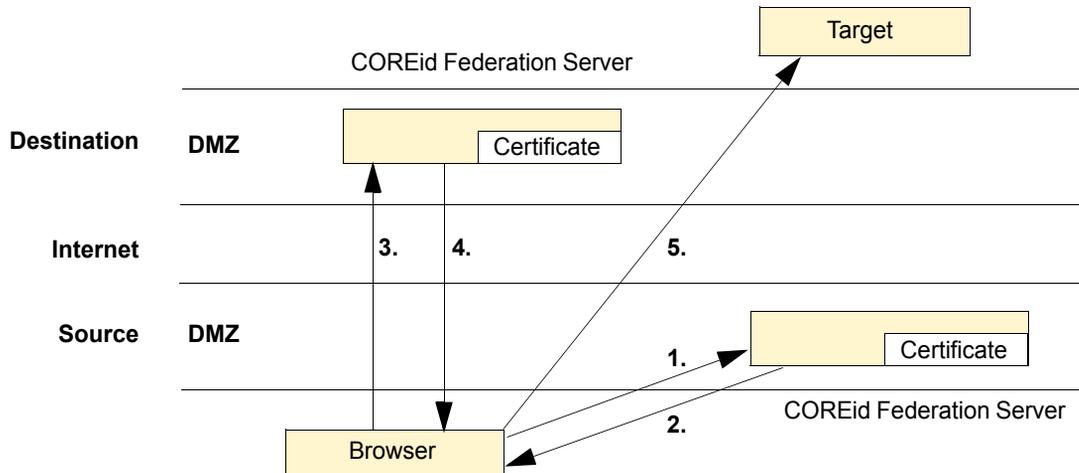
Text encoded with base64 is reversible without a key. This means that the password text is sent in the clear.
2. The source verifies the user name and password against:
 - Its own security repository, if it is a SAML installation other than COREid Federation.
 - The requester ID and the requester password, if it is a COREid Federation installation.

POST Profile Without a Proxy

The POST profile sends the full assertion over HTTPS to the destination. For testing purposes, you may want to configure COREid Federation without using any proxies.

Figure 5 illustrates a COREid Federation configuration using the POST profile and a COREid Federation server in the DMZ.

Figure 5 POST profile installation without a proxy server



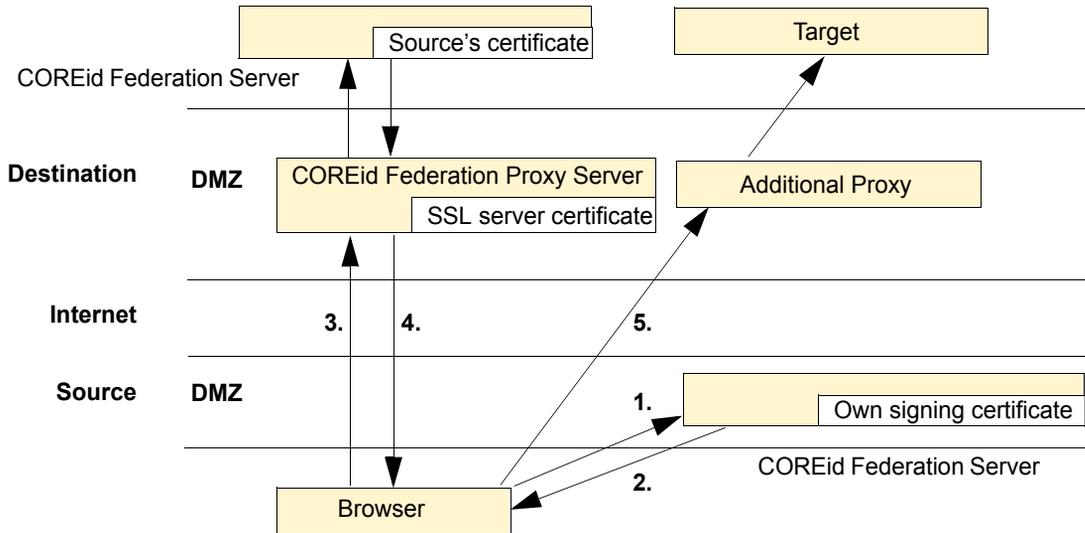
Process Overview: POST profile without a proxy

1. The user requests a transfer through the source domain's COREid Federation server to a destination resource.
2. The source COREid Federation server authenticates the user and returns an HTML form that contains a response with an assertion and the URL of the destination resource.
The response is encrypted using certificate (generated with user's private key).
3. The user's browser posts the form to the destination COREid Federation server's Receiver Service URL.
The Receiver decrypts the response using issuer's public certificate.
4. The destination COREid Federation server extracts the assertion, creates a user session for the assertion, and sends to the user's browser a redirect to the destination resource.
5. The user's browser sends the request to the target resource using the user session created on the destination.

POST Profile Using a Proxy in the Destination DMZ

The POST profile sends the full assertion over HTTPS to the destination. The source and destination are configured to communicate through their SSL ports. When using the POST profile in production, the destination should use a COREid Federation proxy server in the DMZ, as shown in Figure 6.

Figure 6 POST profile installation using a proxy server



Process Overview: POST profile using a destination proxy

1. The user requests a transfer through the source domain's COREid Federation server to a destination resource.

The source COREid Federation authenticates the user and returns an HTML form that contains a response with an assertion and the URL of the destination resource.

The response is encrypted using certificate (generated with user's private key).

2. The user's browser posts the form to the destination COREid Federation proxy Receiver Service URL.

The COREid Federation proxy forwards the form to the destination Receiver service.

3. The destination COREid Federation decrypts and extracts the assertion, creates a user session for the assertion, and sends to the user's browser a redirect to the resource.

4. The user's browser sends the request to the target resource using the session created on the destination.

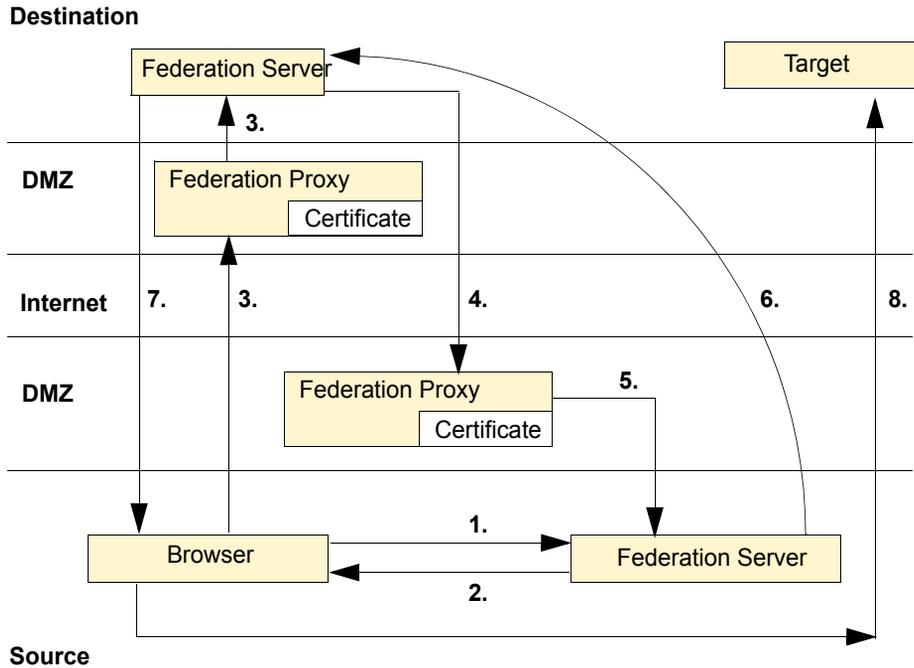
The final routing of the request may pass through an additional proxy (which must be a COREid Federation proxy).

Artifact Profile Using a Proxy in the Source and Destination DMZ

Using the Artifact profile, the source sends an artifact in place of an assertion. The artifact is an identifier that refers to an actual assertion. Upon receiving the artifact, the destination domain requests the full assertion from the issuer.

Figure 7 illustrates a COREid Federation configuration using the Artifact profile and proxies for both the source and destination domains. Note that the decision to use a proxy is up to the administrator.

Figure 7 Artifact profile installation using a proxy server



Process Overview: Artifact profile using proxies for the source and destination

1. The user requests a transfer through the source COREid Federation server to a destination resource.
2. The source COREid Federation authenticates the user, generates an assertion and an artifact, and redirects the browser with the artifact.
3. The user’s browser sends the request to the Receiver service URL, which is on the destination COREid Federation proxy.
The proxy forwards the request to the destination COREid Federation server.
4. The destination COREid Federation server requests the full assertion from the source’s Responder service, which is on the source COREid Federation proxy.
5. The source COREid Federation proxy forwards the request to the source COREid Federation server.
6. The source COREid Federation uses the artifact and retrieves the assertion and the source’s Responder service sends it over a secure connection to the destination.
7. The destination COREid Federation creates a user session and sends to the user’s browser a redirect to the resource.

8. The user's browser sends the request to the target resource using the session created on the destination.

Note: Depending on the configuration of the destination domain, the final routing of the request may pass through an additional proxy which is not a COREid Federation proxy.

Planning Key and Certificate Security Configuration

When you first install COREid Federation, most of the default SAML URLs use SSL, except for the Responder service, which uses the Open port. For initial setup and testing, the source and the destination can use default self-signed certificates that are provided with COREid Federation. However, you will want to ensure that your configuration uses third-party CA certificates before going into production. Some configuration is required for setting up certificates for secure communications. See “Configuring Keys and Certificates” on page 243 for details.

Updating the COREid Federation certificate store involves configuring the certificate store location and password in the COREid Federation Administration Console and at the command line. In addition to configuring the certificate store from the COREid Federation Administration Console, you must:

- Configure the certificate store from the command line using the `keytool` command.
- Update `SHAREid_Install_Dir/conf/server.xml`.

COREid Federation uses a certificate store for the following purposes:

- Providing XML signatures for responses sent using the POST profile.
- Establishing secure connections between the COREid Federation server on the source and the destination.

Exchanging Information Between Domains

Depending on whether you're configuring a domain as either a source or destination site, you'll need to exchange information with other source or destination domains configured in your network:

- Your domain information, which includes the Transfer Service, Responder Service, and Receiver Service URLs. These URLs include the host name and port numbers of your COREid Federation or COREid Federation Proxy Server. See the MyDomain page for a listing of the URLs you will need to provide.

- If you are a destination domain, you need to obtain the CAs of the COREid Federation and COREid Federation Proxy Servers at the source sites.
- If you are a source domain, and are using X.509 client certificate mode for requester authentication, you need to add the CA of the destination COREid Federation server to your Proxy Server. (In your proxy server configuration, you create a ca-bundle.crt file which contains the CA of the destination site). See “Configuring Keys and Certificates” on page 243 for details.

Additional Information Provided by Source Domains to a Destination

You may need to provide the destination with a list of users from your domain, if the destination domain administrator wants to map each user to a unique identity at the destination. This depends on how the destination plans to map your users to identities at the destination. The destination can map a subset of your users to a single identity based on a role, or it can create a 1:1 correspondence between each of your users and corresponding destination identities. However, a destination cannot create many-to-many mappings or one (source)-to-many (destination) mappings.

COREid Federation provides an option for a destination to automatically create new identities when a new user from your domain accesses a destination resource. See “Automatically Mapping User Identities Using SmartMaps” on page 208 for details.

3 Installing COREid Federation

This chapter describes COREid Federation installation, how to perform initial setup using the COREid Federation Administration Console, and starting COREid Federation.

This chapter addresses the following topics:

- “About COREid Federation Installation” on page 80
- “Before Installing COREid Federation” on page 80
- “Installing COREid Federation and the COREid Federation Proxy Server” on page 81
- “Starting Up and Logging In to COREid Federation” on page 87
- “Configuring COREid Federation” on page 89
- “Deploying COREid Federation” on page 91
- “Uninstalling COREid Federation” on page 92

About COREid Federation Installation

COREid Federation installation includes installation of one or more COREid Federation instances and optionally a COREid Federation proxy server, if you do not want your COREid Federation servers installed on server machines in your DMZ ("neutral zone" providing a security barrier between a company's private network and the outside public network) in a production environment.

COREid Federation provides two separate installation programs for installing the COREid Federation Server and a COREid Federation Proxy Server. Depending on the platform on which you want to install COREid Federation, you can run the installation programs from the command line (console mode), or using the InstallShield graphical user interface.

Following installation you can start up the COREid Federation Administration Console and configure your installation for operation as either a source domain, destination domain, or both. The COREid Federation Administration Console provides a Setup Wizard to help you through basic source or destination domain setup and configuration. Following initial setup, you can select other Administration Console options to make additional configuration changes.

Before Installing COREid Federation

Before running the COREid Federation or Proxy Server installation programs:

- Be sure you have the fully qualified DNS name of the machine(s) on which you are installing COREid Federation or the COREid Federation Proxy Server.

You must provide the fully qualified DNS name to ensure that your federation services, URLs, default keys and certificates are configured correctly.

- When installing COREid Federation, be sure to record the name of the host machine on which COREid Federation is installed and the COREid Federation listener ports. You will need to provide this information to all external domains that your domain communicates with. Note that the information you exchange with other domains in your partner network will be affected by whether you also install a COREid Federation Proxy to be used with COREid Federation. See and “Installing COREid Federation and the COREid Federation Proxy Server” on page 81 and “About Putting the COREid Federation Server in the DMZ” on page 51

- Ensure that the source and destination machine times are synchronized.

Your site and your partners’ sites may want to use a product to synchronize each host machine to an atomic clock. Or, you can increase the assertion validity period for an assertion after installing COREid Federation. See “Configuring Assertion Profiles” on page 139 for details.

Installing COREid Federation and the COREid Federation Proxy Server

The following procedures describe installation of COREid Federation on Windows, Linux, and Solaris. Oracle recommends that you install and run COREid Federation as a service. To do this, you must be logged in as an administrator on Windows or as root on Linux or Solaris. When installed as a service, COREid Federation can be set to start automatically when the COREid Federation server machine is rebooted.

There are three ways to start the COREid Federation Server on Windows:

- From the Windows Administrative Tools Services Control panel
- From the command line
- From the Windows Start Menu

Note: When you start COREid Federation via the command line or the Windows Start menu, its started/stopped status will not be reflected in the Services control panel. You can check the COREid Federation status by examining commands displayed in the window in which COREid Federation is launched.

Oracle recommends you run COREid Federation as a service and that you start and stop it from the Services control panel. If using COREid Federation as a service, messages for the server are written to

```
SHAREid_Install_Dir/logs/stderr.log
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

Installing COREid Federation

COREid Federation software can be installed on Windows, Linux, or Solaris platform machines. See “Supported Platforms” on page 47 for details.

To install the COREid Federation Server on Windows

1. Go to the COREid Federation download site provided when you purchased COREid Federation.
2. Download the COREid Federation Server installation file and extract it (if compressed).

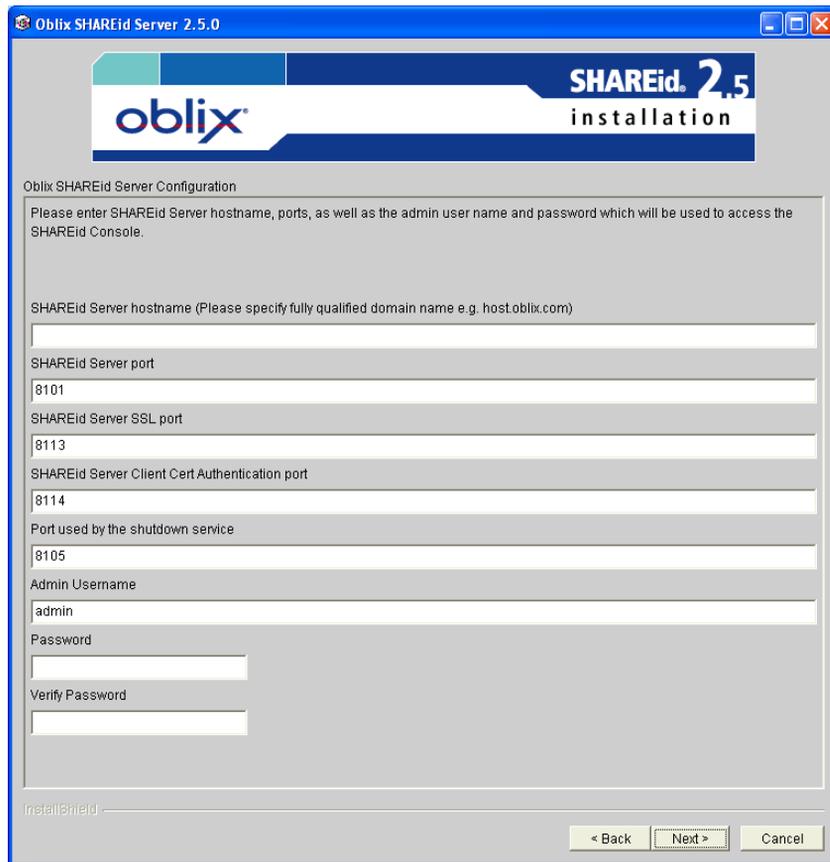
3. Double-click the installation executable.

Note: To run COREid Federation as a service (recommended), be sure you are logged in with administrator privileges.

4. When the Welcome screen appears, click Next.
5. Proceed through the introductory screens. When prompted, select an installation directory and click Next.

The default installation directory is <drive>:\Program Files\Oblix\SHAREid.

The COREid Federation installation program displays the following page:



The screenshot shows the 'Oblix SHAREid Server 2.5.0' installation configuration window. The window title is 'Oblix SHAREid Server 2.5.0'. The main header features the 'oblix' logo on the left and 'SHAREid. 2.5 installation' on the right. Below the header, the text reads 'Oblix SHAREid Server Configuration' and 'Please enter SHAREid Server hostname, ports, as well as the admin user name and password which will be used to access the SHAREid Console.' The configuration fields are as follows: 'SHAREid Server hostname (Please specify fully qualified domain name e.g. host.oblix.com)' with an empty text box; 'SHAREid Server port' with '8101'; 'SHAREid Server SSL port' with '8113'; 'SHAREid Server Client Cert Authentication port' with '8114'; 'Port used by the shutdown service' with '8105'; 'Admin Username' with 'admin'; 'Password' with an empty text box; and 'Verify Password' with an empty text box. At the bottom right, there are three buttons: '< Back', 'Next >', and 'Cancel'. The bottom left corner of the window contains the text 'InstallShield'.

6. For the COREid Federation server configuration parameters, provide the following information:

COREid Federation host—The fully qualified domain name for the COREid Federation host. Note that if you do not supply a fully qualified host name, COREid Federation will not issue an error. However, not supplying a fully qualified host name will result in errors later on.

Important: If you're using a load balancer to distribute requests among COREid Federation instances installed on more than one machine, specify the load balancer hostname here (and on all machines where you install a COREid Federation instance), instead of the hostname of the machine where you are installing a particular COREid Federation instance.

Open port—The non-secure listener port for COREid Federation. (The default port is 8101.)

SSL port—The SSL listener port for COREid Federation. (The default port is 8113.)

Client certificate authentication port—The listener port used when COREid Federation is configured for client certificate authentication. (The default port is 8114.)

Administrator user name—The user name of the COREid Federation administrator. (The default is admin.) This user logs in to the COREid Federation Administration Console and configures domain information.

Administrator password—The password that the administrator uses to log in to COREid Federation.

7. Click Next.
8. Confirm your installation parameters and click Next.
9. When installation is complete, read the ReadMe file and click Next.
10. Read the instructions on starting the COREid Federation server and click Finish.

To install COREid Federation on Linux or Solaris

1. Go to the COREid Federation download site provided when you purchased COREid Federation.
2. Download the COREid Federation installation file and extract it (if compressed).
3. Run one of the following commands, as appropriate:

```
./COREid_Federation_2_5_linux_Server
```

or

```
./COREid_Federation_2_5_sparc-s2_Server
```

Use the `-is:tempdir` switch to change the temporary folder location used to store installation logs. To run in console mode, add the `-console` switch.

Note: To run COREid Federation as a service (recommended), be sure that you are installing as root.

4. At the Welcome screen, click Next.
5. Proceed through the introductory screens. When prompted, provide the user name that the COREid Federation servers will be running as, for example, nobody.
6. Provide the Group name, for example, enter nobody and click Next.
7. Pick an installation directory, for example:

/home/shareid

Make sure the right permissions are set. For example, you may need to run the following command at the operating system command line to set permissions:

```
chmod -R o+rxw
```

8. Click Next.
9. Provide the following:

COREid Federation host—The fully qualified domain name for the COREid Federation host. Note that if you do not supply a fully qualified host name, COREid Federation will not issue an error. However, not supplying a fully qualified host name will result in errors later on.

Open port—The non-secure listener port for COREid Federation. (The default port is 8101.)

SSL port—The SSL listener port for COREid Federation. (The default port is 8113.)

Client certificate authentication port—The listener port used when COREid Federation is configured for client certificate authentication. (The default port is 8114.)

Administrator user name—The user name of the COREid Federation administrator. This user logs in to the COREid Federation Administration Console and configures domain information.

Administrator password—The password the administrator uses to log in to COREid Federation.

10. Click Next.

A dialog appears to indicate that the COREid Federation program files are installing. When installation is complete, a readme page appears.

11. Click Next.

Instructions appear regarding how to start the COREid Federation server.

Installing the COREid Federation Proxy Server

The COREid Federation Proxy Server installation script prompts for the COREid Federation server's listener port. You must install the COREid Federation Server before the COREid Federation Proxy Server so that COREid Federation Server ports are known.

To install the COREid Federation Proxy Server on Windows

1. Go to the COREid Federation download site provided when you purchased COREid Federation.
2. Download the COREid Federation Proxy Server installation file and extract it (if compressed).
3. Double-click the COREid Federation Proxy Server installation icon.

Note: To run the COREid Federation Proxy Server as a service (recommended), be sure you are logged in with administrator privileges.

4. When the Welcome screen appears, click Next.
5. Proceed through the introductory screens. When prompted, select an installation directory and click Next.
The default installation directory is \Program Files\Obliv\SHAREid Proxy.
6. Select the COREid Federation Proxy Server listener port and click Next (The default port is 8101.)
7. Indicate if the listener port is enabled for SSL and click Next. (The default port is 8113.)
8. When the confirmation screen appears, click Next.
9. When installation is complete, read the Read Me screen and click Next.
10. Read the instructions on how to start the COREid Federation Proxy Server and click Finish.

To install the COREid Federation Proxy Server on Linux or Solaris

1. Go to the COREid Federation download site provided when you purchased COREid Federation.
2. Download the COREid Federation Proxy Server installation file and extract it (if compressed).

3. From the command line, run one of the following, as appropriate:

```
./COREid_Federation_2_5_linux_Proxy_Server
```

or

```
./COREid_Federation_2_5_sparc-s2_Proxy_Server
```

To automatically create a temp folder for the installation logs, use the `-is:tempdir` switch. To run in console mode, add the `-console` switch.

Note: To run COREid Federation as a service (recommended), be sure that you are installing as root.

4. At the Welcome screen, click Next.
5. Proceed through the introductory screens. When prompted, provide the user name that the COREid Federation Proxy Server is running as, e.g., Nobody.
6. Specify the group for the COREid Federation Proxy Server user, e.g., Nobody.
7. Select an installation directory and click Next.

If you specify a new directory, the installer creates it for you.

8. Specify the following:

Proxy port number—Port number that the COREid Federation Proxy Server listens on. (The default port is 8101.)

Proxy port SSL—Is this COREid Federation Proxy Server listener port enabled for SSL. (The default port is 8113.)

Host name—Fully qualified host name for the COREid Federation Proxy Server. Note that if you do not supply a fully qualified host name, COREid Federation will not issue an error. However, not supplying a fully qualified host name will result in errors later on.

COREid Federation port—COREid Federation Server listener port. (The default port is 8101.)

SSL for COREid Federation—Is this listener port enabled for SSL. (The default port is 8113.)

9. Click Next.

The COREid Federation Proxy Server is installed.

10. When the COREid Federation Proxy Server has finished installing, read the information in the Read Me screen and click Next.

The installer presents information on how to start the COREid Federation Proxy Server.

11. Click Finish.

Starting Up and Logging In to COREid Federation

You must start the COREid Federation server before logging in to COREid Federation. After starting COREid Federation, check *SHAREid_Install_Dir/logs/shareid.log* for errors before starting to use COREid Federation.

Note that if you start COREid Federation from command line or the Windows Start menu, the COREid Federation service status will not be reflected accurately in the MMC Services window. Similarly, if you start COREid Federation as a service from the MMC Services window (COREid Federation Server service name) and stop COREid Federation from the Windows Start menu, the COREid Federation stop procedure will issue an error.

To start or stop the COREid Federation Server on Windows

1. To start the server, from the Start menu, click:

Start > Programs > COREid Federation Server (*port*) > Start COREid Federation.

To start the COREid Federation Server as a service, open the Services control panel, select the COREid Federation service name, and click Start.

There is a time lag between starting the COREid Federation service and the application becoming available. You may want to examine the Windows Performance Monitor and wait for CPU usage to drop to zero before accessing COREid Federation after starting the server.

Alternatively, you can start COREid Federation as a process from the command line:

```
SHAREid_Install_Dir/bin/catalina.bat start
```

2. To stop the COREid Federation Server, from the Start menu, click Start > Programs > COREid Federation Server (*port*) > Stop COREid Federation.

To start or stop the COREid Federation Server on Linux or Solaris

1. To start the server, enter the following command:

```
SHAREid_Install_Dir/SHAREid/bin/SHAREid start
```

There is a time lag between starting the COREid Federation service and the application becoming available.

2. To stop the COREid Federation Server, enter the following command:

```
SHAREid_Install_Dir/SHAREid/bin/SHAREid stop
```

To start the COREid Federation Administration Console on Windows

1. From the Start menu, click Start > Programs > COREid Federation Server (*port*) > COREid Federation Administration.

Alternatively, you can point to the login URL:

`http://machine name:openport/shareid/`

or

`https://machine name:sslport/shareid/`

where *machine name* is the server where COREid Federation is installed and *port* is the COREid Federation Server host listener port.

You must log in directly to the COREid Federation Server. Do not log in through the COREid Federation Proxy Server.

2. Log in using the username and password supplied during installation.

The COREid Federation Administration Console appears.

oblix Oblix SHAREid™

MAIN

SETUP

ADMINISTRATION

- IdMBridge
- Login
- Encryption
- Source Assertions
- Destination Mappings
- Domains
- Attribute Sharing
- Audits and Logs
- Assertion Store

HELP

- About
- Logout

Welcome to SHAREid 2.5

Use the links on the left as follows:

Setup is a first-time configuration wizard that is provided to simplify initial configuration of your SHAREid System. The setup wizard is optional but recommended for users who are new to SHAREid.

IdMBridge is for configuring a SHAREid engine that communicates with the COREid / LDAP directory / RDBMS / SiteMinder that processes user identities and authorizations.

Login is for configuring how users authenticate locally to SHAREid.

Certificates is for configuring a store for the SSL and digital signatures that are required for secure communication with other domains.

Source Assertions enables you to configure your domain as a source. A source domain is where a request for a resource originates. If your users want to access resources on other domains, you are a source. Source assertions contain information that destination domains use to authenticate your users.

Destination Mappings is for configuring your domain as a destination. If users from other sites want to access your resources, you are a destination domain. Using destination mappings, you map assertion attributes to local user identities.

Domains enables you to configure properties for your domain and other source and destination domains, for example, the hosts and ports used for communication.

Attribute Sharing allows clients within the local domain to use SHAREid to determine if a user from another domain has attributes with specified values. This is used by the COREid Access Server for authorization of user requests based on attribute values.

Audits and Logs enables you to determine if log and audit data is collected. Text descriptions of system events are written to a log file, and generated and received assertions are written to an audit file.

Assertion Store enables you to choose the method for storing assertions.

Internet

To start the COREid Federation Administration Console on Linux or Solaris

1. Point to the login URL:

`http://machine name:openport/shareid/`

or

`https://machine name:sslport/shareid/`

where *machine name* is the COREid Federation host and *port* is the host's listener port.

You must log in directly to the COREid Federation Server. Do not log in through the COREid Federation Proxy Server.

2. Log in using the username and password supplied during installation.

The COREid Federation Administration Console appears.

Configuring COREid Federation

Basic setup for COREid Federation consists of configuring your domain as a source, a destination, or both.

- A source domain is the location that authenticates users who want to access protected resources on a remote Web site.
- A destination domain is the location that contains the application resources that users want to access.

You can use the COREid Federation Administration Console to configure COREid Federation as either a source domain, destination domain, or both. The Administration Console's Setup option provides a wizard-based approach to COREid Federation configuration, performing the most basic operations to configure your domain and get the domain up and running. Following initial setup, you can choose other Administration Console options to make additional changes, customize or tailor your domain configuration for more specialized, advanced or custom operation.

See Chapter 4, "Configuring Source Domains" on page 95, for more information and details on configuring COREid Federation for operation as a source domain. See Chapter 5, "Configuring Destination Domains" on page 169, for more information and details on configuring COREid Federation for operation as a destination domain.

Chapter 6 provides information on specialized and advanced COREid Federation configuration options such as SmartMarks and SmartWalls, login and error message customization, assertion store load balancing and failover setup, and COREid Federation configuration to use X.509 Authentication-based Attribute Sharing Profiles.

Deploying COREid Federation

As part of deploying COREid Federation in a network of trusted source and destination domains, you'll need to exchange information between sites and configure your source and destination domains accordingly. For example, you may need to provide the destination with a list of users from your domain, if the destination domain administrator wants to map each user to a unique identity at the destination. This depends on how the destination plans to map your users to identities at the destination. The destination can map a subset of your users to a single identity based on a role, or it can create a 1:1 correspondence between each of your users and corresponding destination identities. However, a destination cannot create many-to-many mappings or one (source)-to-many (destination) mappings.

COREid Federation provides an option for a destination to automatically create new identities when a new user from your domain accesses a destination resource. See “Automatically Mapping User Identities Using SmartMaps” on page 208 for details.

Before deploying COREid Federation in a production environment, you also need to establish cross-domain trust by setting up authentication and exchanging keys or certificates between the various source and destination domains in your network. To establish trust between the domains, you'll need to perform the following tasks:

- Some configuration is required for setting up certificates for secure communications. Change the keystore password and replace the default self-signed certificates with CA certificates. See “Configuring Keys and Certificates” on page 243 for details. When you first install COREid Federation, most of the default SAML URLs use SSL, except for the Responder service, which uses the Open port. For initial setup and testing, the source and the destination can use default self-signed certificates that are provided with COREid Federation. However, you will want to ensure that your configuration uses third-party CA certificates before going into production.
- If you are using the Artifact profile with Basic authentication for the Requester, change the Requester password.

If you are a source domain, see “Configuring a Destination Domain for a Source Domain” on page 155 for details on configuring the Requester for an external destination domain. If you are a destination domain, see “Configuring MyDomain” on page 184 for details on configuring the Requester for MyDomain.

- Regenerate the encryption key.

If you are a source domain, see “Configuring Passwords and the Certificate Store” on page 136 for details. If you are a destination domain, see “Configuring Password and Certificate Store Encryption” on page 196 for details.

As mentioned previously, you can get more information on making changes to COREid Federation configuration in the following two chapters: Chapter 4, “Configuring Source Domains,” and Chapter 5, “Configuring Destination Domains” .

Chapter 7 provides information on configuring keys and certificates for COREid Federation Servers and COREid Federation Proxy Servers at both source and destination domains, if installed. Chapter 8 provides more information on configuring COREid Federation auditing and logging,

Appendix A provides more information on using SSL and client certificate authentication in COREid Federation. Appendix B provides information on troubleshooting COREid Federation problems and tuning COREid Federation configuration for optimal performance.

Uninstalling COREid Federation

Before uninstalling COREid Federation, stop the COREid Federation server. You can then run the COREid Federation uninstaller program to remove COREid Federation and associated files and programs.

Important: If you do not stop the COREid Federation server, you will need to restart the COREid Federation host machine after completing the uninstall. If you have to uninstall COREid Federation and you are using a COREid IdMBridge, you must manually delete the policies that COREid Federation creates in the COREid Access System. See “Policies Created in COREid” on page 69 for details.

To uninstall COREid Federation on Windows Platforms

1. Open the Windows Control Panel.
2. Open the Add/Remove Programs control panel.
3. Select the COREid Federation Server 2.5.X option from the list of installed programs.
4. Click the Change/Remove button.
5. This displays the COREid Federation InstallShield uninstaller program dialog box.
6. Click Next to proceed with uninstalling COREid Federation and all other associated files.
7. When you’ve finished, remove the COREid Federation installation directory and any remaining extraneous files that may still remain after the uninstall.

To uninstall COREid Federation on Linux or Solaris Platforms

1. From the command line, change directories to your current COREid Federation install directory.
2. Change to the `_uninstSHAREid` subdirectory.
3. From this directory, run the following command:
`./uninstaller.bin`
To run in console mode, add the `-console` switch.
4. Follow the prompts to confirm the COREid Federation uninstall.
5. When you've finished, remove the COREid Federation installation directory and any extraneous files that may still remain after the uninstall.

4

Configuring Source Domains

As a source domain, you are responsible for authenticating users who request access of resources that reside on other external service provider or destination domains. From the COREid Federation Administration Console, you can configure settings for your local domain, when operating as a source domain, including configuring the IdMBridge (how COREid Federation finds information about local users), defining user login and creating assertion profiles, and specifying the destination domains that users may access.

This chapter covers the following topics:

- “Using the COREid Federation Administration Console” on page 96
- “About Source Domain Configuration” on page 98
- “Using the Setup Wizard” on page 99
- “Source Domain Administration from the COREid Federation Console” on page 112
- “Configuring the IdMBridge” on page 113
- “Configuring a Login Method” on page 133
- “Configuring Passwords and the Certificate Store” on page 136
- “Configuring Assertion Profiles” on page 139
- “Configuring Domains” on page 147

Using the COREid Federation Administration Console

All operations to configure a COREid Federation Server for operation as a source domain, destination domain, or both, may be performed using the COREid Federation Administration Console. The Administration Console provides a Web browser-based interface through which you can select navigation bar menu options to perform specific COREid Federation configuration tasks. To use the Administration Console, you must have already started the COREid Federation Server on the machine you want to configure.

To start the COREid Federation Administration Console

1. From an open Web browser window, enter the login URL:

`http://machine name:openport/shareid/`

or

`https://machine name:sslport/shareid/`

where *machine name* is the server where COREid Federation is installed and *port* is the COREid Federation host listener port.

Note: On a Windows machine, you can also start the Administration Console from the Start menu, clicking Start > Programs > COREid Federation Server (*port*) > COREid Federation Administration.

You must log in directly to the COREid Federation server. Do not log in through the COREid Federation proxy server, if installed.

2. Log in using the username and password supplied during installation.
The COREid Federation Administration Console Main page appears.

The screenshot shows the Oblix SHAREid 2.5 administration console. The top header includes the 'oblix' logo and 'Oblix SHAREid™'. The left navigation menu is organized into sections: MAIN, SETUP, ADMINISTRATION (with sub-items: IdMBridge, Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, Assertion Store), and HELP (with sub-items: About, Logout). The main content area is titled 'Welcome to SHAREid 2.5' and contains the following text:

Use the links on the left as follows:

- Setup** is a first-time configuration wizard that is provided to simplify initial configuration of your SHAREid System. The setup wizard is optional but recommended for users who are new to SHAREid.
- IdMBridge** is for configuring a SHAREid engine that communicates with the COREid / LDAP directory / RDBMS / SiteMinder that processes user identities and authorizations.
- Login** is for configuring how users authenticate locally to SHAREid.
- Certificates** is for configuring a store for the SSL and digital signatures that are required for secure communication with other domains.
- Source Assertions** enables you to configure your domain as a source. A source domain is where a request for a resource originates. If your users want to access resources on other domains, you are a source. Source assertions contain information that destination domains use to authenticate your users.
- Destination Mappings** is for configuring your domain as a destination. If users from other sites want to access your resources, you are a destination domain. Using destination mappings, you map assertion attributes to local user identities.
- Domains** enables you to configure properties for your domain and other source and destination domains, for example, the hosts and ports used for communication.
- Attribute Sharing** allows clients within the local domain to use SHAREid to determine if a user from another domain has attributes with specified values. This is used by the COREid Access Server for authorization of user requests based on attribute values.
- Audits and Logs** enables you to determine if log and audit data is collected. Text descriptions of system events are written to a log file, and generated and received assertions are written to an audit file.
- Assertion Store** enables you to choose the method for storing assertions.

The bottom of the screenshot shows a taskbar with an 'Internet' icon.

The Administration Console displays a left-side navigation bar from which you can select menu options to perform specific COREid Federation configuration tasks. The option labelled MAIN is the page first displayed when you start up the Administration Console. Text on this page provides a summary description of each main selection in the navigation bar. Below the Main starting page, the option labelled SETUP provides an option to run the COREid Federation Setup Wizard, which performs a step-by-step initial configuration of a domain. Below SETUP, is a section labelled ADMINISTRATION, which provides options to select and perform individual COREid Federation configuration and setup tasks. From these options, you can configure new or update existing configuration items.

The remaining sections of this chapter, provide instructions on using the Setup Wizard for source domain configuration followed by domain configuration using individual Administration menu selections. To obtain additional information while using the Administration Console, click the Help button to display a PDF copy of this guide. In addition, when you've finished COREid Federation configuration, you can click the Logout menu option to close and exit the Administration Console.

About Source Domain Configuration

As a source domain, you can communicate with one or more destination domains. The steps for configuring a source domain are summarized below.

Task overview: configuring a source domain

1. Configure the IdMBridge to communicate with a data repository.

The IdMBridge is the COREid Federation component that communicates with your user data repository. COREid Federation uses this information to authenticate users and construct assertions.

For a source domain, the IdMBridge can communicate with an LDAP directory, COREid, a relational database, or SiteMinder.

2. Specify how users authenticate to COREid Federation.
3. Configure assertion profiles.

An assertion profile is a template that tells COREid Federation what to put in an assertion. The destination domain extracts data from the assertion and maps it to an identity at the destination.

4. Configure your local domain.
5. Add accessible destination domains.

You must configure at least one destination domain accessible from your source domain.

6. Exchange configuration information with each destination domain your local source domain can access. This is required so that the destination can accept assertions from your domain. See “Exchanging Information Between Domains” on page 77 for more information.

Methods Available for Configuration

The COREid Federation application provides two methods for source domain setup:

- From the COREid Federation Administration Console, select the Setup Wizard option, recommended for first-time users of COREid Federation, that performs basic COREid Federation configuration for operation as a Source domain.
- Following initial COREid Federation setup, choose the COREid Federation Administration Console’s options (located in the Administration section of the left-side navigation bar), pertaining to specific tasks to reconfigure or customize COREid Federation configuration. In some cases, the Administration section menu options provide additional configuration options and parameters not available in the Setup Wizard.

After initial setup of COREid Federation, certain tasks that are presented in the COREid Federation Administration Console’s Setup Wizard, for example, configuring user login, do not need to be performed again. You can perform other tasks such as adding destination domains and creating new assertion profiles, or updating current COREid Federation configuration settings, using the corresponding Administration menu option specific to each task.

Using the Setup Wizard

If you are configuring your local domain for the first time, you may want to use the COREid Federation Administration Console’s Setup Wizard option first. This wizard guides you through all of the steps required for basic domain setup. For a source domain, the configuration steps are the following:

1. Configure the IdMBridge to communicate with your user identity management system.
2. Configure how users log in to COREid Federation.
3. Configure an assertion to send to destination domains.
4. Configure your local domain (MyDomain).

5. Configure your first destination domain.

Note: When you access COREid Federation for the first time, it takes a while for pages to appear after clicking the links or the Next button in the Setup Wizard. This is due to .JSP page compilation that occurs when the pages are accessed for the first time.

The following sections describe the operations

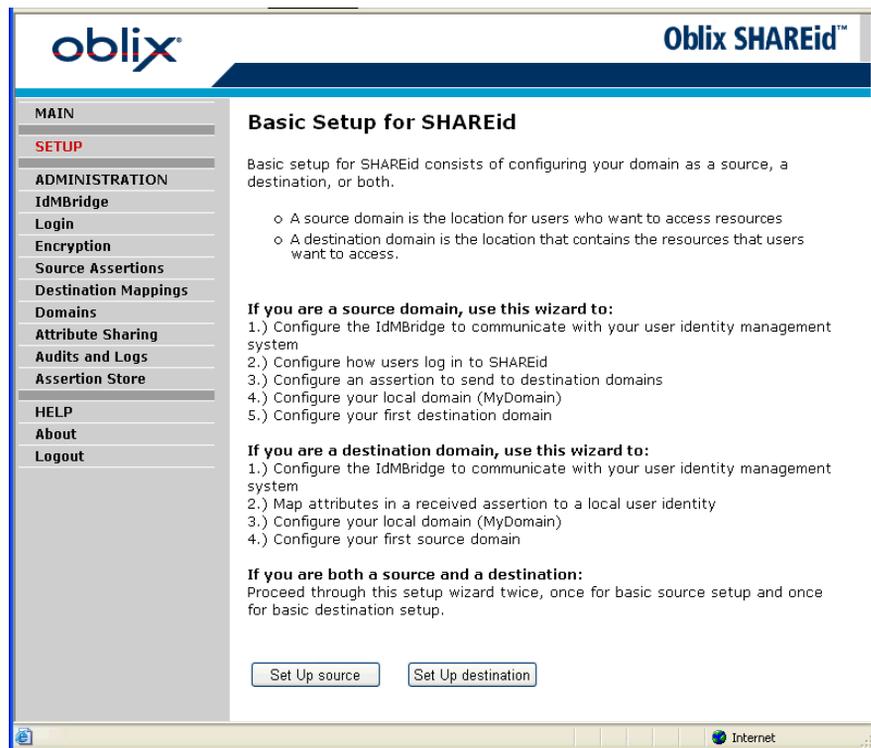
To start the Setup Wizard

1. Open the COREid Federation Administration Console.

See “Starting Up and Logging In to COREid Federation” on page 87 for details.

2. Click the SETUP link.

The first page of the Setup wizard appears.

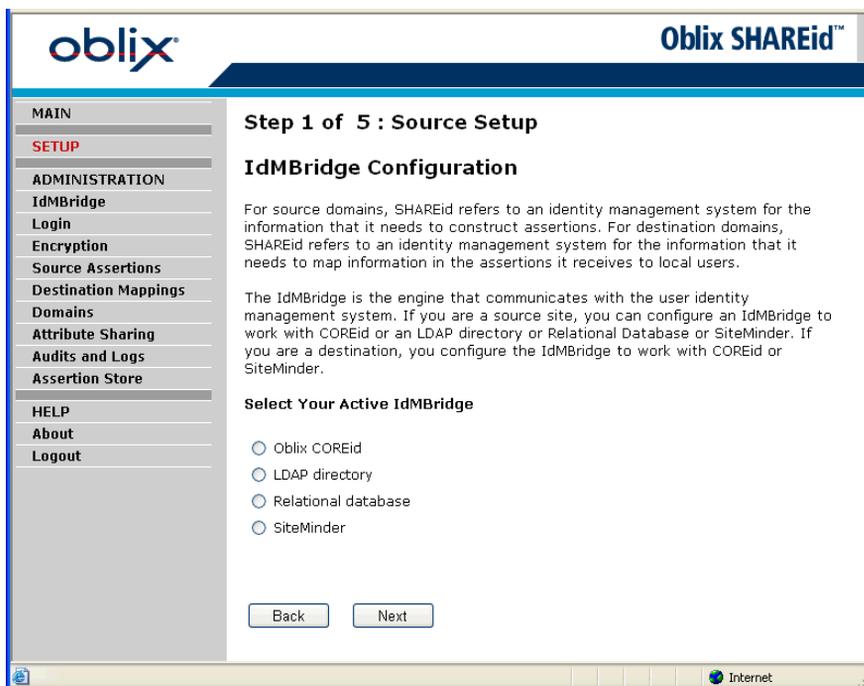


3. To configure your installation as a source domain, click Set Up Source.

Note: To set up a domain to operate both as source and a destination, go through this Setup Wizard twice, once for basic source setup and once for basic destination setup.

Step 1: Configure the COREid Federation IdMBridge

After selecting the Set Up Source option, you are prompted to select which type of IdMBridge you want to configure.



4. Select one of the available options:

- Oracle COREid
- LDAP directory
- Relational Database (RDBMS)
- SiteMinder

5. Click Next.

The COREid Federation Administration Console now displays web pages specific to the IdMBridge type you selected, to let you configure the IdMBridge's specific parameter settings.

Refer to the section “Configuring the IdMBridge” on page 113 for a description of configuration parameters provided for each IdMBridge type: Oracle COREid, LDAP Directory, RDBMS, or SiteMinder.

Step 2: Configure Login using the Setup Wizard

After configuring the IdMBridge, the COREid Federation Administration Console displays a web page to let you configure how users can connect or log into COREid Federation.

The screenshot shows the Oblix SHAREid™ administration console. The left sidebar contains a navigation menu with the following items: MAIN, SETUP (highlighted in red), ADMINISTRATION, IdMBridge, Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, Assertion Store, HELP, About, and Logout. The main content area is titled "Step 2 of 5 : Source Setup" and "Configure Login".

When SHAREid validates a user's identity, one of the following scenarios may apply:

- The user points the browser to a SHAREid login page.
- The user clicks a link to a resource on the destination domain, and is redirected back to the source's SHAREid. SHAREid challenges the user for login credentials. For this scenario to be supported, the SmartMarks function must also be enabled. See the help or the SHAREid documentation for details.
- The user accesses a resource that is protected by an identity management system supported by the SHAREid IdMBridge, for example, the Oblix COREid System. In the case of login to COREid, a single sign-on cookie is set. The user can then access SHAREid enabled resources without logging in again.
- The user logs in to a local portal. The portal sets a header variable that is used by SHAREid to identify the user in the SHAREid data store (for example, LDAP). The user can then access SHAREid-enabled resources without logging in again.

This page identifies the method that SHAREid uses to challenge users for login credentials.

To present the user with a login dialog for authentication, you can select a basic login dialog or Web login form as the login method. If you choose a Web Login Form, a default login form will be provided. You will be able to customize this form. Click the Help link for details.

Authentication Method

Basic Login Dialog

Information to Appear in the Realm Field on this Dialog

Web Login Form

External Credentials
(space-separated list of header variables set by an external authentication mechanism)

SSO Domain

SSO Domain (e.g., ".company.com")

Buttons: Back, Next

System tray: Done, Internet

- From the login page, select the type of login you wish to support.

Method	Description
Basic Login Dialog	Users enter a user ID and password in a pop-up window supplied by the Web server.
Web Login Form	<p>Users supply credentials in a login form. COREid Federation provides a default form that you can modify. See “Customizing the Login Form” on page 209 for details.</p> <p>For a Web login form, supply a path to the form. The default path is:</p> <p><i>SHAREid_Install_Dir</i>/webapps/shareid/login.jsp</p> <p>where <i>SHAREid_Install_Dir</i> is the directory where COREid Federation is installed.</p>
External Credentials	<p>A challenge method outside COREid Federation is used. When the user attempts to access a COREid Federation-enabled resource, COREid Federation looks for a header variable set by the external application.</p> <p>Use this field to specify one or more header variables that are set by the external application and that can be used to identify the user in the COREid Federation user data repository. Use a space to separate header variables.</p>

Note: If you change the authentication method that COREid Federation uses, restart the COREid Federation Server.

For additional information on Login configuration options and also performing Login configuration from the Administration Console’s Login menu option, see “Configuring a Login Method” on page 133, later in this chapter. Also refer to “Planning COREid Federation Source Authentication” on page 61 in Chapter 2, “Planning a COREid Federation installation.”

- Optionally, specify the SSO Domain to be used to set SSO cookies.

The SSO domain is the DNS domain to be used to set the SSO cookie at either the source or destination site. If you do not specify an SSO domain, by default, COREid Federation will determine the cookie domain from the COREid Federation Server host name. For example, for a COREid Federation host of shareid.company.com, the SSO domain used will be .company.com, and for shareid.division.company.com, it will be .division.company.com

- Click Next.

Step 3: Add an Assertion profile using the Setup Wizard

After configuring the parameters for the Configure Login page, the Administration Console displays a web page to let you configure an assertion profile.

The screenshot shows the Oblix SHAREid Administration Console interface. The top navigation bar includes the Oblix logo and 'Oblix SHAREid™'. A left sidebar menu lists various sections: MAIN, SETUP (highlighted), ADMINISTRATION (with sub-items: IdMBridge, Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, Assertion Store), and HELP (with sub-items: About, Logout). The main content area is titled 'Step 3 of 5 : Source Setup' and 'Add Assertion Profile'. It contains a descriptive paragraph, a form with fields for Assertion Profile Name, Description, Issuer, Subject Name Qualifier, Subject Format (set to 'None'), and User Attribute for Subject. Below this is a section 'Add Assertion Attributes to Your Assertion Profile' with another descriptive paragraph and a table for 'Assertion Attribute Mappings'. The table has columns for Assertion Attribute, Attribute in Data Store, Name Space, Optional Type, In SSO Assertions, and Allowed Values. An 'Add New Row' button is at the bottom right of the table. At the bottom of the page are 'Back' and 'Next' buttons. The browser status bar at the very bottom shows 'Done' and 'Internet'.

Step 3 of 5 : Source Setup

Add Assertion Profile

When users request resources, the source site provides the destination site with an assertion that attests to the user's identity. An assertion profile defines the contents of the assertion that is sent to the destination.

Assertion Profile Name

Description

Issuer (e.g. http://host.company.com)

Subject Name Qualifier

Subject Format

User Attribute for Subject

Add Assertion Attributes to Your Assertion Profile

In the following fields, you configure attribute mapping in the assertion profile. To configure an attribute mapping, you create a one-to-one correspondence (a mapping) between assertion attributes that a destination domain administrator gives to you and user attributes in your domain's data store. The destination domain's administrator must tell you the names of the assertion attributes.

Assertion Attribute Mappings					
Assertion Attribute	Attribute in Data Store	Name Space	Optional Type	In SSO Assertions	Allowed Values
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text"/>

9. Define at least one assertion profile for a new source domain.

The Setup Wizard assists you with creating the first profile. You can add more assertion profiles after you finish the Setup Wizard by selecting the Administration Source Assertions menu option. The following table provides a description of the entries provided on the Assertion Profile Setup Wizard page:

Field	Description
Assertion profile name	Any unique name. For example, you could specify a name that indicates the name of the destination domain that will receive the assertions that are created using this profile. Or, the name could indicate the association being made between an attribute stored in a user data repository and the assertion attribute, for example: Mail-to-SubjectName
Description	Text describing or summarizing contents of the profile.
Issuer	Name of the SAML authority who issued the assertion. Identify the issuer with an unambiguous name. You can use a name or the uniform resource identifier (URI) of your host domain, for example: <code>http://host.domain.com</code> where <i>host.domain.com</i> is the DNS host name for the COREid Federation host, for example: <code>http://company.com</code> .
Subject name qualifier	Optional subject name qualifier allows the SAML-compliant server to determine the namespace for a user. The source domain may find this field to be useful for informational purposes. For example, an assertion for <code>jsmith@oblix.com</code> , may contain J. Smith's department. Example: <code>ou=Department,o=Company,c=US</code>

Field	Description
Subject format (Default is None)	<p>Format of the Subject who is the user who is 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 a list of attribute formats:</p> <p>None—No format is specified.</p> <p>Email address—The NameIdentifier is formatted as an email address.</p> <p>X.509 subject name—The NameIdentifier is in the form of a DN, for example, cn=myname,dc=example,dc=com.</p> <p>Windows domain—A Windows domain qualified user name formatted as DomainName\UserName, for example, oblix\jsmith.</p> <p>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.</p> <p>Other—Other is a format value that is not one of the predefined values defined in the SAML spec. A format of Other allows any other URI values that SAML partners might use. Hypothetical examples: "<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 for Other is specified in the shareid-config.xml file.</p>
User Attribute for Subject	<p>This is the local user attribute or RDBMS field name whose value will be used as the value for the Subject assertion element. (More specifically, this field determines the value of the NameIdentifier sub-element of the assertion's Subject element.) The Subject element identifies the source domain user.</p> <p>If this field is left blank, the DN of the user's directory entry is used with LDAP IdMBridge, and COREid or SiteMinder IdMBridges using LDAP data stores. For RDBMS IdMBridges or SiteMinder IdMBridges accessing a database, the user's login name is used.</p>
Assertion attribute mapping table	See individual field description in table below.

10. In the Assertion Attributes table, add one or more user attributes and the assertion attributes that they map to, for example.

Assertion Attribute—UserMail

Attribute in Data Store—mail

The Assertion Attribute mapping table configures the AttributeStatement element in an assertion. If this table is empty, the destination domain uses the SubjectName assertion element when it maps your users to destination identities.

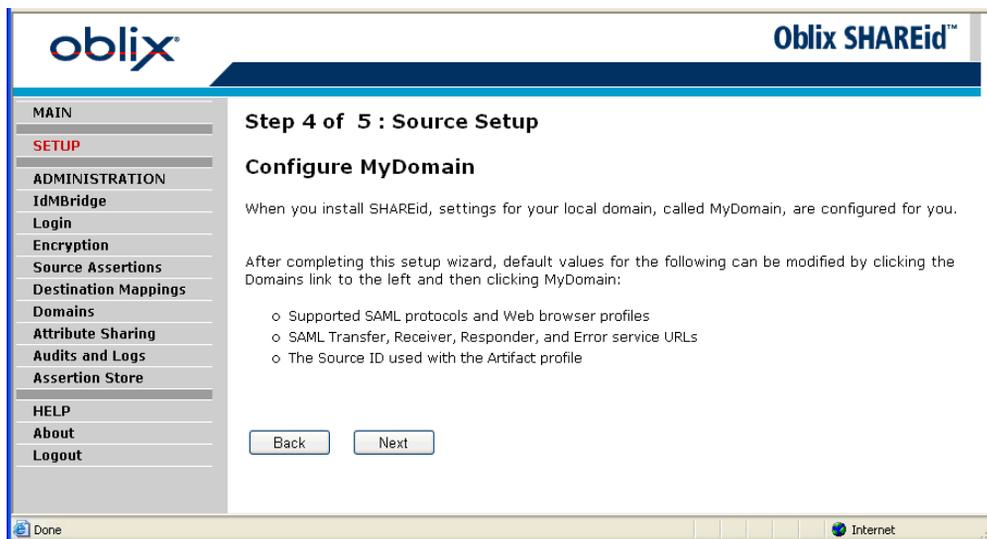
The following table describes each of the fields you can define for an assertion attribute mapping.

Field	Description
Assertion Attribute	Name to be provided in the Assertion Attribute statement. This attribute is usually provided by the destination domain.
Attribute in Data Store	Attribute or field name in local data store where you can retrieve attribute value
Name Space	Optional for SSO Assertions, otherwise needs to be specified. Allows you to specify namespaces if the assertion profile may be used by multiple applications.
Optional Type	Defines the xsi:type attribute of the <Attribute> in the statement. The types are defined in the XML Schema specification. If the type is omitted it will be assumed to be string.
In SSO Assertion?	Select this checkbox if you want to include the attribute in assertions used in Web-browser profiles (Artifact or POST). Selecting this checkbox is not required for assertions using the attribute sharing profile or authorization queries.
Allowed Values	Restrict attribute values returned in an assertion to only those specified in this list.

11. When you've finished defining an assertion profile and assertion attribute mappings, click Next.

Step 4: Configure MyDomain using the Setup Wizard

After configuring the parameters for the Add Assertion Profile page, the Administration Console displays a web page to confirm automatic configuration of MyDomain default parameters.



This display just reminds you that when you installed COREid Federation, certain defaults were initially set for the local domain. Default configuration options set during installation are the following:

- SAML protocols and Web browser profiles
- URLs for SAML transfers
- Error services that are used by your COREid Federation instance.

You can change these local domain (MyDomain) configuration settings later, for example, if you want to add a COREid Federation Proxy, by choosing the Domains menu option from the COREid Federation Administration Console and then selecting the MyDomain configuration from a list of configured domains.

For more information on MyDomain configuration, see “Configuring Domains” on page 147.

12. Click Next to continue to the next step in the Setup Wizard.

Step 5: Configure a Destination Domain using the Setup Wizard

After completing step 4, the Administration Console displays a web page (step 5) to let you configure your first destination domain.



Note that you can skip this section of the Setup Wizard if you do not yet have enough information about your destination domain to configure it.

If you choose to configure a destination domain using the Setup Wizard, you can choose from the following options to specify the type of destination domain that your local domain (MyDomain) will communicate with:

- **COREid Federation**—The destination domain has installed COREid Federation. If you select the COREid Federation destination domain type, the Configure a COREid Federation Destination Domain page appears.

From this page, you can specify the name of the COREid Federation destination domain (e.g., example.com), the fully qualified DNS name of the machine hosting the federation services (for example, host1.example.com), and the port numbers that the destination host listens on. When you click Next, the Setup Wizard uses this information to configure the COREid Federation URLs for this domain.

- **Other SAML-compliant products**—The destination domain has installed a product that supports the SAML 1.0 or 1.1 protocol, but does not have COREid Federation.

If you selected a destination domain that uses another SAML-compliant product, the Configure a Destination Domain page appears.

- **Skip**—The destination domain does not yet have configuration information that you can use, so you will configure it later using the Administration > Domain menu options.

If you choose Skip, the End of Source Setup page appears.

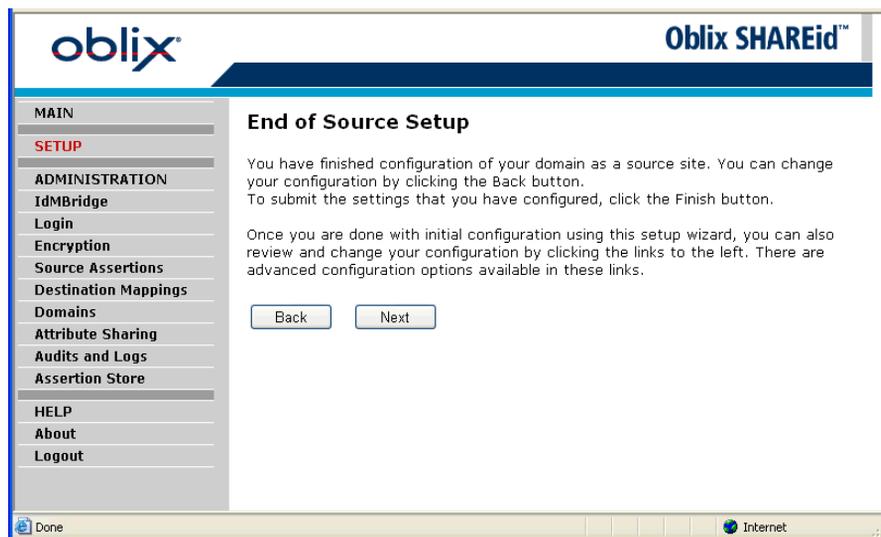
13. After choosing to configure a destination domain as part of this step of the wizard, or choosing Skip, click Next.

If you chose to configure a destination domain in step 5, COREid Federation now uses this information to configure the SAML services URLs for this domain.

Note: To configure other information for this domain after finishing the Setup Wizard, click the Administration > Domains link from the left-side navigation bar. For more information on configuring domains, see “Configuring a Destination Domain for a Source Domain” on page 155.

Setup Wizard - End of Source Setup

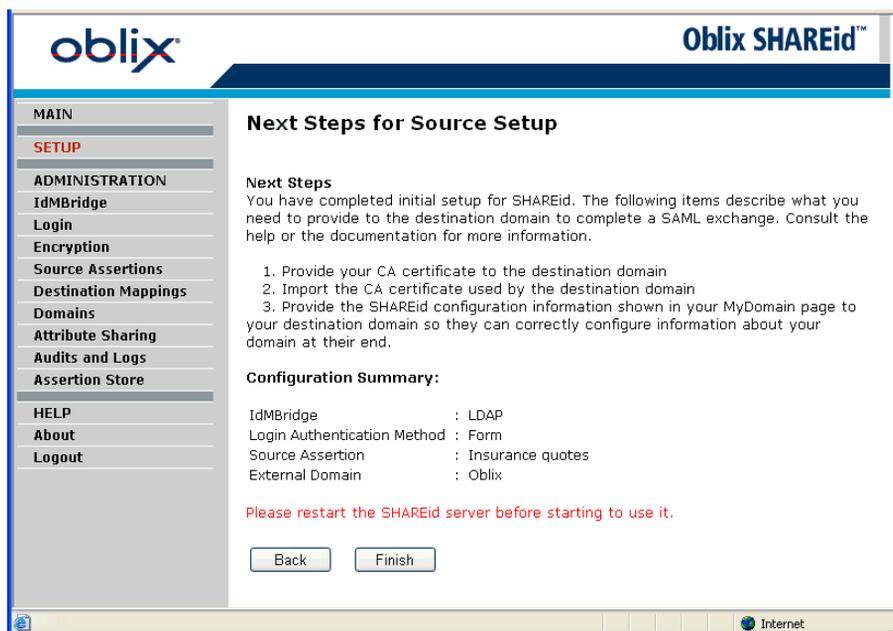
After completing Step 5, the Setup Wizard now displays the End of Source Setup page:



14. Click Next if you're done making changes to your source domain configuration.

Note: If you click Back from the End of Source Setup page, the wizard presents the full configuration page for this destination. See “Configuring a Destination Domain for a Source Domain” on page 155 for details on the fields on this page.

The Setup Wizard displays a Web page providing a summary of the information that you configured using the Setup Wizard.



Once you have completed COREid Federation configuration, you need to exchange the following information with your partner domain:

- Your domain information, which includes the Transfer Service, Responder Service, and Receiver Service URLs. These URLs include the host name and port numbers of your COREid Federation or COREid Federation Proxy Server.
- If you are also a destination domain, you need to obtain the CAs of the COREid Federation and COREid Federation Proxy Servers at the source sites.
- If you are using the X.509 client certificate mode for requester authentication, you need to add the CA of the destination COREid Federation server to your Proxy Server. (In your proxy server configuration, you create a ca-bundle.crt file which contains the CA of the destination site).

You also need to configure trust with the destination domain, as described in “Configuring Keys and Certificates” on page 243.

15. After capturing the summary configuration information, click Finish to exit the Setup Wizard and return to the main Administration Console page.

Note that you’ll also need to restart your local COREid Federation server for the configuration changes to take effect.

After running Setup, if you need to close down COREid Federation, do not force the shutdown by clicking End Now if presented with a Program Not Responding dialog. Allow the program to finish its own shutdown processes.

If you and the destination have completed configuration, you can test sending an assertion. See “Sending Test Assertions to a Destination” on page 145 for details.

Source Domain Administration from the COREid Federation Console

In addition to using the Setup Wizard, the COREid Federation provides source domain configuration from individual navigation bar menu options available from the Administration Console’s left-side navigation bar. For first-time configuration of COREid Federation, it is recommended that you use the Setup Wizard. (See “Using the Setup Wizard” on page 99.) The Wizard ensures that you perform all of the tasks required for initial configuration. After performing initial setup, use the standard administration pages rather than the Setup Wizard.

The remaining sections of this chapter describe configuration of domains using the menu options available in the Administration section of the navigation bar. The sections are included in the order in which the menu options appear in the navigation bar:

- **IdMBridge**—for a source domain, configure the user data repository from which the IdMBridge finds information about your local users. Options available are: Oracle COREid System, an LDAP directory, a relational database, or SiteMinder.

See “Configuring the IdMBridge” on page 113 for more information on configuring an IdMBridge using the IdMBridge menu option. See “Step 1: Configure the COREid Federation IdMBridge” on page 101 for information on configuring an IdMBridge using the Setup Wizard.

- **Login**—for a source domain, configure how users authenticate locally to COREid Federation. See “Configuring a Login Method” on page 133 for information on using this option.
- **Encryption**—configure a store for the SSL and digital signatures that are required for secure communication with other domains. Two submenu options are provide. The Certificates option lets you generate a new key for encrypting COREid Federation login sessions and passwords. The Password option lets you specify passwords and the location of the certificate store. See “Configuring Passwords and the Certificate Store” on page 136 for more information.
- **Source Assertions**—for source domains, configure assertion profiles that define the information placed in assertions. The profiles determines the information that destinations can use to authenticate users from your domain. The profiles also specify mappings of attributes provided by a destination domain to local identity information stored in the source domain’s user data

repository. See “Configuring Assertion Profiles” on page 139 for more information.

- **Destination Mappings**—for destination domains, map attributes from assertions received from a source domain to local user identities at the destination. For more information on using this option, refer to “About Destination Domain Configuration” on page 172.
- **Domains**—configure properties for your domain (MyDomain) and other source and destination domains with which your domain will communicate. For more information on configuration of your local domain as a source domain, see “Configuring Domains” on page 147.
- **Attribute Sharing**—configure X.509 Certificate-authenticated clients within the local domain to use COREid Federation to determine if a user from another domain has attributes with specified values. Verification of these attribute values from the client’s home domain is used by the COREid Access Server for authorization of user requests of protected resources. For more information on using this option, refer to “Configuring COREid Federation to Use X.509 Attribute Sharing Profiles” on page 221.
- **Audits and Logs**—configure and enable COREid Federation audit and system log recording. COREid Federation writes text descriptions of system events to a log file. When enabled, generated and received assertions are written to an audit file or database. For more information, see Chapter 8, “About COREid Federation Audit and System Logs” on page 277.
- **Assertion Store**—configure COREid Federation source or destination domains to store assertion information in local files or a database. Most commonly used in situations where you want to load-balance COREid Federation Servers and need to provide a common data store each domain server can access. For more information, see Chapter 6, “Setting up COREid Federation Load Balancing and Failover” on page 215.

Configuring the IdMBridge

The IdMBridge on a source site is used to authenticate users and populate federation assertions. You can configure the Oracle COREid System, an LDAP directory, a relational database, or SiteMinder to be the user data repository from which the IdMBridge finds information about your local users. The following sections provide information on configuring each of the IdMBridge types from the COREid Federation Administration Console.

For background information on specific IdMBridges, see Chapter 2, “Planning a COREid Federation Installation” on page 45:

- “Planning for a COREid IdMBridge (Source or Destination Domains)” on page 56

- “Planning for an LDAP IdMBridge (Source Domain Only)” on page 52.
- “Planning for a COREid IdMBridge (Source or Destination Domains)” on page 56.
- “Planning for a SiteMinder IdMBridge (Source or Destination Domain)” on page 61.

Setting the Active IdMBridge

If you used the Setup Wizard, you would have configured an IdMBridge to use COREid, an LDAP directory, a relational database, or SiteMinder and the IdMBridge you configured would be designated the active IdMBridge, by default.

To change the active IdMBridge

1. From the COREid Federation Administration links, click IdMBridge.

The IdMBridge Configuration page appears.



2. Select which IdMBridge you want to activate, that is, a bridge configured to access a Oracle COREid System, an LDAP directory, a relational database, or SiteMinder.

3. Click Submit.

Depending on the IdMBridge selected, the Console displays additional screens showing IdMBridge details.

4. Verify that the IdMBridge information is complete and click Submit.
5. Restart the COREid Federation server.

Configuring a COREid IdMBridge

You can configure a COREid IdMBridge when you first set up COREid Federation using the Setup Wizard or choose the Administration IdMBridge > COREid menu option from the Administration Console's Navigation Bar.

For an overview of the COREid IdMBridge, see “Choosing an IdMBridge” on page 52. The following information and procedure assumes that COREid has already been configured for use with COREid Federation. See “Planning for a COREid IdMBridge (Source or Destination Domains)” on page 56 for details.

Note: When you configure a COREid IdMBridge using the Setup Wizard, the COREid Federation Administration Console first prompts you to specify COREid AccessGate and Access Server configuration parameters. (See “To configure the COREid AccessGate and Access Server” on page 118.) Then, if you've specified an Open and Simple Connection type, it prompts you for the COREid Master Access Administrator configuration, as shown below on the IdMBridge: COREid Configuration page.

For more information on using the Setup Wizard, see “Using the Setup Wizard” on page 99.

To configure the COREid IdMBridge

1. From the COREid Federation Administration links, click IdMBridge.

The IdMBridge Configuration page appears in the right-hand pane, and IdMBridge choices appear below the IdMBridge link in the left-hand pane: COREid, LDAP, RDBMS, or SiteMinder.

2. In the left-hand navigation panel, click COREid.

The IdMBridge: COREid Configuration page appears.

The screenshot shows the Oblix SHAREid web interface. The left-hand navigation panel is expanded to show the 'ADMINISTRATION' section, with 'IdMBridge' selected. The main content area is titled 'IdMBridge: COREid Configuration'. It contains the following sections:

- Introduction:** The following information controls SHAREid's communication with the COREid System at your local site. Before you configure your IdMBridge for COREid, you need to configure an AccessGate in the Access System Console, and the Access Management Service must be turned on in the Access Server and AccessGate.
- Note:** Note that you can configure primary and secondary servers using the COREid Access System.
- Master Administrator Information:** This section contains three input fields: 'Master Admin Name', 'Master Admin Password', and 'Confirm Master Admin Password'.
- SAML Authorization Result:** This section contains a dropdown menu for 'Authorization result for unprotected resources' with the value 'Allow' selected.

At the bottom of the form, there are two buttons: 'Submit' and 'Reset'.

3. Provide the Master Access Administrator login ID and password.

This administrator can perform any task in the Access System except creating other Master Access Administrators. See the *NetPoint/COREid Administration Guide* for details. You can obtain the Master Access Administrator login ID from your COREid Access System administrator.

The Master Access login information is used to bind to the user data repository for authentication and obtaining user attributes for assertions. It is also used when COREid Federation automatically creates COREid Federation policies and authentication schemes.

- Optionally, change the setting in the SAML Authorization Result field (option not available from the Setup wizard).

The SAML Authorization Result field value determines what happens if a user from a source site tries to access a resource on a destination site that is not protected by the COREid Access System. Options available are to Allow access (the default), explicitly Deny access, or Indeterminate. You can obtain information about how to set this field from your COREid Access System administrator.

- Click Submit.

The IdMBridge: COREid: Server Configuration page appears.

The screenshot shows the Oblix SHAREid™ web interface. The left sidebar contains a navigation menu with categories: MAIN, SETUP, ADMINISTRATION, IdMBridge (with sub-items: COREid, Server Config, LDAP, RDBMS, SiteMinder), Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, Assertion Store, HELP, About, and Logout. The main content area is titled "IdMBridge: COREid: Server Configuration". It contains the following text: "The following information controls SHAREid's communication with the COREid System at your local site. Before you configure your IdMBridge for COREid, you need to configure an AccessGate in the Access System Console, and the Access Management Service must be turned on in the Access Server and the AccessGate." Below this is a note: "Note that you can configure primary and secondary servers using the COREid Access System. Use this page and the COREid link to the left to configure the IdMBridge for COREid." The "General" section includes four input fields: "AccessGate Name" (value: accessclientdefault), "AccessGate Password", "Confirm Password", and "Host Name Where the Access Server is Installed" (value: localhost). The "Port Number That the Access Server Listens To" field has the value 6021. The "Transport Security Connection Type" section has a red warning: "* If you change the connection type, then please restart the SHAREid server before using the transfer service." and three radio buttons: "Open" (selected), "Simple", and "Cert". Below this is the instruction: "For Simple or Cert Mode, enter the global passphrase" followed by "Global Passphrase" and "Confirm Passphrase" input fields. At the bottom, there are "Submit" and "Reset" buttons. The status bar at the bottom shows "Done" and "Internet".

To configure the COREid AccessGate and Access Server

1. Provide the AccessGate name.

This is a unique, descriptive name for this AccessGate that was provided during its configuration. This AccessGate is used by COREid Federation to communicate with your COREid installation and must be added to the COREid Access System by your Access System administrator.

2. Provide the AccessGate password.

This is a unique password that verifies and identifies the component. This password should have already been created by your Access System Administrator when they configured the AccessGate.

3. Provide the host name where the Access Server is installed.

This is the fully qualified domain name of the machine where the COREid Access Server is installed.

4. Provide the port number that the Access Server listens to.

Configure the transport security connection type

1. Select Open, Simple, or Cert.

These are the transport security modes that can be used between all Access Servers and associated AccessGates and COREid Federation.

2. For Simple or Cert mode, provide the pass phrase that was configured during installation of the AccessGate by your COREid Access System Administrator.

3. Click Submit.

If you selected Cert Mode, the Install Certificate page appears.

The screenshot displays the Oblix SHAREid web interface. The top header features the Oblix logo on the left and 'Oblix SHAREid™' on the right. A left-hand navigation menu lists various sections: MAIN, SETUP, ADMINISTRATION (with sub-items IdMBridge, COREid, Server Config, LDAP, RDBMS, SiteMinder), Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, Assertion Store, HELP, About, and Logout. The main content area is titled 'IdMBridge: COREid: Server Configuration: Install Certificate'. It contains a 'Generate request' button and three textboxes for entering certificate information: 'Please give the certificate key in the textbox below', 'Please give the certificate in the textbox below', and 'Please give the certificate chain in the textbox below'. At the bottom of the form are 'Submit' and 'Reset' buttons. The browser's status bar at the bottom right shows 'Internet'.

To install the certificate

1. From the page titled IdMBridge: COREid: Server Configuration: Install Certificate, click Generate Request.

The Request Certificate page appears.

Request Certificate

This information will be incorporated into your certificate request. Please do not leave any field blank - otherwise your request will not be generated.

Country Name (2 letter code only)

State or Province Name (full name)

Locality Name (eg, city)

Organization Name (eg, company)

Organizational Unit Name (eg, section)

Common Name (eg, hostName.domainName.com)

Email Address

2. Complete the following fields:

Field	Value
Country Code (two-letter code)	The two-letter country code, for example, US.
State or Province Name (full name)	The full name of the state or province, for example, California.
Locality Name (e.g., city)	The name of the township or city.
Organization Name	The name of your company or organization.
Organizational Unit name	The name of a workgroup or division.
Common Name	A fully qualified domain name to identify the requesting server for which this certificate will be generated. This must be the COREid Federation Server domain name.
Email Address	The email address of a contact person.

3. Click Submit.

The Request Certificate Details page appears.

4. Copy and save the information in the certificate request box.

Copy the information labeled Base 64 encoded certificate. Do not copy the information labeled Base 64 encoded certificate with CA certificate chain.

You will use this information in a certificate request that you send to a CA. Use your corporate standards for submitting certificate requests.

5. Once you receive a certificate from a CA, return to the page IdMForge: COREid: Server Configuration: Install Certificate, then copy and paste the information in the certificate to the fields on this page.

Completing COREid IdMForge Configuration

1. Click Submit when you've finished making entries for the IdMForge configuration.
2. Restart the COREid Federation Server to set the COREid IdMForge as the active IdMForge for your COREid Federation installation.

Configuring an LDAP IdMForge

You can configure an LDAP IdMForge when you first set up COREid Federation using the Setup Wizard or choose the Administration > IdMForge > LDAP menu option from the Administration Console's Navigation Bar. Before configuring the LDAP IdMForge, your LDAP directory must already be configured for use with COREid Federation. For an overview of the LDAP IdMForge, see "Planning for an LDAP IdMForge (Source Domain Only)" on page 52.

To configure the LDAP IdMForge

1. From the COREid Federation Administration Console navigation panel, click the IdMForge menu option.

The Administration Console expands the IdMForge navigation bar menu to show the different IdMForge types available: Oracle COREid, LDAP directory, relational database (RDBMS), and SiteMinder. In addition, the Administration Console displays the IdMForge Configuration page, where you can select the relational database (RDBMS) option if you have not already set this as the active IdMForge.

2. Click the LDAP menu option from the left-hand navigation bar.

The IdMBridge: LDAP Configuration page appears.

oblix **Oblix SHAREid™**

MAIN

SETUP

ADMINISTRATION

- IdMBridge**
 - COREid
 - LDAP**
 - RDBMS
 - SiteMinder
- Login
- Encryption
- Source Assertions
- Destination Mappings
- Domains
- Attribute Sharing
- Audits and Logs
- Assertion Store

HELP

- About
- Logout

IdMBridge: LDAP Configuration

The following information controls the SHAREid IdMBridge communication with the LDAP directory. After submitting your changes, click the IdMBridge link to enable LDAP as the active identity management system that SHAREid communicates with.

Connection URLs to the Directory Host and Port

To specify more than one URL for failover, you can provide a space-separated list of URLs.

Connection Url

Example for a secure connection: ldaps://directoryhost.company.com:port.

Example for a non-secure connection: ldap://directoryhost.company.com:port

Other Directory Information

Bind DN

Example: cn=shareid,dc=mycompany,dc=com

Bind Password

Confirm Bind Password

Search Base DN

This is the top directory node for storing user identities. Example: dc=mycompany,dc=com

Search Filter

This filter limits search results to entries that match the filter. Example: (&(objectclass=inetorgperson)(uid=%userid%))

Maximum Connections

Internet

3. Specify the entries you want to define LDAP access for user data.

The following table provides a description of the fields that define an LDAP IdMBridge configuration.

Field	Value
Connection URL	Specify one or more connection URLs, for example: <code>ldaps://directoryhost.company.com:port</code> where <code>directoryhost.company.com:port</code> is the URL (including the port number) for the directory. (Supply more than one connection URL for failover purposes.) Separate each URL with a blank space. Note that if you configure the connection URL to use SSL on a non-SSL port, for example, <code>ldaps://directoryhost.example.com:398</code> instead of <code>ldap://directoryhost.example.com:389</code> logins to COREid Federation will hang. You will need to correct the connection URL from the Administration Console and restart the COREid Federation Server.
Bind DN, Bind Password	Specify the DN (and the corresponding password) of the entry that COREid Federation uses to bind to the directory, for example: <code>cn=shareid,dc=mycompany,cd=com</code>
Search Base DN	top directory node where user entries are stored, for example: <code>dc=mycompany,dc=com</code>
Search Filter	LDAP filter to use for user searches, for example: <code>(&(objectclass=inetorgperson)(uid=%userid%))</code> See “Using LDAP Filters” on page 54 for more information on constructing LDAP filters you can use to narrow the scope of directory searches.
Maximum connections	Specifies the maximum number of connections in the pool maintained by the LDAP IdMBridge. This sets an upper limit on the number of concurrent requests using LDAP. Additional requests will wait until a connection is returned to the pool. Set the maximum high enough to handle the maximum anticipated concurrent COREid Federation requests. (The default is 10.)

Note: The Maximum connections option is only available when you configure LDAP IdMBridge configuration parameters using the Administration IdMBridge > LDAP menu option.

4. When you’re finished with the LDAP configuration, click Submit.

5. Restart the COREid Federation server to set the LDAP IdMBrige as the active IdMBrige for your COREid Federation installation.

Configuring an RDBMS IdMBrige

You can configure an RDBMS IdMBrige when you first set up COREid Federation using the Setup Wizard or choose the Administration IdMBrige > RDBMS menu option from the Administration Console's Navigation Bar.

This section assumes that you've already configured your RDBMS database for use with COREid Federation. For an overview of the RDBMS IdMBrige, see "Planning for an RDBMS IdMBrige (Source Domain Only)" on page 60.

Configure the RDBMS IdMBrige Database Connection Configuration

1. From the COREid Federation Administration Console Navigation panel, click IdMBrige.

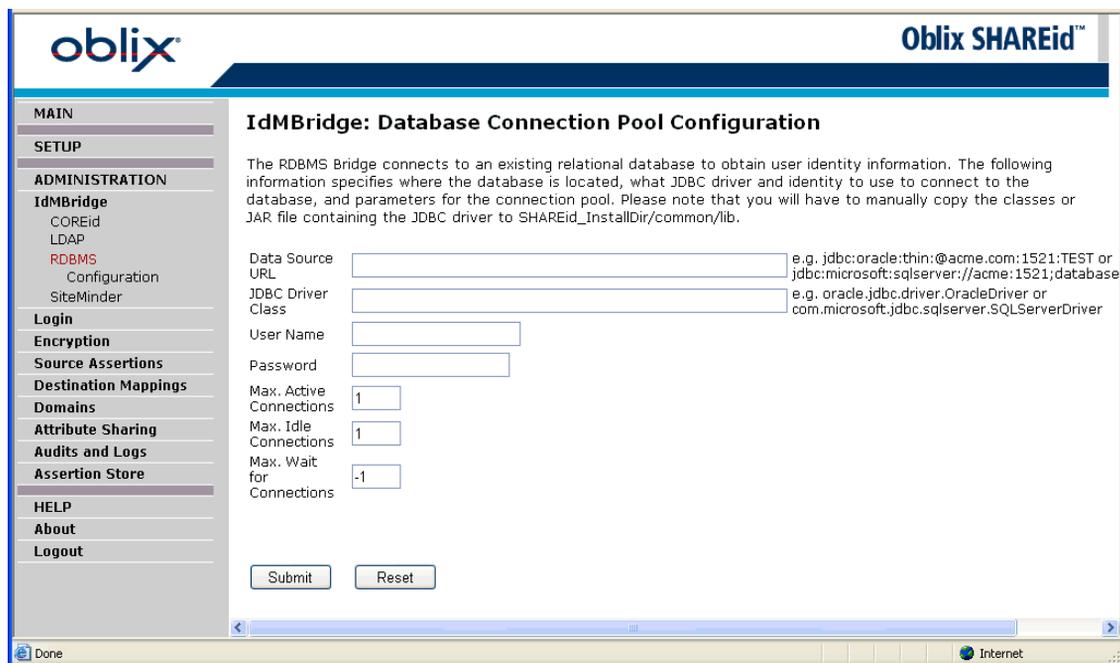
The Administration Console expands the IdMBrige navigation bar menu to show the different IdMBrige types available: Oracle COREid, LDAP directory, relational database (RDBMS), and SiteMinder.

In addition, the Administration Console displays the IdMBrige Configuration page, where you can select the relational database (RDBMS) option if you have not already set this as the active IdMBrige.

Database Connection Configuration

- Click the IdMForge > RDBMS menu option in the left-hand navigation panel.

The IdMForge: Database Connection Configuration page appears.



oblix Oblix SHAREid™

MAIN

SETUP

ADMINISTRATION

- IdMForge**
 - COREid
 - LDAP
 - RDBMS**
 - Configuration
 - SiteMinder
- Login
- Encryption
- Source Assertions
- Destination Mappings
- Domains
- Attribute Sharing
- Audits and Logs
- Assertion Store

HELP

- About
- Logout

IdMForge: Database Connection Pool Configuration

The RDBMS Bridge connects to an existing relational database to obtain user identity information. The following information specifies where the database is located, what JDBC driver and identity to use to connect to the database, and parameters for the connection pool. Please note that you will have to manually copy the classes or JAR file containing the JDBC driver to SHAREid_InstallDir/common/lib.

Data Source URL	<input type="text"/>	e.g. jdbc:oracle:thin:@acme.com:1521:TEST or jdbc:microsoft:sqlserver://acme:1521;database
JDBC Driver Class	<input type="text"/>	e.g. oracle.jdbc.driver.OracleDriver or com.microsoft.jdbc.sqlserver.SQLServerDriver
User Name	<input type="text"/>	
Password	<input type="password"/>	
Max. Active Connections	<input type="text" value="1"/>	
Max. Idle Connections	<input type="text" value="1"/>	
Max. Wait for Connections	<input type="text" value="-1"/>	

Note: Using the Setup Wizard, this configuration page corresponds to page 1 of 2 of the Setup Wizard's RDBMS IdMForge pages.

- Specify entries on this page to define the RDBMS IdMBridge database connection parameters.

The following table provides a description of database connection parameters required for an RDBMS IdMBridge configuration.

Field	Description
Data Source URL	<p>Vendor-specific URL to be passed to the JDBC driver to establish a connection to IdMBridge database, for example with an Oracle database:</p> <pre>jdbc:oracle:thin:@acme.com:1521:TEST</pre> <p>or, for Microsoft SQL Server:</p> <pre>jdbc:microsoft:sqlserver://acme:1521;databaseName=Northwind</pre> <p>The general syntax for this parameter with an Oracle database (Oracle 9i) is:</p> <pre>jdbc:oracle:<driver type>:@<hostname or TCP/IP address>:<port number>:<database name></pre> <p>where driver type is thin or thick. For Microsoft SQL Server, the general syntax is the following:</p> <pre>jdbc:microsoft:sqlserver://<hostname>:<JDBC port number>;databaseName=<database instance name></pre>
JDBC Driver Class	<p>The fully qualified Java class name of the JDBC driver to be used, for example, with an Oracle database:</p> <pre>oracle.jdbc.driver.OracleDriver</pre> <p>or, for Microsoft SQL Server:</p> <pre>com.microsoft.jdbc.sqlserver.SQLServerDriver</pre> <p>NOTE: You need to copy the associated Jar files to your <SHAREid Install>/common/lib folder for the specific JDBC driver you are using. For example, using the JDBC driver from Microsoft, the corresponding class jar files are msbase.jar, mssqlserver.jar, and msutil.jar. For more information on specific JDBC drivers to use in your environment, check your platform vendor's documentation. After copying the Jar files, you need to restart your COREid Federation Server.</p>

Field	Description
User Name	User name to be passed to the JDBC driver to establish a connection. (Associated user must have Read access to the RDBMS user table.)
Password	Password to be passed to the JDBC driver to establish a connection.
*Maximum Active Connections	The maximum number of active connections that can be allocated from the connection's database pool at the same time, or zero (0) for no limit. The default setting is 1.
Maximum Idle Connections	The maximum number of active connections that can remain idle in the pool, without extra ones being released, or zero for no limit. The default setting is 1.
Maximum Wait for Connections	The maximum number of milliseconds that the pool will wait (when there are no available connections) for a connection to be returned before throwing an exception, or -1 to wait indefinitely. The default setting is -1.

Note: The Maximum Active Connections option is only available when you configure RDBMS IdMBridge configuration parameters using the Administration IdMBridge > RDBMS menu option.

4. When you've finished configuring the database connection parameters on this page, click Submit.

RDBMS Configuration

1. Click the IdMForge > RDBMS > Configuration menu option in the left-hand navigation panel.

The IdMForge: RDBMS Configuration page appears:

The screenshot shows the Oblix SHAREid web interface. The top navigation bar includes the Oblix logo and 'Oblix SHAREid™'. A left-hand navigation menu lists various sections: MAIN, SETUP, ADMINISTRATION (with sub-items: IdMForge, COREid, LDAP, RDBMS, Configuration, SiteMinder), Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, and Assertion Store. Below the menu is a HELP section with About and Logout links. The main content area is titled 'IdMForge: RDBMS Configuration'. It contains the following text: 'The RDBMS Bridge connects to an existing relational database to obtain user identity information. The following information specifies how users are stored in your database.' Below this is a paragraph: 'Select a table name that has all the identity attributes in its columns. Only login id and password columns are configured here. Other columns can be mapped to assertion attributes in the assertion profile configuration.' There are four input fields: 'Table', 'Login id column', 'Password column', and 'Password Digest algorithm' (which is a dropdown menu currently set to 'None'). At the bottom of the form are 'Submit' and 'Reset' buttons. The status bar at the very bottom shows 'Done' on the left and 'Internet' on the right.

Note: Using the Setup Wizard, this configuration page corresponds to page 2 of 2 of the Setup Wizard's RDBMS IdMForge pages.

2. Specify entries on this page to configure parameters for the RDBMS associated with this IdMForge, specify entries for all the fields on this second page.

The following table provides a description of parameters required for the RDBMS IdMForge database configuration.

Field	Description
Table	Name of the table (or view) that contains all the user identity or credential attributes you want to include in an assertion profile.
Login id column	Name of the column that stores user names or login IDs of users making COREid Federation resource requests.
Password column	Name of the column that stores passwords corresponding to the user names or login IDs of users that making requests. Entry in this field is not required if external authentication is used for login.

Field	Description
Password Digest algorithm	Access authentication scheme to use to secure passwords. Set this entry to match the scheme used by your database. Options available are None, MD5, and SHA. (The default is None.)

3. After finishing your entries, click Submit.
4. Restart the COREid Federation server to set the RDBMS IdMBrige as the active IdMBrige for your COREid Federation installation.

Configuring a SiteMinder IdMBrige

If you want to use a SiteMinder installation as the user data repository for COREid Federation, you can configure a SiteMinder IdMBrige when you first set up COREid Federation using the Setup Wizard or by choosing the Administration IdMBrige > SiteMinder menu option from the Administration Console's Navigation Bar.

Note: For more information on using the Setup Wizard, see "Using the Setup Wizard" on page 99.

Before configuring the SiteMinder IdMBrige, your SiteMinder installation must already be configured for use with COREid Federation. For more information about using the SiteMinder IdMBrige, see "Planning for a SiteMinder IdMBrige (Source or Destination Domain)" on page 61 for details. Also see Appendix B on page 291 for SiteMinder site installation and configuration required for operation with COREid Federation.

To configure the SiteMinder IdMBrige

1. From the COREid Federation Administration Console's Navigation panel, click IdMBrige.

The Administration Console expands the IdMBrige navigation bar menu to show the different IdMBrige types available: Oracle COREid, LDAP directory, relational database (RDBMS), and SiteMinder.

In addition, the Administration Console displays the IdMBrige Configuration page, where you can select the SiteMinder option if you have not already set this as the active IdMBrige.

- Click the IdMBrige > SiteMinder menu option in the left-hand navigation panel.

The IdMBrige: SiteMinder Configuration page appears (top portion of configuration page shown below):

oblix Oblix SHAREid™

MAIN

SETUP

ADMINISTRATION

- IdMBrige
- COREid
- LDAP
- RDBMS
- SiteMinder

Login

Encryption

Source Assertions

Destination Mappings

Domains

Attribute Sharing

Audits and Logs

Assertion Store

HELP

About

Logout

IdMBrige: SiteMinder Configuration

The following information controls the SHAREid IdMBrige communication with the SiteMinder Policy Server and its user repository. After submitting your changes, click the IdMBrige link to enable SiteMinder as the active identity data store that SHAREid communicates with.

SiteMinder Agent Configuration

This agent must be configured through the SiteMinder Administration interface.

Agent Name:

Agent Secret:

Confirm Agent Secret:

Connection to SiteMinder Policy Servers

You may configure multiple replicated Policy Servers for failover and load-balancing.

Host	Ports			Connections			Timeout (seconds)
	Authentication	Authorization	Accounting	Min	Max	Step	
Defaults	44443	44442	44441	1	1	1	20
<input type="text"/>							

Done Internet

The steps to configure the SiteMinder IdMBrige from the COREid Federation Administration Console are:

- Specify Agent information—SiteMinder Agent and shared secret
 - Specify SiteMinder Policy Information—SiteMinder Host and AAA ports
 - Specify Automatic Policy Creation Information—Administrator ID and password, domain, and user directory.
- In the “SiteMinder Agent Configuration” section of the form, specify the SiteMinder Agent Name and Secret (also confirm the Secret).

4. In the “Connection to SiteMinder Policy Servers” section, specify the parameters for SiteMinder Policy Servers configured for use with COREid Federation.

The following table describes the SiteMinder Policy Server entries:

Method	Description
Host	The host where the Policy Server is installed.
Ports	Authentication: Policy Server port used for authentication requests. Authorization: Policy Server port used for authorization requests. Accounting: Policy Server port used for accounting requests.
Connections	Min: The minimum number of agent connections to the Policy Server. Max: The maximum number of agent connections to the Policy Server. Step: The number of connections to the Policy Server opened by the agent at a time.
Timeout (seconds)	The time in seconds that the agent will wait for a response from the Policy Server before it returns a failure.

The remaining lower portion of the SiteMinder Configuration page is shown

below:

The screenshot shows the Oblix SHAREid™ web interface. The top navigation bar includes the 'oblix' logo and 'Oblix SHAREid™'. A left sidebar contains a menu with categories: MAIN, SETUP, ADMINISTRATION (with sub-items: IdMBridge, COREid, LDAP, RDBMS, SiteMinder), Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, Assertion Store, HELP (with sub-items: About, Logout). The main content area is titled 'Automatic Policy Creation' and contains a descriptive paragraph: 'The SiteMinder Bridge automatically creates the policy objects that it requires. It does this by logging into SiteMinder as an administrator. You must also specify the SiteMinder domain to contain the policy objects and the SiteMinder-configured name of the user directory for this domain.' Below this are five input fields: 'Admin ID', 'Admin Password', 'Confirm Admin Password', 'Domain', and 'User Directory'. The second section is titled 'Assertion Mapping using a Secondary IdMBridge' and contains a descriptive paragraph: 'On a destination site, the SiteMinder Bridge maps an incoming SSO assertion to a local SiteMinder user. The authentication scheme used for this requires a user name. If this name is present in the assertion, either a SubjectName or an Attribute value, it can be used directly and no secondary bridge is required. If the assertion does not have the user name, then a secondary LDAP or RDBMS Bridge must be used to map the assertion SubjectName and Attribute values to a user, and then obtain the user name from an attribute or row for the mapped user. The secondary bridge must be configured in its own IdMBridge Configuration page.' Below this are three radio button options, each with a corresponding input field: 'None' (User Name Attribute from Assertion), 'LDAP Bridge' (User Name Attribute in Directory), and 'RDBMS Bridge' (User Name Column in Database). At the bottom of the form are 'Submit' and 'Reset' buttons. The browser's status bar at the bottom shows 'Internet'.

5. In the “Automatic Policy Creation” specify the parameters required to log into SiteMinder as an administrator and specify the SiteMinder domain that contains the policy objects and the SM-configured name of the user directory for this domain.
6. If your SiteMinder configuration requires it, first specify the IdMBridge type, LDAP or RDBMS, and then the corresponding IdMBridge parameters in the “Assertion Mapping using a Secondary Bridge” section.

These parameters specify the mapping in the assertion SubjectName and Attribute values to a user, which then obtain the user name from an attribute or row for the mapped user from the secondary LDAP or RDBMS Bridge.

The Assertion Mapping using a Secondary IdMBridge options on the SiteMinder IdMBridge configuration page indicates the source of the user name used in the SiteMinder authentication on the destination site. The interpretation of this parameter is determined by the Secondary Bridge setting

- None: An attribute from the received SSO assertion.
- LDAP: An LDAP attribute in the user's directory entry.

- RDBMS: A column in the user database table.

If the `UserNameAttribute` entry is omitted or empty and the Secondary Bridge selection is `None`, COREid Federation uses the `SubjectName` attribute (the value of `<NameIdentifier>` of the assertion's `<Subject>`) as the user name. The `UserNameAttribute` must be set for the LDAP or RDBMS Secondary Bridges.

7. After finishing your entries, click `Submit`.
8. Restart the COREid Federation server to set the `SiteMinder IdMBridge` as the active `IdMBridge` for your COREid Federation installation.

Following setup of the `SiteMinder IdMBridge`, you need to configure login, assertion profiles, and destination domains for use in COREid Federation operations using the `SiteMinder IdMBridge`, as you would for any other COREid Federation `IdMBridge`.

Configuring a Login Method

As a source domain, COREid Federation must authenticate your local users. This enables COREid Federation to create assertions for these users. You can configure user login methods and authentication when you configure COREid Federation the first time using the `Setup Wizard`, or later, selecting `Administration` menu options from the COREid Federation `Administration Console`'s navigation bar.

For an overview of login methods, see “About User Login and Single Sign-On” on page 61.

To configure a basic login method

1. From the COREid Federation Administration Console's Navigation panel, click the Administration > Login menu option.

The Configure Login page appears.

Configure Login

When SHAREid validates a user's identity, one of the following scenarios may apply:

- The user points the browser to a SHAREid login page.
- The user clicks a link to a resource on the destination domain, and is redirected back to the source's SHAREid. SHAREid challenges the user for login credentials. For this scenario to be supported, the SmartMarks function must also be enabled. See the help or the SHAREid documentation for details.
- The user accesses a resource that is protected by an identity management system supported by the SHAREid IdMBridge, for example, the Oblix COREid System. In the case of login to COREid, a single sign-on cookie is set. The user can then access SHAREid enabled resources without logging in again.
- The user logs in to a local portal. The portal sets a header variable that is used by SHAREid to identify the user in the SHAREid data store (for example, LDAP). The user can then access SHAREid-enabled resources without logging in again.

This page identifies the method that SHAREid uses to challenge users for login credentials.

To present the user with a login dialog for authentication, you can select a basic login dialog or Web login form as the login method. If you choose a Web Login Form, a default login form will be provided. You will be able to customize this form. Click the Help link for details.

Authentication Method

Basic Login Dialog

Information to Appear in the Realm Field on this Dialog

Web Login Form

External Credentials
(space-separated list of header variables set by an external authentication mechanism)

SSO Domain

SSO Domain (e.g., ".company.com")

User Timeout

Active Idmbridge LDAP

Session Timeout (seconds)

Idle Timeout (seconds)

- From the Configure login page, select the type of login you wish to support.

Method	Description
Basic Login Dialog	Users enter a user ID and password in a pop-up window supplied by the Web server.
Web Login Form (the default)	<p>Users supply credentials in a login form. COREid Federation provides a default form that you can modify. See “Customizing the Login Form” on page 209 for details.</p> <p>For a Web login form, supply a path to the form. The default path is:</p> <p><i>SHAREid_Install_Dir/webapps/shareid/login.jsp</i></p> <p>where <i>SHAREid_Install_Dir</i> is the directory where COREid Federation is installed.</p>
External Credentials	<p>A challenge method outside COREid Federation is used. When the user attempts to access a COREid Federation-enabled resource, COREid Federation looks for a header variable set by the external application.</p> <p>Use this field to specify one or more header variables that are set by the external application and that can be used to identify the user in the COREid Federation user data repository. Use a space to separate header variables.</p>

Note: If you change the authentication method that COREid Federation uses, you must restart the COREid Federation Server to have the changes take effect.

- If you are using an LDAP or RDBMS IdMBridge, you can also configure timeouts for the directory:
 - Session timeout**—This is the amount of time that an LDAP or RDBMS session is permitted to exist, whether or not the user is actively making use of the session. The default is 3600 seconds.
 - Idle timeout**—This is the amount of time that an active LDAP or RDBMS session is permitted to exist if there is no activity on the part of the user. The default is 3600 seconds.

Note: These two options are only available when configuring the Login page using the Administration > Login menu option, not using the Setup Wizard.

- When you’ve completed your selections, click Submit.

Configuring Passwords and the Certificate Store

COREid Federation provides a default certificate store and self-signed certificates. From the COREid Federation Administration Console, you need to enter the certificate store password and specify the location of the certificate store.

The COREid Federation Administration Console also allows you to change the encryption key used to secure the session passwords. A periodic update of these passwords increases security. COREid Federation encrypts the following:

- The password that protects the COREid Federation certificate store.
- The passwords that protect COREid Federation sessions, the password for the COREid Master Access Administrator when using the COREid IdMBridge, and the user session cookie.

Important: Run the keytool command to change keystore passwords.

Changing the Session Password Encryption Key

For security purposes, you may want to periodically change the key used to encrypt various passwords, such as the Master Administrator password, the certificate store password, and so on. This key is also used to encrypt the user's session cookie.

Note that if you re-encrypt the session password, the password value remains the same, only the encrypted value changes. For example, if the password is oblix before regeneration, it would remain oblix after regeneration.

To update the session password encryption

1. From the COREid Federation Administration Console, click Encryption > Sessions/Passwords.

The Regenerate Session and Password Encryption Key page appears.



2. To regenerate the key, click Update.
3. Restart the COREid Federation server to use the new generated keys.

Configuring the Certificate Store

A database known as the certificate store contains the keys and certificates that are used for SSL and for digitally signing a protocol response when the SAML POST profile is used. By default, the certificate store is located in the /conf folder of your COREid Federation installation directory:

```
SHAREid_Install_Dir/conf
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

COREid Federation creates a default certificate store that contains keys and certificates. You can use this certificate store, or you can use a custom store, for example, a store that already contains existing keys and certificates.

In addition to configuring the certificate store, you configure the keys and certificates that COREid Federation uses. See “Configuring Keys and Certificates” on page 243 for details.

To configure the default certificate store

1. To change the password of the certificate store, you must first go to the command line of the machine where COREid Federation is installed and run the keytool command to update the password of the keystore file.

The certificate store password is required to perform any operation on the keystore, for example, a listing. See “Configuring Keys and Certificates” on page 243 for details.

2. You must also update the password in the following file:

`SHAREid_Install_Dir/conf/server.xml`

where `SHAREid_Install_Dir` is the directory where COREid Federation is installed. The default password is `changeit`.

3. After changing the password from the command line and in `server.xml`, go to the COREid Federation Administration Console, click the Encryption link, then click Certificates.

The Certificate Store page appears.



The screenshot shows the Oblix SHAREid Administration Console interface. The top navigation bar includes the Oblix logo and the text "Oblix SHAREid™". A left-hand menu lists various sections: MAIN, SETUP, ADMINISTRATION (with sub-items IdMBridge, Login, Encryption, Certificates, Sessions/Passwords), Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, and Assertion Store. Below this is a HELP section with sub-items About and Logout. The main content area is titled "Certificate Store" and contains the following text: "Use this page to enter information about the certificate store that SHAREid uses for SSL and digital signatures. If you want to use different certificate stores for SSL and digital signatures, click Help." It then states: "SHAREid creates a default certificate store that contains a default self-signed certificate with the shareid alias." and "By default, the certificate store is located in the /conf folder of your SHAREid installation directory." A paragraph follows: "To change the password for this certificate store, first run the keytool utility to change the password for the keystore, then update the password in the field below that SHAREid uses to access the keystore. The certificate store and key passwords are expected to be the same. See documentation for details." Below this text are four input fields: "Path to the certificate store" (containing "C:/Program Files/Oblix/SHAREid/conf/keystore"), "Signing Key Alias" (containing "shareid"), "Certificate store and Signing key Password" (masked with dots), and "Confirm Password" (masked with dots). A red asterisk warning reads: "*Please restart the SHAREid server after submitting this change." At the bottom of the form are "Submit" and "Reset" buttons. The browser's address bar at the bottom right shows "Internet".

4. Provide a path to the certificate store.

The default is `SHAREid_Install_Dir/conf`.

5. Provide a password for the certificate store and signing key.

6. Click Submit.
7. Restart the COREid Federation server.

Note: You can configure separate certificates for HTTPS encryption and for signing the SAML response when the POST profile is used. See “Maintaining Separate Signing and CA Keys” on page 251 for details.

Configuring Assertion Profiles

When a user from a source domain requests a resource in a destination domain, the source domain provides the destination domain with an assertion. The assertion attests to the user's identity. An assertion profile defines the information that is used to build the assertion.

Note: For an overview of assertion profiles, see “About Assertions” on page 24 in Chapter 1. Introduction, and “Planning Assertion Profiles to use with COREid Federation” on page 64 in Chapter 2. Planning a COREid Federation Installation. For information about basic assertion setup using the Setup Wizard, see “Step 3: Add an Assertion profile using the Setup Wizard” on page 104.

If you are a source site, you need to configure at least one assertion profile. The profile determines the information that is put into the protocol assertion that destinations can use to authorize users from your domain. When you configure an assertion profile, you identify user data to be placed in the assertion. Note that you and your destination domains must agree on the attribute names to be used in assertions prior to configuring the assertions.

When your users attempt to access a resource on the destination domain and they are using COREid Federation, the destination domain uses one or a combination of attributes to map the user to a local identity at the destination site to make authorization decisions. The destination can use other attributes, for example, the user's title and email, in the assertion for tracking purposes as well as authorization.

To view, modify, or delete an assertion file

From the COREid Federation Administration Console, you can view and optionally modify an existing assertion profile.

1. From the COREid Federation Administration Console navigation bar menu, click Source Assertions.
2. Select the Source Assertions > View All menu option.
The Administration Console displays a list of all currently defined assertion profiles.
3. To view and optionally modify an assertion profile, click its hyperlinked name.
4. To delete an assertion, click the checkbox for the assertion that you want to delete and click the Delete button.

To create an assertion profile

1. From the COREid Federation Administration Console navigation bar menu, click Source Assertions.
A Configure Assertions page appears in the right-hand pane and links for View All and Add Assertion appear in the left-hand navigation bar menu.

2. Click Add Assertion.

The Add Assertion Profile page appears.

oblix Oblix SHAREid™

MAIN

SETUP

ADMINISTRATION

IdMBridge

Login

Encryption

Source Assertions

View All

Add Assertion

Destination Mappings

Domains

Attribute Sharing

Audits and Logs

Assertion Store

HELP

About

Logout

Add Assertion Profile

When users request resources, the source site provides the destination site with an assertion that attests to the user's identity. An assertion profile defines the contents of the assertion that is sent to the destination.

Assertion Profile Name:

Description:

Issuer: (e.g. http://host.company.com)

Subject Name Qualifier:

Subject Format:

User Attribute for Subject:

Add Assertion Attributes to Your Assertion Profile

In the following fields, you configure attribute mapping in the assertion profile. To configure an attribute mapping, you create a one-to-one correspondence (a mapping) between assertion attributes that a destination domain administrator gives to you and user attributes in your domain's data store. The destination domain's administrator must tell you the names of the assertion attributes.

Assertion Attribute	Attribute in Data Store	Name Space	Optional Type	In SSO Assertions	Allowed Values
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text"/>

Advanced Options

Assertion Signing

Include a certificate in the signature

Assertion Validity Period

seconds before the time assertion generated

seconds after the time assertion generated

Delimited Data

In addition to supporting and passing multi-valued attributes, SHAREid can also support delimited data to provide multiple values for assertion attributes. To use the delimited data option, select yes below and specify the character to be used.

Data is delimited: Yes No

Delimiter character:

- Specify the assertion profile name and other parameters to configure a new assertion profile.

The following table provides a description of fields provided on the Add Assertion Profile page:

Field	Description
Assertion profile name	Any unique name. For example, you could specify a name that indicates the name of the destination domain that will receive the assertions that are created using this profile. Or, the name could indicate the association being made between your user store attribute and the assertion attribute, for example: Mail-to-SubjectName
Description	Text describing or summarizing contents of the profile.
Issuer	Name of the SAML authority who issued the assertion. Identify the issuer with an unambiguous name. You can use a name or the uniform resource identifier (URI) of your host domain, for example: <code>http://host.domain.com</code> where <i>host.domain.com</i> is the DNS host name for the COREid Federation host, for example: <code>http://company.com</code>
Subject name qualifier	Optional subject name qualifier allows the SAML-compliant server to determine the namespace for a user. The source domain may find this field to be useful for informational purposes. For example, an assertion for <code>jsmith@oblix.com</code> , may contain J. Smith's department. Example: <code>ou=Department,o=Company,c=US</code>

Field	Description
Subject format	<p>Format of the Subject who is the user who is 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 a list of attribute formats:</p> <p>None—No format is specified.</p> <p>Email address—The NameIdentifier is formatted as an email address.</p> <p>X509 subject name—The NameIdentifier is in the form of a DN, for example, cn=myname,dc=example,dc=com.</p> <p>Windows domain—A Windows domain qualified user name formatted as DomainName\UserName, for example, oblix\jsmith.</p> <p>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.</p> <p>Other—Other is a format value that is not one of the predefined values defined in the SAML spec. A format of Other allows any other URI values that SAML partners might use. Hypothetical examples: "<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 for Other is specified in the shareid-config.xml file.</p>
User Attribute for Subject	<p>This is the local LDAP user attribute or RDBMS field name whose value will be used as the value for the Subject assertion element. (More specifically, this field determines the value of the NameIdentifier sub-element of the assertion's Subject element.) The Subject element identifies the source domain user.</p> <p>If this field is left blank, the DN of the user's directory entry is used with the LDAP IdMBridge, and COREid or SiteMinder IdMBridges using LDAP data stores. For RDBMS IdMBridges or SiteMinder IdMBridges accessing a database, the user's login name is used.</p>
Assertion attribute mapping table	See individual field description in table below.

The following table describes each of the fields you can define for an assertion attribute mapping.

Field	Description
Assertion Attribute	Name to be provided in the Assertion Attribute statement. This attribute is usually provided by the destination domain.
Attribute in Data Store	Attribute or field name in local data store where you can retrieve attribute value
Name Space	Optional for SSO Assertions, otherwise needs to be specified. Allows you to specify namespaces if the assertion profile may be used by multiple applications.
Optional Type	Defines the xsi:type attribute of the <Attribute> in the statement. The types are defined in the XML Schema specification. If the type is omitted it will be assumed to be string.
In SSO Assertions?	Select this checkbox if you want to include the attribute in assertions used in Web-browser profiles (Artifact or POST). Selecting this checkbox is not required for assertions using the attribute sharing profile or authorization queries.
Allowed Values	Restrict attribute values returned in an assertion to only those specified in this list.

4. The following table describes Advanced Options you can specify for Assertion profiles:

Field	Description
Assertion Validity Period	
XX seconds before assertion generated	Because of clock skew, it is possible that a time stamp on an assertion will precede the time on the clock of the destination host. As a result, this field enables you to indicate an amount of time that the assertion is valid prior to the time stamp indicating its creation time.
XX seconds after assertion generated	Indicates the amount of time that the assertion can be used from the time that the assertion was created. Note that you and the destination domain may want to use a product that synchronizes your host machines to an atomic clock.
Delimited Data	
Data is delimited	Indicate whether multi-value attributes in data store are delimited.

Field	Description
Delimiter character	<p>If multi-valued attributes in data store are delimited, specify the delimiter that COREid Federation needs to use to properly retrieve and parse the values.</p> <p>For example, if a data stores multiple values of blue, green, and red as "blue,green,red", the comma (",") character is the delimiter. Specify the comma in this field as the delimiter character COREid Federation needs to use to return each of the individual attribute values.</p> <p>Delimiters are specified as part of each Assertion Profile, so it is possible for each profile to have a different delimiter. Delimiters are stored in the shareid-config.xml file.</p>

- When you've finished defining an assertion profile and assertion attribute mappings, click the Submit button.

Sending Test Assertions to a Destination

If you have configured your domain and a destination domain, and the destination domain has completed configuration for itself and your domain, you can create a COREid Federation-enabled link that transfers you to a protected resource on the destination.

The general syntax for the link to send a test assertion is the following:

```
<a href="https:source_host:source_port/shareid/saml/ObSAMLTransferService?DOMAIN=domain&METHOD=method&TARGET=https:destination_host:destination_port/path_to_the_resource">link text</a>
```

where:

- source_host* is the host name that you configured for your COREid Federation Server when it was installed. (See your MyDomain configuration for details.)
- source_port* is the SSL port that you configured for your COREid Federation Server when it was installed. (See your MyDomain configuration for details.)
- domain* is the destination domain name that you configured in the COREid Federation Administration Console.
- method* is Artifact or POST, depending on which profile you will be testing (usually determined by the destination). Both source and destination must accept the profile.
- destination_host* is the fully qualified host name for the destination that you configured in the COREid Federation Administration Console.
- destination_port* is the port for the destination that you configured in the COREid Federation Administration Console.

- *path_to_the_resource* is a path that allows the user to access the destination resource. The destination domain must supply the path.

Note: The combination of destination host, destination_port, and path to the resource specifies a target URL that can point to any target resource, usually that of the application you are trying to access at the destination site.

- *link text* is information that you might include on the HTML page in the form of a link for users to select to access the destination resource or you could enter directly in your Web browser's URL address field.

To illustrate a COREid Federation-enabled link, the following link calls a transfer service as represented by source.oblix.com. This link posts an assertion to the SupplierNet domain. The target is a protected parts ordering application at SupplierNet, as represented by http://destination.oblix.com:92. The target URL for this assertion is the following:

```
<a href="https://source.oblix.com:8113/shareid/saml/ObSAMLTransferService?DOMAIN=SupplierNet&METHOD=post&TARGET=http://destination.oblix.com:92/index.html">Click here to access the SupplierNet destination</a>
```

The structure of this URL is as follows:

- **a href**—The HTML code used to create a link.
- **“https://source.oblix.com...”**—The source domain's Transfer Service URL.
- **DOMAIN**—The domain name configured for the destination.
- **METHOD**—Artifact or POST.
- **TARGET**—The destination resource, in the example above, the parts ordering application.
- **Click here to. . .**—The text of the link displayed to the user.

If you log in to COREid Federation using the login method that you configured, and the identity that you use for logging in is a valid user entry in your directory, COREid Federation should send the assertion to the destination. Successful completion of the transfer depends on the destination's configuration. If running the test returns an error URL page, you may want to enable debug messages to be included in the system logs and then check for errors and other messages that may indicate the source of the problem. See “About COREid Federation Audit and System Logs” on page 277 for details.

Configuring Domains

From the COREid Federation Administration Console, you can configure a domain for your site, which is always called MyDomain. You can also configure partner sites or domains to operate as either source or destination domains as part of your network.

When you installed COREid Federation, defaults were initially set for the local domain, setting parameters, for example, to specify SAML protocols and Web browser profiles and the URLs for SAML transfers and error services that are used by your COREid Federation instance. You can change local domain (MyDomain) configuration settings by choosing the Administration > Domains menu option from the COREid Federation Administration Console and then selecting the MyDomain submenu option.

To view, modify, or delete, or other currently configured domains you can select the Administration > Domains menu option and choose the View All Domains submenu option. To add a new domain to your network, you can choose the Add Other Domains submenu option to add other COREid Federation or non-COREid Federation based domains.

For more information on COREid Federation network and domain configuration, see “About COREid Federation” on page 18. Also refer to Chapter 3 for more information on COREid Federation network and domain configuration planning and requirements.

To view, modify, or delete a domain

From the COREid Federation Administration Console, you can view, modify, or delete an existing domain.

1. From the COREid Federation Administration Console’s navigation bar, click the Domains option.
2. Select the Domains > View All Domains menu option.

The Administration Console displays a list of all currently configured domain.

3. To view and modify an existing domain, click its hyperlinked name.
4. To delete a configured domain, click the checkbox next to the domain you want to delete and click the Delete button.

Configuring MyDomain

Whether you are a source domain, a destination domain, or both, your local domain is always called MyDomain. MyDomain is configured automatically after you install and run through the Setup Wizard to specify options such as SAML protocols, protocols supported and the URLs used by COREid Federation for SAML transfers and error services. You can change the settings for MyDomain, but cannot add or delete MyDomain.

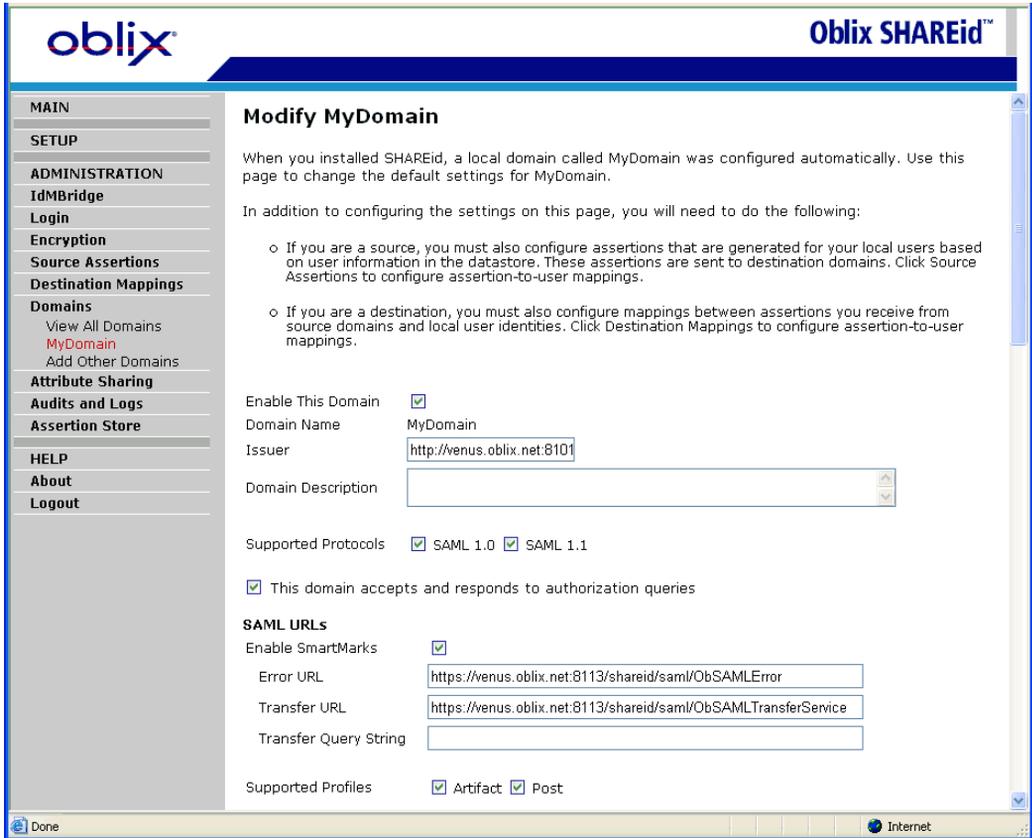
Note: COREid Federation generates the service URLs (Requester, Responder, Transfer, and Receiver) based on the host and port information that you supplied during COREid Federation Server installation. In general, you should not change these URLs unless your configuration changes, for example, if you install a proxy.

To modify MyDomain configuration

1. From the COREid Federation Administration links, click Domains.
The Configure Domains page appears.

2. Click MyDomain.

The Modify MyDomain page appears (shown in the following three screen displays):



Under General Information, the following items appear:

Field	Description
Enable this domain	Checking this field enables users in your domain to access resources on a destination domain if you are a source domain. Enables users from other domains to access your resources if you are a destination.
Domain name	This is always MyDomain for the local domain.
Issuer	This is the name of the authority who issues the assertion. The Issuer field can be a URI, for example: example.shareid.com The Issuer field configured for MyDomain (when MyDomain is a source) must match the Issuer field used in the profile that the destination domain configures for this source.

Field	Description
Domain Description	A description of this domain.
Supported Protocols	By default, COREid Federation creates assertions using the SAML 1.1 protocol, but your domain may need to communicate with a domain that still uses SAML 1.0. COREid Federation responds to requests using the protocol specified in the requests. For example, if the destination domain uses SAML 1.0, the source COREid Federation responds with SAML 1.0 if the domain configuration specifies that SAML 1.0 is supported. COREid Federation uses the highest protocol configured for both domains, if both are enabled.
This domain accepts and responds to authorization queries	<p>Enables or disables authorization decision queries for MyDomain. This feature is not directly related to Web single sign-on. None of the Web SSO profiles uses the SAML authorization query. Other SAML applications might use this. You set this field if your domain's SAML Responder will process authorization decision queries from SAML Requesters. If you want to enable authorization queries, you would also check the same checkbox in the destination domain's configuration page. The destination domain administrator would also check this box for MyDomain on the destination.</p> <p>For the Artifact profile, the source domain Responder service determines if a destination Requester is permitted to request an authorization decision as follows:</p> <ul style="list-style-type: none"> • An HTTP request contains a header that matches the user name and the Requester password configured for the domain. • An HTTPS request is made, and COREid Federation performs SSL client certificate authentication and then looks up the domain with a Requester ID or Subject DN that matches the Subject DN field of the certificate.

Under the SAML URLs section, the following items appear:

Field	Description
Enable SmartMarks	<p>Enables a feature that redirects a non-authenticated user who tries to access a protected resource to a URL where authentication can occur. See "Redirecting Users to a Login Form" on page 198 for details.</p> <p>Select Enable SmartMarks if you want to use this feature.</p>
Error URL	<p>This component enables your domain to receive errors from a destination and present customized error messages to your users. See "Error Reporting" on page 210 for details.</p> <p>Your error URL is the location for receiving redirected error information from a destination.</p>

Field	Description
Transfer URL	<p>The URL for your Transfer service. The Transfer service initiates the process that enables a user to access resources on another domain.</p> <p>Even if you use a proxy, users can go directly to the Transfer service on the COREid Federation server, for example:</p> <p><code>https://shareid.source.com:8113/shareid/saml/ObSAMLTransferService</code></p> <p>Instead of:</p> <p><code><https://shareid-proxy.source.com:8113/shareid/saml/ObSAMLTransferService></code>.</p>
Transfer Query String	<p>This field is only relevant for a destination domain.</p> <p>This string is required to transfer users to the source domain for authentication if SmartMarks is used. See “Redirecting Users to a Login Form” on page 198 for details.</p>
Supported Profiles	<p>Determines if Artifact or POST, or both, are supported. See “Web Browser Profiles for COREid Federation SAML” on page 32 and “Planning Assertion Profiles to use with COREid Federation” on page 64 for details.</p> <p>COREid Federation uses the METHOD parameter in your Transfer Service URL to determine what profile to use. If the METHOD is not present, the default profile is Artifact.</p>

The following display shows the Source and Destination Domain configuration sections of the Modify MyDomain page:

The screenshot displays the Oblix SHAREid web interface. The top navigation bar includes the Oblix logo and the text "Oblix SHAREid™". A left-hand sidebar menu lists various administrative options: MAIN, SETUP, ADMINISTRATION (with sub-items IdMBridge, Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, and Assertion Store), and HELP (with sub-items About and Logout). The main content area is titled "Configure This Domain as a Source" and contains two sections: "Post Profile" and "Artifact Profile".

Configure This Domain as a Source

Post Profile
Signature Verification for Assertions Generated Using the Post Profile

Signing Certificate Subject DN:
Signing Certificate Issuer DN:

Artifact Profile

Responder URL:
Source ID:

Configure This Domain as a Destination.

Receiver URL:

Artifact Profile

Requester Authentication: Select X.509 Certificate only if Responder URL of source domain uses Client Certificate authentication port.

Basic

Requester Id:
Requester Password:
Confirm Password:

X.509 Certificate

Signing Certificate Subject DN:

The browser's address bar at the bottom shows "Internet".

Under Configure This Domain as a Source, the following items are configurable:

Field	Description
Signing Certificate Subject DN	<p>By default, the signing certificate store is located in: <i>SHAREid_Install_Dir/conf</i></p> <p>For the POST Profile, the source signs the SAML response using a signing key in a certificate. The destination reads the issuer name in the assertion, and uses the value in the Certificate Subject DN field configured for this source domain to verify the signature.</p> <p>To view the contents of the COREid Federation certificate store located in <i>SHAREid_Install_Dir/conf</i>, type:</p> <pre>keytool -list -v keystore keystore</pre> <p>The default password is “changeit”. An example of a subject DN is the following: cn=example.oblix.net,ou=Security,o=Oblix,l=Cupertino,st=California,c=US</p>
Signing Certificate Issuer DN	<p>By default, the signing certificate store is located in: <i>SHAREid_Install_Dir/conf</i></p> <p>For the POST Profile, the source signs responses using a signing key in a certificate. The destination reads the issuer name in the assertion, and uses the value in this Certificate Issuer DN field configured for this source domain to verify the signature.</p>
Responder URL	<p>For the Artifact profile, when a user from a source domain requests a resource on a destination, the destination queries the source about the user. The Responder service on the source sends a reply.</p> <p>If you change the Responder URL, you must regenerate the source ID, since it is based on this URL. If you change the Responder URL, delete the information in the Source ID field and click Submit to regenerate the source ID.</p> <p>Example of a Responder URL: https://example.oblix.net:2113/shareid/saml/ObSAMLResponderService</p>

Field	Description
Source ID	<p>For the Artifact profile, the destination uses the Source ID to look up the source domain when preparing to request an assertion. The source domain includes its source ID in the artifact it sends to the destination. When the destination requests a full assertion, it uses the source domain's Source ID in the request.</p> <p>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.</p> <p>You provide your SourceID to destination domains.</p>

Note: For more information about field entries in the “Configure This Domain as a Destination” section, refer to “About Destination Domain Configuration” on page 172.

The following display shows the Attribute or Authorization Authority configuration sections of the Modify MyDomain page:

The screenshot displays the Oblix SHAREid web interface. The top navigation bar includes the 'oblix' logo and 'Oblix SHAREid™'. A left-hand sidebar contains a menu with categories: MAIN, SETUP, ADMINISTRATION (with sub-items: IdMBridge, Login, Encryption, Source Assertions, Destination Mappings, Domains (View All Domains, MyDomain, Add Other Domains), Attribute Sharing, Audits and Logs, Assertion Store), and HELP (with sub-items: About, Logout). The main content area is titled 'Configure This Domain as a Attribute or Authorizat on Authority'. Below the title, it states: 'To service SAML queries, MyDomain must have an assertion mapping configured that maps the Subject of the query to a local user to obtain attributes or an authorization decision.' There is a 'Subject Mapping' dropdown menu set to 'Minimal Mapping'. A second section is titled 'Configure This Domain for Loopback Testing'. It states: 'In loopback, MyDomain is both the source and destination of a SAML transfer. Consequently it needs an assertion profile to construct the SSO assertion and an assertion mapping to map the assertion to a local user.' Below this, there are two dropdown menus: 'Assertion Profile' set to 'Minimal Profile' and 'Assertion Mapping' set to 'Minimal Mapping'. At the bottom of the configuration area are 'Submit' and 'Reset' buttons. The browser's address bar at the bottom shows 'Internet'.

In the last section of the Modify MyDomain page, the following items are configurable:

Field	Description
Configure This Domain as a Attribute or Authorization Authority	Selection indicates whether domain is attribute or authorization authority. To service SAML attribute or authorization queries, MyDomain must have an assertion mapping configured that maps the Subject of the query to a local user to obtain attributes or an authorization decision.
Configure This Domain for Loopback Testing	Checkbox indicates whether the domain is configured for loopback testing. In loopback, MyDomain is both the source and destination of a SAML transfer. Consequently it needs an assertion profile to construct the SSO assertion and an assertion mapping to map the assertion to a local user.

3. When you've finished defining an assertion profile and assertion attribute mappings, click the Submit button.
4. Restart the COREid Federation server.

Configuring a Destination Domain for a Source Domain

From the COREid Federation Administration Console, you also configure partner sites or domains to operate as either source or destination domains as part of your network.

To view, modify, or delete currently configured domains you can select the Administration > Domains menu option and choose the View All Domains submenu option. To add a new domain to your network, you can choose the Add Other Domains submenu option to add other COREid Federation or non-COREid Federation based domains.

When you configure a non-COREid Federation domain, all configuration fields are presented on one page. When you configure a COREid Federation domain, configuration is divided into two pages. On the first page, you supply the COREid Federation server host and ports. You then click a Next button to allow COREid Federation to generate the SAML service URLs.

For more information on COREid Federation network and domain configuration, see “About COREid Federation” on page 18. Also see Chapter 2. “Planning a COREid Federation Installation” on page 45 for more information on COREid Federation network and domain configuration planning and requirements.

To add a new COREid Federation destination domain

1. From the COREid Federation Administration navigation bar links, click Domains.

The Navigation bar menu expands to show View All Domains, MyDomain, and Add Other Domains menu options.

2. Click Add Other Domain.

The Navigation bar menu expands to show the two Add Other Domain options: COREid Federation and Non-COREid Federation.

3. Click COREid Federation.

The Add Other COREid Federation Domain page appears.

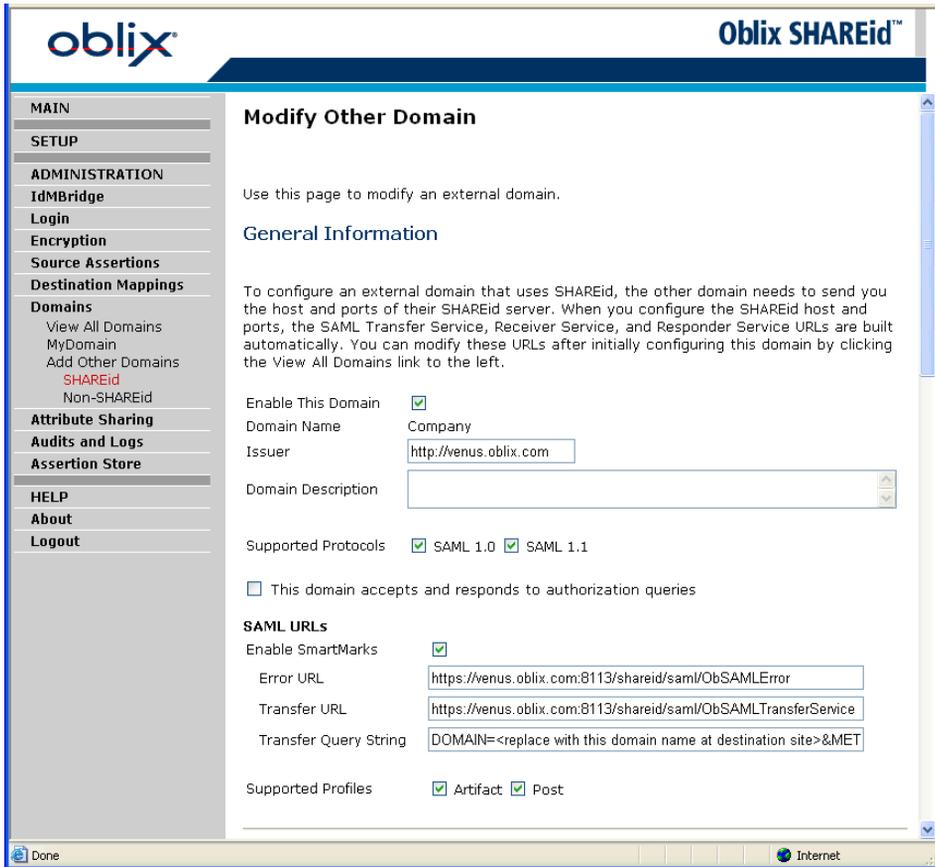
The screenshot shows the Oblix SHAREid administration interface. The top navigation bar includes the Oblix logo and the text 'Oblix SHAREid™'. A left-hand navigation menu lists various sections: MAIN, SETUP, ADMINISTRATION (with sub-items IdMBridge, Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, Assertion Store), HELP, About, and Logout. The 'Domains' section is expanded, showing 'View All Domains', 'MyDomain', 'Add Other Domains', 'SHAREid', and 'Non-SHAREid'. The main content area is titled 'Add Other SHAREid Domain' and contains the following text: 'Use this page to configure external domains. If users from these domains want to access your resources, they are source domains. If your users want to access resources on the external domain, configure the domain as a destination.' Below this is a 'General Information' section with the text: 'To configure an external domain that uses SHAREid, the external domain needs to send you the host and ports of their SHAREid server'. This section includes two input fields: 'Domain Name' and 'Fully qualified hostname' (with an example 'host.company.com'). The 'SHAREid Ports' section follows, with the text: 'If you configure:'. It lists three options: '- Open port only - all SAML URLs use the open port.', '- Open and SSL ports - Configures most SAML URLs to use the server's SSL port. The Responder URL will use the open port.', and '- Open and client certificate authentication port - Configures most SAML URLs to use the open port. The Responder URL will use the client certificate port.' Below these are three input fields: 'Open Port:', 'SSL Port:', and 'Client certificate authentication port:'. At the bottom of the form are 'Submit' and 'Reset' buttons. The browser's status bar at the bottom shows 'Done' and 'Internet'.

4. Complete the General Information fields as described in the following table:

Field	Value
Domain Name	<p>The name of the destination domain. This can be any unique name that you want to provide.</p> <p>If the domain's COREid Federation server is placed behind a proxy, use a domain name appropriate for the proxy.</p>
Fully qualified host name	<p>The machine where COREid Federation is installed at the destination domain. The administrator at the destination domain provides this information.</p> <p>If the domain's COREid Federation server is placed behind a proxy, use the fully qualified host name of the proxy. The proxy hosts the relevant URLs that your domain needs to use.</p>
COREid Federation ports	<p>The ports where the destination domain's COREid Federation listens. The administrator at the destination domain provides this information.</p> <p>If the domain's COREid Federation server is placed behind a proxy, use the ports for the proxy. The proxy hosts the relevant URLs that your domain needs to use.</p>

5. Click Submit.

The Modify Other Domain page appears:



6. Configure the General Information fields on this page as follows:

Field	Value
Enable this domain	Checking this field allows users to access resources on this domain once configuration is complete.
Domain Name	This field repeats the name that you provided on the first configuration pages.
Issuer	This is the name of the SAML authority who issued the assertion. The issuer field does not need to be configured for a destination domain. It is relevant to a source domain in the Assertion Profile page, where this field can be used for SmartWalls. See “Restricting Users to Their Local Domain Using SmartWalls” on page 204 for details.
Domain Description	A description that allows you to identify this domain.

Field	Value
Supported protocols	By default, COREid Federation creates assertions using the SAML 1.1 protocol, but your domain may need to communicate with a domain that still uses SAML 1.0. COREid Federation responds to requests using the protocol specified in the requests. For example, if the destination domain uses SAML 1.0, the source COREid Federation responds with SAML 1.0 if the domain configuration specifies that SAML 1.0 is supported. COREid Federation uses the highest protocol configured for both domains.
This domain accepts and responds to authorization queries	<p>Enables or disables authorization decision queries from this destination domain. This feature is not directly related to Web single sign-on. None of the Web SSO profiles uses the SAML authorization query. Other SAML applications might use this. You set this field if this domain's SAML Requester sends authorization decision queries to your SAML Responder. In this case you would also check this checkbox on the MyDomain configuration page, and the destination domain administrator would do the same for MyDomain on the destination.</p> <p>For the Artifact profile, the source Responder determines if a destination Requester may request an authorization decision as follows:</p> <ul style="list-style-type: none"> • An HTTP request contains a header that matches the user name and the Requester password configured for the domain. • An HTTPS request is made, and COREid Federation performs SSL client certificate authentication and then looks up the domain with a Requester ID or Subject DN that matches the Subject DN field of the certificate.
Enable SmartMarks	<p>Enables the redirection of a non-authenticated user who tries to access a protected resource to a URL where authentication can occur. See “Redirecting Users to a Login Form” on page 198 for details.</p> <p>Both source and destination domains play a role in SmartMarks authentication.</p>
Error URL	This field is not relevant to a source administrator when configuring a destination.
Transfer URL	This field is not relevant to a source administrator when configuring a destination.

Field	Value
Transfer Query String	<p>This field is not relevant to a source administrator when configuring a destination.</p> <p>If the SmartMarks feature is used, the destination administrator inputs a transfer query string. See “Redirecting Users to a Login Form” on page 198 for details.</p> <p>This field is only relevant for a destination domain administrator who is configuring MyDomain.</p>
Supported Profiles	<p>Determines if Artifact or POST, or both, are supported. See “Web Browser Profiles for COREid Federation SAML” on page 32 and “Choosing Artifact or POST Profiles to use with SAML” on page 70 for details.</p> <p>COREid Federation uses the METHOD parameter in your Transfer Service URL to determine what profile to use. If the METHOD is not present, the default profile is Artifact.</p>

7. Scroll down on the page and complete the Configure this Domain as a Destination fields as follows:

Field	Value
Assertion Profile	<p>The assertion profile that your COREid Federation Server is to use when sending assertions to this domain. See “Configuring Assertion Profiles” on page 139 for details.</p>

Field	Value
Requester Authentication	<p>For the Artifact profile, the source Responder service uses this information to authenticate requests from a destination. You obtain this information from the destination administrator.</p> <p>Options:</p> <ul style="list-style-type: none"> • Basic authentication—This is what the destination has configured in the Artifact profile Requester Authentication field. The source protects the Responder service with the user name and password that the destination configures. You can configure your Responder to use Basic authentication over open ports for testing purposes. For production, the destination needs to import your CA certificate and SSL must be used. • X.509 certificate authentication—The Signing Certificate Subject and Issuer DN fields must match the destination domain’s certificate. Note that if you and the destination administrator have already exchanged COREid Federation CA certificates, you can extract the Signing Certificate Subject DN from the certificate in <i>SHAREid_Install_Dir/conf</i> using the <code>keytool -list -v</code> command. To be able to use this option, be sure that your responder URL uses the client certificate port configured during installation. <p>For Basic authentication using SSL and for client certificate authentication, additional configuration is required. See “Configuring Keys and Certificates” on page 243 for details.</p>
Receiver URL	<p>A URL that you obtain from the destination domain’s administrator. The Receiver service at the destination domain processes assertions.</p> <p>See Table 2 on page 33 for details</p>

8. Scroll down on the page to the Configure This Domain as a Attribute or Authorization Authority section.
9. If the domain you are configuring is an attribute authority for an attribute sharing profile, specify the domain’s namespace.
10. Click Submit.

To add a new non-COREid Federation destination domain

1. From the COREid Federation Administration links, click Domains.

The Navigation bar menu expands to show View All Domains, MyDomain, and Add Other Domains menu options.

2. Click Add Other Domains.

The Navigation bar menu expands to show the two Add Other Domain options: COREid Federation and Non-COREid Federation.

3. Click Non-COREid Federation.

The Add Non-COREid Federation Domain page appears (top section shown in the following display):

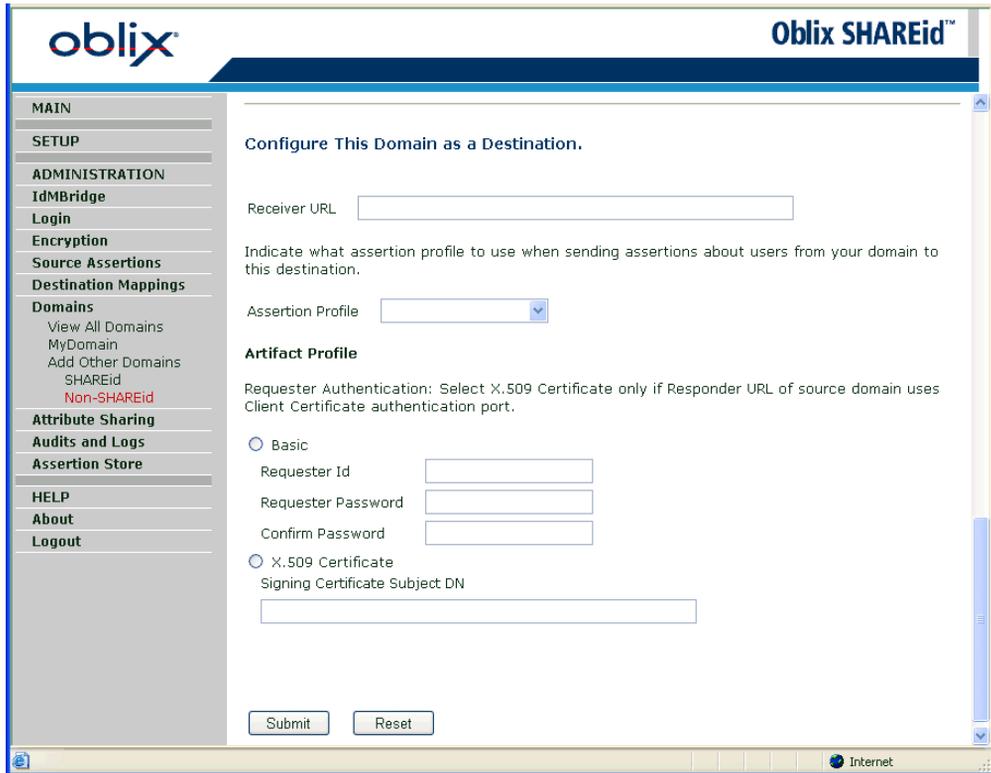
The screenshot shows the Oblix SHAREid administration interface. The top navigation bar includes the Oblix logo and the text "Oblix SHAREid™". A left-hand navigation menu lists various sections: MAIN, SETUP, ADMINISTRATION (with sub-items: IdMBrige, Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, Assertion Store), and HELP (with sub-items: About, Logout). The "Domains" sub-item is highlighted in red. The main content area is titled "General Information" and contains the following text: "Use this page to configure external domains. If users from these domains want to access your resources, they are source domains. If your users want to access resources on the external domain, configure the domain as a destination." Below this text, there are several configuration options: "Enable This Domain" (checkbox), "Domain Name" (text input), "Issuer" (text input), "Domain Description" (text area), "Supported Protocols" (checkboxes for SAML 1.0 and SAML 1.1), and "This domain accepts and responds to authorization queries" (checkbox). Under the "SAML URLs" section, there are "Enable SmartMarks" (checkbox), "Error URL" (text input), "Transfer URL" (text input), and "Transfer Query String" (text input). At the bottom, "Supported Profiles" includes checkboxes for "Artifact" and "Post". The interface is displayed in a browser window with a taskbar at the bottom showing the "Internet" icon.

4. Under General Information, configure the following fields:

Field	Description
Enable this Domain	Checking this field allows users to access resources on this domain once configuration is complete.
Domain Name	The name of the destination domain. This can be any unique name that you want to provide.
Issuer	This is the name of the SAML authority who issued the assertion. The issuer field does not need to be configured for a destination domain. It is relevant to a source domain in the Assertion Profile page, where this field can be used for SmartWalls. See “Restricting Users to Their Local Domain Using SmartWalls” on page 204 for details.
Domain Description	A text description of this domain.
Supported Protocols	By default, COREid Federation creates assertions using the SAML 1.1 protocol, but your domain may need to communicate with a domain that still uses SAML 1.0. COREid Federation responds to requests using the protocol specified in the requests. For example, if the destination domain uses SAML 1.0, the source COREid Federation responds with SAML 1.0 if the domain configuration specifies that SAML 1.0 is supported. COREid Federation uses the highest protocol configured for both domains.
This domain accepts and responds to authorization queries	<p>Enables or disables authorization decision queries from this destination domain. This feature is not directly related to Web single sign-on. None of the Web SSO profiles uses the SAML authorization query. Other SAML applications might use this. You set this field if this domain’s SAML Requester sends authorization decision queries to your SAML Responder. In this case you would also check this checkbox on the MyDomain configuration page, and the destination domain administrator would do the same for MyDomain on the destination.</p> <p>For the Artifact profile, the source Responder determines if a destination Requester may request an authorization decision as follows:</p> <ul style="list-style-type: none"> • An HTTP request contains a header that matches the user name and the Requester password configured for the domain. • An HTTPS request is made, and COREid Federation performs SSL client certificate authentication and then looks up the domain with a Requester ID or Subject DN that matches the Subject DN field of the certificate.

Field	Description
Enable SmartMarks	<p>Enables the redirection of a non-authenticated user who tries to access a protected resource to a URL where authentication can occur. See “Redirecting Users to a Login Form” on page 198 for details.</p> <p>Both source and destination domains play a role in SmartMarks authentication.</p>
Error URL	<p>This field is not relevant to a source administrator when configuring a destination.</p>
Transfer URL	<p>This field is not relevant to a source administrator when configuring a destination.</p>
Transfer Query String	<p>This field is not relevant to a source administrator when configuring a destination.</p> <p>The destination domain inputs the source domain’s Transfer Query String when configuring the source domain. The string required for transfer to the source domain if SmartMarks is used. See “Redirecting Users to a Login Form” on page 198 for details.</p> <p>This field is only relevant for a destination domain administrator who is configuring MyDomain.</p>
Supported Profiles	<p>This field refers to supported Web browser profiles. See “Web Browser Profiles for COREid Federation SAML” on page 32 for details.</p>

The following display shows the Configure This Domain as a Destination section of the Non-COREid Federation Domain page:



5. Under Configure this Domain as a Destination, configure the following fields:

Field	Description
Assertion Profile	The name of the assertion profile that your COREid Federation is to use when sending assertions to this domain. See "Configuring Assertion Profiles" on page 139 for details.

Field	Description
Requester Authentication	<p>For the Artifact profile, the source Responder service uses this information to authenticate requests from a destination. You obtain this information from the destination administrator.</p> <p>Options:</p> <ul style="list-style-type: none"> • Basic authentication—This is what the destination has configured in the Artifact profile Requester Authentication field. The source protects the Responder service with the user name and password that the destination configures. You can configure your Responder to use Basic authentication over open ports for testing purposes. For production, the destination needs to import your CA certificate and SSL must be used. • X.509 certificate authentication—The Signing Certificate Subject and Issuer DN fields must match the destination domain's certificate. Note that if you and the destination administrator have already exchanged COREid Federation CA certificates, you can extract the Signing Certificate Subject DN from the certificate in <i>SHAREid_Install_Dir/conf</i> using the <code>keytool -list -v</code> command. To be able to use this option, be sure that your responder URL uses the client certificate port configured during installation. <p>For Basic authentication using SSL and for client certificate authentication, additional configuration is required. See "Configuring Keys and Certificates" on page 243 for details.</p>
Receiver URL	<p>A URL that you obtain from the destination domain's administrator. The Receiver service at the destination domain processes assertions.</p> <p>See Table 2 on page 33 for details.</p>

6. Scroll down on the page to the Configure This Domain as a Attribute or Authorization Authority section.
7. If the domain you are configuring is an attribute authority for an attribute sharing profile, specify the domain's namespace.

To modify or delete a destination domain

1. From the COREid Federation Administration links, click Domains.

The Navigation bar menu expands to show View All Domains, MyDomain, and Add Other Domains menu options.

2. Click View All Domains.

The COREid Federation Administration Console displays a list of all currently defined assertion profiles.

3. Click a link for the domain you wish to modify.

Note: If you want to delete a domain, click the checkbox next to the domain that you want to delete and click the Delete button.

4. Configure the fields as described in “To add a new COREid Federation destination domain” on page 156.

5 **Configuring Destination Domains**

As a destination domain, users authenticated at source domains request access to resources that reside on your site. You use COREid Federation to map these users to local users (defined within your domain and grant permissions to access resources based on policies defined in your Identity Management System (Oracle COREid or SiteMinder).

This chapter provides information on configuring a destination domain to protect your Web site resources and specify authentication and authorization for source domains that will access your site. This chapter addresses the following topics:

- “Using the COREid Federation Administration Console” on page 170
- “About Destination Domain Configuration” on page 172
- “Using the Setup Wizard” on page 175
- “Destination Domain Administration from the COREid Federation Console” on page 177
- “Configuring an IdMBridge” on page 178
- “Configuring Assertion Mappings” on page 179
- “Configuring MyDomain” on page 184
- “Configuring a Source for a Destination Domain” on page 189
- “Configuring Password and Certificate Store Encryption” on page 196

Using the COREid Federation Administration Console

All operations to configure a COREid Federation server for operation as a source domain, destination domain, or both, are performed using the COREid Federation Administration Console. The Administration Console provides a Web browser-based interface through which you can select navigation bar menu options to perform specific COREid Federation configuration tasks. To use the Administration Console, you must have already started the COREid Federation Server on the machine you want to configure.

To start the COREid Federation Administration Console

1. From an open Web browser window, enter the login URL:

`http://machine name:openport/shareid/`

or

`https://machine name:sslport/shareid/`

where *machine name* is the server where COREid Federation is installed and *port* is the COREid Federation host listener port.

Note: On a Windows machine, you can also start the Administration Console from the Start menu, clicking Start > Programs > COREid Federation Server (*port*) > COREid Federation Administration.

You must log in directly to the COREid Federation server. Do not log in through the COREid Federation proxy, if installed.

2. Log in using the username and password supplied during installation.
The COREid Federation Administration Console Main page appears.

The screenshot shows the Oblix SHAREid 2.5 administration console. The top header includes the Oblix logo and the text 'Oblix SHAREid™'. The left sidebar contains a navigation menu with the following items: MAIN, SETUP, ADMINISTRATION (with sub-items: IdMBridge, Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, Assertion Store), and HELP (with sub-items: About, Logout). The main content area is titled 'Welcome to SHAREid 2.5' and contains the following text:

Use the links on the left as follows:

Setup is a first-time configuration wizard that is provided to simplify initial configuration of your SHAREid System. The setup wizard is optional but recommended for users who are new to SHAREid.

IdMBridge is for configuring a SHAREid engine that communicates with the COREid / LDAP directory / RDBMS / SiteMinder that processes user identities and authorizations.

Login is for configuring how users authenticate locally to SHAREid.

Certificates is for configuring a store for the SSL and digital signatures that are required for secure communication with other domains.

Source Assertions enables you to configure your domain as a source. A source domain is where a request for a resource originates. If your users want to access resources on other domains, you are a source. Source assertions contain information that destination domains use to authenticate your users.

Destination Mappings is for configuring your domain as a destination. If users from other sites want to access your resources, you are a destination domain. Using destination mappings, you map assertion attributes to local user identities.

Domains enables you to configure properties for your domain and other source and destination domains, for example, the hosts and ports used for communication.

Attribute Sharing allows clients within the local domain to use SHAREid to determine if a user from another domain has attributes with specified values. This is used by the COREid Access Server for authorization of user requests based on attribute values.

Audits and Logs enables you to determine if log and audit data is collected. Text descriptions of system events are written to a log file, and generated and received assertions are written to an audit file.

Assertion Store enables you to choose the method for storing assertions.

The bottom of the screenshot shows a taskbar with an Internet Explorer icon and the text 'Internet'.

The Administration Console displays a left-side navigation bar from which you can select menu options to perform specific COREid Federation configuration tasks. The option labelled MAIN is the page first displayed when you start up the Administration Console. Text on this page provides a summary description of each main selection in the navigation bar. Below the Main starting page, the option labelled SETUP provides an option to run the COREid Federation Setup Wizard, which performs a step-by-step initial configuration of a domain. Below SETUP, is a section labelled ADMINISTRATION, which provides options to select and perform individual COREid Federation configuration and setup tasks. Using these options, you can define new configuration items or update existing ones.

The remaining sections of this chapter provide instructions on using the Administration Console to perform specific destination domain configuration operations. To obtain additional information while using the Administration Console, click the Help button to display a PDF copy of this guide. In addition, when you have finished COREid Federation configuration, you can click the Logout menu option to exit the Administration Console.

About Destination Domain Configuration

Basic setup for COREid Federation consists of configuring your domain as a source, a destination, or both.

- A source domain is the location for users who want to access resources on a remote Web site.
- A destination domain is the location that contains the Web site resources that users want to access.

Note: This chapter discusses configuring COREid Federation as a destination domain. For information on setting up COREid Federation as a source domain, see “Configuring Source Domains” on page 95.

Destination domain configuration consists of:

- Protecting your Web site resources using the COREid Access System or SiteMinder. (For more information on configuring SiteMinder for use with COREid Federation, see “SiteMinder System Configuration for COREid Federation” on page 291.)
- Ensuring that users from source domains can be associated with local user identities.
- Sending information about your domain to the source domains so they can configure their domains to send assertions to your domain.

Task overview: configuring a destination domain

1. Configure communication with your Identity Manager System; for COREid, the AccessGate and Access Server; or using SiteMinder. (For more information on configuring SiteMinder for use with COREid Federation, see “SiteMinder System Configuration for COREid Federation” on page 291.)

Communication between COREid Federation and these components is handled by the IdMBridge.

For a destination domain using COREid, the user data repository is the COREid System (including the COREid Server, WebPass, Access Server, Access Manager, and AccessGate). You can also use SiteMinder to manage the user data repository as well as authorize access to protected resources.

2. Map attributes in a received assertion to a local user identity.

This enables the source domain user to assume the identity of a local user who is authorized to access the requested resource.

3. Configure your local domain (MyDomain).

Your local domain always has the name of MyDomain.

4. Configure source domains.

You must identify at least one source domain to COREid Federation. The source domain can use COREid Federation or another SAML-compliant product.

Your domain (MyDomain) uses the information about the source domain when receiving assertions from the source, and when requesting a full assertion from the source when the Artifact profile is used.

5. Exchange information with the source domain.

You and the source domain administrator must exchange configuration information. See “Exchanging Information Between Domains” on page 77.

Configuration Prerequisites

Before configuring COREid Federation, you need to be sure that resources are protected using policies that you configure from the Access Manager and Access System Console (if you’re using COREid as the IdMBridge for your destination site). There are three types of policies you may need to configure:

- Policies for users from source domains who have authenticated to the source domain.
- Policies for users from your local domain.

- Policies for users from source domains who have not yet authenticated to the source domain. See “Redirecting Users to a Login Form” on page 198 for details on this topic.

Note: If you’re using the SiteMinder IdMBridge for your destination domain, refer to Appendix B on page 291 for more information on the types of policies you need to configure using SiteMinder.

Also, you may need to create local user identities so that the users from source domains can be mapped to these identities. Using COREid, these user identities can be created manually in the COREid User Manager, or they can be created automatically (on demand) when new users from source domains request access to your resources. For information on creating on-demand user identities, see “Automatically Mapping User Identities Using SmartMaps” on page 208.

Note that if you are using COREid, you must have at least one identity defined in COREid to which you can map users from the source domain. You can map many source users to one destination identity, or you can create a 1:1 correspondence between source and destination identities. You cannot, however, map one source user to many destination identities or configure a many-to-many mapping.

Methods Available for Configuration

The COREid Federation application provides two methods for destination domain setup:

- From the COREid Federation Administration Console, select the Setup Wizard option, recommended for first-time users of COREid Federation, that performs basic COREid Federation configuration for operation as a Destination domain.
- Following initial COREid Federation setup, choose the COREid Federation Administration Console’s options (located in the Administration section of the left-side navigation bar), pertaining to specific tasks to reconfigure or customize COREid Federation configuration. In some cases, the Administration section menu options provide additional configuration options and parameters not available in the Setup Wizard.

Using the Setup Wizard

If you are configuring your local domain for the first time, use the COREid Federation Setup Wizard. This wizard guides you through all of the steps required for basic domain setup.

Note: When you access COREid Federation for the first time, it takes a while for pages to appear after clicking the links or the Next button in the Setup Wizard. This is due to jsp page compilation that occurs when the pages are accessed for the first time.

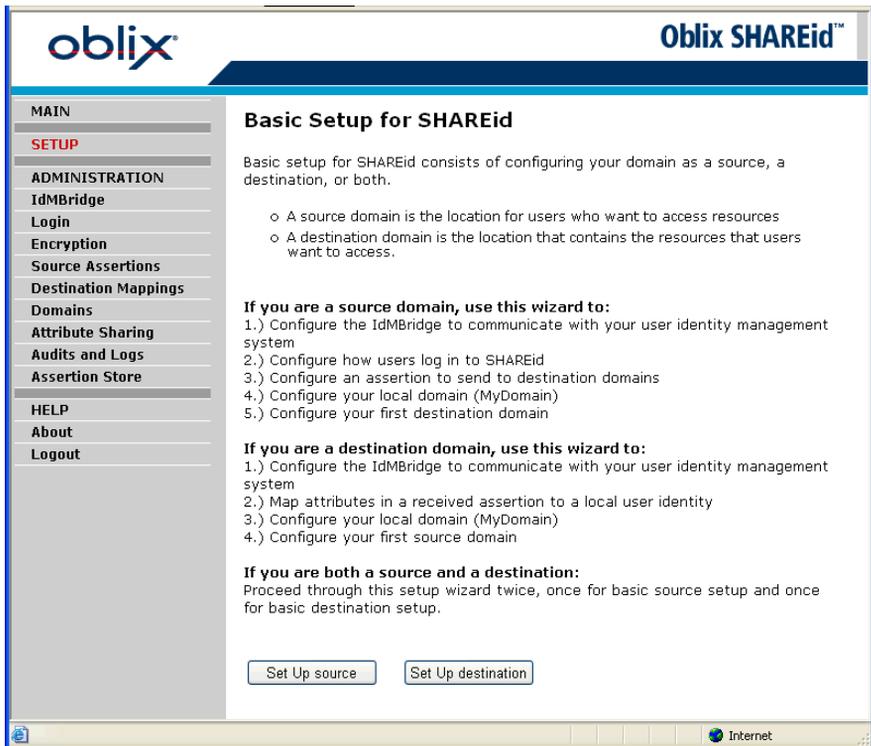
If you are configuring your local domain for the first time, you may want to use the COREid Federation Setup Wizard option first. This wizard guides you through all of the steps required for source or destination domain setup.

To start the Setup Wizard

1. Open the COREid Federation Administration Console.
See “Starting Up and Logging In to COREid Federation” on page 87 for details.

2. Click the SETUP link.

The first page of the Setup Wizard appears.



3. To configure your installation as a destination domain, click Set Up Destination.

The Setup Wizard now takes you through the configuration steps to complete an initial destination domain configuration.

When you've completed all the steps, the End of Destination Setup page appears.

4. After capturing the summary configuration information on this page, click Finish to exit the Setup Wizard and return to the main Administration Console page.

Once you have completed COREid Federation configuration of your domain, you will need to exchange the following information with other partner source domains in your network:

- Your domain information, which includes the Transfer Service, Responder Service, and Receiver Service URLs. These URLs include the host name and port numbers of your COREid Federation or COREid Federation Proxy Server.

- You need to obtain the CAs of the COREid Federation and COREid Federation Proxy Servers at the source sites.

Note: After running Setup, if you need to close down COREid Federation, do not force the shutdown by clicking End Now if presented with a Program Not Responding dialog. Allow the program to finish its own shutdown processes.

Destination Domain Administration from the COREid Federation Console

In addition to using the Setup Wizard, the COREid Federation provides destination domain configuration from individual navigation bar menu options available from the Administration Console's left-side navigation bar. For first-time configuration of COREid Federation, it is recommended that you use the Setup Wizard. (See "Using the Setup Wizard" on page 175.) The Wizard ensures that you perform all of the tasks required for initial configuration. After performing initial setup, use the standard administration pages rather than the Setup Wizard.

The remaining sections of this chapter describe configuration of destination domains using the menu options available in the Administration section of the navigation bar. The sections are included in the order in which the menu options appear in the navigation bar:

- **IdMBridge**—for destination domains, configure the user data repository from which the IdMBridge finds information about local users. Options that can be used for a destination domain are COREid or SiteMinder.

See "Configuring an IdMBridge" on page 178 for more information on configuring an IdMBridge using the IdMBridge menu option.

- **Login**—Only configured for source domains.
- **Encryption**—configure a store for the SSL and digital signatures that are required for secure communication with other domains. Two submenu options are provide. The Certificates option lets you generate a new key for encrypting COREid Federation login sessions and passwords. The Password option lets you specify passwords and the location of the certificate store. See "Configuring Password and Certificate Store Encryption" on page 196 for more information.
- **Source Assertions**—only configured for source domains, defines assertion profiles that specify the information that needs to be placed in assertions. Profiles determines the information that destinations can use to authenticate users from your domain. The profiles also specify mappings of attributes provided by a destination domain to local identity information stored in the

source domain's user data repository. See “Configuring Assertion Profiles” on page 139 for more information.

- **Destination Mappings**—for destination domains, map attributes from assertions received from a source domain to local user identities at the destination. For more information on using this option, refer to “Configuring Assertion Mappings” on page 179.
- **Domains**—configure properties for your domain (MyDomain) and other source and destination domains with which your domain will communicate. For more information on configuration of your local domain as a destination domain, see “Configuring MyDomain” on page 184.
- **Attribute Sharing**—configure X.509 Certificate-authenticated clients within the local domain to use COREid Federation to determine if a user from another domain has attributes with specified values. Verification of these attribute values from the client's home domain is used by the COREid Access Server for authorization of user requests of protected resources. For more information on using this option, refer to “Configuring COREid Federation to Use X.509 Attribute Sharing Profiles” on page 221.
- **Audits and Logs**—configure and enable COREid Federation audit and system log recording. COREid Federation writes text descriptions of system events to a log file. When enabled, generated and received assertions are written to an audit file or database. For more information, see Chapter 8, “About COREid Federation Audit and System Logs” on page 277.
- **Assertion Store**—configure COREid Federation source or destination domains to store assertion information in local files or a database. Most commonly used in situations where you want to load-balance COREid Federation Servers and need to provide a common data store each domain server can access. For more information, see Chapter 6, “Setting up COREid Federation Load Balancing and Failover” on page 215.

Configuring an IdM Bridge

As a destination domain, COREid Federation receives assertions about users from source domains and maps the subject of the assertion to a local identity. Optionally, you can extract other attributes in an assertion for user authentication, for record-keeping, and for user authorization.

Destination domains use the COREid or SiteMinder Identity Management System as the authoritative source of data for mapping assertions to users. The COREid or SiteMinder system must be installed and configured prior to configuring the COREid Federation IdM Bridge. (See Appendix B, “SiteMinder System Configuration for COREid Federation” on page 291 for more information.)

The fields and values you provide on this page are the same as those provided when configuring the COREid Federation IdMBridge for a source domain using COREid or SiteMinder. See “Configuring a COREid IdMBridge” on page 115 or “Configuring a SiteMinder IdMBridge” on page 129 for details.

For an overview of the COREid IdMBridge, see “Planning for a COREid IdMBridge (Source or Destination Domains)” on page 56. For more information on the SiteMinder IdMBridge, see “Planning for a SiteMinder IdMBridge (Source or Destination Domain)” on page 61 and “Configuring a SiteMinder IdMBridge” on page 129.

Configuring Assertion Mappings

You must define at least one assertion mapping for your domain to be able to interoperate with a source domain. You can add more assertion mappings after you finish the Setup Wizard.

Assertion-to-user mappings should be unique. If you create a non-unique mapping, you may get unexpected results. If a mapped user has unexpected values in their LDAP entry, you may want to use the LDAP search tool to verify the mapping. For an overview of assertion mappings, see “About SAML Assertion Elements and Attributes” on page 65.

Note: If you’re using the COREid IdMBridge and you modify an assertion mapping for a destination domain and restart the destination’s COREid Federation server, COREid Federation updates the corresponding COREid Access System authentication scheme. However, if the COREid Access Server is down when the COREid Federation server is restarted, the authentication scheme is disabled. The COREid administrator needs to manually enable the scheme in the Access Manager Console to re-enable the authentication scheme.

To add new assertion mappings

1. From the COREid Federation Administration links, click Destination Mappings.
2. Click Add Mappings

The Console displays the Add Assertion-to-User Configuration page. Note that the specific page that is displayed depends on the particular destination domain IdMBrige you have configured and is set as the active IdMBrige.

The following display shows the Assertion-to-User Mappings page that appears when you are using the COREid IdMBrige:

oblix **Oblix SHAREid™**

MAIN
SETUP
ADMINISTRATION
IdMBrige
Login
Encryption
Source Assertions
Destination Mappings
View All
Add Mappings
Domains
Attribute Sharing
Audits and Logs
Assertion Store
HELP
About
Logout

Add Assertion-to-User Mappings

When an assertion has been validated, the assertion needs to be mapped to a local identity. This mapping uses an attribute in the received assertion to determine who the local user is.

Mapping Name

Description

Authentication Scheme Level
Level Used by the COREid Access System

Top Directory Node
Node Used in Searches for Matching Users (the search base)
(Example: dc=mycompany,dc=mygroup,dc=com)
Person Object Class Used by the COREid Access System

Assertion Attributes
The following mappings allow assertions to be matched to local users.
Use SubjectName as Assertion Attribute if "User Attribute for Subject" (or the equivalent) is configured in the corresponding assertion profile at the source site. If the SubjectName is an X.509 DN, you can use components of the DN as SubjectName.COMPONENT, e.g., SubjectName.CN or SubjectName.OU.
If using the SmartWalls feature you may need to also enter Domain as the Assertion Attribute. See documentation for more details.

Mapping Attributes	
Assertion Attribute	User LDAP Attribute
<input type="text"/>	<input type="text"/>
<input type="button" value="Add New Row"/>	

Internet

3. Specify a name for the mapping.

You may want to use the name of the source domain for which these mappings will apply. Or, if this mapping will apply to assertions from multiple source domains, the name could indicate the kind of mapping being done, for example, SubjectName-to-Mail.

4. Write a description for this assertion mapping.

5. If you are using a COREid IdMBridge for your destination domain, input an integer for an authentication scheme level.

Using COREid, you configure authentication scheme levels in the COREid Access System Console. The security level reflects the degree of security used to protect transport of credentials from the user.

The security level affects single sign-on capability. If a user is authenticated for a resource at a specified level, the user is automatically authenticated for other resources in the same or in different policy domains if the resources have the same or lower security level as the original resource.

See the *NetPoint Administration Guide* for more information about the security level using COREid.

6. If you are using a COREid IdMBridge for your destination domain, specify the top directory node where searches for a user will start.

7. Specify the person object class.

For COREid, this is the name of the structural object class configured for the User Manager in the COREid System, for example, inetOrgPerson.

8. Configure what assertion attributes map to what destination attributes.

You and the source domain administrator must coordinate on the assertion attributes to be mapped.

If multiple assertion attributes are specified in an assertion mapping, an “AND” relationship is used when matching these attributes to a local user identity. For example, if the following assertion attributes are mapped:

Assertion attribute 1—SubjectName

LDAP attribute 1—Mail

Assertion attribute 2—Workgroup

LDAP attribute 2—Ou

An assertion generated for `jsmith@example.com` with a Workgroup of Sales will match the directory entry for a user with a Mail attribute of `jsmith@example.com` and an ou attribute value of Sales.

The following table provides a summary of the fields included on the Add Assertion-to-User Mapping page:

Field	Value
Mapping Name	A name for this assertion mapping. You use the mapping name when you configure a source domain.
Description	A description for this assertion mapping.
Authentication Scheme Level	Set level of security used to protect transport of credentials from user. If a user is authenticated for a resource at a specified level, the user is automatically authenticated for other resources in the same or in different policy domains if the resources have the same or lower security level as the original resource.
Top Directory Node	The top node in the directory where searches begin.
Mapping Attributes	These are pairs of assertion and LDAP attributes, for example: Assertion attribute—SubjectName LDAP attribute—Mail See “About SAML Assertion Elements and Attributes” on page 65 for details.

9. Click Submit.

To configure assertion mappings

1. From the COREid Federation Administration Console, click Destination Mappings.

In the right-hand pane, the Configure Assertion-to-User Mappings for a Destination Domain page appears. In the left-hand pane, the View All and Add Mappings links appear.

2. Click Add Mapping.

The Add Assertion-to-User Mapping page appears.

3. Click Submit.

To modify an assertion mapping

1. From the COREid Federation Administration Console, click Destination Mappings.

In the right-hand pane, the Configure Assertion-to-User Mappings for a Destination Domain page appears. In the left-hand pane, the View All and Add Mappings links appear.

2. Click View All.

3. Select the mapping and click Modify.

To delete an assertion mapping

1. From the COREid Federation Administration Console, click Destination Mappings.

In the right-hand pane, the Configure Assertion-to-User Mappings for a Destination Domain page appears. In the left-hand pane, the View All and Add Mappings links appear.

2. Click View All.

3. Select the mapping that you wish to delete and click Delete.

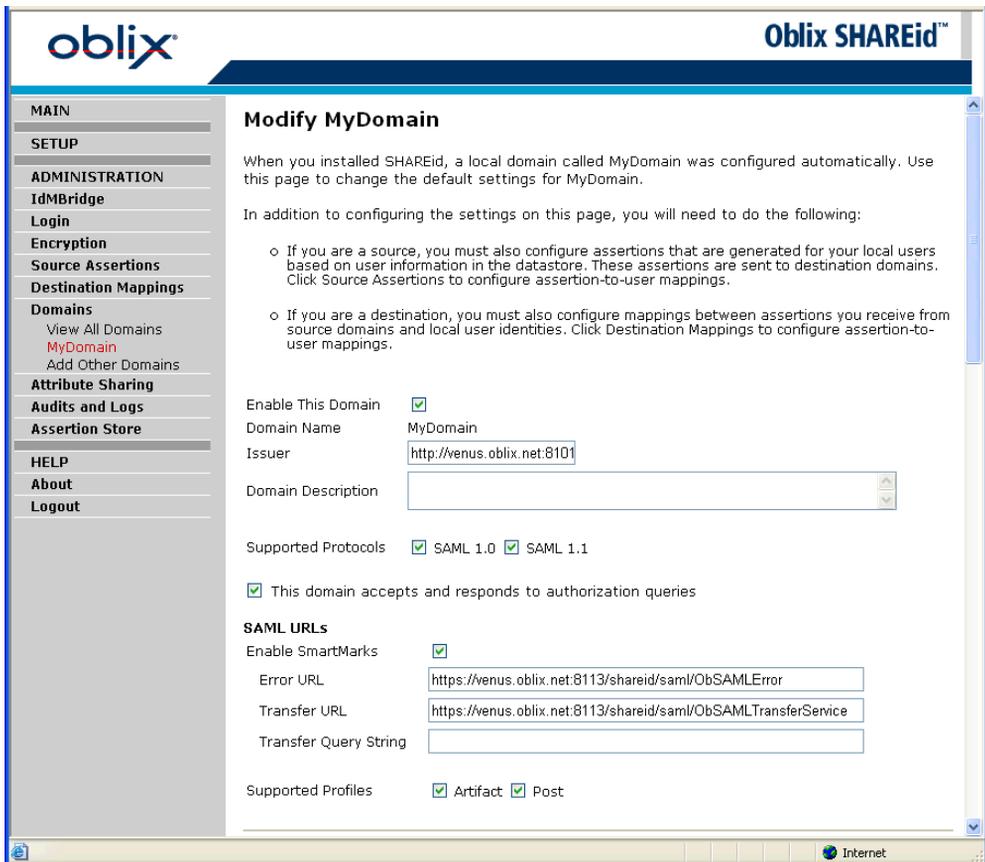
Configuring MyDomain

Whether you are a source domain, a destination domain, or both, your local domain is always called MyDomain. You can modify, but cannot add or delete MyDomain. When you installed COREid Federation, a number of defaults were set for your MyDomain installation, including the COREid Federation services URLs (Transfer, Receiver, and so on) that are configured based on the host and port numbers you supplied during installation.

To change MyDomain Configuration

1. From the COREid Federation Administration links, click Domains > MyDomain.

The Modify MyDomain Configuration page appears.



oblix Oblix SHAREid™

Modify MyDomain

When you installed SHAREid, a local domain called MyDomain was configured automatically. Use this page to change the default settings for MyDomain.

In addition to configuring the settings on this page, you will need to do the following:

- o If you are a source, you must also configure assertions that are generated for your local users based on user information in the datastore. These assertions are sent to destination domains. Click Source Assertions to configure assertion-to-user mappings.
- o If you are a destination, you must also configure mappings between assertions you receive from source domains and local user identities. Click Destination Mappings to configure assertion-to-user mappings.

Enable This Domain

Domain Name MyDomain

Issuer

Domain Description

Supported Protocols SAML 1.0 SAML 1.1

This domain accepts and responds to authorization queries

SAML URLs

Enable SmartMarks

Error URL

Transfer URL

Transfer Query String

Supported Profiles Artifact Post

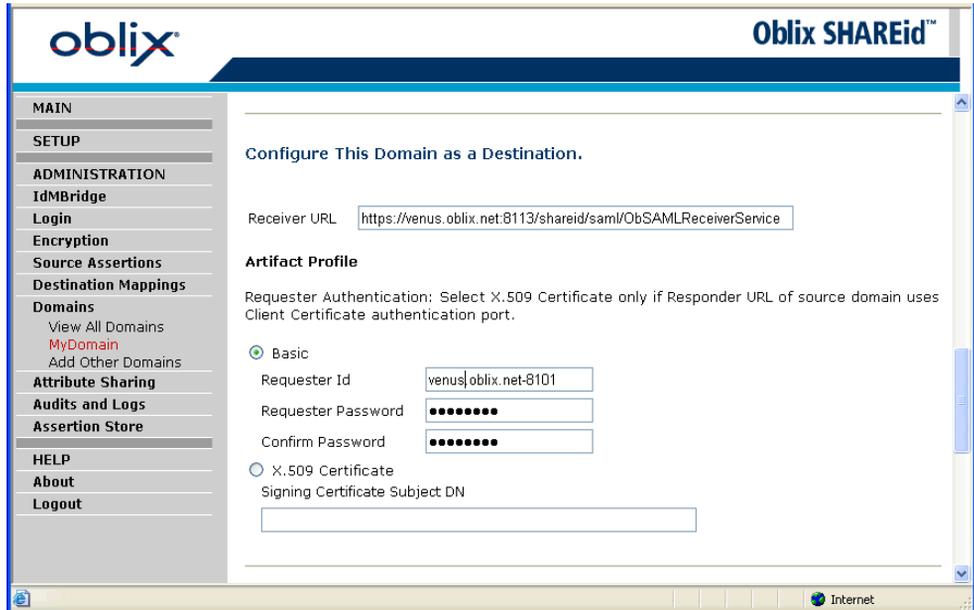
Internet

2. Configure the following:

Field	Description
Enable this domain	Checking this field allows users to access resources on this domain.
Domain name	This is always MyDomain for the local domain.
Issuer	This is the name of the SAML authority who issued the assertion. The issuer field does not need to be configured for a destination domain. It is only relevant to a destination domain administrator when configuring a source domain. See “Configuring a Source for a Destination Domain” on page 189 for details.
Supported Protocols	By default, COREid Federation creates assertions using the SAML 1.1 protocol, but your domain may need to communicate with a domain that still uses SAML 1.0. COREid Federation responds to requests using the protocol specified in the requests. For example, if the destination domain uses SAML 1.0, the source COREid Federation responds with SAML 1.0 if the domain configuration specifies that SAML 1.0 is supported. COREid Federation uses the highest protocol configured for both domains.
This domain accepts and responds to authorization queries	<p>Enables or disables authorization decision queries for MyDomain. This feature is not directly related to Web single sign-on. None of the Web SSO profiles uses the SAML authorization query. Other SAML applications might use this. You set this field if you want your domain’s SAML Requester to send authorization decision queries to SAML Responders. In this case the source domain or domains would also check the field “This domain accepts and responds to authorization queries” checkbox for their domain.</p> <p>This field is only relevant if you intend to support authorization decisions queries, as described in the SAML protocol specification.</p>
Enable SmartMarks	<p>Enables the redirection of a non-authenticated user who tries to access a protected resource to a URL where authentication can occur. See “Redirecting Users to a Login Form” on page 198 for details.</p> <p>Both source and destination domains play a role in SmartMarks authentication.</p>

Field	Description
Error URL	<p>This component receives error information from a destination, parses it, and puts the it into a custom form for presentation to users. See “Error Reporting” on page 210 for details.</p> <p>The destination’s error URL can be used with the SmartMarks feature to redirect errors generated at the destination for display to a local (destination) user. See “Redirecting Users to a Login Form” on page 198 for details.</p>
Transfer URL	This field is only relevant for a source domain.
Transfer Query String	<p>The query string required for transfer back to the source domain if SmartMarks login is used. See “Redirecting Users to a Login Form” on page 198 for details.</p> <p>An example of the Transfer Query String: DOMAIN=<i>DomainB</i>&METHOD=<post or artifact>&TARGET= where <i>DomainB</i> is the name of the domain used by the source domain to access your (destination) server.</p>
Supported Profiles	<p>This field refers to supported Web browser profiles. See “Web Browser Profiles for COREid Federation SAML” on page 32 for details.</p> <p>COREid Federation uses the METHOD parameter in your Transfer Service URL to determine what profile to use. If the METHOD is not present, the default profile is Artifact.</p>

3. Scroll down the page to the Configure This Domain as a Destination section.



4. Under Configure This Domain as a Destination, the following items are configurable:

Field	Description
Requester Authentication	<p>This field defines how your domain authenticates to a source domain. You give this information to the source administrator. For the Artifact profile, the source Responder service uses this information to authenticate requests from a destination.</p> <p>Options:</p> <ul style="list-style-type: none"> • Basic authentication—If you select Basic authentication, the source protects the Responder service with the user name and password that the destination configures. You supply the user name and password to the source domain. You can configure your Requester to use Basic authentication over open ports for testing purposes. For production, you need to import the source’s CA certificate and SSL must be used. • X.509 certificate authentication—The Signing Certificate Subject and Issuer DN fields refer to the subject and issuer DNs in your COREid Federation signing certificate. You input your signing certificate subject DN here and provide it to the source domain. You can extract the signing certificate subject and issuer DNs from the certificate in <i>SHAREid_Install_Dir/conf</i> using the <code>keytool -list -v</code> command. To be able to use this option, be sure that your responder URL uses the client certificate port configured during installation. <p>For Basic authentication using SSL and for client certificate authentication, additional configuration is required. See “Configuring Keys and Certificates” on page 243 for details.</p>
Receiver URL	<p>Upon receiving an assertion, the Receiver service at the destination domain processes the assertion.</p> <p>See Table 2 on page 33 for details.</p>

Configuring a Source for a Destination Domain

If at least one COREid Federation source domain has already been configured, the COREid Federation Setup Wizard lets you configure basic properties such as the source domain's COREid Federation host and ports and add the source domain to your destination configuration.

Using the Setup Wizard, when you configure a non-COREid Federation domain, all configuration fields are presented on one page. When you configure a COREid Federation domain, configuration is divided into two pages. On the first page, you supply the COREid Federation server host and ports. You then click the Next button to allow COREid Federation to generate the SAML service URLs.

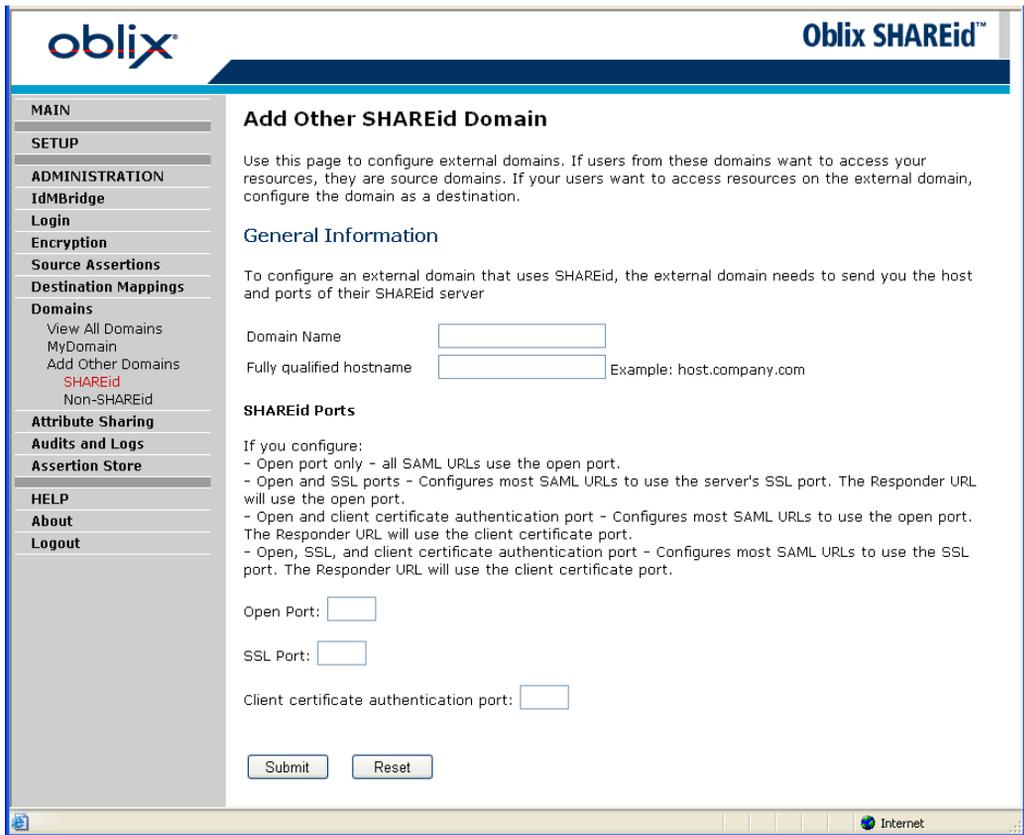
Upon completing the Setup Wizard, you can set additional source domain configuration options by selecting the domain using the Administration Console's Domain > View All Domains menu option.

To add a new COREid Federation source domain

1. From the COREid Federation Administration Console, click Domains.
2. Click Add Other Domains.

3. Click COREid Federation.

The Add Other COREid Federation Domain page appears:



4. Complete the General Information fields on this page as follows:

Field	Value
Domain Name	<p>The name of the source domain. This can be any unique name that you want to provide. You may want to provide a URI, for example:</p> <p style="text-align: center;">source.example.com</p> <p>If the source domain's COREid Federation server is placed behind a proxy, use a domain name appropriate for the proxy.</p> <p>This name can play a role in the SmartWalls feature. See "Restricting Users to Their Local Domain Using SmartWalls" on page 204 for details.</p>

Field	Value
Fully qualified host name	The machine where COREid Federation is installed at the destination domain. The administrator at the source domain provides this information. If the source domain's COREid Federation server is placed behind a proxy, use the fully qualified host name of the proxy. The proxy hosts the relevant URLs that your domain needs to use.
COREid Federation ports	The ports where the source domain's COREid Federation listens. The source domain administrator provides this information. If the source domain's COREid Federation server is placed behind a proxy, use the ports for the proxy. The proxy hosts the relevant URLs that your domain needs to use.

5. Click Submit.

The Modify Other Domain page appears:

oblix Oblix SHAREid™

Modify Other Domain

Use this page to modify an external domain.

General Information

To configure an external domain that uses SHAREid, the other domain needs to send you the host and ports of their SHAREid server. When you configure the SHAREid host and ports, the SAML Transfer Service, Receiver Service, and Responder Service URLs are built automatically. You can modify these URLs after initially configuring this domain by clicking the View All Domains link to the left.

Enable This Domain

Domain Name

Issuer

Domain Description

Supported Protocols SAML 1.0 SAML 1.1

This domain accepts and responds to authorization queries

SAML URLs

Enable SmartMarks

Error URL

Transfer URL

Transfer Query String

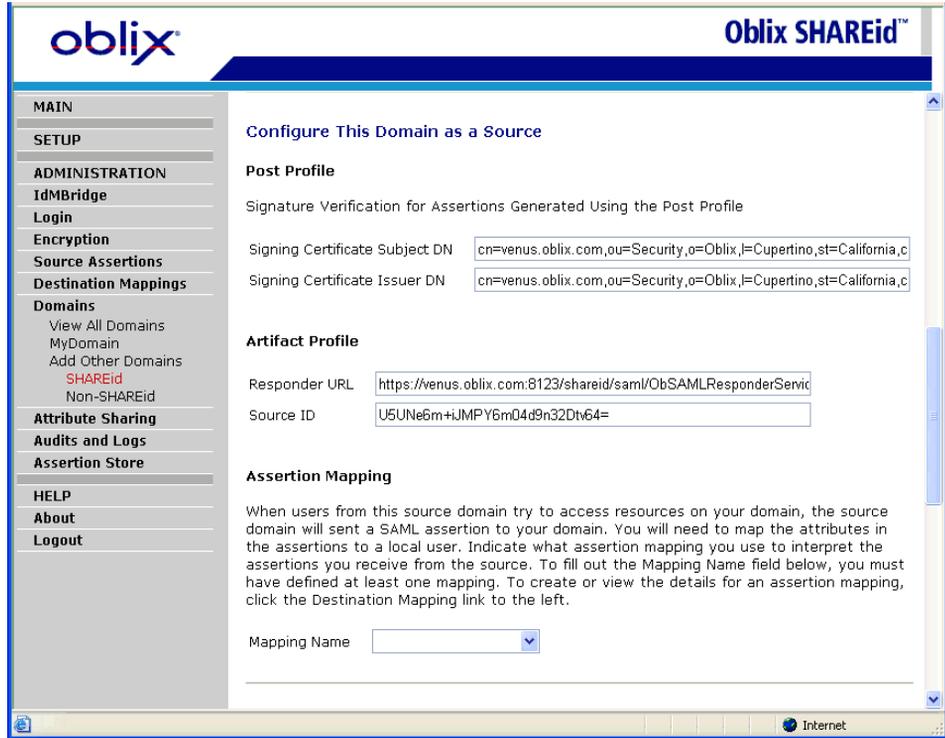
Supported Profiles Artifact Post

6. Configure the General Information on this page as follows:

Field	Value
Enable this domain	Checking this field allows users from this domain to access resources on your domain.
Domain Name	This is a read-only field that repeats the name you assigned on the first page of domain configuration.
Issuer	<p>Input the value that the source domain administrator configured in the Issuer field (this is the issuer field in the source's definition of MyDomain).</p> <p>The source domain's issuer is relevant when configuring SmartWalls. See "Restricting Users to Their Local Domain Using SmartWalls" on page 204 for details.</p>
Domain Description	A description that allows you to identify this domain.
Supported protocols	<p>By default, COREid Federation creates assertions using the SAML 1.1 protocol, but your domain may need to communicate with a domain that still uses SAML 1.0. COREid Federation responds to requests using the protocol specified in the requests. For example, if the destination domain uses SAML 1.0, the source COREid Federation responds with SAML 1.0 if the domain configuration specifies that SAML 1.0 is supported. COREid Federation uses the highest protocol configured for both domains.</p>
This domain accepts and responds to authorization queries	<p>Enables the source domain to authorization decision queries from your destination domain. This feature is not directly related to Web single sign-on. None of the Web SSO profiles uses the SAML authorization query. Other SAML applications might use this. You set this field if you want this domain's SAML Responder to process authorization decision queries from your SAML Requester. In this case you would also check the field "This domain accepts and responds to authorization queries" checkbox for MyDomain.</p> <p>For the Artifact profile, the source domain Responder service determines if a destination Requester is permitted to request an authorization decision as follows:</p> <ul style="list-style-type: none"> • An HTTP request contains a header that matches the user name and the Requester password configured for the domain. • An HTTPS request is made, and COREid Federation performs SSL client certificate authentication and then looks up the domain with a Requester ID or Subject DN that matches the Subject DN field of the certificate.

Field	Value
Enable SmartMarks	<p>This enables the source domain to use the SmartMarks feature. See “Redirecting Users to a Login Form” on page 198 for details.</p> <p>Both source and destination domains play a role in SmartMarks authentication.</p>
Error URL	<p>This component receives error information from a destination, parses it, and puts the it into a custom form for presentation to users. See “Error Reporting” on page 210 for details.</p> <p>For a source domain, the error URL is used for receiving error information generated at the destination and redirected to the source. Get the source’s error URL from the source domain administrator.</p>
Transfer URL	<p>The URL for the Transfer service. The Transfer service initiates the process that enables a user to access resources on another domain. See Table 2 on page 33 for details.</p> <p>You get the Transfer URL from the source domain administrator.</p>
Transfer Query String	<p>The query string is required for a destination domain to transfer a user back to the source domain if SmartMarks login is used. See “Redirecting Users to a Login Form” on page 198 for details.</p> <p>Example of a Transfer Query String: <code>DOMAIN=<i>DomainB</i>&METHOD=<post or artifact>&TARGET=</code></p> <p>where <i>DomainB</i> is the domain name that the source domain uses to access your (destination) server.</p>
Supported Profiles	<p>Select the profile that will be used by this source domain. See “Web Browser Profiles for COREid Federation SAML” on page 32 for details.</p> <p>COREid Federation uses the METHOD parameter in your Transfer Service URL to determine what profile to use. If the METHOD is not present, the default profile is Artifact.</p>

7. Scroll down to the Configure this Domain as Source section.



8. Complete the Configure this Domain as a Source fields as follows:

Field	Value
Signing Certificate Subject DN	<p>You obtain this information from the source domain administrator.</p> <p>For the POST Profile, the source signs assertions using a signing key in a certificate. The destination reads the issuer name in the assertion, and uses the value in this Certificate Subject DN field configured for this source domain to verify the signature.</p> <p>Example: cn=example.oblix.net,ou=Security,o=Oblix, l=Cupertino,st=California,c=US</p>
Signing Certificate Issuer DN	<p>You obtain this information from the source domain administrator.</p> <p>For the POST Profile, the source signs assertions using a signing key in a certificate. The destination reads the issuer name in the assertion, and uses the value in this Certificate Issuer DN field configured for this source domain to verify the signature.</p>

Field	Value
Responder URL	This is the Artifact Profile Responder URL. The source domain administrator provides this information.
Source ID	<p>Get the Source ID from the source domain administrator. For the Artifact profile, the destination uses the Source ID to look up the source domain when preparing to request an assertion. The source domain includes its source ID in the artifact it sends to the destination. When the destination requests a full assertion, it uses the source domain's Source ID in the request.</p> <p>The Source ID is generated automatically from a SHA-1 digest of the Responder URL. The Source ID is only meaningful for the Artifact profile. A checkbox on this page enables you to indicate that this domain supports the Artifact profile.</p>
Assertion Mapping	The name of the assertion mapping that you configured to use when you receive assertions from this source domain. See "Configuring Assertion Mappings" on page 179 for details.

9. Scroll down on the page to the Configure This Domain as a Attribute or Authorization Authority section.
10. If the domain you are configuring is an attribute authority for an attribute sharing profile, specify the domain's namespace.
11. Click Submit.

To add a new non-COREid Federation source domain

1. From the COREid Federation Administration Console, click Domains.
The Configure Domains page appears.
2. Click Add Other Domain.
The Add Other Domains page appears.
3. Click Non-COREid Federation.
The Add Non-COREid Federation Domain page appears.
4. Configure the fields as described in "To add a new COREid Federation source domain" on page 189.

Note: When you configure the Source ID for a non-COREid Federation domain, you can provide either 30 character hex or 28 character base64 source IDs.

To modify a source domain

1. From the COREid Federation Administration Console, click Domains.
The Configure Domains page appears.
2. Click View All Domains.
The View All Domains page appears.
3. Click a link for the domain you wish to modify.
4. Configure the fields as described in “To add a new COREid Federation source domain” on page 189.

To delete a source domain

1. From the COREid Federation Administration Console, click Domains.
The Configure Domains page appears.
2. Click View All Domains.
The View All Domains page appears.
3. Click the checkbox next to the domain that you want to delete and click the Delete button.

Configuring Password and Certificate Store Encryption

COREid Federation provides a database known as the certificate store that contains the keys and certificates used for SSL and digital signatures. Configuring a certificate store is identical for source and destination domains. COREid Federation also maintains encrypted passwords that are used to secure various COREid Federation sessions.

Using COREid Federation, you can need to specify where the certificate store is located and enter keystore password information. See “Configuring Passwords and the Certificate Store” on page 136 for details.

6 Advanced Configuration

COREid Federation provides advanced features for securing your domain and processing user requests. This chapter provides information on the following advanced configuration topics:

- “About Advanced Configuration” on page 198
- “Redirecting Users to a Login Form” on page 198
- “Restricting Users to Their Local Domain Using SmartWalls” on page 204
- “Automatically Mapping User Identities Using SmartMaps” on page 208
- “Customizing the Login Form” on page 209
- “Error Reporting” on page 210
- “Setting up COREid Federation Load Balancing and Failover” on page 215
- “Configuring COREid Federation to Use X.509 Attribute Sharing Profiles” on page 221

About Advanced Configuration

Advanced configuration includes:

- Enabling redirection of non-authenticated users to a login form.
- Ensuring that domains can only send assertions about their own users.
Using COREid Federation, you can prevent rogue domains from sending assertions. SmartWalls ensures that the domain of an assertion issuer matches the domain of the assertion subject.
- Automatically creating local identities for users from source domains.
The destination domain can use COREid IdentityXML functionality to create a local identity for a source user who does not yet have a destination domain identity.
- Customizing the Web login form that is displayed to source domain users.
- Providing custom error messages to users.
A destination can send error messages to a source domain, where these messages can be customized and displayed to users.
- Setting up COREid Federation load balancing and failover
You can configure multiple COREid Federation Servers to distribute load. To load balance COREid Federation Servers, you also need to set up a common data store to store, query, and update assertion information.
- Setting up COREid Federation to use X.509 Attribute Sharing profiles
You can configure COREid Federation source and destination domains to use X.509 certificates and attribute sharing profiles to authenticate and authorize resource access across domains.

Redirecting Users to a Login Form

A user typically authenticates before requesting a resource. However, a user can request a resource using an expired session token or without having obtained a session token. For example, the user can bookmark an application and select the bookmark before authenticating to the source domain. In this case, the destination would want to redirect the user to a login form.

Process overview: Example of accessing a resource before authenticating

1. The user authenticates to the source domain's COREid Federation.
2. The user requests points to a resource on the destination Web site.

3. The user bookmarks the resource.
4. The user logs out, or closes the browser, or the user's session is ended by an idle time-out.
5. The user re-launches the browser.
6. The user selects the bookmark.

In this scenario, the user requests a resource by selecting a bookmark for the resource before authenticating. (Note that the user could have also directly requested the resource by typing in the URL before authenticating locally.) In a default COREid Federation configuration, the user would be challenged to log in to the destination because the request would not be accompanied by an assertion. The user may not be granted access because he or she may not have been given a user name and password for local login at the destination.

The SmartMarks feature enables a destination to redirect a non-authenticated user to an authentication URL before either serving the resource or denying access to it. Note that the SmartMarks process can be applied to local users as well as source domain users.

SmartMarks can be used by any source site, regardless of whether a COREid or an LDAP IdMBridge is used. However, SmartMarks currently is not certified for source domains that use external (portal) authentication.

The following sections assume that you can create policy domains, policies, and authentication rules using the Access System. See the *NetPoint Administration Guide* for details.

Note: To use SmartMarks if you have installed a COREid Federation proxy server, be sure that the proxy server and the destination resources being accessed do not reside on the same machine.

About SmartMarks Authentication Schemes

In the SmartMarks authentication process, a source domain can be any SAML-compliant installation. The destination is always a COREid Federation installation. SmartMarks works as follows:

- If a source domain user has not logged in or the user's session has timed out, SmartMarks redirects the user back to the source domain for authentication.
- SmartMarks redirects users who belong to the destination domain to a destination domain login URL.

Using SmartMarks, the requested resource is protected by a *SmartMarks Login authentication scheme*. COREid Federation automatically configures this authentication scheme in the COREid Access System.

The following is an example of a SmartMarks authentication scheme:

Name—COREid Federation SmartMarks

Description—Created by COREid Federation. You may modify the level and plug-in of this scheme as required for local logins.

Level—1

Challenge Method—Form

Challenge Parameters—creds: userid password

SSL Required—no

Challenge Redirect—<https://shareid-host.example.com:8113/shareid/saml>

Enabled—yes

Plug-ins—examples below:

- `credential_mapping obMappingBase="o=Company,c=US", obMappingFilter="(&(&(objectclass=gensiteorgperson)(uid=%userid%))(|(! (obuseraccountcontrol=*)) (obuseraccountcontrol=ACTIVATED)))"`
- `validate_password obCredentialPassword="password"`

In this authentication scheme, the level and plug-ins are based on the COREid Basic Over LDAP authentication scheme that is optionally created during Access System setup. For the local login feature of SmartMarks to work, the credentials used in the plug-ins must match the credentials provided in the login method configured for COREid Federation, for example, user name and password. In the example above, the credentials are:

- In the credential mapping plug-in, the credential is %userid%.
- In the validate password plug-in, the credential is password.

You can modify the SmartMarks plug-ins specified in the authentication scheme to match the login method, if necessary. However, you should not change the challenge redirect. The challenge redirect ensures that COREid Federation participates in user authentication.

Note: The same SmartMarks authentication scheme is used for all resources that are protected by SmartMarks.

SmartMarks Login Process

The following describes the process for SmartMarks login.

Process overview: How SmartMarks Login works

1. A user requests a resource that is protected by a COREid Access System policy on the destination.

2. A COREid WebGate at the destination intercepts the resource request.

An WebGate is a plug-in that intercepts HTTP requests for Web resources and forwards them to the Access Server for authentication and authorization.

3. The WebGate checks the policy protecting the resource.

The policy contains an authentication rule which in turn contains a SmartMarks authentication scheme. This authentication scheme is configured automatically during COREid Federation installation. The authentication scheme enables the AccessGate to redirect the user to the SmartMarks login facility on the destination domain.

4. The SmartMarks Login facility checks for the presence of an ObSAMLDomain cookie:

For a source domain user—If the user belongs to a source domain, the cookie exists if the user previously accessed the destination domain resource. During the previous session, the destination domain's Receiver service set the ObSAMLDomain cookie in the user's browser.

If an ObSAMLDomain cookie is present, the SmartMarks facility directs the user's browser to the source domain's Transfer service.

Note that the ObSAMLDomain cookie is updated after every successful transfer from the source to the destination. If no transfers occur, the cookie expires after one year.

For a destination domain user—COREid Federation does not put an ObSAMLDomain cookie in a local user's browser.

If the ObSAMLDomain cookie is not present, the Transfer Form facility determines that the user is local, and it redirects the user to a local login form.

Note: The ObSAMLDomain cookie is updated whenever a user accesses a resource where this cookie is set. If not updated the cookie expires after one year. The cookie expiration date is set in the shareid-config.xml file.

SmartMarks Login for Source Domain Users

This section discusses how SmartMarks login works after the user is redirected for authentication.

Process overview: Authenticating the user at the source domain

1. The user is redirected to the source domain's Transfer service.
2. The user is challenged for login.
3. After the user is authenticated, they are transferred to the target resource on the destination.

SmartMarks Login for Destination Domain Users

This section discusses SmartMarks login on the destination domain after COREid Federation determines that the user belongs to the destination.

Process overview: Authenticating the user at the destination domain

1. The user fills out a login form.
2. The destination's COREid Federation authenticates the user and redirects the user to the target resource Web server.

Configuring SmartMarks as a Source Domain

When the SmartMarks login process redirects users from the destination domain, the redirection is received by the source domain's Transfer service.

COREid Federation automatically protects the Transfer service. If your domain uses an LDAP IdMBridge, the scheme that protects the Transfer service is configured when you complete the login configuration page in the COREid Federation Administration Console. If your domain uses a COREid IdMBridge, the Transfer service is protected by a combination of the login configuration settings and Transfer Form policies that COREid Federation configures automatically.

To configure SmartMarks as a source domain

1. From the COREid Federation Administration Console, select Domains > MyDomain, and be sure that Enable SmartMarks is selected.
2. Copy the Transfer Service URL for MyDomain.
3. Give the destination domain your Transfer Service URL.
4. If you have not yet configured the destination domain that will be implementing SmartMarks, from the COREid Federation Administration Console, select Domains > Add Other Domain, and configure this domain.

Configuring SmartMarks as a Destination Domain

For a destination domain, the SmartMarks feature consists of creating a COREid policy domain that uses the SmartMarks authentication scheme to protect different groups of resources. You can protect each group of resources in a unique way. For example, you can give users at source domains access to some resources and deny your local users access to those resources.

To differentiate use of the SmartMarks login scheme, you can combine an authentication rule that includes the SmartMarks login scheme with other policy conditions, for example, different authorization requirements.

To configure SmartMarks as a destination

1. From the COREid Federation Administration Console, select Domains > MyDomain, and select Enable SmartMarks.
2. Enable SmartMarks for the source domain as follows.
Domains > Add Other Domain > COREid Federation or Non-COREid Federation
or
Domains > View All Domains > select the domain link
Click Enable SmartMarks.
3. Obtain the source domain's Transfer Service URL from the source domain administrator.
4. In the COREid Federation Administration Console, specify the URL of the source domain's Transfer service.
Assuming that this is an existing domain, go to:
Domains > View All Domains > select the domain link
Enter the source domain's Transfer Service URL, for example:
`https://saml.a.com/shareid/saml/ObSAMLTransferService`
5. Configure the Transfer Query String, for example:
`DOMAIN=DomainB&METHOD=<post or artifact>&TARGET=`
where *DomainB* is the destination domain.
6. In the COREid Access System, create an authentication rule that contains the SmartMarks authentication scheme.
7. Use that rule in a policy that protects the resources to be covered by the SmartMarks authentication scheme.
8. Create "authorization inconclusive" or "authorization failure" redirect URLs for the policy that protects these resources.

Implementation Notes for SmartMarks

Note that for some configurations, SmartMarks only works with COREid Federation basic and form login, not with authentication using an external mechanism. Restrictions are as follows:

- If a source domain has a COREid Federation server, but no proxy server, external authentication credentials do not work.
- If a source domain has a COREid Federation server installed behind a proxy, but the proxy is not protected by a WebGate, external authentication credentials do not work.

Restricting Users to Their Local Domain Using SmartWalls

SmartWalls functionality enables destination domains to verify that an assertion issuer and an assertion subject are from the same domain. That is, you can ensure that assertions are rejected if the subject of the assertion is from domain X but the assertion was issued from domain Y. SmartWalls prevents a source domain from issuing assertions about a user from any other source domain. This prevents users from accessing protected resources from any domain other than the correct source domain, and it prevents other domains from sending assertions on behalf of a user from the correct source domain.

SmartWalls functionality consists of a set of best practices for structuring your directory tree and mapping assertion attributes. SmartWalls is based on the presence of information about the assertion issuer in an assertion. SmartWalls functionality is based on the assumption that source domain X cannot produce assertions that claim to be issued from domain Y without possessing domain Y's private keys or certificates.

When the POST profile is used, COREid Federation expects to receive a certificate with a particular subject and issuer. The domain information in an assertion is based on the assertion's signature. For the POST profile, when the assertion signature is verified, domain information is also verified. For the Artifact profile, an SSL connection requires a server certificate. The URL in the certificate must match the URL that the destination is actually connected to. The destination verification is based on the SAML Responder authentication that occurs when SSL is enabled. Note that if you use the Artifact profile, you at least need server-side SSL authentication for SmartWalls to be effective.

COREid Federation uses the Issuer attribute of the assertion to identify the domain of the issuer, and then uses configured information about the domain (the signer and issuer DN or the requester ID and password) to verify that the issuer is authentic.

How Destination Domains Implement SmartWalls

Destination domains implement SmartWalls in one of two ways:

- Ensuring that the user's directory entry contains an attribute with the value of the source domain.
- Structuring the directory tree so that the user's entry occurs in a branch reserved for the source domain.

The destination administrator maps an assertion attribute that identifies the assertion issuer to a local LDAP attribute. The LDAP attribute can either be part of the user's directory entry, or it can be a branch of the directory tree. Either way, the subject of the assertion must be mapped to a directory entry that identifies the subject's domain. If the domain in the assertion does not match the domain in your local (destination) directory, the assertion mapping fails.

Note that the source and destination domains must agree on an assertion attribute to identify the user and an attribute to identify the source domain. The destination domain uses these attributes to map the source user to a destination identity. For example:

Assertion attribute—Domain

Destination LDAP attribute—Domain

About the Issuer Statement and the Domain Attribute

All properly constructed assertions have an Issuer attribute to identify who issued the assertion, similar to the following:

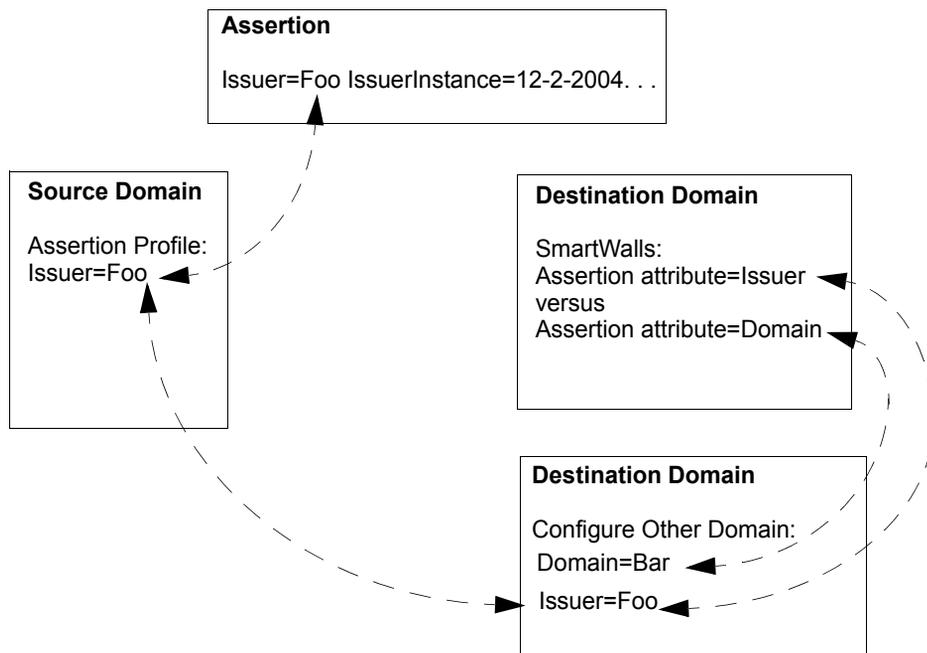
```
<Assertion Issuer="shareid.oblix.com" IssueInstance="2002-05-28">
```

When the source domain administrator configures an assertion profile, he or she supplies a value in the Issuer field. This value is used in the assertion. The destination domain administrator, when configuring this source domain, must also supply this value in the Issuer field for the source.

When the destination administrator also supplies a value in the Domain field when configuring the source. COREid Federation recognizes an assertion attribute called Domain. This attribute does not appear in the assertion. The attribute takes its value from what the destination administrator specifies in the Domain field when configuring the source.

Figure 8 illustrates the Issuer and Domain assertion attributes.

Figure 8 Issuer and Domain assertion attributes



Since the Domain attribute is under control of the destination, its value may be less apt to change than the Issuer attribute.

Attribute Mappings for SmartWalls

As described in “About Assertion Mappings” on page 66, assertion mappings enable you to identify assertion attributes and set up correspondences between those attributes and local LDAP attributes. The mapping allows you to match source domain users with user identities at your local (destination) domain.

To configure SmartWalls, you include an attribute that identifies the assertion issuer in the assertion mapping. As a destination domain administrator, you have two choices:

- Add an attribute for the assertion issuer to the LDAP entry for each user identity created at the destination domain.

To configure attribute mappings for SmartWalls, the destination domain must structure its directory appropriately. User entries must have an attribute that identifies the user's source domain.

Either the Issuer or the Domain attribute can be used. The Issuer attribute appears in the assertion, and is configured by the source domain in its assertion profile. The Domain attribute is a special-purpose attribute that never actually appears in the assertion but is recognized by COREid Federation. The destination administrator specifies the value for the Domain attribute when configuring the source domain in COREid Federation. The configuration page contains a Domain field where this value is specified.

You might opt to use Domain if the Issuer attribute value is less helpful than the Domain attribute value, for instance, if the source domain administrator specified the Issuer field to be a random number.

- Organize the directory tree so that searches are filtered by the source user's domain.

In this case, all users from a particular domain are stored in one branch of the directory tree. The same assertion mapping applies as in the case where the domain information is added to each user's directory entry.

Task overview: Configuring SmartWalls via an attribute in the user's destination directory entry

1. The destination administrator creates an attribute for each source user entry in the directory to specify the assertion issuer for that user.
2. The destination administrator sets up the assertion mapping to map the Domain or Issuer assertion attribute to that LDAP attribute.

Example:

Assertion attribute—Domain

Destination LDAP attribute—Domain

In this example, the value of Domain will match what the source domain specified in the Issuer field when configuring MyDomain.

Task overview: Configuring SmartWalls by structuring the destination directory tree

1. Organize your LDAP directory tree so that each source domain has its own subtree.
2. In the COREid Federation Administration Console, configure an assertion mapping for the Domain or Issuer assertion attribute.

3. Use the mapped attribute for the Domain or Issuer in the LDAP directory mapping base, for example:

`ou=%Domain%,o=Company,c=US`

For example, suppose the source domain users are entered in the destination's directory in the following subtrees:

`ou=PartnerN, o=Company, c=US`

where *PartnerN* is the domain name configured for the source. The mapping base for each subtree could be the following:

`ou=FirstPartner, o=MyCompany, c=US` contains users for FirstPartner

`ou=SecondPartner, o=MyCompany, c=US` contains users for secondPartner.

4. Add entries for source domain users to the appropriate subtree.

This can be done manually or using the SmartMaps feature.

Automatically Mapping User Identities Using SmartMaps

By default, a destination domain administrator must manually create local user identities to which source domain users can be mapped. To simplify the management of user identities, the SmartMaps feature automates creation of a new user identity when an unknown user from a source domain requests a resource.

If your domain is a destination that has the COREid product installed, the SmartMaps feature consists of automatically creating a new profile in COREid based on data in an assertion.

SmartMaps functionality supports three operations:

Invoking IdentityXML—The COREid System's IdentityXML functionality provides a programmatic interface for carrying out actions that a user can perform when accessing a COREid application from a browser. Using custom logic, an IdentityXML workflow can be invoked to create a new user in COREid. The workflow can be multi-step with approvals or single step. If multi-step approval is used, a message can be returned to notify the user at the source domain that approval for access is in process.

Presenting a login page—Users from source domains who have unknown identities can be presented with a login page. The login page takes the credentials for the destination identity and adds the credentials for the user's source domain identity to the destination user profile. Subsequent attempts to access destination resources during the established session occur without prompting the user to authenticate.

Invoking custom logic—The destination domain administrator can invoke custom logic to handle unknown source identities.

Important: The SmartMaps function requires custom code that may only be created by a representative from Oracle. For assistance with implementing SmartMaps, contact your Oracle support representative.

Customizing the Login Form

By default, COREid Federation provides two types of login form:

- A login form that administrators use when logging in to the COREid Federation Administration Console.

This login form is located in

SHAREid_Install_Dir/webapps/shareid/admin-login.jsp

where *SHAREid_Install_Dir* is the directory where COREid Federation was installed.

- A login form presented to all source domain users who need to authenticate to COREid Federation.

This login form is located in

SHAREid_Install_Dir/webapps/shareid/login.jsp

where *SHAREid_Install_Dir* is the directory where COREid Federation was installed.

This form is used for both standard user login and login using the SmartMarks Transfer form.

You can customize these login forms the same way you would modify any authentication or login form used on the Web.

Note: When customizing the login form for LDAP authentication, the login credential must be set to “password.” This is a hard-coded value in COREid Federation.

Error Reporting

COREid Federation provides an error reporting feature that destination domains can use when communicating with a source domain. When an error occurs that pertains to a source domain's attempt to establish communication with a destination domain—for example, to send an assertion—you can redirect error information to the source. The source domain can be configured to receive redirected error information from a destination domain, parse it, and present an error message to the source domain user. From the user's point of view, it is as if the source domain originally issued the error message.

To use the error reporting feature, the destination domain must use COREid Federation. The source domain can use any SAML-compliant application. However, if the source domain does not use COREid Federation, the source domain administrator must define and implement the following:

- A method of receiving the redirected error information
- Parsing the redirected information
- Presenting the resulting error message to source domain users.

Error Redirection for Destination Domains

To implement error redirection as a destination domain administrator, you must convey the following information to the source domain.

Task overview: Error redirection for destination domains

1. The source and destination domain administrators discuss using the error redirection service.
2. The source domain administrator writes code to parse the redirected error code that the destination sends.

It is up to the source domain administrator to decide how to present the error message to the source domain users.

One method of receiving the information and presenting it to users is to use a static HTML page. COREid Federation uses a servlet and an HTML page to receive redirected error information and present it to users.

3. The destination obtains the following information from the source domain:
 - Their error redirection URL configured in the COREid Federation Administration Console. This is the URL of the code that receives the redirected error information.
 - For COREid Federation installations, COREid Federation provides an `ObsSAMLError` servlet. Error information is redirected to this servlet

unless the administrator replaces it with a custom servlet. For details, see “The ObSAMLError Servlet and Template” on page 211.

- For other SAML installations, this is the URL of their error redirection process—that is, the servlet or other code that receives the error information directed to it for further processing.

Error Redirection for Source Domains

The COREid Federation error reporting feature enables you to receive error information from destination domains when an error pertaining to your domain occurs at the destination.

COREid Federation provides an error reporting servlet called ObSAMLError and a template HTML form called ObSAMLError.html. You can customize this form to report and present the error to your user.

Task overview: Error redirection for source domains

1. Contact the destination domain administrator to coordinate error redirection information.
2. Communicate the Error URL for your servlet.

This is the URL to which the destination domain will redirect error information. This is the Error URL field in the MyDomain page of the COREid Federation Administration Console.

3. Write code to parse the error information.

Note: Non-COREid Federation source domains will need to implement a custom error redirection service.

The ObSAMLError Servlet and Template

The ObSAMLError error reporting servlet receives the redirected error information from the destination site, parses it, and puts the information into the ObSAMLError.html template form to be presented to your user. The servlet returns the HTML error page to the user. By default, the ObSAMLError.html form resides in

SHAREid_Install_Dir/oblix/lib/ObSAMLError.html

where *SHAREid_Install_Dir* is where you installed COREid Federation.

Here is the HTML form template:

```
<html>
<head><title>Web Intersite Signon Error</title></head>
<body>
<h1>Web Intersite Signon Error</h1>
<p>%ErrMsg%</p>
<p>To resolve this problem, call Tech Support, ext.555</p>
</body>
</html>
```

Note that the example error message text is a placeholder that you would want to replace with the phone number of a local technical support number.

When the ObSAMLError servlet receives the redirected error information, it replaces the %ErrMsg% placeholder with the textual explanation of the error condition sent in the redirection.

You can customize this form to tailor the information that is presented to your users.

Configuring Error Redirection as the Destination

As the destination domain administrator, you configure your domain to recognize the error URL for each source domain. When an error pertaining to that domain occurs, your COREid Federation COREid Federation server redirects the error information to that URL. The source domain's servlet or process can then report the error to its users.

To configure error redirection as the destination domain

1. From the COREid Federation Administration Console, click
Domains > Add Other Domain > COREid Federation or Non-COREid Federation
or
Domains > View All Domains > select the domain link
2. Enter the source domain's error redirection URL in the Error URL of the domain configuration page.

To disable error redirection as the destination domain

1. From the COREid Federation Administration Console, click
Domains > Add Other Domain > COREid Federation or Non-COREid Federation
or
Domains > View All Domains > select the domain link

2. Delete the redirection error URL from the Error URL field.

Error Redirection Scenario

The following example assumes that the source—Domain A—and the destination—Domain B—are COREid Federation installations that use the Artifact profile.

Source (Domain A)

1. A user from Domain A logs in to Domain A.
An ObSSOCookie is set for the user.
2. The user requests a resource from a Web server, target.b.com, in Domain B.
3. Domain A's Transfer Service gets the destination domain name (DOMAIN=Domain B) and Web server (TARGET=https://target.b.com/resources/mine.html) for the resource from the URL for the request.
4. Using the ObSSOCookie created for the user's session, the source Transfer Service creates an assertion and sends an artifact to Domain B.
5. The source's Transfer Service redirects the user to target.b.com in Domain B (saml.b.com).

```
https://saml.b.com/  
ObSAMLReceiverService&SAMLart=artifact&TARGET=https://  
target.b.com/resources/mine.html
```

Destination Site (Domain B)

6. Domain B's Receiver matches the sourceID in the artifact to Domain A.
7. Domain B's Receiver cannot find the ResponderURL for Domain A.
8. Domain B's Receiver finds the error URL for Domain A (https://saml.a.com/ObSAMLError) and redirects the error information to Domain A's error URL.

```
https://saml.a.com/ObSAMLError?ErrID=RVE003&ErrP1=artifact&  
ErrMsg=CONFIG%20ERROR:%20No%20configured%20SAML%20Respo  
nder%20URL%20for%20the%20source%20of%20the%20artifact%20  
artifact
```

Source Site (Domain A)

9. The ObSAMLError servlet on Domain A receives the error information from Domain B and unpacks the following:
 - The error ID
ErrID=RVE003
 - Error message parameter

ErrP1=*artifact*

- The error message:

ErrMsg=CONFIG%20ERROR:%20No%20configured%20SAML%20Responder%20URL%20for%20the%20source%20of%20the%20artifact%20artifact

No configured SAML Responder URL for the source of the artifact

10. The ObSAMLError servlet on the Domain A substitutes the actual error message for the dummy one in ObSAMLError.html, the error presentation form, and returns the error page to the user:

Web Intersite Signon Error: No configured SAML Responder URL for the source of the artifact.

Options for Error Message Customization

There are three options for customizing the error messages returned to the ObSAMLError servlet:

- Modify ObSAMLError.html so that you provide local contact information and branding, for example, “An error has occurred with inter-domain login. Call extension 12345 for information.”
- Customize the source domain’s shareid-messages.properties file to report different messages, for example, in French or German. ObSAMLError inserts the customized error messages (instead of the ErrMsg parameter value) into the ObSAMLError.html page.

This file is located in:

SHAREid_Install_Dir/oblix/webapps/shareid/WEB-INF/classes

where *SHAREid_Install_Dir* is the directory where SHAREid is installed.

- Replace the ObSAMLError servlet with another program that can manage the error information.

Setting up COREid Federation Load Balancing and Failover

In environments where you need to increase throughput, or you have specific requirements to provide continuous site operation, you can use one of any number of commercial solutions available to provide COREid Federation load balancing and failover. Typical load balance scenarios can be set up to distribute requests to multiple or “clustered” COREid Federation servers based on specific load distribution algorithms and remove configured COREid Federation Servers from service if particular server machines fail at your site.

To create the different COREid Federation server “instances” that you want to load balance, you need to first install COREid Federation on each server machine to which you want to distribute requests. You would then typically choose one machine on which you would set up a source or destination domain master configuration. Following that you would copy the “master” COREid Federation domain configuration to each of your other load balanced machines to replicate the same configuration on each server.

The following procedure describes specific steps to setting up COREid Federation Servers for use in a load balancing/failover configuration:

1. Choose a “master” server machine for your site and install COREid Federation.
2. Import the certificate issued for your load balancer’s host name, e.g., shareidload.oblix.com) into the “master” COREid Federation Server machine’s keystore. For more information, see Chapter 7. “Configuring Keys and Certificates” on page 243.

Note: Importing the load balancer’s certificates here will pass SSL sessions back to the individual COREid Federation servers in your network. For other SSL session handling and certificate installation options, refer to the your load balancer vendor’s documentation.

3. Set up the COREid Federation Server configuration for operation as a source domain, destination domain, or both. Load balancing COREid Federation also requires that you set up a common data store so that multiple COREid Federation Servers can store, query, and update assertions from the same source. (See “Setting up a Common Assertion Store Database” on page 216 for more information.)
4. Install COREid Federation on the other server machines in your network that you want to run COREid Federation. COREid Federation should be installed

in the same drive and directory path location as your “master” COREid Federation Server machine.

Note: You install COREid Federation to copy and properly register the underlying COREid Federation software on each machine in your network, but you do not need to perform COREid Federation Server configuration (Step 3) since configuration files will be copied from your master COREid Federation Server in the next step. This procedure assumes that all the server machines that you want to include in your load-balanced COREid Federation configuration provide the same hardware and operating system software platform.

5. Make a copy of the entire COREid Federation installation directory tree (including all subdirectories and files) on your master COREid Federation Server machine and use these files to copy (overwrite) the COREid Federation installation directory on each of the other COREid Federation Server machines in your network.
6. Stop and restart all COREid Federation Servers in your network.
7. Perform any additional configuration steps required by your load balancer hardware or software.

Note: If you make subsequent changes to your master COREid Federation Server configuration, you’ll need to repeat steps 5 and 6 to replicate the configuration changes across all the other COREid Federation Servers in your network.

Setting up a Common Assertion Store Database

To support COREid Federation load balancing and failover, you need to set up common shared database where multiple COREid Federation servers can access the same set of assertions for either a source or destination domain. COREid Federation translates information from COREid Federation Java objects generated for assertions and store it in rows of a table that is created in a MySQL database.

The following procedure describes the specific installation and configuration steps necessary to set up an assertion store database to enable COREid Federation load balancing and failover:

1. Download MySQL, version 4.1, available from mysql.com, and install it on a computer that all the COREid Federation servers in your source or destination domain can access.
2. Start MySQL and create or select a database that you want to use to store assertions.

Note: Keep track of the username, database name and other installation information regarding your MySQL database as you'll need to specify database connection parameters in the COREid Federation Administration Console when setting up COREid Federation to use the database to store assertions. COREid Federation can typically run with a small to mid-size database. Assertion information stored in the database is transient, so the size and number of rows stored in the MySQL database at any one time depend on the overall transaction rate for assertion and resource requests and how long assertions live before they expire (and are deleted from the database).

3. Run the COREid Federation `create_tables.sql` script (by default, located in the directory:

`<InstallDir>Program Files\Oblix\SHAREid\oblix\sql-scripts\mysql`

This script creates two database tables, `oblix_shd_assertions` and `oblix_shd_audit_archive`, each having almost the identical table structure and fields to store assertion information:

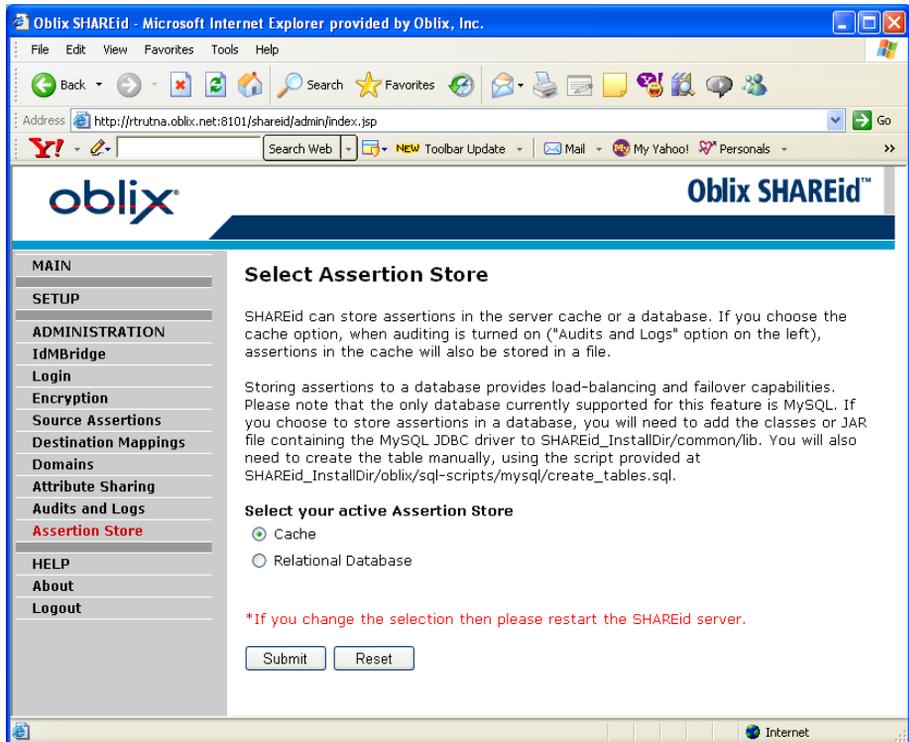
Col #	Column Name	Type	Allow NULLs
1	Assertionid	Varchar(255)	No
2	Artifact	Varchar(255)	Yes
3	Requesterid	Varchar(255)	Yes
4	Subject	Varchar(255)	Yes
5	Issuer	Varchar(255)	Yes
6	Assertion	Text Blob	No
7	Issuedon	Datetime	Yes
8	Expireson	Datetime	Yes

The first table, `oblix_shd_assertions`, stores active assertion information that only remains in the database as long as the information is needed by COREid Federation. The mapping of database table columns to COREid Federation Java assertion objects is specified in a separate file, `assertion.hbm.xml`.

The second table, `oblix_shd_audit_archive`, provides persistent storage of assertion information for auditing purposes. This table is populated when auditing is enabled from the Administration Console Audits and Logs configuration page and you have also selected the Relational Database option from the Assertion Store configuration page.

4. Start and log into the COREid Federation Administration Console.
5. From the COREid Federation Administration Console, select the Assertion Store option from the left-hand navigation pane.

The Select Assertion Store Configuration page appears.



6. Select the Relational Database option and click Submit.

Note: By default, assertion and artifact information exists only in cache. You can also choose to store the cached information to a file, by choosing the Enable Auditing option from the Administration Console's Audit and Log page. For more information, see Chapter 6. "COREid Federation Auditing and Logging" on page 277.

The Database Connection Configuration for storing Assertions page appears.

The screenshot displays the 'Database Connection Configuration for Storing Assertions' page in the Oblix SHAREid™ Administration Console. The page has a sidebar menu on the left with options like MAIN, SETUP, ADMINISTRATION, and HELP. The 'Assertion Store' option is highlighted in red. The main content area contains a form with the following fields and values:

- Data Source URL: (e.g., jdbc:mysql://acme.com:3306)
- JDBC Driver Class: (e.g., org.gjt.mm.mysql.Driver)
- User Name:
- Password:
- Max. Active Connections: (value: 1)
- Max. Idle Connections: (value: 1)
- Max. Wait for Connections: (value: -1)

At the bottom of the form are 'Submit' and 'Reset' buttons. The browser's address bar shows 'Internet'.

7. Specify the parameters to use to establish a connection to the MySQL database that will store assertions for either a source or destination domain. Also, specify values for Maximum Active Connections, Maximum Idle

Connections, and Maximum Wait for Connections parameters to control database pooling active and idle connections and wait times.

The following table provides a description of database connection parameters required for the Assertion Store database configuration. (These are the same options as those you specify for an RDBMS IdMBridge database connection.)

Field	Description
Data Source URL	Vendor-specific URL to be passed to the JDBC driver to establish a connection to IdMBridge database, for example: <code>jdbc:mysql://yttrium.com:3306/TEST</code> The general syntax for this parameter is: <code>jdbc:mysql://hostname:port/dbinstance</code> where hostname is the TCP/IP address or TCP/IP host name of the server to which you are connecting, port is the number of the TCP/IP port, and dbinstance is the name of the Database instance used.
JDBC Driver Class	The fully qualified JDBC driver class name used, for example: <code>org.gjt.mm.mysql.Driver</code>
User Name	User name to be passed to the JDBC driver to establish a connection.
Password	Password to be passed to the JDBC driver to establish a connection.
Maximum Active Connections	The maximum number of active connections that can be allocated from the connection's database pool at the same time, or zero (0) for no limit. The default is 1.
Maximum Idle Connections	The maximum number of active connections that can remain idle in the pool, without extra ones being released, or zero for no limit. The default is 1.
Maximum Wait for Connections	The maximum number of milliseconds that the pool will wait (when there are no available connections) for a connection to be returned before throwing an exception, or -1 to wait indefinitely. The default is -1.

8. When you have finished making your entries, click Submit.

You can now exit the Administration Console.

9. Restart the COREid Federation Server to have the Assertion store configuration changes take effect.

Configuring COREid Federation to Use X.509 Attribute Sharing Profiles

COREid Federation also supports single sign-on using a variation of the SAML attribute query/response protocol (sstc-saml-x509-authn-attribute-protocol-profile) in which retrieval of user attributes is authenticated using an X.509 certificate. This section describes the overall process and steps of configuring source domains and destination domains to implement single sign-on using this profile. (See “Single Sign-On Process Using the X.509 Attribute Sharing Profile” on page 38.)

Identity Provider (Source or Home) Domain Configuration

To configure the COREid Federation Server for the Identity Provider (Source or Home) domain, you need to perform the following steps:

1. Set up the Subject Mapping to map the certificate subject to a local user using the COREid Federation Administration Console’s Destination Mappings menu option.
2. Go to the MyDomain Configuration page (using the COREid Federation Administration Console’s Domains > MyDomain menu option) and select the Subject Mapping defined in step 1.
3. Set up an Assertion profile to use attribute sharing (Administration Console’s Source Assertions menu option).
4. Select the profile defined in Step 3 as the assertion profile to use for the destination domain (selected from the COREid Federation Administration Console’s Domains > View All Domains menu option).

The following sections provide more detailed instructions for performing each of these steps.

Step 1: Set up the Subject Mapping

1. From the COREid Federation Administration Console, select the Destination Mappings > Add Mappings option from the left-side navigation pane.

The Administration Console displays the Add Assertion to User Mapping page. From this page you can map the subject name specified in assertions or attribute queries to a local user defined in the identity provider (source or home) domain.

2. Enter a name for the mapping in the Mapping Name field and optionally provide a description to appear when you display a list of mappings using the Destination Mappings > View All option.
3. Optionally change the settings for the Authentication Scheme Level (used by the COREid Access System).

4. Specify the top directory node (and associated options) used in searches to match users.
5. In the Assertion Attributes section and the Mapping Attributes table, enter the pair of values by which you want to map Assertion subjects to local users.

Typically you would map a CN component of the assertion SubjectName (value of the Name Identifier in the Subject element) to the cn user attribute in your local user profile store although other mappings are possible depending on the format of the certificate SubjectDN and the available user profile attributes.

In the Mapping Attributes section, if you need to map the entire SubjectName string to a user attribute, you just enter “SubjectName” in the Assertion Attribute field. An example of an entire SubjectName string is “cn=John Smith,ou=customer, ou=Division,o=Company,c=US”.

The following screen display provides an example:

oblix **Oblix SHAREid™**

MAIN
SETUP
ADMINISTRATION
 IdMBridge
 Login
 Encryption
 Source Assertions
 Destination Mappings
 View All
 Add Mappings
 Domains
 Attribute Sharing
 Audits and Logs
 Assertion Store
HELP
 About
 Logout

Add Assertion-to-User Mappings

When an assertion has been validated, the assertion needs to be mapped to a local identity. This mapping uses an attribute in the received assertion to determine who the local user is.

Mapping Name:

Description:

Authentication Scheme Level
 Level Used by the COREid Access System:

Top Directory Node
 Node Used in Searches for Matching Users (the search base):
 (Example: dc=mycompany,dc=mygroup,dc=com)
 Person Object Class Used by the COREid Access System:

Assertion Attributes
 The following mappings allow assertions to be matched to local users.

Use SubjectName as Assertion Attribute if "User Attribute for Subject" (or the equivalent) is configured in the corresponding assertion profile at the source site. If the SubjectName is an X.509 DN, you can use components of the DN as SubjectName.COMPONENT, e.g., SubjectName.CN or SubjectName.OU.

If using the SmartWalls feature you may need to also enter Domain as the Assertion Attribute. See documentation for more details.

Mapping Attributes	
Assertion Attribute	User LDAP Attribute
<input type="text" value="SubjectName.CN"/>	<input type="text" value="cn"/>
<input type="button" value="Add New Row"/>	

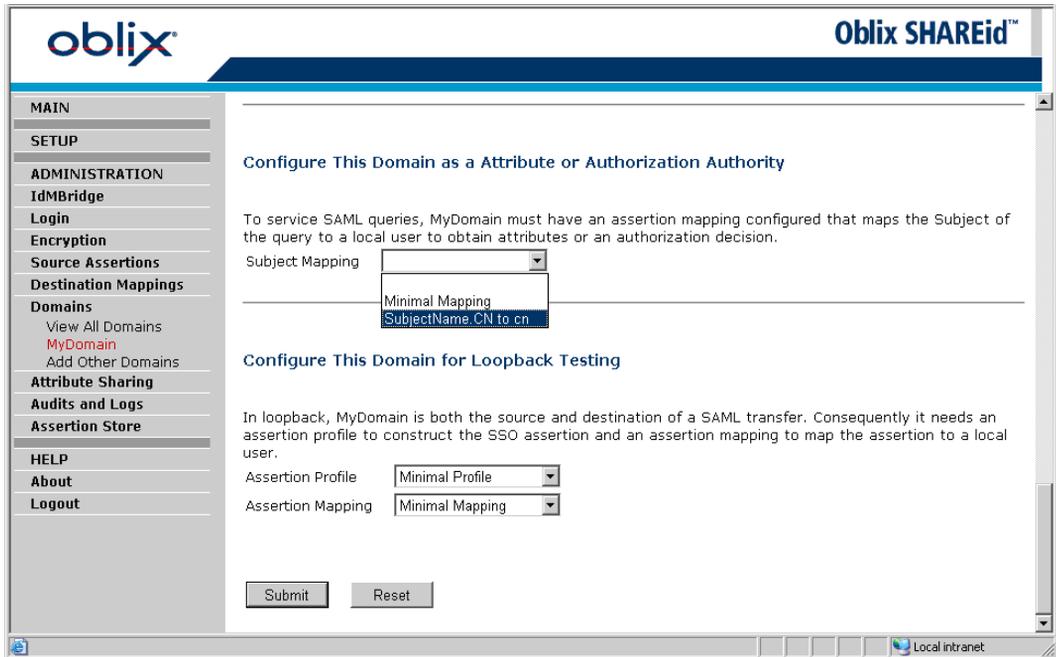
Done Local intranet

6. You can specify additional mapping attributes by clicking the Add New Row button.
7. When you're finished adding mapping attribute entries, click the Submit button.

Step 2: Select the Subject Mapping for MyDomain

1. From the COREid Federation Administration Console, select the Domains > MyDomain menu option from the left-side navigation pane.

The Administration Console displays the Modify MyDomain page.



From this page you can configure this domain as an attribute authority. In the subject mapping field, you specify the mapping used to correlate incoming attribute queries to a local user.

Step 3: Set up an Assertion Profile

In this step, you set up an Assertion Profile to define the assertion attributes sent in response to the Attribute Query and how the assertion will be generated.

1. From the COREid Federation Administration Console, select the Source Assertions > Add Assertion menu option.

The Administration Console displays the Add Assertion Profile page.

oblix **Oblix SHAREid™**

MAIN
SETUP
ADMINISTRATION
 IdMBridge
 Login
 Encryption
Source Assertions
 View All
 Add Assertion
 Destination Mappings
 Domains
 Attribute Sharing
 Audits and Logs
 Assertion Store
HELP
 About
 Logout

Add Assertion Profile

When users request resources, the source site provides the destination site with an assertion that attests to the user's identity. An assertion profile defines the contents of the assertion that is sent to the destination.

Assertion Profile Name:

Description:

Issuer: (e.g. http://host.company.com)

Subject Name Qualifier:

Subject Format:

User Attribute for Subject:

Add Assertion Attributes to Your Assertion Profile

In the following fields, you configure attribute mapping in the assertion profile. To configure an attribute mapping, you create a one-to-one correspondence (a mapping) between assertion attributes that a destination domain administrator gives to you and user attributes in your domain's data store. The destination domain's administrator must tell you the names of the assertion attributes.

Assertion Attribute	Attribute in Data Store	Name Space	Optional Type	In SSO Assertions	Allowed Values
<input type="text" value="Role"/>	<input type="text" value="title"/>	<input type="text" value="http://source.comp"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="Accountant"/> <input type="text" value="Manager"/>

Advanced Options

Assertion Signing

Include a certificate in the signature

Assertion Validity Period

seconds before the time assertion generated

seconds after the time assertion generated

Delimited Data

In addition to supporting and passing multi-valued attributes, SHAREid can also support delimited data to provide multiple values for assertion attributes. To use the delimited data option, select yes below and specify the character to be used.

Data is delimited Yes No

Delimiter character:

Internet

2. On this page, describe the assertion and specify mapping of attributes requested by a service provider or destination domain to those in your domain's local data store.

In this example, the value of the "title" attribute in your local data store is used to populate the "Role" attribute values in the assertion. The Allowed Values entries specify that only the values of Accountant and Manager may be included in the assertion for an Attribute statement.

3. When you have finished your entries, click Submit.

Note: If multi-valued attributes in data store are delimited, you can specify, in the Delimiter field, the delimiter that COREid Federation needs to use to properly retrieve and parse the values. For example, if a data stores multiple values of blue, green, and red as "blue,green,red", the comma (",") character is the delimiter. Specify the comma in this field as the delimiter character that COREid Federation needs to use to return each of the individual attribute values.

Delimiters are specified as part of each Assertion Profile, so it is possible for each profile to have a different delimiter. Delimiters are stored in the shareid-config.xml file.

Step 4: Select the Profile for the Destination Domain

In this step, you select the Assertion Profile to define the assertion attributes for the Attribute Query and determine how the assertion will be generated.

1. From the COREid Federation Administration Console, select the Domains > View All Domains menu option.
2. Click the link corresponding to the destination domain that is sending the attribute query.

COREid Federation displays a page showing the current configuration of the selected destination domain.

3. Scroll down to the Configure this Domain as a Destination section.

4. In the Assertion Profile field, select the assertion profile defined in Step 1 from the list of defined profiles:

oblix **Oblix SHAREid™**

Configure This Domain as a Destination.

Receiver URL

Indicate what assertion profile to use when sending assertions about users from your domain to this destination.

Assertion Profile

Artifact Profile

Requester Authentication: Select X.509 Certificate only if Responder URL of source domain uses Client Certificate authentication port.

Basic

Requester Id

Requester Password

Confirm Password

X.509 Certificate

Signing Certificate Subject DN

Local intranet

5. Click the Submit button.

Service Provider/Destination Configuration

At the destination domain, you need to perform the following configuration.

From the COREid Access System Console:

1. Define the authentication scheme for SSL client certificate authentication with the user's Web browser.
2. Add the authorization scheme.
3. Add a policy domain to protect the target site's resources and also do the following:
 - Add an authorization rule.
 - Set plug-in parameters for the authorization rule.
 - Create an authorization expression using the new authorization rule.

From the COREid Federation Administration Console:

1. Configure the Attribute Sharing Service so COREid can access COREid Federation to determine whether users from another domain have specified attribute values.
2. On the Attribute Sharing page, specify the subject name mapping to home domains for users. This mapping will identify which domain will be contacted as the authority to obtain attribute values for the requesting user.
3. Make sure that for each configured domain, the attribute namespace is correct.

This information is configured in the Configure This Domain as an Attribute or Authorization Authority section for each external domain (COREid Federation or non-COREid Federation) that might act as an attribute authority.

Define Authentication Scheme

Define a new X.509 Certificate authentication scheme in COREid.

1. From the COREid Access System Console, add a new authentication scheme by selecting the Access System Configuration > Authentication Management menu option and clicking the Add button.
2. Specify a name to identify the Authentication scheme, then specify a level and select the X.509Cert Challenge method and SSL Required options.

The following display shows the New Authentication Scheme page.

Oblix • NetPoint

System Configuration System Management Access System Configuration

Logged in user: Master Admin

General Plugins Steps Authentication Flow

Define a new authentication scheme

Name X.509 Authentication with SAML Attributes

Description

Level 5

Challenge Method None Basic X509Cert Form Ext

Challenge Parameter

SSL Required No Yes

Challenge Redirect

Enabled No Yes

Update Cache

Save Cancel

NetPoint BEA Ready Realm Configuration Internet

This scheme tells WebGate and the Web server to perform SSL client certificate authentication in communication with a user's Web browser.

Now, add COREid Federation plug-ins to the new authentication scheme you have defined in COREid.

3. From the COREid Access System Console, choose the Authentication Management option.
4. Click the Plug-Ins tab.

The Console displays a message: “No plug-ins have been defined for the authentication scheme. Please click on Modify button to add plug-ins.”

5. Click the Modify button.

The Console displays the Plug-Ins for Authentication Scheme page.

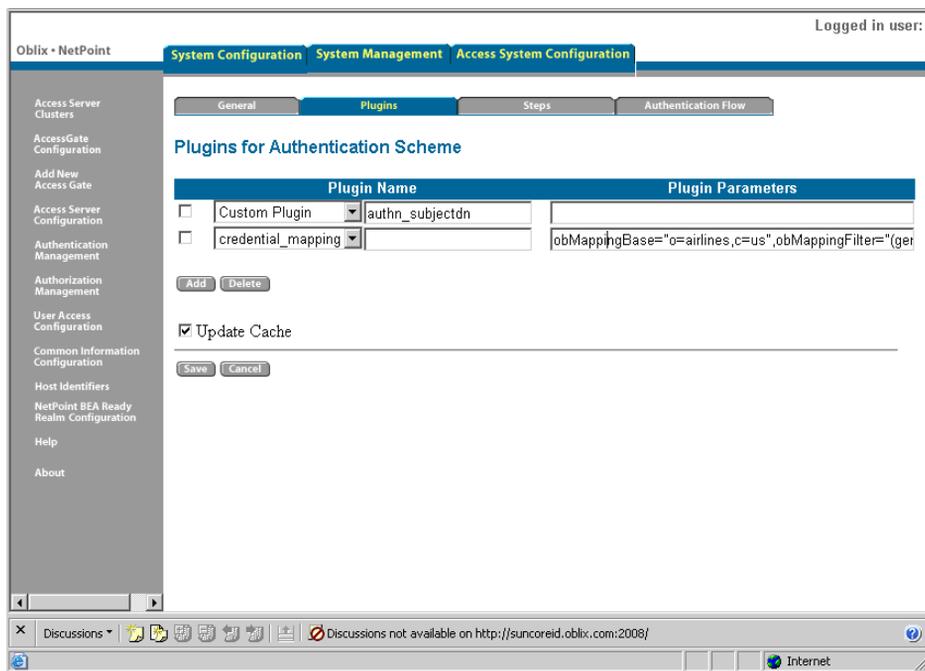
6. Select the Custom Plug-In option and specify the name of the COREid Federation attribute authorization plug-in: “authn_subjectdn”.

The authn_subjectdn extracts the SubjectDN from the user's certificate and sets the SubjectDN header for the authorization plug-in.

Select the credential_mapping plug-in and specify plug-in parameters to set a dummy authenticated user for the Access System.

Note: You can obtain the plug-in parameters to use here by copying the same parameter string used in the NetPoint None Authentication Scheme.

The following display shows how the page will appear after you have added and configured the two plug-ins:



Add COREid authorization scheme

Define a new authorization scheme in COREid.

1. From the COREid Access System Console, choose the Access System Configuration > Authorization Management option.
2. Click the Add button.
3. Specify a name for the new Attribute sharing authorization scheme, for example, SAML Attribute Scheme.

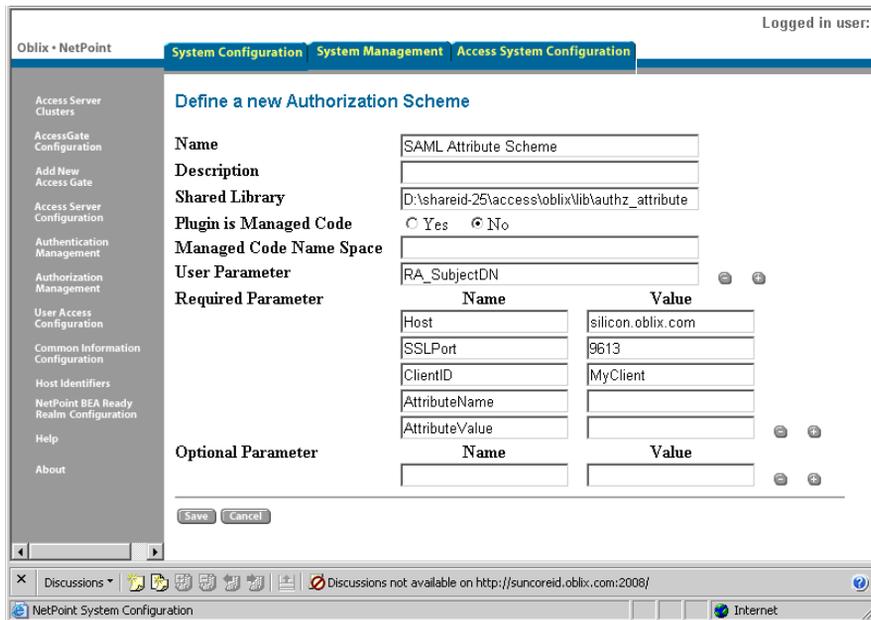
In addition, specify the following parameters:

- **Shared Library**—full path to the authorization plug-in installed with the COREid Access Server, for example:
`c:\access\oblix\lib\authz_attribute`
- **Plug-In is Managed Code**—keep the default selection, No.
- **User Parameter**—enter the parameter **RA_SubjectDN** required to tell the Access Server that this is a Reverse Action to be retrieved from the WebGate SubjectDN header.

- **Required Parameters**—enter the names and values as described in the following table:

Name Field	Corresponding Value field
Host	Fully-qualified hostname you configured when you installed COREid Federation Server.
SSLPort	SSL Port you specified when you installed COREid Federation Server.
ClientID	Corresponding ClientID value identifies this plug-in to the COREid Federation Attribute Sharing Service (configured on COREid Federation Administration Console's Attribute Sharing page).
AttributeName	Leave blank for now (to be filled in by authorization rules that use this scheme; tells the Attribute Sharing Service which attribute to check for the subject).
AttributeValue	Leave blank for now (to be filled in by authorization rules that use this scheme; tells the Attribute Sharing attribute value to check for the subject).

The following display shows an example of the completed Authorization Scheme:



4. When you have finished, click Save.

Add a Policy Domain

Create a new Policy Domain in the COREid Access Manager to protect resources on the destination domain.

1. From the COREid Access Manager, click Create Policy Domain in the left side navigation bar.

Access Manager displays a Create Policy Domain page.

2. In the Name field, enter a short alphanumeric string identifying the domain.

Optionally, in the Description field, type a brief description of this policy domain.

The following display shows an example of the new Policy Domain page.

The screenshot shows the 'Create Policy Domain' page in the COREid Access Manager. The page is titled 'Create Policy Domain' and is part of the 'Access Manager' interface. The user is logged in as 'Master Admin'. The page has a navigation menu on the left with options like 'Search', 'My Policy Domains', 'Create Policy Domain', 'Access Tester', 'Help', and 'About'. The main content area has tabs for 'General', 'Resources', 'Authorization Rules', 'Default Rules', and 'Policies'. The 'General' tab is active, showing a 'Name' field with the text 'SHAREid Attribute Sharing Policy Domain' and a 'Description' field. There are 'Save' and 'Cancel' buttons at the bottom of the form. The browser window title is 'Save this policy domain.' and the address bar shows 'http://suncoreid.oblix.com:2008/'.

3. Save the new policy domain.

Set up Protected Resources

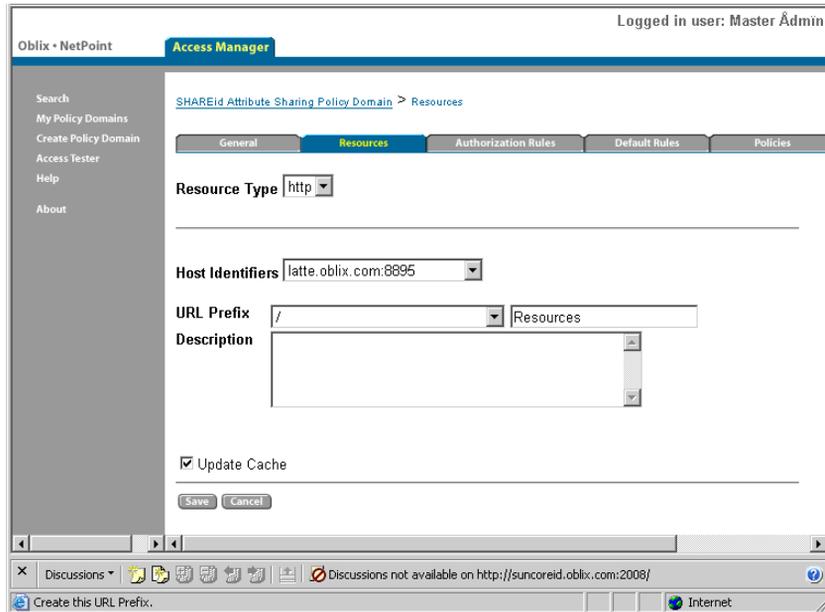
Specify resources to protect.

1. From the COREid Access Manager, click the Resources tab.

Access Manager displays a page in which you can specify resources you want to protect with a policy.

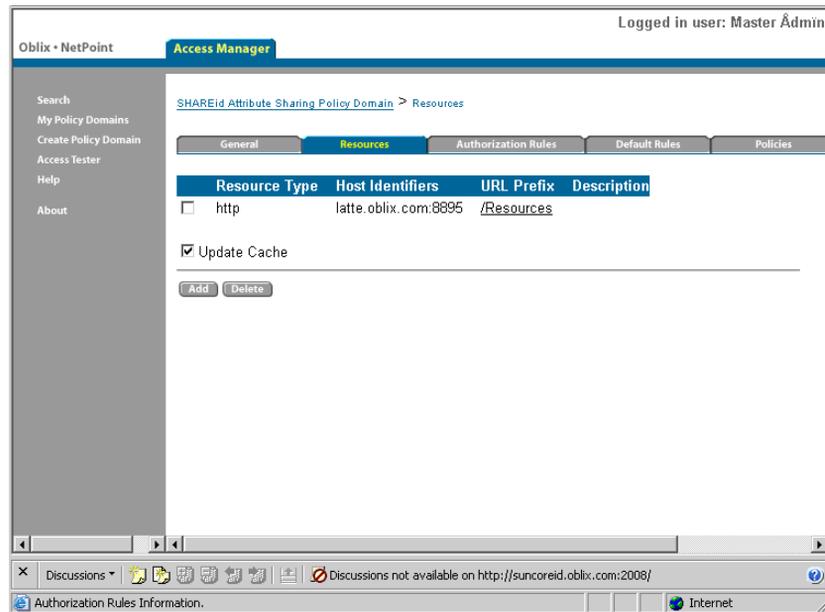
- Specify the resource type, host, URL, and resource description information.

The following display shows an example of a completed Resource specification:



- Save the resource specification.

The Access Manager now lists your new resource:

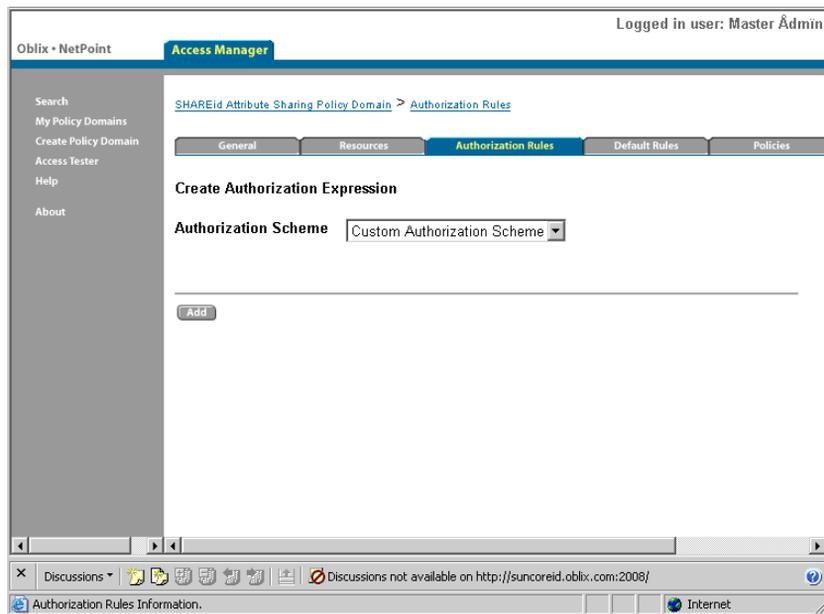


Add an Authorization Rule

Add an authorization rule (using the authorization scheme for a particular attribute name and value) to the policy domain you just created. Note that you may define different authorization rules using the same authorization scheme, but for different attributes and values.

1. From the COREid Access Manager, select My Policy Domains.
2. Select the policy domain you just created for protecting a resource.
3. Click the Authorization Rules tab.
4. Click Add.

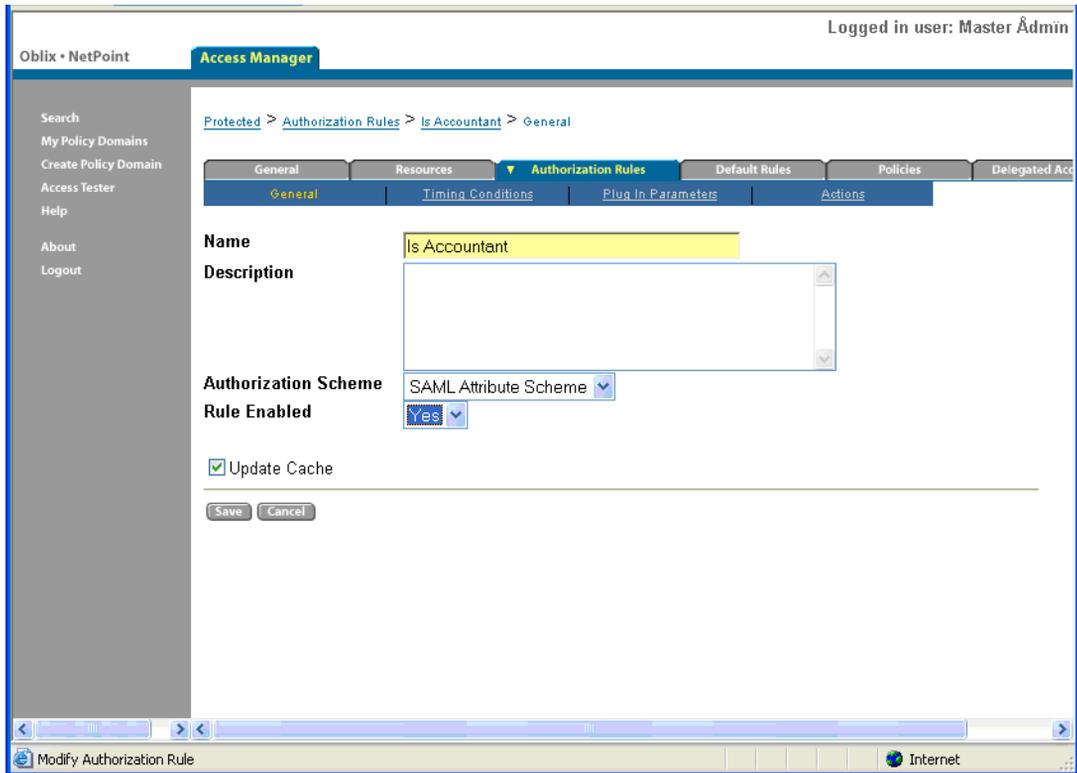
The Access Manager displays the following page.



5. From the Select Authorization Scheme field, choose “Custom Authorization Scheme.”
6. Click Add.
The Access Manager now displays a page asking you to identify the rule and specify an authorization scheme to use.
7. Specify a name and optionally a description to identify the authorization rule.
8. In the Authorization Scheme field, select the X.509 authentication scheme you defined to provide client certificate authorization (from the previous example, SAML Attribute Scheme).

9. Select Yes in the Rule Enabled field.

The following display shows an example of the completed Authorization Rule page.



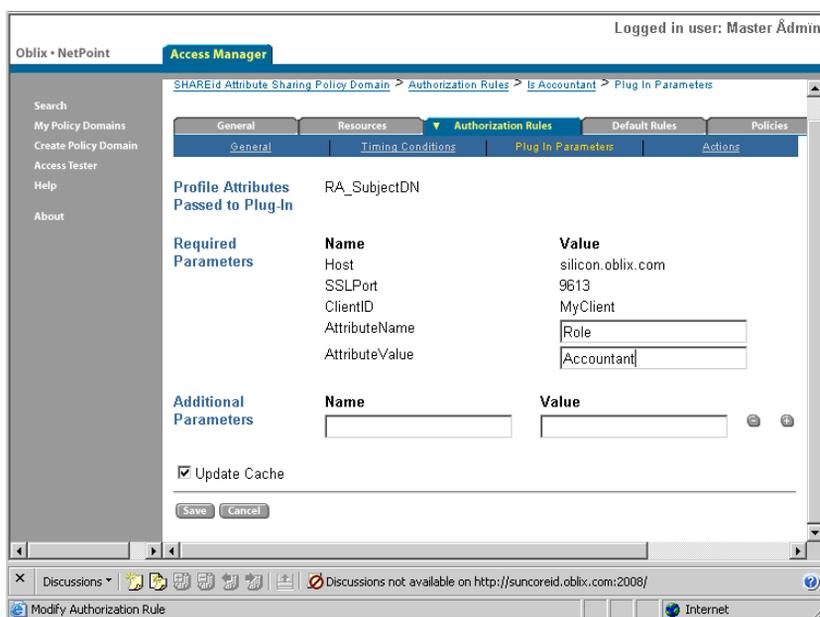
10. Click Save.

Set the plug-in parameters for the authorization rule

From the Authorization Rules tab of the Policy Domain display, set the plug-in parameters for the authorization rule:

1. Click the Plug-In Parameters tab from the Policy Domain page display.
2. Click Modify.

From this page you can specify the attribute name and value you want to use to authorize a user:



In this example, the attribute name is set to Role and the attribute value is set to Accountant. This means the Destination domain will only authorize users for which the user's attribute authority (source domain) returns an attribute value of Accountant.

Add the Authorization Rule to an Authorization Expression

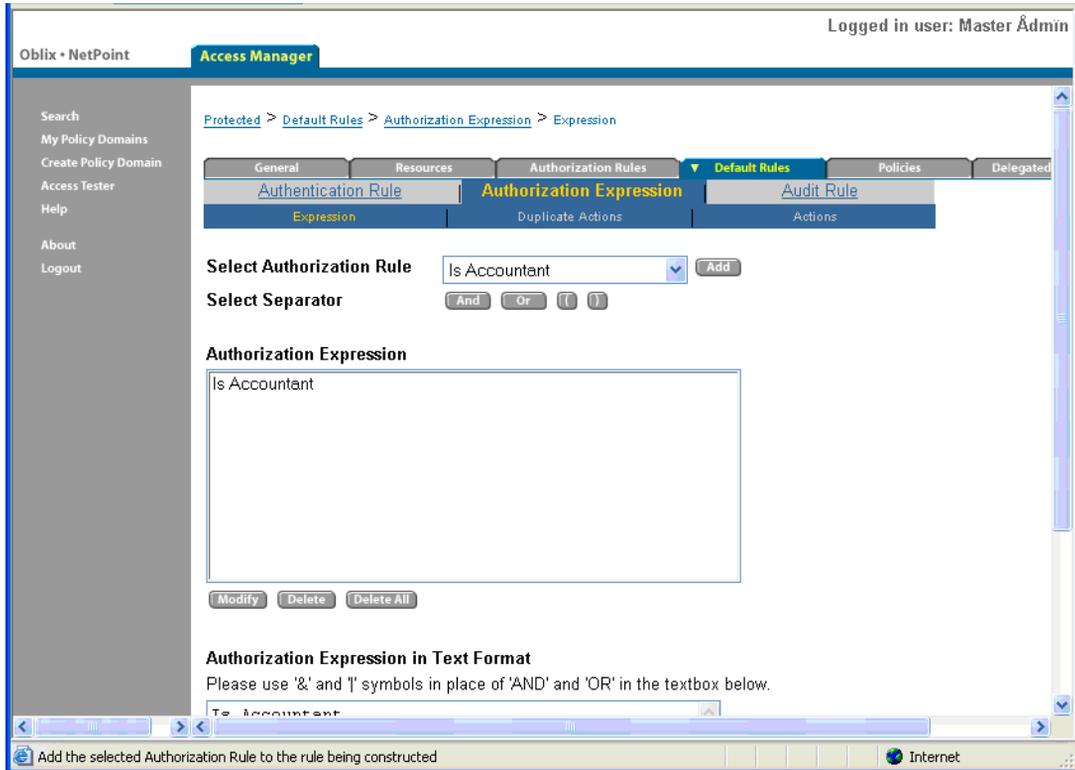
From the Policy Domain display, add the authorization rule to an authorization expression.

1. Click the Default Rules tab.
2. Click the Authorization Expression link.
3. Click Add.

The Access Manager displays a page in which you can select authorization rules you have defined and add them to an authorization expression.

4. Select an authorization rule in which you've defined an attribute name and value to check authorization of users with the X.509 attribute sharing profile.

For example, you can choose the "Is Accountant" rule to authorize users for which the user's attribute authority returns a value of "Accountant" for the attribute name "Role" as shown below.



5. When you have finished adding authorization rules to the expression, click Save.

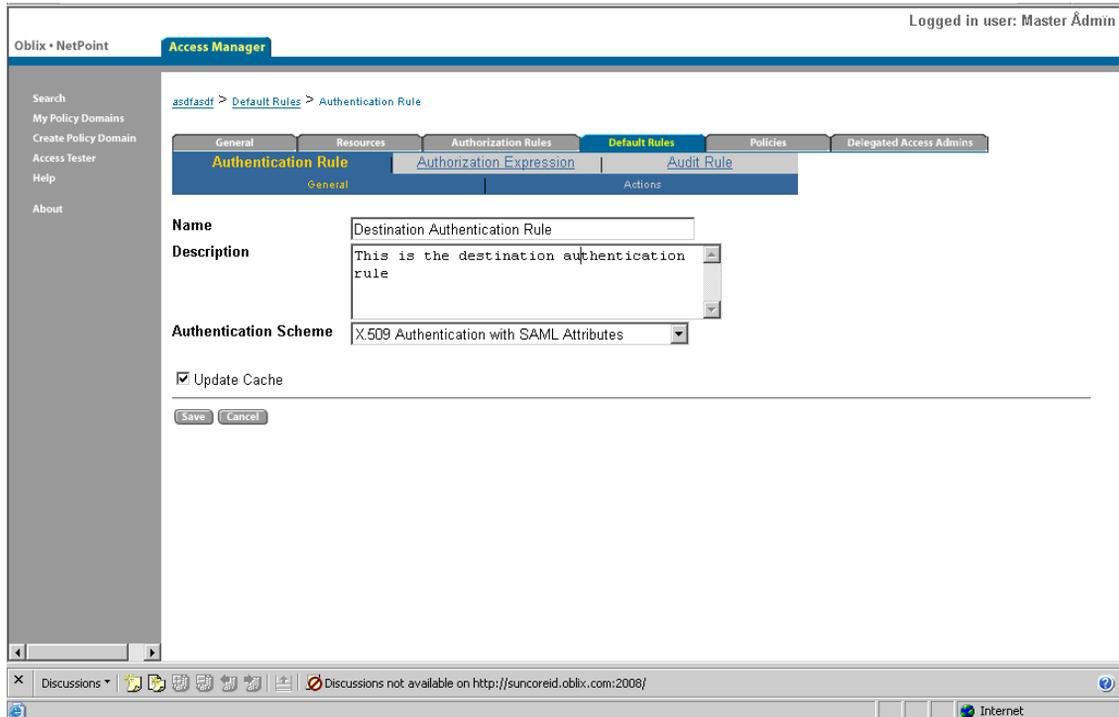
6. Now, click the Default Rules tab.

7. Click the Authentication Rule link.

The Access Manager displays the Default Rules > Authentication Rule page along with a message indicating that no rules have been defined yet.

8. Click the Add button.

The Access Manager now displays a page in which you can create a new authentication rule.



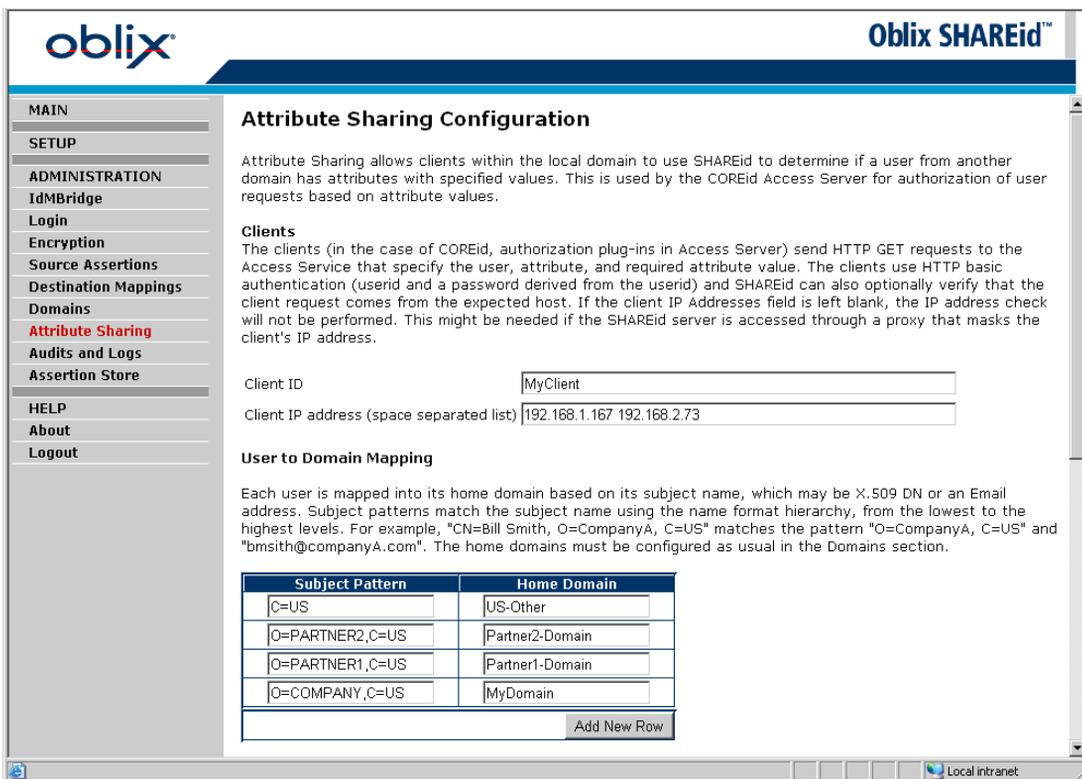
9. Specify a name and optional description to describe the authentication rule.
10. In the Authentication Scheme field, select the X.509 authentication scheme you defined earlier to provide client certificate authorization.
11. Click Save.
12. Exit the Access Manager and go to the Access System Console.
13. Navigate to the WebGate Configuration page for the WebGate protecting the resource to be accessed by users with the attribute sharing profile.
14. Configure the WebGate for the attribute sharing scheme to have a primary HTTP cookie domain that is a substring of the WebGate hostname.
For example, if the hostname is venus.oblix.com, the cookie domain you would enter is venus.
15. Save the WebGate configuration and now go back to the COREid Federation Administration Console and complete the configuration of Attribute Sharing at the COREid Federation destination.

Configure Attribute Sharing

Configure Attribute Sharing on the service provider or destination domain.

1. From the COREid Federation Administration Console, select the Attribute Sharing option.

The Administration Console displays the Attribute Sharing Configuration page in which you can specify how COREid Federation authenticates its clients and where it sends its attribute queries:



The screenshot shows the Oblix SHAREid Administration Console. The left sidebar contains a navigation menu with the following items: MAIN, SETUP, ADMINISTRATION, IdMBridge, Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing (highlighted in red), Audits and Logs, Assertion Store, HELP, About, and Logout. The main content area is titled "Attribute Sharing Configuration".

Attribute Sharing Configuration

Attribute Sharing allows clients within the local domain to use SHAREid to determine if a user from another domain has attributes with specified values. This is used by the COREid Access Server for authorization of user requests based on attribute values.

Clients

The clients (in the case of COREid, authorization plug-ins in Access Server) send HTTP GET requests to the Access Service that specify the user, attribute, and required attribute value. The clients use HTTP basic authentication (userid and a password derived from the userid) and SHAREid can also optionally verify that the client request comes from the expected host. If the client IP Addresses field is left blank, the IP address check will not be performed. This might be needed if the SHAREid server is accessed through a proxy that masks the client's IP address.

Client ID:

Client IP address (space separated list):

User to Domain Mapping

Each user is mapped into its home domain based on its subject name, which may be X.509 DN or an Email address. Subject patterns match the subject name using the name format hierarchy, from the lowest to the highest levels. For example, "CN=Bill Smith, O=CompanyA, C=US" matches the pattern "O=CompanyA, C=US" and "bmsith@companyA.com". The home domains must be configured as usual in the Domains section.

Subject Pattern	Home Domain
<input type="text" value="C=US"/>	<input type="text" value="US-Other"/>
<input type="text" value="O=PARTNER2,C=US"/>	<input type="text" value="Partner2-Domain"/>
<input type="text" value="O=PARTNER1,C=US"/>	<input type="text" value="Partner1-Domain"/>
<input type="text" value="O=COMPANY,C=US"/>	<input type="text" value="MyDomain"/>

2. Set the Client ID to match the ClientID authorization plug-in parameter.

Add the Subject Pattern to Domain mappings

Specify the mapping of users to their respective home domains based on Certificate subject patterns.

1. On the same Attribute Sharing page, scroll down to the User to Domain Mapping section.
2. In the subject pattern and Home Domain fields, add the Subject to Domain mappings for the client certificate subject patterns for which you want to check authorization from your site:

In the Subject pattern field, specify subject name patterns for users, for example, O=OBLIX, C=US. In the Home Domain fields, specify the corresponding domains that act as the attribute authority for matching subjects.

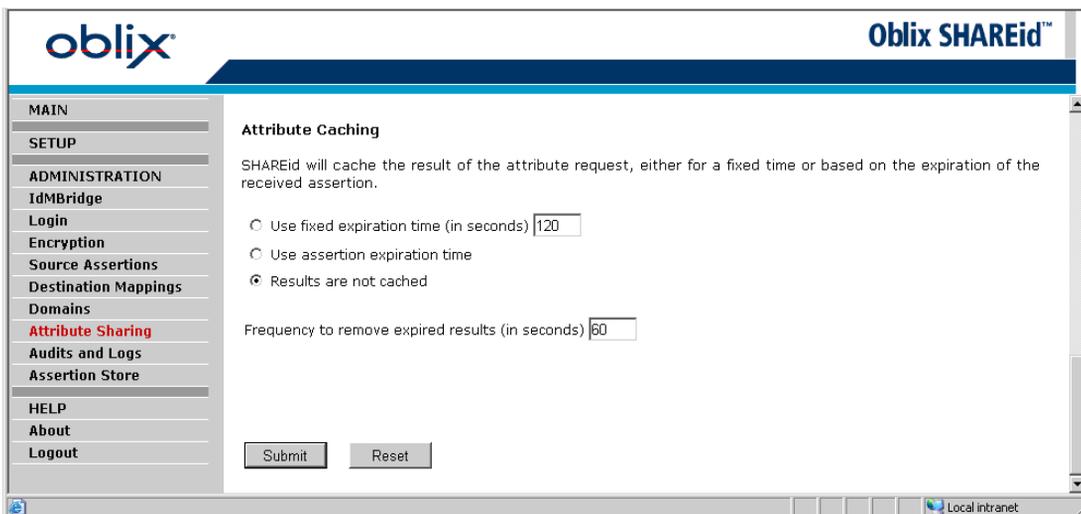
Note: COREid Federation searches for subject pattern matches from the lowest to highest levels of the name format hierarchy.

Set Attribute Caching

On the Attribute Sharing page, specify options for caching attribute query results.

1. On the same Attribute Sharing page, scroll down to the Attribute Caching section.

The Administration Console displays the following page to specify Attribute caching options.



The screenshot shows the Oblix Administration Console interface. The top header features the 'oblix' logo on the left and 'Oblix SHAREid™' on the right. A vertical navigation menu on the left lists various sections: MAIN, SETUP, ADMINISTRATION (with sub-items: IdMBridge, Login, Encryption, Source Assertions, Destination Mappings, Domains, Attribute Sharing, Audits and Logs, Assertion Store), HELP, About, and Logout. The 'Attribute Sharing' item is highlighted in red. The main content area is titled 'Attribute Caching' and contains the following text: 'SHAREid will cache the result of the attribute request, either for a fixed time or based on the expiration of the received assertion.' Below this text are three radio button options: 'Use fixed expiration time (in seconds)' with a text input field containing '120', 'Use assertion expiration time', and 'Results are not cached' (which is selected). A second text input field for 'Frequency to remove expired results (in seconds)' contains '60'. At the bottom of the form are 'Submit' and 'Reset' buttons. The browser's status bar at the bottom right shows 'Local intranet'.

Specifying attribute caching will allow users to access the same protected resource without requiring a new attribute query, while the attribute authorization result is cached.

After you've first set up your COREid Federation installation and are configuring and testing Identity and Service Provider settings, you may want to select "Results are not cached", so you will be able to immediately test the effects of changes in configuration. When you're ready to deploy single sign-on in a production environment, you can re-enable caching.

2. When you've finished click the Submit button to save your entries.

7 **Configuring Keys and Certificates**

Before deploying COREid Federation in production, you must configure secure connections between source and destination domains. A destination and a source establish a secure connection by exchanging certificates and communicating over secure ports. Once you have exchanged certificates, a trust relationship exists between source and destination that permits the destination to assume that an assertion sent by the source domain can be trusted, and by association, to assume that the contents of the assertion are authentic.

This chapter addresses the following topics:

- “About Configuring Secure Connections” on page 244
- “Configuring Certificates for the POST Profile on the COREid Federation Server” on page 254
- “Artifact Profile Using Basic Authentication on the COREid Federation Server” on page 260
- “Artifact Using Client Certificate Authentication on the COREid Federation Server” on page 264
- “Configuring Certificates for the Proxy” on page 270
- “Changing the Certificate Store Location and Passwords” on page 273
- “Configuring Certificates for the LDAP Data Source” on page 274

About Configuring Secure Connections

Configuring secure connections consists of the following:

- For the Artifact profile, you configure an authentication method in the COREid Federation administration console.

You can choose Basic authentication using SSL or client certificate authentication for the Artifact Profile.

- For both Artifact and POST, you establish trust between communicating domains.

Establishing trust consists of obtaining your own certificates, and installing certificates from and providing certificates to other domains.

Note: Restart the COREid Federation server after any changes that you make to your certificate store or CA certificates.

About the Certificate Stores

COREid Federation uses its own certificate store and a default self-signed certificate with the alias `shareid`. The alias is configured through the COREid Federation administration console. For production environments, you must replace the default self-signed certificate with a third party certificate from a vendor such as Verisign or Thawte.

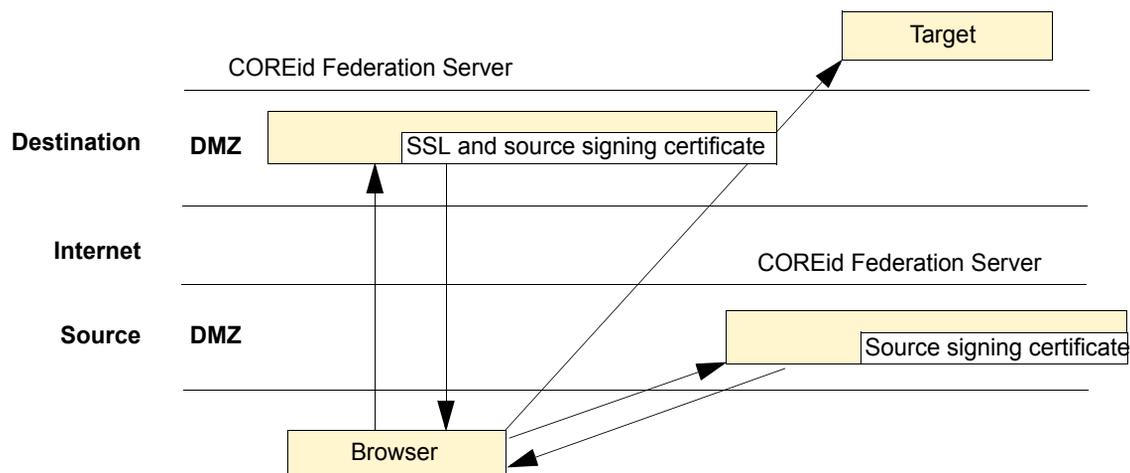
The Tomcat `server.xml` file points to the COREid Federation certificate store (or another acceptable authority) that is needed for secure connections among the various components of a COREid Federation installation. By default, COREid Federation and Tomcat use the same certificate store.

POST Profile Certificate Overview

For the POST profile, the source domain has a certificate and keys to protect the SAML services. The certificate is placed in the DMZ, either in the source's COREid Federation proxy or the COREid Federation server, depending on which resides in the DMZ. The source domain signs responses that contain assertions using a private key, and the destination receives the signed response and verifies the signature using the source domain's public key. For the destination to be able to use the source domain's public key, the destination must import the source's certificate into its COREid Federation certificate store. The destination must also have an SSL server certificate for the SSL handshake between the user's browser and the destination.

Figure 9 illustrates using the POST profile if no proxy is used.

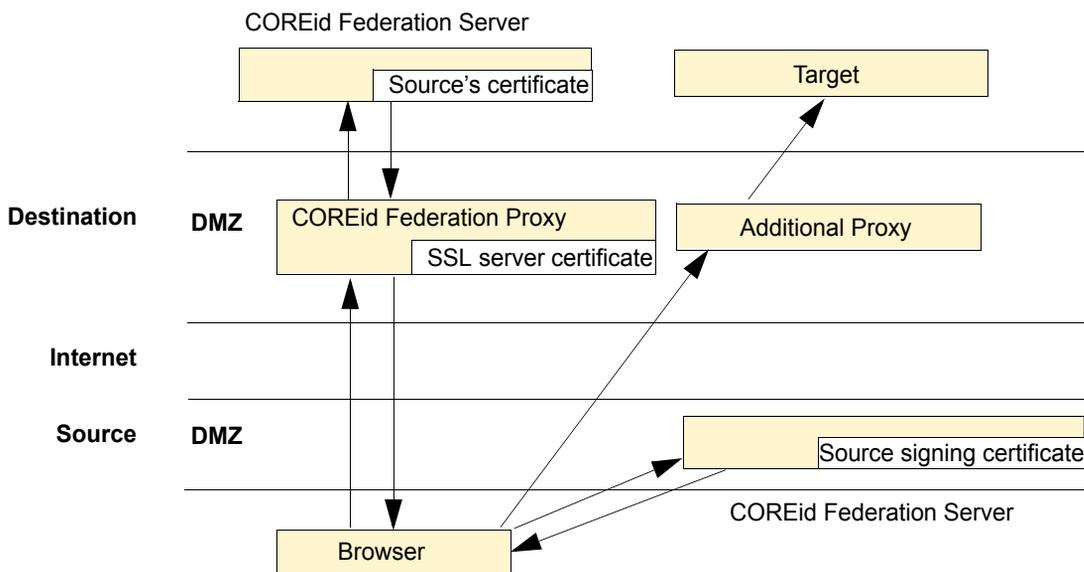
Figure 9 POST profile installation without a proxy server



If a proxy is used, the proxy must contain an SSL server certificate. The source's signature is verified by the destination's COREid Federation server behind the DMZ.

Figure 10 illustrates using the POST profile with a proxy on the destination.

Figure 10 POST profile installation using a proxy server



Artifact Profile Certificate Overview

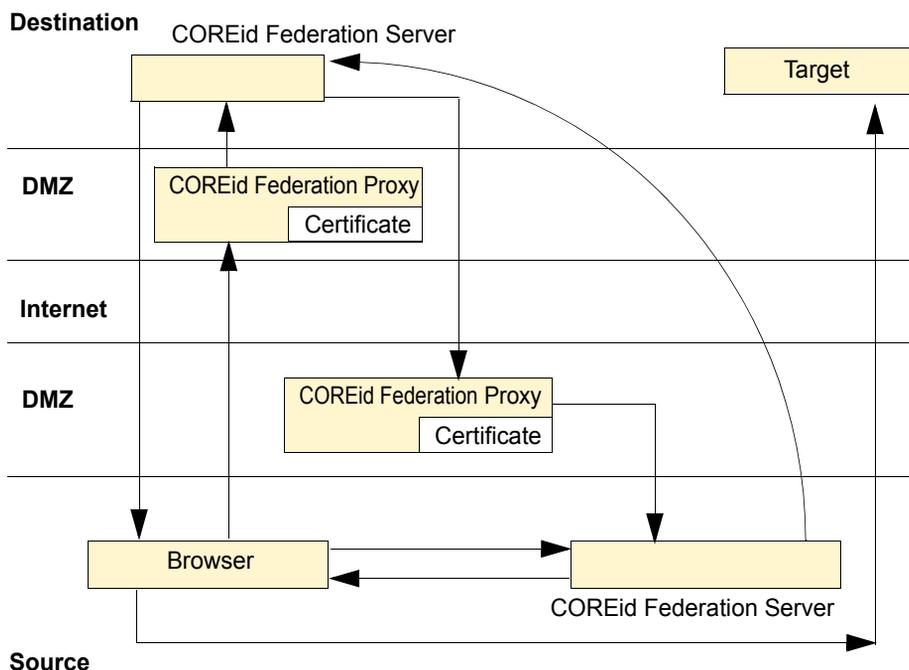
For domains using the Artifact profile with Basic authentication using SSL, the destination needs to import the source CA certificate. This type of trust is similar to that established for the POST profile.

If you configured client certificate authentication, two-way trust must be established. For two-way trust, the source has its own key pair and certificate, and it imports the destination's certificate into the COREid Federation certificate store. The destination domain imports the certificate for the source into the COREid Federation certificate store.

There are many possible permutations for certificate configuration using the Artifact profile, depending on whether Basic over SSL or Client Certificate authentication is configured, and on whether a proxy is used on the source, the destination, or both.

Figure 11 provides a generic illustration for certificate configuration using the Artifact profile.

Figure 11 Artifact profile installation using a proxy server



Default Versus Third-Party Certificates

For initial setup and testing of COREid Federation, you can configure secure connections using self-signed certificates that are provided with COREid Federation. For production, you replace these with certificates issued by a third-party vendor.

Your choice of certificates may affect the messages sent to users when they access resources on a destination. Communication between a browser and a source Transfer service is secured by server-side SSL on the Transfer service. For server-side SSL, you must have a key pair and a certificate in the COREid Federation server or in the proxy that processes the Transfer service URL. When a Web browser encounters a certificate it does not recognize, it issues a warning. This is normal HTTPS behavior. Each browser is configured to trust a set of CAs from specific vendors. If the user's browser connects to a Web server using SSL, and that Web server was not certified by one of these trusted CAs, the browser will display a security alert. For example, on Internet Explorer, the user has three options:

- To temporarily trust the CA for the single transaction
- To reject the transaction

- To import the CA's certificate and mark it as trusted, preventing future security alerts.

COREid Federation cannot prevent these security alerts if the browser does not trust the server's CA.

To ensure that your browser users never get these security alerts, use certificates from a commercial CA, or provide your users with instructions on how to import and trust the CA.

Locations for Certificates and the Keytool Command

COREid Federation uses the following directories for managing keys and certificates:

- The COREid Federation certificate store used for signing (and for SSL, if the default certificate store is used for both purposes):

SHAREid_Install_Dir/conf/keystore

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

- The keytool command that you use to manage keys and certificates:

SHAREid_Install_Dir/jdk/bin/keytool/keytool.exe

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

- The CA certificates used for ldaps that are trusted by COREid Federation's Tomcat servlet container are stored in:

SHAREid_Install_Dir/jdk/jre/lib/security/cacerts

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed. This directory is also used by the source domain for storing the destination's certificate when the two domains have implemented the Artifact profile with client certificate authentication.

Certificate Configuration Overview for Artifact and POST

The following tables summarize what certificates belong on what servers for the Artifact and POST profiles. For source domains that use the LDAP IdMBridge, the CA certificate for the LDAP directory server must also be imported into a JDK keystore file.

Table 1 Summary of certificate configuration for the POST profile

POST profile	If the destination has no proxy	If the destination has a proxy
<p>...and the Source has no proxy</p>	<ul style="list-style-type: none"> • (A.) Each source exports the self-signed or third party CA signing certificate from its COREid Federation certificate store in <i>SHAREid_Install_Dir/conf/keystore</i>. • (B.) Each destination imports the source signing certificate into its COREid Federation certificate store in <i>SHAREid_Install_Dir/conf/keystore</i>. • (C.) The destination COREid Federation server must also have an SSL server certificate. The destination uses the key in the COREid Federation certificate store (in <i>SHAREid_Install_Dir/conf/keystore</i>) to generate this certificate. The destination imports the certificate into the keystore. • (D.) The source may optionally have an SSL server certificate. 	<ul style="list-style-type: none"> • See A. • See B. • The destination COREid Federation proxy must have an SSL server certificate in <i>SHAREid_Proxy_Install_Dir/conf/ssl.crt/server.crt</i>. To get the SSL server certificate, the destination generates the request using <code>openssl</code>. • See D.
<p>...and the source has a proxy</p>	<ul style="list-style-type: none"> • See A. • See B. • See C. • Optionally, the source COREid Federation proxy has an SSL server certificate in <i>SHAREid_Proxy_Install_Dir/conf/ssl.crt/server.crt</i>. To get the SSL server certificate, the destination generates the request using <code>openssl</code>. 	<ul style="list-style-type: none"> • See A. • See B. • The destination (required) and the source (optional) COREid Federation proxy have an SSL server certificate in <i>SHAREid_Proxy_Install_Dir/conf/ssl.crt/server.crt</i>. To get the SSL server certificate, the destination generates the request using <code>openssl</code>.

Table 2 summarizes certificate configuration for the Artifact profile when Basic authentication over SSL is used:

Table 2 Certificates for Artifact using Basic authentication over SSL

Artifact profile using Basic over SSL	Destination (with or without a proxy)
...and the Source has no proxy	<ul style="list-style-type: none"> • COREid Federation client and SSL server certificates are installed, per the descriptions in Table 1. • The destination imports the certificate of the CA that issued the source SSL server certificate into the COREid Federation certificate store <i>SHAREid_Install_Dir/conf/keystore</i>.
...and the source has a proxy	<ul style="list-style-type: none"> • COREid Federation client and SSL server certificates are installed, per the descriptions in Table 1. • The destination imports the certificate of the CA that issued the source <i>proxy</i> SSL server certificate into the COREid Federation certificate store <i>SHAREid_Install_Dir/conf/keystore</i>.

Table 3 summarizes certificate configuration for the Artifact profile when client certificate authentication is used:

Table 3 Certificates for Artifact using Client Certificate authentication

Artifact profile using Client Certificate Authentication	Destination (with or without a proxy)
...and the Source has no proxy	<ul style="list-style-type: none"> • See all of the steps for configuration with no proxies, Table 2, plus the following bullets. • The destination's COREid Federation server must have an SSL client certificate. To get the SSL client certificate, the destination's COREid Federation certificate store (in <i>SHAREid_Install_Dir/conf/keystore</i>) has a default key in it that is used to generate a certificate that the destination imports into the keystore. • The source imports the certificate of the CA that issued the destination SSL client certificate into the COREid Federation certificate store (<i>SHAREid_Install_Dir/conf/keystore</i>) and the certificate database (<i>SHAREid_Install_Dir/jdk/jre/lib/security</i>)

Table 3 Certificates for Artifact using Client Certificate authentication

<p>...and the source has a proxy</p>	<ul style="list-style-type: none">• See all of the steps for no proxy on the source, this table, plus the following bullets.• The source imports into the COREid Federation proxy certificate database <i>SHAREid_Proxy_Install_Dir/conf/ssl.crt/ca_bundle.crt</i> the certificate of the CA that issued the destination's SSL client certificate.• The source also edits the proxy's <i>SHAREid_Proxy_Install_Dir/ssl.conf</i> file. See "Configuring Certificates for the Proxy" on page 270 for details.
---	---

Maintaining Separate Signing and CA Keys

By default, COREid Federation uses the same certificate for signing and for HTTPS connections. If you want to use different certificates for these purposes, you update the file:

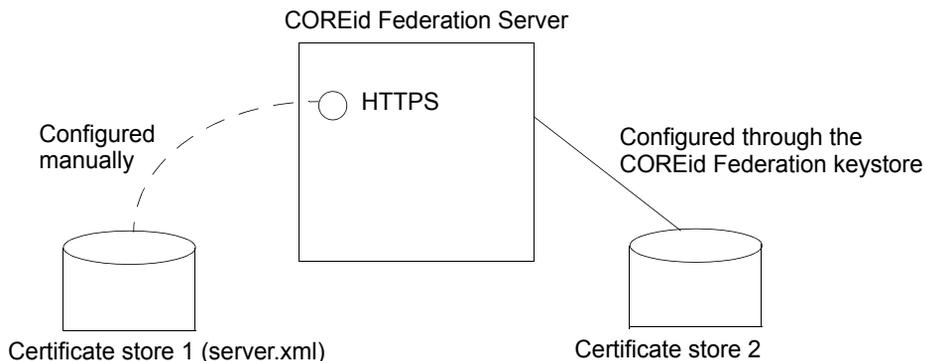
SHAREid_Install_dir/conf/server.xml

This is a Tomcat file. To maintain separate signing and CA keys, you:

- Create a new keystore file with a different key and certificate from the original COREid Federation keystore file.
- Change the keystoreFile attributes in *SHAREid_Install_Dir/conf/server.xml* to the path of the new keystore file.

Tomcat uses the first key entry in this file for client certification. As a result, Oracle recommends that you only keep one key in this file. Note that if you use a separate certificate for the signing key and the SSL key, you must use the same password for both the certificate store and for the signing key.

Figure 12 Options for using separate signing and HTTPS keys



If you want to use two different certificate stores, you would update `server.xml` or use the COREid Federation administration console to configure a custom certificate store. If you use separate signing and HTTPS keys in `server.xml`, you must use the same password for both the certificate store and each set of certificates and keys in the store.

The following is a sample of the `server.xml` file:

```
<!-- Define a SSL Coyote HTTP/1.1 Connector on port 8213 -->
- <Connector className="org.apache.coyote.tomcat4.CoyoteConnector" port="8213"
minProcessors="5" maxProcessors="75" enableLookups="true" acceptCount="100"
debug="0" scheme="https" secure="true" useURISValidationHack="false"
disableUploadTimeout="true">
<Factory className="org.apache.coyote.tomcat4.CoyoteServerSocketFactory"
clientAuth="false" keystoreFile="conf/keystore" keystorePass="changeit"
protocol="TLS" />
</Connector>
- <!-- Define a SSL Client Certificate Coyote HTTP/1.1 Connector on port 8214 -->
- <Connector className="org.apache.coyote.tomcat4.CoyoteConnector" port="8214"
minProcessors="5" maxProcessors="75" enableLookups="true" acceptCount="100"
debug="0" scheme="https" secure="true" useURISValidationHack="false"
disableUploadTimeout="true">
<Factory className="org.apache.coyote.tomcat4.CoyoteServerSocketFactory"
clientAuth="true" keystoreFile="conf/keystore" keystorePass="changeit"
protocol="TLS" />
</Connector>
```

In this file, if you keep the SSL server certificate and key in the COREid Federation certificate store, but you want to change the keystore password, you need to change the `keystorePass` attributes in `server.xml`.

To use a custom certificate store for SSL and the COREid Federation certificate store for the POST profile signing keys

1. In the server.xml file, edit the keystoreFile parameter to point to the new certificate store.
2. Optionally, in the server.xml file, edit the keystorePass parameter.
3. Create the new certificate store and be sure that it contains a certificate and a key.

If you specify a name for a file that does not yet exist in the keystore parameter, the keytool command will create the new keystore. For example:

```
keytool -keystore NewKeystore -storepass newPass -genkey -alias newKey
```

creates a file named NewKeystore with password called newPass holding a new key with the alias newKey.

Guidelines for Certificate Configuration

When setting up certificates, keep the following in mind:

- Back up your certificate store files before deleting or importing certificates.
- Use an alias other than shareid for your production certificates to prevent the environment from becoming unclear.

The default self-signed certificate uses the shareid alias. You can perform initial tests using this alias.

- After updating, importing, or exporting certificates, be sure to update the configuration for MyDomain and your partner domain, as appropriate, in the COREid Federation administration console. For example, if you are a destination and have imported a source CA certificate, you need to update the configuration for the source domain in the COREid Federation administration console.
- Be sure you update the passwords and alias in the COREid Federation administration console (in Encryption > Certificates) as well as on the command line.
- Restart the COREid Federation server after importing certificates or changing the certificate store.

Configuring Certificates for the POST Profile on the COREid Federation Server

For the POST profile, the destination must import the source's certificate. The source can give the destination either COREid Federation's default self-signed certificate (for testing purposes), or the certificate of the CA that certified the source domain's certificate.

For additional security, you may wish to configure a secure connection between the user's browser and COREid Federation at the source domain.

Note: The COREid Federation certificate store can contain many certificates. When you export and import certificates, change the name of the certificate file so that you do not overwrite any aliases already in the store. You should back up the certificate store. Note that if you import a domain's certificate, you should use the name of the domain to identify the certificate. If you import a CA certificate, use the name of the CA.

Configuring a Test Environment for POST on the COREid Federation Server

The following procedures are adequate for setting up secure connections for a test environment where the COREid Federation server resides in the DMZ (no proxy is used), and the POST profile is used for sending and receiving assertions.

To configure a test environment as a source using POST on the COREid Federation server

1. In the COREid Federation administration console, click Domain > MyDomain.
2. In the Modify MyDomain page, configure MyDomain as a source domain that uses the POST profile by adding the DNs for the signing certificate Subject and Issuer.
3. To view the contents of the COREid Federation certificate store located in *SHAREid_Install_Dir/conf*, type:

```
keytool -list -v -keystore keystore_file_name
```

4. To export the default self-signed certificate in the COREid Federation certificate store, from the operating system command line, go to the directory containing the keytool command, located in:

```
SHAREid_Install_Dir/jdk/bin
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

5. From this directory, export the default self-signed certificate by running the following command:

```
keytool -export -alias shareid -keystore keystore -file  
path-of-exported-cert.pem -rfc
```

where

- *keystore* is the name of the COREid Federation keystore (the default name is keystore)
- *path-of-exported-cert.pem* is a path with a unique name, for example, MyDomain.pem.

Note that on Windows, the directions of the slashes is reversed (“\”).

Note: The initial certificate store password is changeit. Change this password as soon as possible. See “Configuring Passwords and the Certificate Store” on page 136 and “Changing the Certificate Store Location and Passwords” on page 273 for details.

6. Send the exported certificate to the destination domain using a secure method that you and the destination have agreed upon.
7. In the COREid Federation administration console, configure the signing certificate subject and issuer DN for MyDomain using the information in your certificate.

See “Configuring a Destination Domain for a Source Domain” on page 155 for details.

8. In the COREid Federation administration console, select Encryption > Certificates and be sure that the information on this page matches your current configuration.

To configure a test environment as a destination using POST on the COREid Federation server

1. To import the source domain's certificate, from the operating system command line, go to the keytool command, located in:

```
SHAREid_Install_Dir/jdk/bin
```

where *SHAREid_Install_Dir* is the name of the directory where COREid Federation was installed. Note that on Windows, the direction of the slashes is reversed (“\”).

2. From this directory, import the certificate sent to you from the source domain into the COREid Federation certificate store by running the following command:

```
keytool -import -alias shareid-src -keystore keystore -storepass password -file path-of-imported-cert.pem
```

where

- *shareid-src* is the alias for the certificate (for example, the name of the CA or domain)
- *keystore* is the name of the COREid Federation certificate store (the default path is *SHAREid_Install_Dir/conf*)
- *-storepass password* sets the certificate store password (the default name is *changeit*)
- *path-of-imported-cert.pem* is a unique name, for example, *OtherDomain.pem*.

Note: The initial certificate store password is *changeit*. Change this password as soon as possible. See “Configuring Passwords and the Certificate Store” on page 136 for details.

3. In the COREid Federation administration console, configure the source domain, and be sure the signing certificate subject and issuer DNs match the information in the source certificate.

Use this command to view the certificate information:

```
keytool -list -v -keystore keystore_file_name
```

4. In the COREid Federation administration console, select Encryption > Certificates and be sure that the information on this page matches your current configuration.
5. Restart the COREid Federation server.

Configuring a Production Environment for POST Operations

The following procedures describe setting up secure connections for a production environment where the COREid Federation server resides in the DMZ (no proxy is used), and the POST profile is used.

Before the source exports its certificate as explained in “Configuring a Test Environment for POST on the COREid Federation Server” on page 254, the source domain must first delete its old keys and certificates and replace them with new keys and a certificate obtained from a third-party CA.

To configure a production environment using POST on the COREid Federation server

1. Go to the following directory:

```
SHAREid_Install_Dir/jdk/bin
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

2. From this directory, to delete your old keys, run the following command:

```
SHAREid_Install_Dir/conf/keytool -delete -alias shareid -keystore keystore -storepass changeit
```

where:

- *SHAREid_Install_Dir* is the directory where COREid Federation is installed.
- *keystore* is the full path to the keystore, for example, *SHAREid_Install_Dir*/conf/keystore.
- *changeit* is the keystore password.

3. From the same directory, generate a new key pair:

```
keytool -genkey -alias shareid -keyalg rsa -keystore keystore -keypass password -storepass password -validity 365 -dname 'cn=fully qualified domain name,ou=production,o=example,c=us'
```

where:

- *shareid* is the alias for the keystore. If you deleted the default COREid Federation alias as shown in the preceding step, you can use this alias for the new keystore.
- *keystore* is the name of the keystore.
- *-keypass password* is the password that protects the keys (the default is *changeit*).

- `-storepass password` is the password that protects the keystore (the default is `changeit`).
- `365` is the number of days until this key pair expires.
- `'cn=fully qualified domain name,ou=production,o=example,c=us'` is the Distinguished Name (DN) of the subject, enclosed in quotes on Windows and double quotes on Linux or Solaris. An example of the fully qualified domain name:

`shareid.example.com`

Note that the attributes required for the `-dname` parameter depend on the requirements of the CA issuing the certificate. For example, the CA may require attributes for locality (l), state (st), country (c), and so on.

4. From the same directory, generate a Web server certificate request:

```
keytool -certreq -alias shareid -keystore keystore -keypass password -storepass password -file certreq.pem
```

where

- *keystore* is the full path to the keystore, for example, `SHAREid_Install_Dir/conf/keystore`.
- `-keypass password` is the key password
- `-storepass password` is the keystore password

5. Submit the `.pem` file to a CA.

You can submit this Web server certificate request to any third-party CA. If the CA permits, request a key for the following purposes:

- Digital signature
- SSL client
- SSL server

6. Import the certificate generated by the CA:

```
keytool -import -alias shareid -keystore keystore -storepass password -file path-of-certificate
```

where

- *keystore* is the full path to the keystore, for example, `SHAREid_Install_Dir/conf/keystore`.
- *password* is the keystore password

7. Verify the certificate's Subject DN and Issuer DN by viewing the contents of the keystore:

```
keytool -list -v -keystore keystore -storepass password
```

where

- *keystore* is the full path to the keystore, for example, *SHAREid_Install_Dir/conf/keystore*.
- *password* is the keystore password

8. To import the source domain's certificate, from the operating system command line, go to the keytool command, located in:

```
SHAREid_Install_Dir/jdk/bin
```

where *SHAREid_Install_Dir* is the name of the directory where COREid Federation was installed. Note that on Windows, the direction of the slashes is reversed (“\”).

9. For verification of the digital signature in the assertion, either the source site certificate (used for signing) or the certificate of the CA that issued it needs to be imported into the destination keystore by running the following command:

```
keytool -import -alias alias -keystore keystore -storepass password -file  
path-of-imported-cert.pem
```

where

- *alias* should match the domain name for the source site domain configured at the destination if you are importing the source site certificate. If you are importing the certificate of the CA that issued the source domain's certificate, the alias can be any name.
- *keystore* is the full path to the keystore, for example, *SHAREid_Install_Dir/conf/keystore*.
- `-storepass password` sets the certificate store password (the default name is `changeit`)
- *path-of-imported-cert.pem* is the path of the .pem file being imported, for example, `OtherDomain.pem`

The initial certificate store password is `changeit`. Change this password as soon as possible. See “Configuring Passwords and the Certificate Store” on page 136 for details.

10. In the COREid Federation administration console, select Encryption > Certificates and be sure that the information on this page matches your current configuration.
11. Restart the COREid Federation server.

12. To finish configuring your keys and certificates, follow the steps in “Configuring a Test Environment for POST on the COREid Federation Server” on page 254.

Artifact Profile Using Basic Authentication on the COREid Federation Server

When you first install COREid Federation, the security required for the Artifact profile is not yet configured. To set up COREid Federation using the Artifact profile with Basic authentication of the requester using SSL, the following is required:

- The source domain must export the default self-signed certificate, or the source must tell the destination what CA the source is using. If both domains use the same CA, the export-import process is only required one time. However, if different CAs are used, both domains must export their certificates and import their partner’s.
- The source domain must enter the destination’s Requester ID and password when creating the destination domain. See “Configuring a Destination Domain for a Source Domain” on page 155 for details.

For the Artifact profile, this is the information that the Responder service in the source domain uses to authenticate a SAML request from a destination.

Note: The COREid Federation certificate store can contain many certificates. When you export and import certificates, change the name of the certificate file so that you do not overwrite any aliases already in the certificate store. It is recommended that you make a backup of the certificate store.

Testing Basic Over SSL with the Artifact Profile

The following procedures describe configuring one-way trust for a test environment. Note that this configuration is required when you select Basic authentication using SSL ports in the COREid Federation administration console.

To configure Basic Over SSL with the Artifact profile for a source COREid Federation server

1. In the COREid Federation administration console, configure MyDomain.
2. Modify the Responder URL in MyDomain to use the following:

`https://host:sslPort/rest of URL`

where *rest of URL* represents the remaining information in the Responder URL.

3. Export the default certificate with the shareid alias and send the certificate to the destination.

This certificate is created during installation and is in the COREid Federation certificate store. From the operating system command line, go to the COREid Federation certificate store, located in:

```
SHAREid_Install_Dir/jdk/bin
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed. Note that on Linux or Solaris, the direction of the slashes is reversed (“\”).

4. From this directory, export the default self-signed certificate by running the following command:

```
keytool -export -alias shareid -keystore keystore -file  
path-of-exported-cert.pem -rfc
```

where

- *keystore* is the name of the COREid Federation keystore (the default name is keystore)
- *path-of-exported-cert.pem* should use a unique name, for example, MyDomain.pem.

Note that on Windows, the directions of the slashes is reversed (“\”).

Note: The initial certificate store password is changeit. Change this password as soon as possible. See “Configuring Passwords and the Certificate Store” on page 136 and “Changing the Certificate Store Location and Passwords” on page 273 for details.

5. Send the exported certificate to the destination domain using a secure method that you and the destination have agreed upon.
6. In the COREid Federation administration console, configure the destination domain that you want to communicate with.

See “Configuring a Destination Domain for a Source Domain” on page 155 for details.

7. In the COREid Federation administration console page for configuring a destination domain, enter the Requester ID and password for this destination domain.

You obtain these values from the destination domain administrator.

8. In the COREid Federation administration console, select Encryption > Certificates and be sure that the information on this page matches your current configuration.
9. Restart the COREid Federation server.

To configure Basic Over SSL with the Artifact profile for a destination COREid Federation server in a test environment

1. From the COREid Federation administration console, click Domains > MyDomain.
2. Select basic authentication and supply a Requester ID and password for MyDomain.

The default Requester ID is *hostname-Open Port*

where *hostname-Open Port* is the name of the COREid Federation host and its open listen port, and the default password is *changeit*.

Note: The initial password is *changeit*. Change this password as soon as possible. See “Configuring Passwords and the Certificate Store” on page 136 for details.

3. Click View All Domains, click a link for the appropriate source domain, and modify the Responder URL in the modify domain page to use the following:

`https://host:sslPort/rest of URL`

where *rest of URL* represents the remaining information in the Responder URL for the source domain. You obtain this value from the source domain administrator.

4. To import the source domain’s certificate, from the operating system command line, go to the keytool command, located in:

`SHAREid_Install_Dir/jdk/bin`

where *SHAREid_Install_Dir* is the name of the directory where COREid Federation was installed. Note that on Windows, the direction of the slashes is reversed (“\”).

5. From this directory, import the certificate sent to you from the source domain into the COREid Federation certificate store by running the following command:

```
keytool -import -alias shareid-src -keystore keystore -storepass password
-file path-of-imported-cert.pem
```

where

- *shareid-src* is the alias for the certificate (for example, the name of the CA or domain)
- *keystore* is the name of the COREid Federation certificate store (the default path is *SHAREid_Install_Dir/conf*)
- `-storepass password` sets the certificate store password (the default name is *changeit*)

- *path-of-imported-cert.pem* is an example of a keystore file named, for example, *OtherDomain.pem*.

Note that on Windows, the directions of the slashes is reversed (“\”).

Note: The initial certificate store password is *changeit*. Change this password as soon as possible. See “Configuring Passwords and the Certificate Store” on page 136 for details.

6. Import the certificate into the COREid Federation certificate store, located in:

SHAREid_Install_Dir/conf/keystore

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

The command to import the certificate is as follows:

```
keytool [-v] -keystore SHAREid_Install_Dir/conf -storepass password
-import -alias -import -file
```

where

- *-v* provides verbose output.
- *SHAREid_Install_Dir* is the directory where COREid Federation is installed.
- *password* is the password (the default is *changeit*).
- *alias* is the alias for the certificate (for example, the name of the CA or domain).
- *file* is the certificate file exported by the keytool command at the other domain. This is a self-signed certificate or a certificate provided by a CA.

Example:

```
keytool -v -keystore C:\Program
Files\Obliv\SHAREid_0210_8101\SHAREid\conf\keystore -storepass
changeit -import -alias cert-shareid-smith-8111 -file C:\Program
Files\Obliv\SHAREid_0210_8101\SHAREid\conf\cert-shareid-smith-8111
```

Answer yes to the prompt about trusting this certificate.

This command is needed for HTTPS, so that the SAML Requester on the destination trusts the SAML Responder on the source.

7. In the COREid Federation administration console, select Encryption > Certificates and be sure that the information on this page matches your current configuration.
8. Restart the COREid Federation server.

Artifact Using Basic Over SSL Authentication

The process for configuring one-way trust for a production environment that uses the Artifact profile is as follows:

1. The source domain configures new keys and obtains a third-party CA certificate.

See “To configure a production environment using POST on the COREid Federation server” on page 257 for details. The procedure for configuring new keys and certificates is the same for all environments.

2. Complete the configuration steps for establishing one-way trust with the destination domain.

See “Testing Basic Over SSL with the Artifact Profile” on page 260 for details.

Artifact Using Client Certificate Authentication on the COREid Federation Server

If you opted to use client certificate authentication between source and destination domains, you ensure that both domains trust each other. Client certificate configuration for the Responder URL using the Artifact profile is similar to configuring SSL. However, for X.509 client certificate authentication, the source and the destination both export their certificates, and the source domain’s certificates are imported to the COREid Federation or proxy server certificate database rather than the COREid Federation certificate store.

Note that prior to configuring client certificate authentication, the destination domain provides the source domain with the destination’s certificate subject DN.

Configuring Client Certificate Authentication in a Test Environment

The following procedures describe how to configure client certificate authentication using the default COREid Federation self-signed certificates.

To configure client certificate authentication for a source using the Artifact profile

1. In the destination domain configuration page, modify the Requester Authentication to use X.509 certificates.
2. In the destination domain configuration page, supply the DN of the destination site's certificate in the Subject DN field.

You can get this information from the source domain administrator, who can look up the certificate Subject DN in the COREid Federation administration console. Or you can use the following command to look in the certificate store for this information:

```
keytool -list -v -keystore keystore_file_name
```

3. Modify the Responder URL in the MyDomain configuration page to use the following:

```
https://host:clientCertPort/rest of URL
```

where *rest of URL* represents the remaining information required to specify the Responder URL. Note that the URL may be structured differently for a non-COREid Federation domain.

4. Export the default certificate with the shareid alias and send the certificate to the destination.

This certificate is created during installation and is in the COREid Federation certificate store. From the operating system command line, go to the COREid Federation certificate store, located in:

```
SHAREid_Install_Dir/jdk/bin
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed. Note that on Linux or Solaris, the direction of the slashes is reversed (“\”).

5. From this directory, export the default self-signed certificate by running the following command:

```
keytool -export -alias shareid -keystore keystore -file  
path-of-exported-cert.pem -rfc
```

where

- *keystore* is the name of the COREid Federation keystore (the default name is keystore)

- *path-of-exported-cert.pem* uses a unique name, for example, *MyDomain.pem*.

Note that on Windows, the directions of the slashes is reversed (“\”).

Note: The initial certificate store password is changeit. Change this password as soon as possible. See “Configuring Passwords and the Certificate Store” on page 136 and “Changing the Certificate Store Location and Passwords” on page 273 for details.

6. Send the exported certificate to the destination domain using a secure method that you and the destination have agreed upon.
7. Import the destination’s certificate as follows:

From the operating system command line, go to the following directory:

```
SHAREid_Install_Dir/jdk/jre/lib/security
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

8. From this directory, run the following command, but do not use COREid Federation as the keyword as the alias:

```
../../../../bin/keytool -import -alias shareid-dest -keystore cacerts -file  
dest.keystore
```

where

- *shareid-dest* is the alias for the certificate (for example, the name of the CA or domain).
- *dest.keystore* is a unique file name.

Note: The initial certificate store password is changeit. Change this password as soon as possible. See “Configuring Passwords and the Certificate Store” on page 136 for details.

9. Also import the destination's certificate into the COREid Federation certificate store file located in:

```
SHAREid_Install_Dir/conf
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

The command to import the certificate is as follows:

```
keytool [-v] -keystore keystore -storepass changeit -import -alias alias -file file
```

where

- *-v* provides verbose output.
- *SHAREid_Install_Dir* is the directory where COREid Federation is installed.
- *changeit* is the password (the default is *changeit*).
- *alias* is the alias for the certificate (for example, the name of the CA or domain)
- *file* is the certificate file exported by the keytool command at the other domain. This is a self-signed certificate or a certificate provided by a CA.

Example:

```
keytool -v -keystore C:\Program  
Files\Obliv\SHAREid_0210_8101\SHAREid\conf -storepass changeit  
-import -alias cert-shareid-smith-8111 -file C:\Program  
Files\Obliv\SHAREid_0210_8101\SHAREid\conf\cert-shareid-smith-8111
```

Answer yes to the prompt about trusting this certificate.

This command is needed for HTTPS, so that the SAML Requester on the destination trusts the SAML Responder on the source.

10. Restart the COREid Federation server.

To configure client certificate authentication for a destination using the Artifact profile

1. From the COREid Federation administration console, click Domains > MyDomain, and modify the Requester Authentication to use X.509 certificates.
2. Modify the Responder URL in the MyDomain configuration page to use the following:

`https://host:clientCertPort/rest of URL`

where *rest of URL* represents the remaining information required to specify the Responder URL. Note that the URL may be structured differently for a non-COREid Federation domain.

3. From the operating system command line, go to the COREid Federation certificate store, located in:

`SHAREid_Install_Dir/jdk/bin`

where *SHAREid_Install_Dir* is the name of the directory where COREid Federation was installed.

4. From the COREid Federation certificate store, export your self-signed certificate by running the following command:

`./keytool -export -alias shareid-src -keystore keystore -file src.keystore`

where

- *shareid-src* is the alias (the default alias is *shareid*)
- *keystore* is the name of the COREid Federation keystore (the default name is *keystore*)
- *src.keystore* is a unique alias, for example, the name of the domain this keystore is being generated for.

Note: The initial keystore password is *changeit*. Change this password as soon as possible. See “Configuring Passwords and the Certificate Store” on page 136 for details.

5. Send this certificate to the source domain via a previously agreed-upon method.
6. To import the source domain’s certificate, from the operating system command line, go to the keytool command, located in:

`SHAREid_Install_Dir/jdk/bin`

where *SHAREid_Install_Dir* is the name of the directory where COREid Federation was installed. Note that on Windows, the direction of the slashes is reversed (“\”).

7. From this directory, import the certificate sent to you from the source domain into the COREid Federation certificate store by running the following command:

```
keytool -import -alias shareid-src -keystore keystore -storepass password
-file path-of-imported-cert.pem
```

where

- *shareid-src* is the alias for the certificate (for example, the name of the CA or domain)
- *keystore* is the name of the COREid Federation certificate store (the default path is *SHAREid_Install_Dir/conf*)
- *-storepass password* sets the certificate store password (the default name is *changeit*)
- *path-of-imported-cert.pem* is an example of a keystore file named, for example, *OtherDomain.pem*.

Note that on Windows, the directions of the slashes is reversed (“\”).

Note: The initial certificate store password is *changeit*. Change this password as soon as possible. See “Configuring Passwords and the Certificate Store” on page 136 for details.

8. Import the certificate into the COREid Federation keystore file, located in:

```
SHAREid_Install_Dir/conf
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

The command to import the certificate is as follows:

```
keytool [-v] -keystore SHAREid_Install_Dir/conf -storepass password
-import -alias alias -file file
```

where

- *-v* provides verbose output.
- *SHAREid_Install_Dir* is the directory where COREid Federation is installed.
- *password* is the password (the default is *changeit*).
- *alias* is the alias for the certificate (for example, the name of the CA or domain).
- *file* is the certificate file exported by the keytool command at the other domain. This is a self-signed certificate or a certificate provided by a CA.

Example:

```
keytool -v -keystore C:\Program
Files\Obliv\SHAREid_0210_8101\SHAREid\conf -storepass changeit
-import -alias cert-shareid-smith-8111 -file C:\Program
Files\Obliv\SHAREid_0210_8111\SHAREid\conf\cert-shareid-smith-8111
```

Answer yes to the prompt about trusting this certificate.

This command is needed for HTTPS, so that the SAML Requester on the destination trusts the SAML Responder on the source.

Configuring Client Certificate Authentication in a Production Environment

The process for configuring client certificate authentication for a production environment that uses the Artifact profile on the COREid Federation server is as follows:

1. Both the source and destination configure new keys and obtain a third-party CA certificate.

This process is the same for all environments. See “To configure a production environment using POST on the COREid Federation server” on page 257 for details.

2. Complete the configuration steps described in “Configuring Client Certificate Authentication in a Test Environment” on page 265.

Note that instead of working with the default self-signed certificate, as described in the procedure, the source obtains a CA certificate and exports it, and the destination imports it.

Configuring Certificates for the Proxy

The process for configuring certificates is similar whether you are installing the certificates on a proxy or the COREid Federation server. However, the proxy and the COREid Federation server use different commands to generate the keys and certificate requests, and certificate exchange on the proxy does not require use of a command like keytool for importing and exporting the certificates.

The following sections discuss certificate configuration for the POST profile on a proxy server.

Note: COREid Federation does not provide default self-signed certificates for the proxy.

Generating Keys and Certificates for the Proxy

The following procedure explains how to generate keys and certificates for a Proxy server.

To generate keys and certificates for the proxy

1. Go to the following directory:

Proxy_Install_Dir/bin

where *Proxy_Install_Dir* is the directory where the proxy is installed.

2. Generate the certificate key using the following command:

```
openssl genrsa -out server.key 1024
```

where *server.key* is the name of the key file. Note that *ssl.key* is a literal part of the path.

On Windows, do not specify the `-des` option. On Linux or Solaris, if you provide the `-des` option, a passphrase is required when you start up the proxy server.

3. Generate a Web server certificate request using the following command:

```
openssl req -config openssl.cnf -new -key server.key -out server.csr
```

where *server.key* is the name of the key file and *server.csr* is the name of the certificate request file.

When responding to prompts for this command, do not enter an e-mail address. When prompted for a common name, use the fully qualified domain name of the machine on which the proxy is installed.

4. Submit the `.csr` file to a Certificate Authority.

Ensure that the certificate is enabled for an SSL server.

5. When you receive the approved the certificate, copy the certificate to:

Proxy_Install_Dir/conf/ssl.key/*server.crt*

Copy only the approved certificate, not the CA certificate chain.

Configuring Certificates for the POST Profile

In a configuration where the destination uses a proxy and the source domain places COREid Federation in the DMZ, the destination's COREid Federation proxy must obtain an SSL server certificate.

Other certificate configuration for this scenario is the same as described in “Configuring a Test Environment for POST on the COREid Federation Server” on page 254 and “Configuring a Production Environment for POST Operations” on page 257.

Configuring Client Certificate Authentication for the Artifact Profile

In a configuration where both the source and the destination use a proxy, and Basic authentication over SSL has been configured, the source proxy CA certificate needs to be imported into the destination COREid Federation certificate store.

For the Artifact profile using Client Cert authentication, the source domain administrator must modify made to two files on the proxy: `ssl.conf` and `ca-bundle.crt`.

To modify the proxy configuration files

1. Open the following file:

```
SHAREid_Proxy_Install_Dir/conf/ssl.conf
```

where *SHAREid_Proxy_Install_Dir* is the location on the proxy server where the COREid Federation proxy was installed.

2. Uncomment these lines:

```
#Listen ##REPLACE_SSL_PROXY_CLIENT_CERT_PORT
#ServerName localhost:##REPLACE_SSL_PROXY_CLIENT_CERT_PORT
#SSLCACertificateFile SHAREid_Proxy_Install_Dir/conf/ssl.crt/
ca-bundle.crt
```

3. Replace

```
<virtualHost_Default_:8114>
```

with

```
<virtualHost_default_:client_certificate_port>
```

4. Select an unused port number for the SSL client port with certificate authentication.
5. Replace `##REPLACE_SSL_PROXY_CLIENT_CERT_PORT` with the port number throughout the file.
6. Create the following file:

```
SHAREid_Proxy_Install_Dir/conf/ssl.crt/ca-bundle.crt
```

7. Add the PEM-formatted certificate of the CA that issued the SSL client certificate to the destination domain.

PEM format refers to the base64-encoded certificate enclosed in the statements “-----BEGIN CERTIFICATE-----” and “-----END CERTIFICATE-----”.

8. Restart the proxy.

Verifying SSL and Client Certificate Authentication on the Proxy

To verify the SSL proxy, point your browser to the URL containing the SSL port using HTTPS. For example:

```
https://jsmith.oblix.net:8765/
```

If the configuration is correct, you should be served the Test Page for Apache Installation. You can use the same procedure to verify the client certificate port. In this case, you will receive a pop-up box asking for your client certificate.

Changing the Certificate Store Location and Passwords

Oracle recommends that you use this certificate store for COREid Federation only. A dedicated certificate store obviates any problems that might occur from interaction with other applications and storage of their certificates and key pairs.

However, if you have already set up a certificate store for use by other applications and you must store COREid Federation certificates and key pairs in that certificate store, you can change the default certificate store path. See “Configuring Passwords and the Certificate Store” on page 136 for details.

Note: You must use the same password for the certificate store and SSL server key.

To change the certificate store and key password

1. To change the certificate store password, go to:

```
SHAREid_Install_Dir/jdk/bin/
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

2. Run the following command:

```
keytool -storepasswd -new password -keystore keystore -storepass  
oldpassword
```

where:

- *password* is the new certificate store password.
 - *keystore* is the name of the COREid Federation keystore
 - *oldpassword* is the old certificate store password (the default is *changeit*).
3. Edit the following file:

SHAREid_Install_dir/conf/server.xml

Replace the old password (the default is *changeit*) with the new certificate store password.

4. To change the key password, go to:

SHAREid_Install_Dir/jdk/bin/

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed.

5. From this directory, run the following command:

```
keytool -keypasswd -alias shareid -keypass password -new password
-keystore keystore -storepass password1
```

where:

- *shareid* is the certificate store alias (the default is *shareid*)
- *changeit* is the old password
- *password* is the new password
- *keystore* is the name of the certificate store where the keys are located
- *password1* is the certificate store password.

Note: Use the same password for the keys and the certificate store.

6. From the COREid Federation administration console, click Encryption > Certificates and modify the certificate store and encryption key password.
7. Restart the COREid Federation server.

Configuring Certificates for the LDAP Data Source

If you use an LDAP IdMBridge, and you wish to use LDAPS (secure transmission of LDAP data) between COREid Federation and the directory, you must import the CA certificate for the directory server used by the LDAP IdMBridge into the COREid Federation JDK cacerts keystore file.

There are several reasons why you would want to use LDAPS:

- The directory only provides and ldaps port.
- Your organization's policies require that user data used by COREid Federation must be protected.

Oracle recommends that you use LDAPS for increased security.

To import the CA certificate for the directory server

1. Work with your directory administrator to obtain the directory server's CA certificate.

2. Go to the following directory:

SHAREid_Install_Dir/jdk/bin/

3. From this directory, run the following command:

```
keytool -keystore SHAREid_Install_Dir\jdk\jre\lib\security\cacerts -storepass  
password -import -alias alias -file certificate_file
```

where:

- *SHAREid_Install_Dir* is the directory where COREid Federation is installed.
- *password* is the password (the default is changeit).
- *alias* is the alias for the certificate (for example, the name of the CA)
- *certificate_file* is the certificate file path.

8 COREid Federation Auditing and Logging

COREid Federation allows you to capture and store information about system activity, assertions, and artifacts into audit and system log files. This chapter addresses the following topics:

- “Audit Logging” on page 277
- “System Activity Logging” on page 278

About COREid Federation Audit and System Logs

To monitor COREid Federation activity, you can configure audit and system activity log files. You can enable or disable audit logging from the Audit and Log page in the COREid Federation Administration Console. COREid Federation stores audit and system log information in separate files, based on the type of information collected.

Note: As an alternative to creating and using audit log files to capture assertions and artifact information, you can also store audit information in a relational database. To use this method of storing assertion and artifact information, install and configure the database to be used for the Assertion Store, then choose the RDBMS option from the Assertion Store page in the COREid Federation Administration Console. For more information, see “Setting up a Common Assertion Store Database” on page 216, in Chapter 6. Advanced Configuration.

Audit Logging

By default, assertion and artifact information is only maintained in cache. If you enable audit logging from the Audit and Log page in the COREid Federation Administration Console, Audit files are created and stored in the directory

SHAREid_Install_Dir/auditlogs

One audit file containing assertions that have been generated by or received by the COREid Federation server is created with the file name

`assertion_XXXX`

where *XXXX* is the generation of the server, starting at 0001 and incremented by 1 for each restart.

A second file containing artifacts that are associated with assertions (if the Artifact profile is used) is created with the name

`artifact_XXXX`

where *XXXX* is the generation of the server, starting at 0001 and incremented by 1 for each restart.

Assertion and artifact information is added to the audit files until the COREid Federation server is stopped. A new set of audit files is created each time a COREid Federation server is restarted. When COREid Federation starts, it deletes empty audit files of either type created by the last running instance of COREid Federation.

To enable auditing

1. From the COREid Federation Administration Console, click the Audits and Logs link.
2. From the Configure Audits and Logs page, check Enable Auditing.
3. Click Submit.

This change takes effect immediately.

Note: To save assertion and artifact auditing information to the Assertion Store database (instead of the audit and artifact audit files), you need to also select the Relational Database option from the Assertion Store configuration page.

System Activity Logging

System log files contain information about system status, errors, and debug messages. The default location for log files is

`SHAREid_Install_Dir/logs`

and system log files are created as

`shareid.log, shareid.log.1, shareid.log.2, ...`

where `shareid.log` is the current log file and older archive log files are numbered from 1 to *X* (*latest to earliest*), where *X* is the maximum number of archive log files you choose to maintain. Log files have a default maximum size of 100KB. The COREid Federation server rotates the current log file when it reaches that limit.

COREid Federation uses the log4j package to collect system log information. To configure system logging services, go to the `SHAREid_Install_Dir/oblix/config/shareid-log4j.properties` file. Using this file, you can specify options such as the logging level, maximum log file size, the number of log files to keep, and the content and format of information stored in the log file. Changes to the `shareid-log4j.properties` file are picked up automatically every 60 seconds.

Setting the System Logging Level

You can determine what level of information is logged by editing the log properties file, located in:

SHAREid_Install_Dir/oblix/config/shareid-log4j.properties

The logging level refers to the amount of information that is written to the log file. For example, a logging level of Error records only error messages. A logging level of Info records all error messages, plus information such as each instance of a SAML Responder authenticating a SAML Requester, each instance of setting a login cookie value, redirection events, and reading and writing of assertion files. A logging level of debug records the most information of any logging level.

You control the logging level by editing the `log4j.rootCategory` property in the log properties file. This file can be modified without requiring the server to restart and changes are picked up automatically every 60 seconds.

The `log4j.rootCategory` property can have one of the following values:

Table 4 Logging levels in `shareid-log4j.properties`

Value	Description
Error	Logs all error messages.
Info	Error level logging plus the following: <ul style="list-style-type: none">• Initialization activity (for example, loading of an <code>IdMBridge</code> class)• HTTP requests that the SAML servlets process• SAML request and response messages that were sent and received• SAML Responder authentications of SAML Requesters• Login cookie values that have been set• Redirection events• Reading and writing assertion files.
Debug	Info level logging plus the following: <ul style="list-style-type: none">• Mapped users• Internal debugging messages, including:<ul style="list-style-type: none">Method entry and exit with input and output parametersState changes within methods.

A

COREid Federation Security

This appendix provides background on SSL and client certificate authentication. Topics covered in this appendix are the following:

- “About COREid Federation Configuration for SSL” on page 282
 - “Public and Private Key Pairs” on page 282
 - “Certificates and Certificate Authorities” on page 282
 - “Digital Signatures” on page 283
 - “About Keystores for SSL Using the POST profile” on page 285
 - “About Keystores for the Artifact Profile” on page 286

About COREid Federation Configuration for SSL

COREid Federation provides secure communication using SSL and X.509 client certificate authentication. The following sections explain certificate-based security and how to obtain and install a certificate.

Public and Private Key Pairs

Encryption protects the integrity of data, keeps data confidential, and provides authentication. To provide encryption services, COREid Federation uses *public key encryption*. A *key* is an algorithm that encrypts data. Public key encryption is a special method of encryption that uses two kinds of keys, a public key and a private key.

The public key and the private key are generated as one key pair. Using public key encryption, data is signed with a private key and the signature is verified with a public key.

The primary goal of public key encryption is to ensure that data encrypted with one key cannot be decrypted with the same key. This means that even if the key is intercepted, the data is still secure because the reciprocal key is required to assist the attacker. Without the private signing key, a signature cannot be forged.

Keys are included in certificates. See “Certificates and Certificate Authorities” on page 282 for details.

Certificates and Certificate Authorities

COREid Federation uses key pairs and electronic documents known as *certificates* to protect assertions and to allow domains to trust each other. Certificates identify the user and help to prevent masqueraders from using fake keys. It is the trust established between domains that permits a destination domain to trust users from the source domain.

A *Certificate Authority (CA)* is a trusted authority that validates a user’s identity and issues a certificate that attests to the identity of a user. The CA includes the individual’s public key in the certificate. Only the owner of the certificate has access to the private signing key. The public key is included in the certificate, along with other information about the subject of the certificate. The certificate also includes an expiration date. Recipients of a certificate can verify that the signature on the certificate was encrypted using the certificate owner’s private key.

A certificate can be issued for an individual, a company, or some other entity, including another CA.

The CA digitally signs each certificate that it issues to attest to the authenticity of the certificate. The CA's digital signature allows the certificate to function as a formal introduction for users who know and trust the CA but who do not know the entity identified by the certificate.

The CA issues a certificate based on information that it receives in a certificate request form.

You configure certificates and key pairs for COREid Federation in a local COREid Federation keystore. The source domain configures one key pair and certificate when using the POST profile, and the destination imports the source domain's certificate. When the source domain encrypts an assertion, the destination decrypts it using the source domain's public key.

For the POST profile, a one-way trust relationship exists between the source and the destination. That is, the source exports its certificate and the destination imports it. This allows the destination to decrypt certificates sent by the source domain using the source domain's CA's public key. For the Artifact profile, either a one-way or two-way trust relationship exists between the source and destination. For two-way trust, the source and destination domains exchange certificates. Each administrator imports the other's CA certificate into its local keystore. The COREid Federation server or proxy and application server provide the Responder functions. For the Web or proxy server Responder function, the certificate is installed in the Web server's certificate database. In this case, source domain installs the destination's key pair and certificate in the Web server certificate database.

Digital Signatures

Data that is signed with a signing key is said to contain a digital signature. COREid Federation uses the following kinds of digital signature to authenticate information:

- **The CA's signature**—The Artifact profile uses the SAML protocol (SOAP over HTTPS) to get the response. The trust is in the SSL communications between the source and destination. Specifically, the trust is a result of importing the reciprocal domain's CA certificate (or self-signed certificate) into the keystore and marking it as trusted.
- **An XML signature for signing assertions**—An XML signature is the digest of signed data that has been encrypted by the signer's private key. COREid Federation uses an XML signature for the POST profile, as follows:
 - **As source domain**—To deliver a signed SAML Response. For the POST profile, the response holding the assertion is signed and put in the posted form data.
 - **As destination domain**—The SSL protocol allows the destination domain to verify the signed Response it receives from its partner source domain.

Verification of a Digital Signature

A digital signature is like a sophisticated checksum. It is a compressed form of a message for which the compression is not reversible. Digital signatures are created through a hash of data. A hash is a message-digest algorithm. The hash takes any amount of data as input and digests the message producing a fixed-size signature as the output. A digital signature is unique if the message is unique. If the message is unique, the signature is unique.

Another person can check the signature by hashing their copy of the message (the document), and comparing the resulting hash value to the original document. This process is as follows.

Process overview: Validating a digital signature

1. The sender of the message computes a digest for the message.

For an XML-based assertion, there are transformation algorithms to massage the XML data so it can be signed and verified. The point of the transformation is that both the signer and the verifier have the same sequence of bytes to process.

2. The signer then encrypts the digest with their private key.

SHA and MD5 are examples of algorithms used to produce digital signatures.

3. The verifier recomputes the digest of the data, decrypts the signature (which consists of the digest encrypted with the signer's private key), using the signer's public key, and compares the original and the recomputed digest.

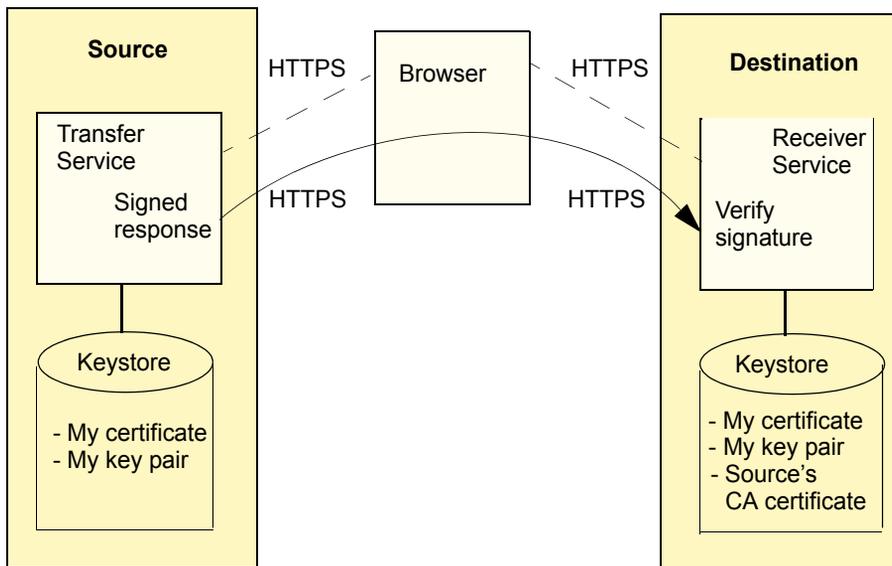
Using this process, a digital signature combines the message-digest algorithm with private and public key cryptography.

About Keystores for SSL Using the POST profile

For the POST profile, the source domain exports the COREid Federation self-signed certificate, or it exports its root CA certificate from the COREid Federation keystore. The destination domain imports the source domain's certificate.

Figure 13 shows the keystores and certificates for the POST profile or the Artifact profile using SSL with Basic authentication.

Figure 13 Keystores for POST or for Artifact using one-way trust



Note that the destination domain must import the source's certificate into the destination keystore. The destination domain must also configure the subject DN from the source's certificate in the COREid Federation Administration Console.

Table 5 provides describes the keystore configurations for the POST profile.

Table 5 POST Profile Services and Keystore Configuration

Service	Functions Provided and Keystore Configuration
<ul style="list-style-type: none"> • Transfer service 	<p>The source Transfer service:</p> <ul style="list-style-type: none"> • Gets user information from the HTTP request • Generates an assertion and adds it to a Response • Signs the Response • Constructs a Transfer Form for the Response • Returns the Response to the user's browser. • The browser posts the Response to the destination's Receiver service. <p>Keystore configuration:</p> <ul style="list-style-type: none"> • The source must have its own certificate and signing key pair in its COREid Federation keystore.
<ul style="list-style-type: none"> • Receiver service 	<p>The destination Receiver service:</p> <ul style="list-style-type: none"> • Receives the Transfer Form and resource request • Verifies the signature <p>Keystore configuration:</p> <ul style="list-style-type: none"> • The destination domain must have the source's CA certificate to verify signature.

About Keystores for the Artifact Profile

Source domains that use the Artifact profile and Basic authentication over SSL must do the following:

- Import your key pair and certificate into your Web server certificate database.
- Configure the destination's Requester ID and SHA-1 digest of the Requester Password in the COREid Federation Administration Console.

The destination must do the following for the Artifact profile using Basic authentication over SSL:

- Import the source domain's CA certificate in the COREid Federation keystore.
- Configure the source's Requester ID and Requester Password in the COREid Federation Administration Console.

Source domains that use the Artifact profile and client certificate authentication do the following:

- Import your key pair and certificate into your Web server certificate database.

- Import the destination domain's certificate into your Web server certificate database.
- Configure the subject DN from the destination's certificate in the COREid Federation Administration Console

Destination domains do the following:

- Import your key pair and certificate into your COREid Federation keystore.
- Import the source domain's certificate into your COREid Federation keystore.

Figure 14 shows the requirements for the Artifact profile using client certificate authentication.

Figure 14 Artifact profile: Keystores for client certificate authentication

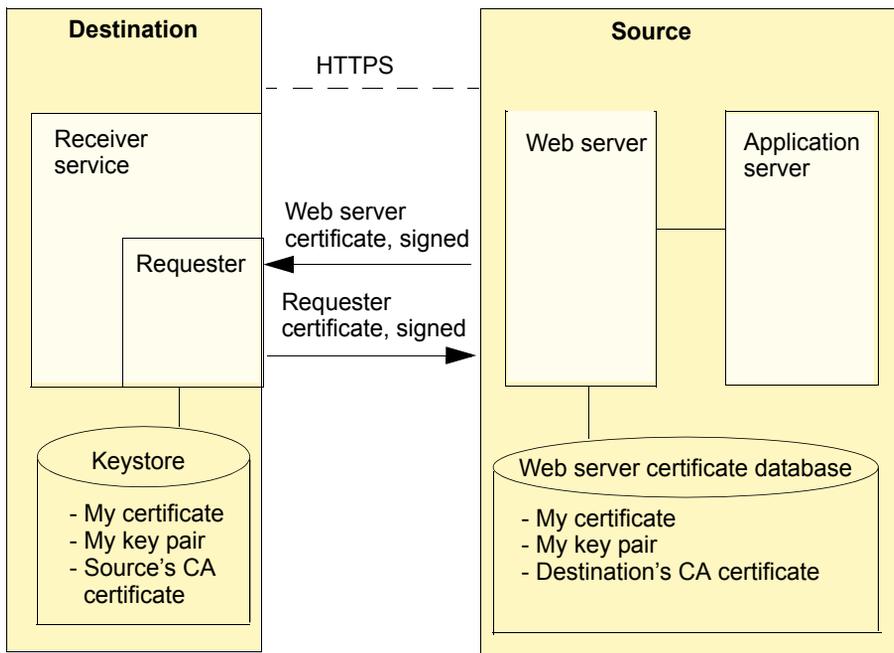


Table 6 summarizes the keys and certificates that are used for the Artifact profile.

Table 6 Artifact profile keystores for client certificate authentication

Service	Functions Provided and Keystore Configuration
Transfer service	<p>The source Transfer service:</p> <ul style="list-style-type: none"> • Gets user information from the HTTPS request. • Creates an assertion for the user. • Creates an artifact for the assertion. • Redirects the browser and the artifact to the destination's Receiver service. <p>No keys or certificates are used. The artifact is sent over HTTPS.</p>
<ul style="list-style-type: none"> • Receiver service 	<p>The destination's Receiver service:</p> <ul style="list-style-type: none"> • Receives and extracts the artifact. • Maps the source ID in the artifact to the source Responder service. • Passes the artifact to its Requester service. <p>This is local to the destination domain. No keys are required.</p>

Table 6 Artifact profile keystores for client certificate authentication

Service	Functions Provided and Keystore Configuration
• Requester component	<p>The destination's Requester:</p> <ul style="list-style-type: none">• Establishes a secure connection with the source domain.• Obtains the artifact from its own Receiver service.• Requests the assertion from the source's Responder service.• Receives the assertion. <p>Keystore configuration:</p> <ul style="list-style-type: none">• The destination domain stores its own certificate and key pair in its COREid Federation keystore. This certificate identifies the destination to the source.• The destination installs the source domain's CA certificate is authenticate information passed from the source Responder service.
• Responder service	<p>The source Responder:</p> <ul style="list-style-type: none">• Establishes a secure connection with the destination.• Receives the request from the destination Requester.• Sends the assertion. <p>Keystore configuration:</p> <ul style="list-style-type: none">• The source stores its certificate and key pair in its Web server certificate database. This identifies the source to the destination Requester.• The source also installs the destination's CA certificate to authenticate the destination.

B SiteMinder System Configuration for COREid Federation

If you want to use a SiteMinder installation as the user data repository for COREid Federation source domains or as the Identity Management System providing destination domain authentication and authorization, you can configure a SiteMinder IdMBridge when you set up COREid Federation from the COREid Federation Administration Console. (Before configuring the SiteMinder IdMBridge, your SiteMinder installation must already be configured for use with COREid Federation.)

This appendix describes the requirements and setup of SiteMinder to be used when you configure a COREid Federation SiteMinder IdMBridge.

Note: For more information on configuring the COREid Federation SiteMinder IdMBridge from the Administration Console, once you've configured your SiteMinder installation, see "Configuring a SiteMinder IdMBridge" on page 129 for source domain configuration, and "Configuring a SiteMinder IdMBridge" on page 129 for destination domain configuration.

This appendix includes the following topics:

- "SiteMinder Requirements" on page 292
- "SiteMinder System Configuration for COREid Federation" on page 293
- "SiteMinder Configuration and Setup for COREid Federation" on page 301

SiteMinder Requirements

To use SiteMinder for a COREid Federation IdMBridge, you need the following SiteMinder system configuration.

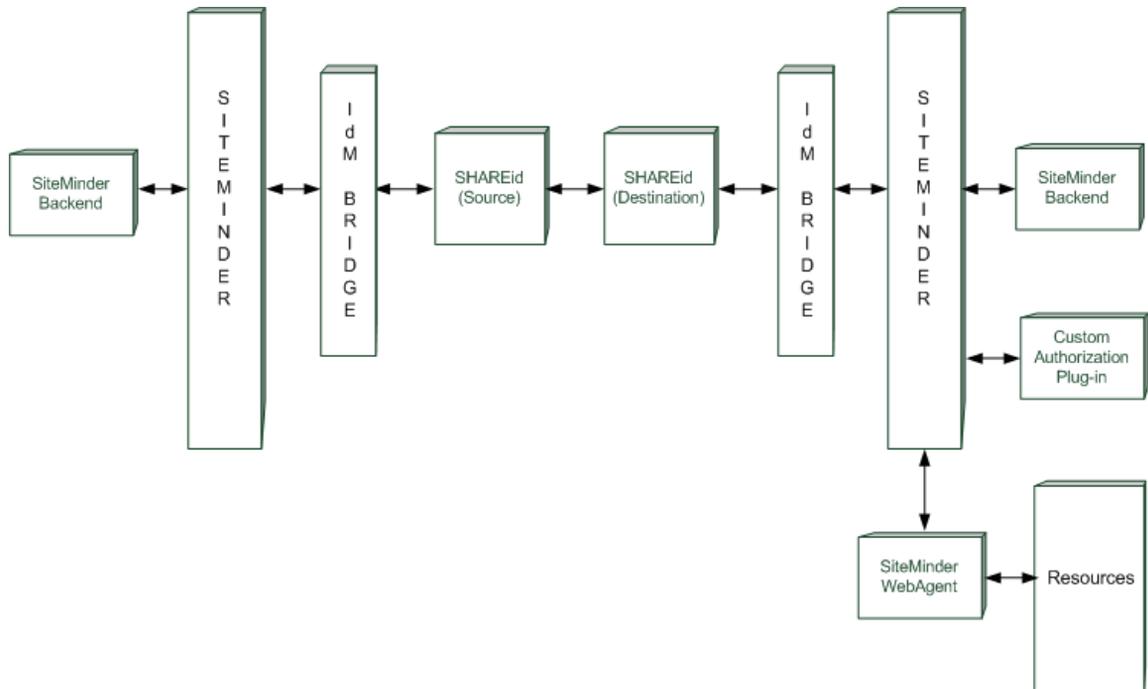
- SiteMinder Policy Server, Version 5.5 with SiteMinder Optional Pack
For SiteMinder Policy Server installation guidelines refer to the SiteMinder policy-server-installation-guide_v55.pdf and policy-server-management_v55.pdf documents. For the Optional pack installation guidelines, refer to the sm-option-pack-guide_v55.pdf document.
- SiteMinder Web Agent, Version 5.0 with patch QMR1 or higher. (Make sure that the corresponding Web Server has been stopped, before applying the patch for WebAgent.
For SiteMinder WebAgent installation guidelines, refer to the sm-web-agent-installation-guide_v50.pdf document. For installation guidelines on the patch, refer to the SiteMinder sm-upgrade-guide_v55.pdf document.
- SiteMinder SDK, Version 5.5. For SiteMinder SDK installation guidelines refer to the SiteMinder sm-sdk-overview_v55.pdf document.

For specific SiteMinder configuration steps required for use with COREid Federation, see “SiteMinder Configuration and Setup for COREid Federation” on page 301

SiteMinder System Configuration for COREid Federation

The following diagram shows the configuration of a typical system in which a SiteMinder installation is used with COREid Federation.

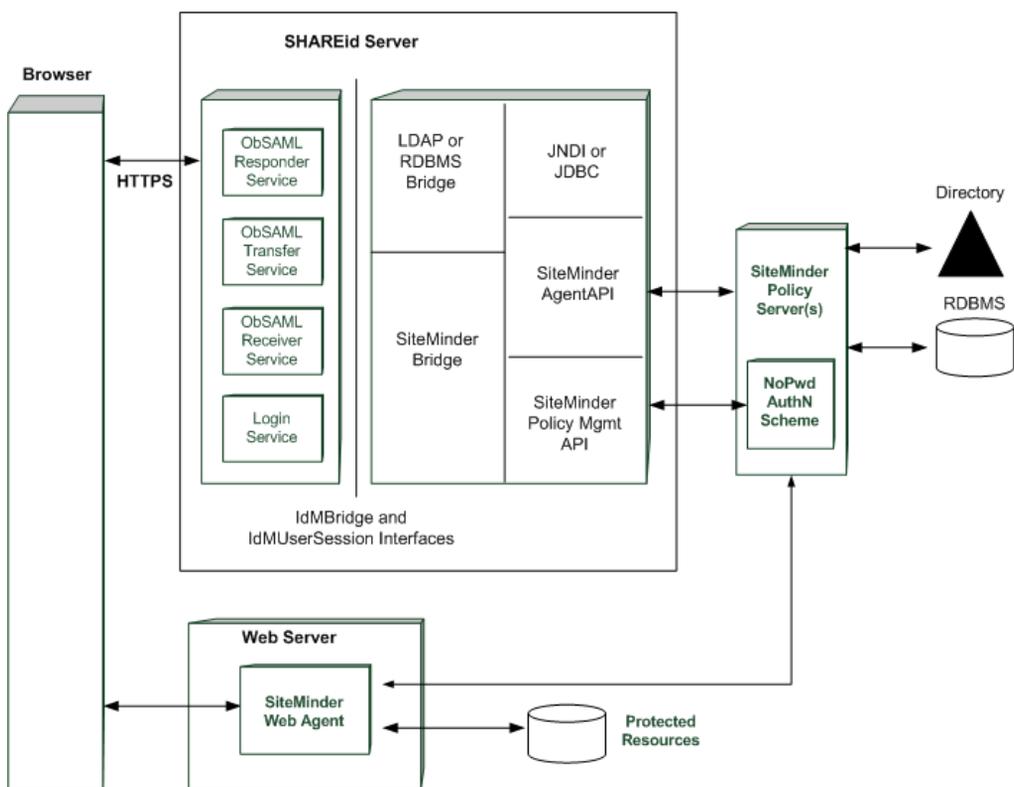
Figure 1 SiteMinder IdMBridge configuration in Source and Destination Domains



SiteMinder can be configured as the COREid Federation IdMBridge to provide the user data repository for source domain configurations, and serve as the authentication and authorization authority for resource access at destination domains

The following diagram shows the configuration of a typical system in which a SiteMinder installation is used with COREid Federation.

Figure 2 COREid Federation Components with the SiteMinder Bridge



The following paragraphs provide a description of components shown in Figure 2.

- **ObSAML Services**—handle requests and transfers between COREid Federation and a requesting user’s Web browser.
- **IdM/IUserSession Interfaces**—used by COREid Federation to authenticate users from credentials such as userid and password, to map incoming SSO assertions to local users, to retrieve user and session attributes, and to determine if resource access by a user is authorized.
- **SiteMinder Bridge**—implements the IdM/IUserSession interfaces using the SiteMinder Agent API.
- **LDAP or RDBMS Bridge**—used to provide the equivalent of the COREid credential mapping function for SiteMinder. SiteMinder authentication schemes require a userid, which may or may not be present in an incoming SSO assertion. In the case where the assertion does not have a userid, assertion data need to be mapped to a user before the SiteMinder authentication can occur. For a user directory, the LDAP Bridge can perform this secondary function; for a user database, the RDBMS Bridge can be used. In these cases,

the SiteMinder Bridge will call the require functionality in these “secondary” bridges to perform the mapping.

- SiteMinder Agent API—connects to a SiteMinder Policy Server, sends authentication and authorization requests to the server, and receives responses from the server. The SiteMinder Bridge uses the Java Agent API, which consists of two jar files (smjavaagentapi.jar and smjavasdk2.jar) and one native library for each platform (smjavaagentapi.dll for Windows, smjavaagentapi.so for Linux and Solaris). The Agent API connections to the Policy Server are secured using an agent secret shared between the agent and the Policy Server, and additional encryption keys generated by the server and distributed to the agent. Connections to multiple Policy Server replicas can be configured for an agent, and the agent will load-balance requests across the replicas and will failover in case a replica fails to respond within a time-out period.
- SiteMinder Policy Management API—allows a client to create, modify, and delete SiteMinder policy objects. The SiteMinder Bridge may use this API to construct the realms, policies, and authentication schemes it uses. Otherwise, the customer will have to set these up through the SiteMinder administration console interface.
- SiteMinder Web Agent—controls access to resources on the destination site. COREid Federation sets the SMSESSION cookie with a session token for the mapped user for use by the Web Agent.

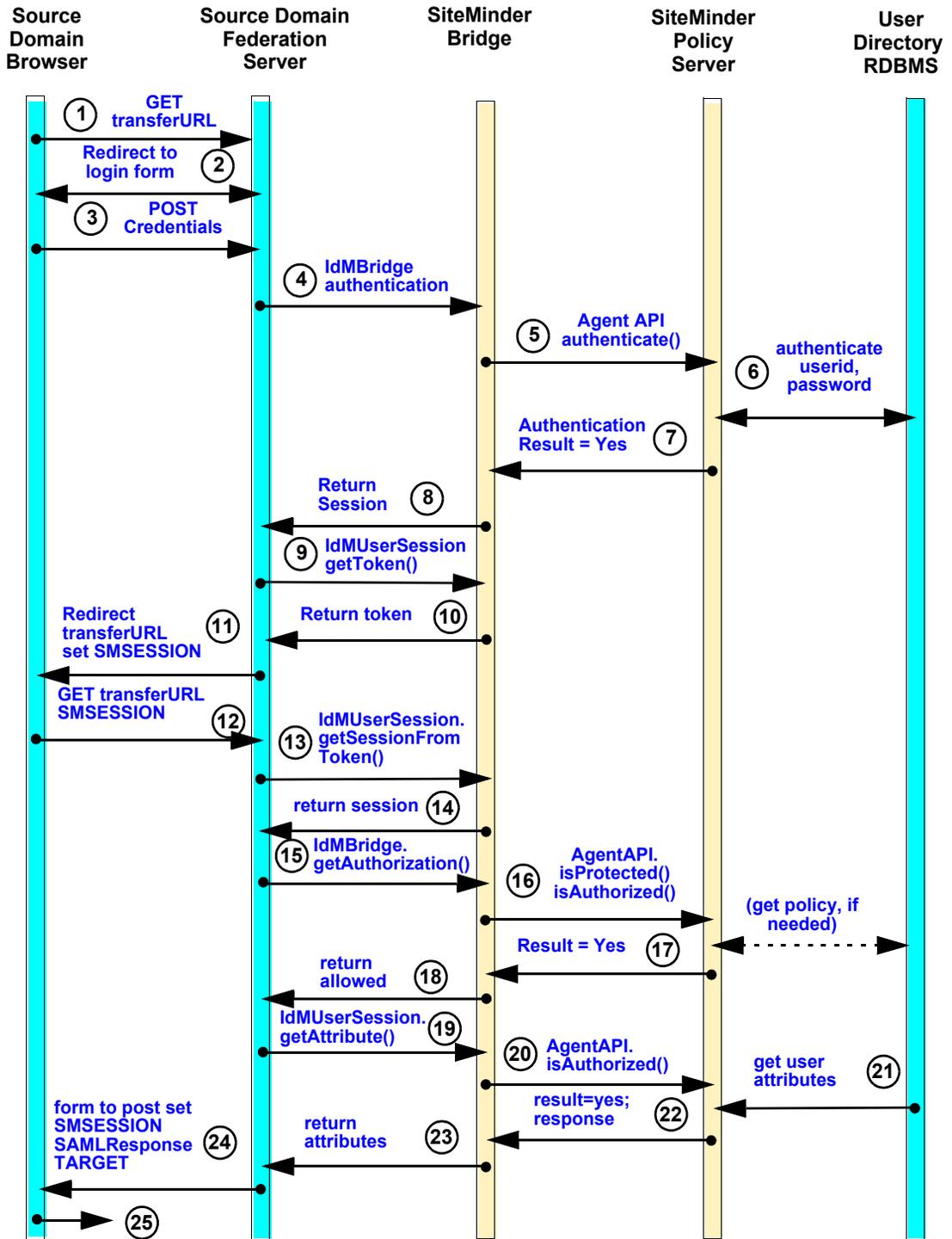
Note: SiteMinder 4.x and 5.x Web Agents need to be patched to 4.xQMR4 or later or 5.xQMR1 or later, respectively, to accept third-party session tokens from custom agents. Also, a configuration flag AcceptTPCookie="yes" must be set in the web agents configuration. See Section 5.2 in the Netegrity SiteMinder SDK Version 5.5 Release Notes for details about how to do this.

- SiteMinder Policy Server(s)—perform authentication and authorization on behalf of SiteMinder agents. Each Policy Server connects to user and policy directories or databases.
- Custom No-Password Authentication Plug-in—required for mapping SSO assertions to SiteMinder users. Normally SiteMinder authentication schemes require a userid and password, but SSO assertions will not provide passwords, so this function is performed by the custom authentication plug-in, written to the SiteMinder Authentication API.

The following illustration shows the process flow for COREid Federation operations using SiteMinder for a source domain configuration.

Figure 3 COREid Federation process flow using SiteMinder at a source

domain



1. The user at his or her browser requests a transfer from the source site to a destination site, for example:

```
https://shareid.source.com/shareid/saml/  
ObsAMLLTransferService?DOMAIN=dest&METHOD=post&  
TARGET=https://target.dest.com/someResource.
```

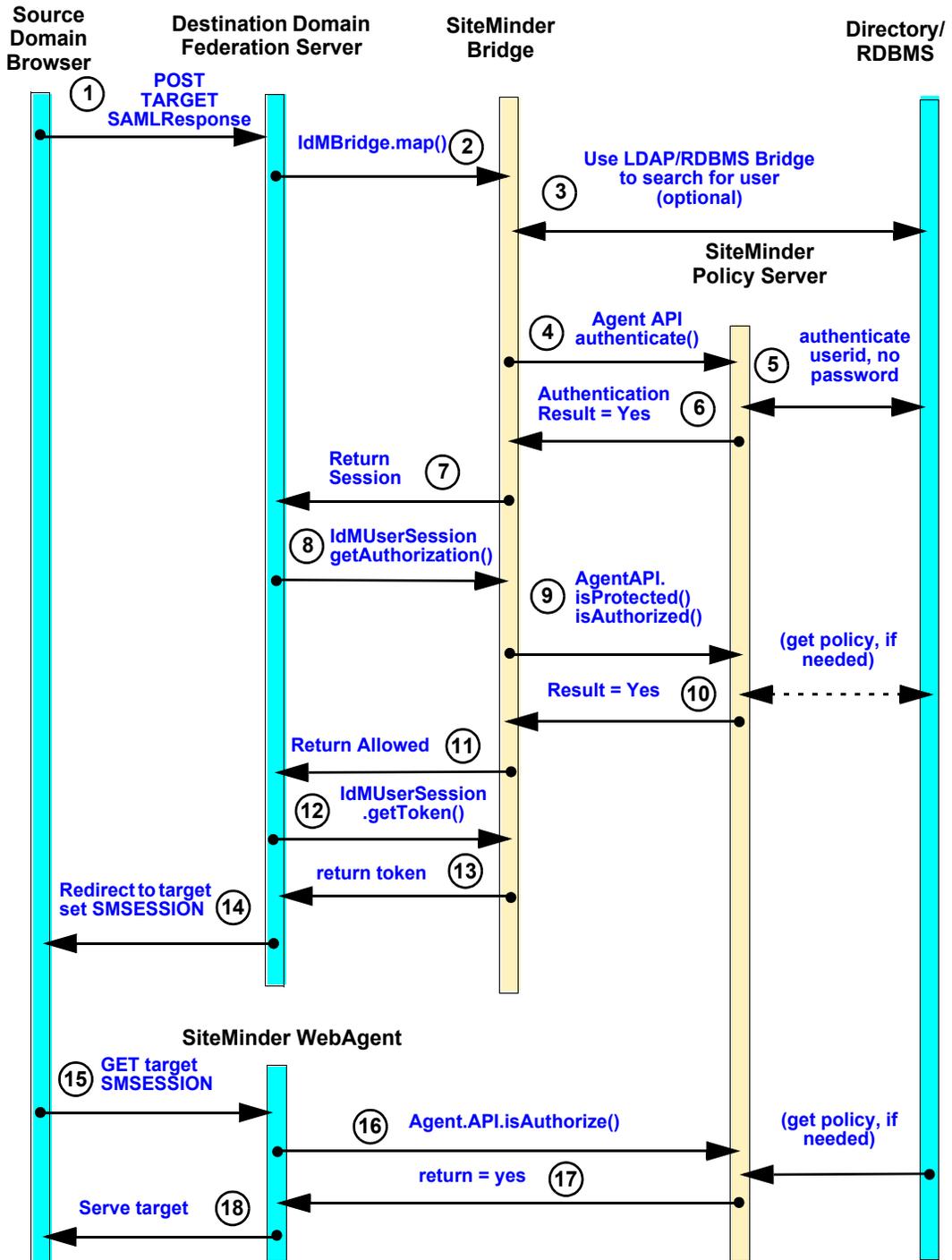
For simplicity, in this example, the browser post profile (BPP) is used, but the operation of the SiteMinder IdMBridge is the same for the browser artifact profile (BAP).

2. Assuming that the user has not previously logged in using the local SiteMinder (and hence does not have an SMSESSION cookie yet), COREid Federation will respond with a login challenge, in this example a redirection to an HTML login form.
3. Users, in response to the login challenge, enter their credentials - a userid and password - which the browser sends in a POST request to the COREid Federation Server.
4. The ObsAMLLLoginService servlet receives the posted credentials and calls IdMBridge.authenticate() with the userid and password.
5. The SiteMinder IdMBridge implementation of IdMBridge.authenticate() calls the AgentAPI.authenticate() method, passing the userid and password and a resource /SHAREid-SMAgent/Login that is protected by a SiteMinder realm that has a Basic or Form-Based Authn authentication scheme.
6. The SiteMinder Policy Server receives the authenticate request from the Agent API and performs the authentication via the user directory or database.
7. Assuming that the credentials are correct, the Policy Server sends a successful response to the Agent API, which returns an AgentAPI.YES result to the SiteMinder IdMBridge.
8. The SiteMinder IdMBridge constructs a SMUserSession object with session information from the Agent API and returns the session object to the ObsAMLLLoginService servlet.
9. ObsAMLLLoginService calls IdMUserSession.getToken() for the newly created session.
10. The SiteMinder IdMBridge implementation of IdMUserSession.getToken() calls AgentAPI.createSSOToken() to return an encrypted string token for the SiteMinder session.
11. ObsAMLLLoginService sets a SMSESSION cookie with the string token and redirects the user's browser back to the Transfer Service.
12. The user's browser sends another GET request for the Transfer Service with the SMSESSION cookie.
13. The ObsAMLLTransferService servlet gets the SMSESSION cookie and calls IdMUserSession.getSessionFromToken().

14. The SiteMinder IdMBridge implementation of IdMUserSession getSessionFromToken() calls AgentAPI.decodeSSOToken() to decrypt the token and reestablish the SiteMinder session.
15. ObSAMLTransferService calls IdMUserSession.getAuthorization() for the user session and the transfer URL to check if the user is authorized by a SiteMinder policy to use the Transfer Service.
16. The SiteMinder IdMBridge implementation of IdMUserSession getAuthorization() calls AgentAPI.isProtected() for the transfer URL, and, if the transfer URL is protected by a SiteMinder policy, it calls AgentAPI.isAuthorized() for the user and the transfer URL. This allows the SiteMinder administrator to control who can use the COREid Federation Transfer Service.
17. If necessary, the Policy Server retrieves the relevant policies from the policy directory or database, determines (in a separate request) if the resource is protected, and if so, applies the policies and returns an indication whether access is allowed or denied.
18. The SiteMinder IdMBridge returns Permitted, Denied, or Indeterminate (if the resource is not protected).
19. Assume that access to the Transfer Service is not denied, ObSAMLTransferService calls IdMUserSession.getAttributes() to get attributes for the construction of the SSO assertion.
20. The SiteMinder IdMBridge implementation of IdMUserSession.getAttributes() calls AgentAPI.isAuthorized() for the user and the resource /SHAREid-SMAgent/Login, which has a policy with responses to return the desired user attributes.
21. The Policy Server, under the direction of the policy associated with the /SHAREid-SMAgent/Login resource, retrieves the user attributes from the user directory or database.
22. The Policy Server returns a result with the user attributes in the response.
23. The SiteMinder IdMBridge returns the user attributes to ObSAMLTransferService, which uses them to construct the SSO assertion as specified by the configured assertion profile for the target domain.
24. ObSAMLTransferService uses the user attributes to construct the SSO assertion as specified by the configured assertion profile for the target domain, embeds the assertion in a SAML response, constructs a form with SAMLResponse and TARGET fields, and returns the form in an HTTP response to the user's browser.
25. The browser sends a POST request of the form data to the destination site.

The following illustration shows the process flow for COREid Federation operations using SiteMinder for a destination domain configuration.

Figure 4 COREid Federation process flow using SiteMinder at a destination domain



1. Continuing with the post profile example, the source site has sent to the user's browser a form with hidden variables TARGET, with the target URL, and SAMLResponse, containing the SSO assertion. JavaScript in the form automatically posts the form data to the destination site.
2. The ObSAMLReceiverService servlet on the destination site receives the posted data, gets the SSO assertion from the SAMLResponse variable, and calls IdMBridge.map() with properties extracted from the assertion.
3. The SiteMinder IdMBridge implementation of IdMBridge.map() may optionally use the assertion properties to search for a user in the SiteMinder user repository. It would need to do this if the assertion properties did not contain the userid required by the applicable SiteMinder authentication scheme. The bridge would use functionality in either the existing LDAP Bridge (for a user directory) or the new RDBMS Bridge (for a user database) to search for the user.
4. Once it has the userid for the user, the SiteMinder IdMBridge calls AgentAPI.authenticate() with the userid, no password, and a resource / SHAREid-SMAgent/LoginNoPwd that is protected by a SiteMinder realm with a special purpose authentication scheme that does not require a password.
5. The SiteMinder Policy Server calls the No Password authentication plug-in supplied with COREid Federation, which authenticates the userid.
6. The Policy Server returns, through the AgentAPI, a result of YES and the resulting session.
7. The SiteMinder IdMBridge creates and returns a SMUserSession object to encapsulate the SiteMinder session.
8. ObSAMLReceiverService calls IdMUserSession.getAuthorization() to determine if the user session is allowed to access the target resource. This is a courtesy check, since the actual access control enforcement is performed by the SiteMinder Web Agent in steps 16 and 17.
9. The SiteMinder IdMBridge implementation of IdMUserSession.getAuthorization() calls AgentAPI.isProtected() for the target URL and, if it is protected by a SiteMinder policy, it calls AgentAPI.isAuthorized() for the SiteMinder user session and the target resource.

Note: SiteMinder policy does not allow one agent to make an authorization request for a resource controlled by another agent. The realm that protects a resource specifies only one agent, so the COREid Federation agent cannot use the realm and associated policies defined for a web agent.

10. The Policy Server applies the relevant policy and returns, through the Agent API, a result of YES if it is protected.
11. The SiteMinder IdMBridge returns "Permitted" or "Indeterminant" to ObSAMLReceiverService, which proceeds with processing the transfer.

12. ObSAMLReceiverService calls `IdmUserSession.getToken()` to get a string token for the user session.
13. The SiteMinder implementation of `IdmUserSession.getToken()` calls `AgentAPI.createSSOToken()` to create the string token for the SiteMinder session.
14. ObSAMLReceiverService sets the `SMSESSION` cookie to the string token and redirects the user's browser to the target URL.
15. The user's browser issues a GET request for the target URL, with the `SMSESSION` cookie, to the web server managing the resource.
16. The SiteMinder Web Agent configured for the target web server intercepts the GET request, extracts the session from the `SMSESSION` cookie, and calls `AgentAPI.isAuthorized()` for the user and the resource.
17. The Policy Server determines that the user is authorized to access the resource, so it returns a result of YES through the AgentAPI (or its equivalent).
18. The SiteMinder Web Agent serves the resource to the user's browser.

SiteMinder Configuration and Setup for COREid Federation

Here are the steps required to configure SiteMinder for operation with COREid Federation:

- Install the SiteMinder SDK on the same machine as your COREid Federation Server.
- “Copy Jar Files and Configure Your Environment for the SiteMinder SDK” on page 302
- “Create a SiteMinder WebAgent Identity” on page 305
- “Create and Configure a SiteMinder User Directory” on page 306
- “Create a SiteMinder Domain” on page 307
- “Changing Agent Cookie Settings (COREid Federation Destination)” on page 309
- “Configure and Enable SiteMinder Logs” on page 310

The following sections provide instructions, program listings, and screen display details for performing these operations. Following SiteMinder configuration, you can use the COREid Federation Administration Console and configure a COREid Federation IdMBridge to use SiteMinder, and then configure Login and Assertion Profiles based on whether you are setting up COREid Federation for a source domain, a destination domain, or both.

Install the SiteMinder SDK

The first step in configuring a source or destination domain where you want to configure a SiteMinder IdMBridge is to install the SiteMinder SDK, Version 5.5, on the same machine as COREid Federation. For SiteMinder SDK installation guidelines refer to the SiteMinder sm-sdk-overview_v55.pdf document available from your software vendor.

After installing the SiteMinder SDK, make a note of the directory in which the SDK is installed as you will need it later when you complete other configuration steps for COREid Federation.

Copy Jar Files and Configure Your Environment for the SiteMinder SDK

Before you begin adding or changing configuration settings from either the COREid Federation or SiteMinder administration consoles, copy the smjavaagentapi.jar and smjavasdk2.jar files from the <SiteMinder SDK Install>\SDK\java directory to the <SHAREid_Home>\webapps\shareid\WEB-INF\lib and <SHAREid_Home>\common\endorsed directories.

Note: For destination sites, you also need to copy the smanapi.dll file provided with COREid Federation to your SiteMinder\bin installation directory.

You need to set platform-specific PATH environment variables for COREid Federation to use to include the path to the installed SiteMinder SDK. On Windows platforms, paths are set in the <SHAREid_HOME>/bin/setenv.bat file. For Solaris, you set paths in a .sh file.

For example, on Windows, you might set the path as follows:

```
set PATH=C:\SiteMinder_5.5\SDK\bin;C:\SiteMinder_5.5\SDK\bin;
%SHAREID_HOME%\oblix\lib;%PATH%
```

For Solaris, you might set the path as follows (making **highlighted** changes):

```
#!/usr/bin/sh
```

```
-----
# script that sets environment variables for Federation server#
-----
```

```
SHAREID_HOME=/export/home1/panacea/test/shareid/source/SHAREid
```

```
SM_LIB_PATH=/export/home1/panacea/test/SDK/bin
```

```
CATALINA_HOME="${SHAREID_HOME}"
```

```
JAVA_HOME="${SHAREID_HOME}/jdk"
```

```
_JAVA_OPTIONS=
```

```
IBM_JAVA_OPTIONS=
```

```

export CATALINA_HOME JAVA_HOME _JAVA_OPTIONS IBM_JAVA_OPTIONS
case `uname` in
    Linux)
if [ -n "${LD_LIBRARY_PATH}" ]; then
    LD_LIBRARY_PATH="${SHAREID_HOME}/oblix/lib"
else
    LD_LIBRARY_PATH="${SHAREID_HOME}/oblix/
        lib:${LD_LIBRARY_PATH}"
fi
    export LD_LIBRARY_PATH
    echo Using LD_LIBRARY_PATH:          ${LD_LIBRARY_PATH}
;;
    SunOS)
if [ -n "${LD_LIBRARY_PATH}" ]; then
    LD_LIBRARY_PATH="${SM_LIB_PATH}:${SHAREID_HOME}/oblix/lib"
else
    LD_LIBRARY_PATH="${SM_LIB_PATH}:${SHAREID_HOME}/oblix/
        lib:${LD_LIBRARY_PATH}"
fi
    export LD_LIBRARY_PATH
    echo Using LD_LIBRARY_PATH:          ${LD_LIBRARY_PATH}
;;
    AIX)
if [ -n "${LIBPATH}" ]; then
    LIBPATH="${SHAREID_HOME}/oblix/lib"
else
    LIBPATH="${SHAREID_HOME}/oblix/lib:${LIBPATH}"
fi
    export LIBPATH
    echo Using LIBPATH:                  ${LIBPATH}
;;
    HP-UX)
if [ -n "${SHLIB_PATH}" ]; then
    SHLIB_PATH="${SHAREID_HOME}/oblix/lib"
else
    SHLIB_PATH="${SHAREID_HOME}/oblix/lib:${SHLIB_PATH}"
fi
    export SHLIB_PATH
    echo Using SHLIB_PATH:              ${SHLIB_PATH}
;;
    *)
esac

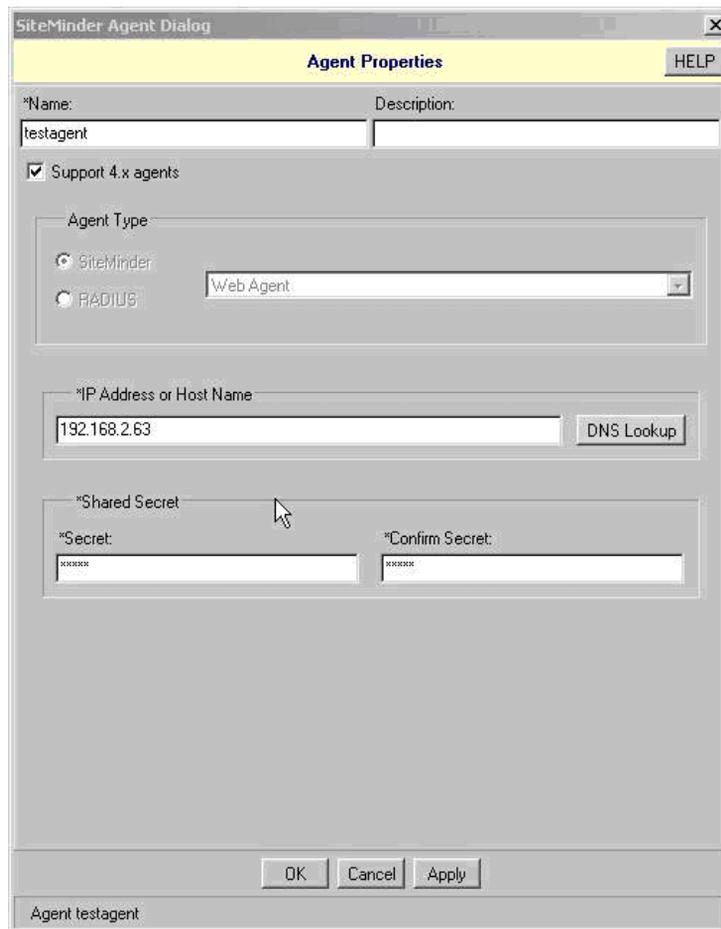
```

Create a SiteMinder WebAgent Identity

SiteMinder Agents control access to protected resources. An Agent grants or denies access by enforcing policies defined through the Policy Server. These policies govern the level of access and which resources can be accessed by a user.

1. In the SiteMinder Administration Console's Agent Configuration dialog, select the System > System Configuration > Agent > Create Agent option.
2. Specify entries in the Name, Description, Host Name, and Shared Secret. Fields. Also select the Enable 'Support 4.x agents' option.

The following screen display provides an example of the WebAgent Identify configuration:



The screenshot shows the 'SiteMinder Agent Dialog' window with the 'Agent Properties' tab selected. The window contains the following fields and options:

- Name:** testagent
- Description:** (empty)
- Support 4.x agents
- Agent Type:** SiteMinder (selected), RADIUS (unselected). A dropdown menu shows 'Web Agent'.
- *IP Address or Host Name:** 192.168.2.63. A 'DNS Lookup' button is next to the field.
- *Shared Secret:** (empty)
- *Secret:** (masked with asterisks)
- *Confirm Secret:** (masked with asterisks)

At the bottom of the dialog are 'OK', 'Cancel', and 'Apply' buttons. A status bar at the very bottom shows 'Agent testagent'.

Refer to the SiteMinder WebAgent Guide for more information and an in-depth description of WebAgent configuration.

3. Click on Apply and Ok.

Create and Configure a SiteMinder User Directory

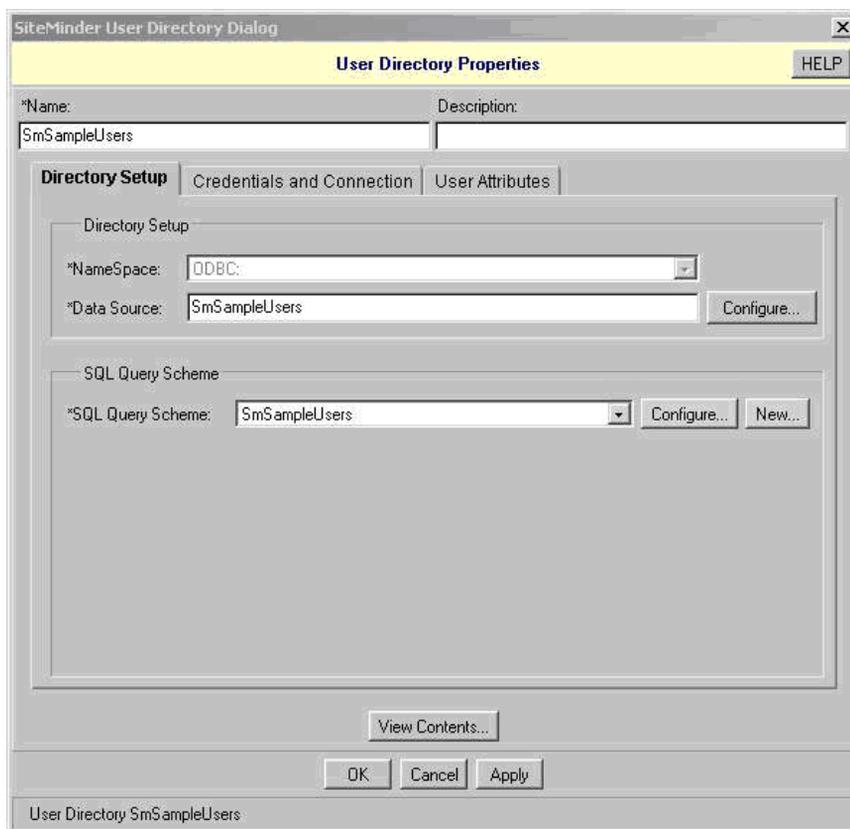
User directories store user data, including organizational information and credentials such as passwords. The Policy Server User Interface allows you to configure connections to existing user directories. The Policy Server uses these connections to verify user identities and retrieve user attributes contained in the directories.

1. In the SiteMinder Administration Console's User Directory Dialog box, select the System > System Configuration > User Dir > Create User Directory option.
2. Specify entries in the Name and Description fields.
3. On the Directory Setup tab, specify an entry in the NameSpace box.
4. Click the Configure button and configure the data source accordingly to specify the location of the user directory repository (LDAP, ODBC, etc.).

Note: The following steps show SiteMinder configuration of a user directory repository provided by an ODBC database (for example, Microsoft Access or SQL Server). Refer to the SiteMinder documentation for additional details and options available when setting up the data source for the user directory repository.

5. In the SQL Query Schema section of the same tab, configure the SQL Query Scheme used.

The following screen display shows an example of the user directory configuration:



Refer to the SiteMinder Policy Design documentation for more information on policy design and configuration options available.

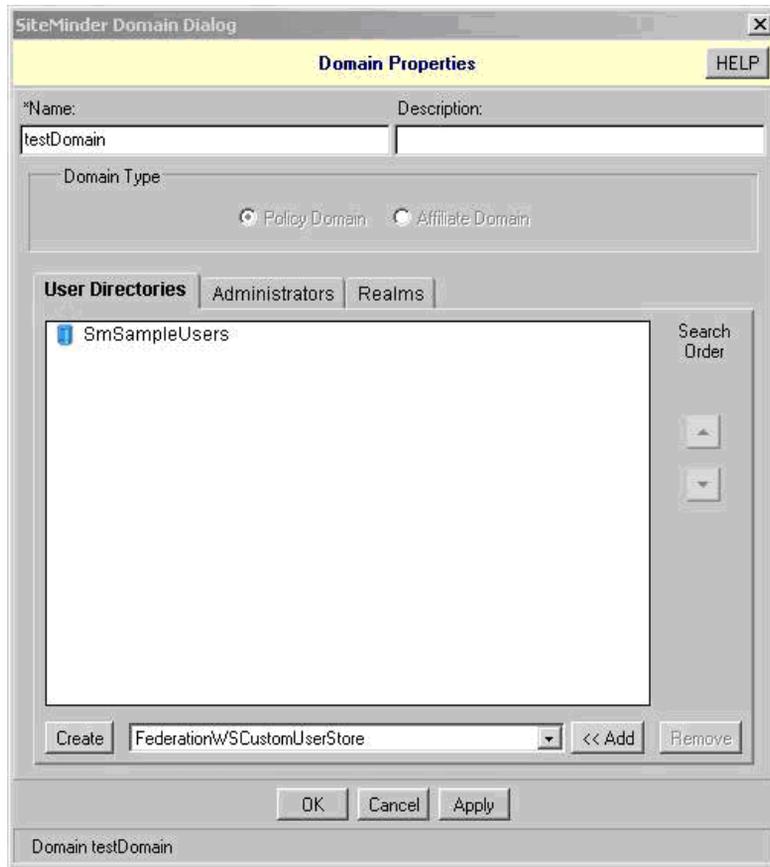
Create a SiteMinder Domain

A policy domain is a logical grouping of resources associated with one or more user directories. The resources in a policy domain can be grouped in one or more realms. A realm is a set of resources with a common security (authentication) requirement. Access to resources is controlled by rules, which are associated with the realm that contains the resource.

1. In the SiteMinder Administration Console's domain dialog box, select the System > System Configuration > Domain > Create Domain option.
2. Specify entries in the Name and Description fields.
3. Add the User Directory created in the previous step.

4. Click Apply and then Ok.

The following screen display shows an example of the SiteMinder domain configuration:



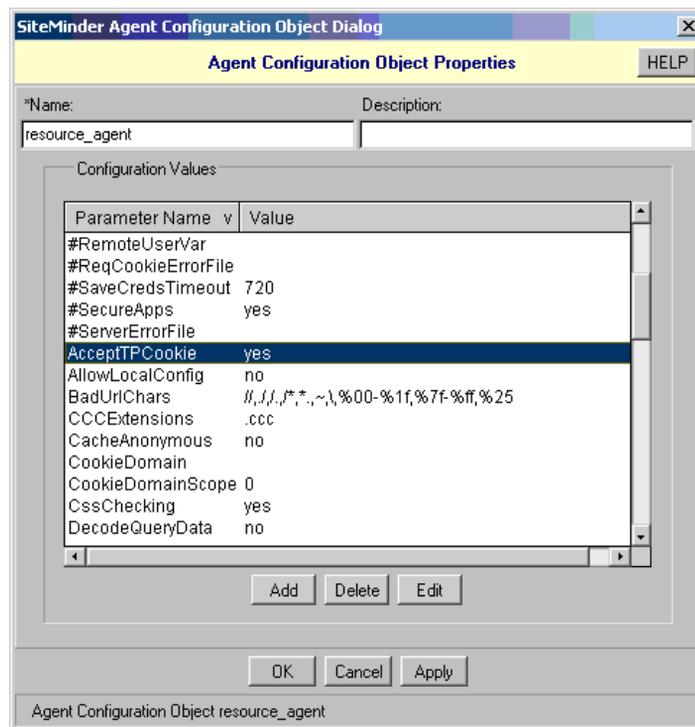
Refer to the SiteMinder Policy Design documentation for more information on available domain configuration options.

Changing Agent Cookie Settings (COREid Federation Destination)

In the SiteMinder Administration Console's Agent Configuration Object dialog, you need to set the value of the AcceptTPCookie property to Yes and also set the AllowLocalConfig property to No.

Note: This step is only required when configuring SiteMinder to be used for a COREid Federation destination domain.

The following screen display shows the location of the parameters in the SiteMinder Agent Configuration Object dialog box:



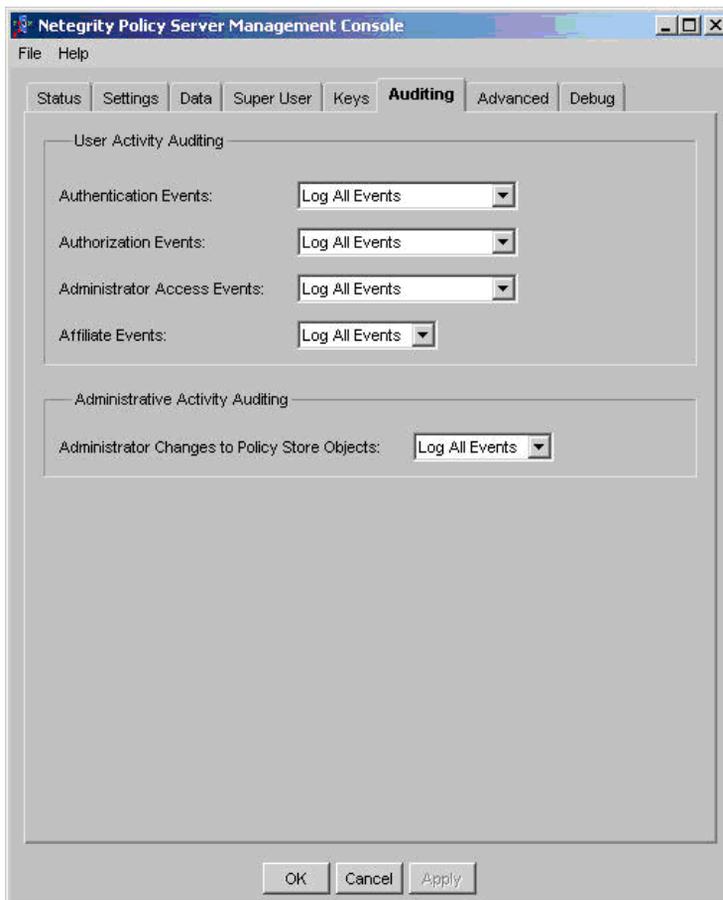
The relevant entries are:

- AcceptTPCookie
- DefaultAgentName
- DefaultPassword
- Logging related entries if logging is enabled

Configure and Enable SiteMinder Logs

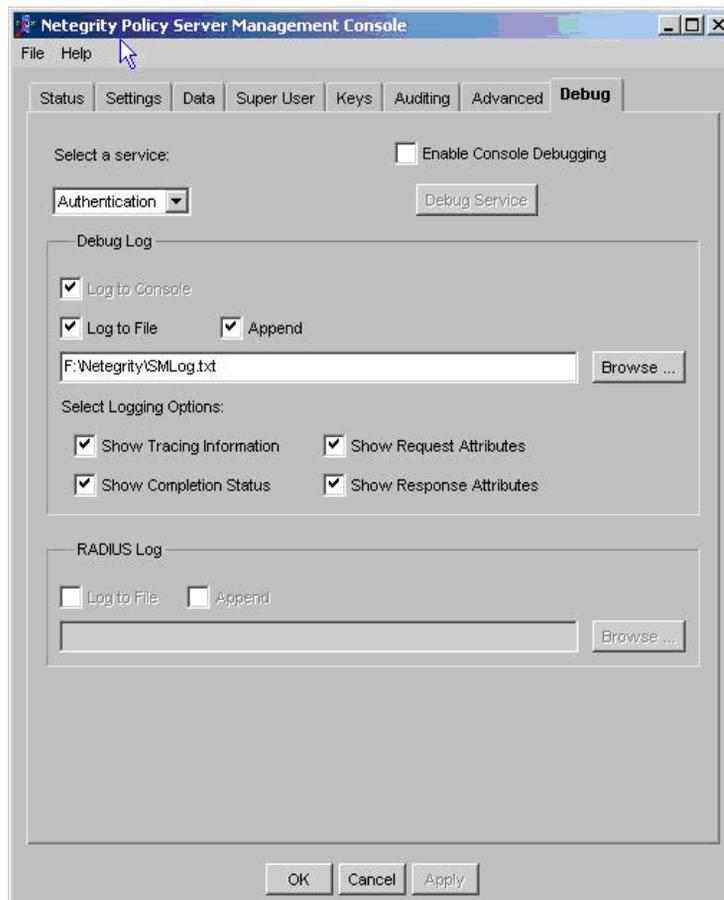
Having log files available is often useful in situations where you need to troubleshoot a problem or monitor behavior. To configure SiteMinder logs:

1. From the Start menu, go to Netegrity Policy Server Management Console.
2. Click on Auditing tab and configure desired log levels as shown in the following screen display:



3. Click on the Debug tab.
4. Select the Log to File option.

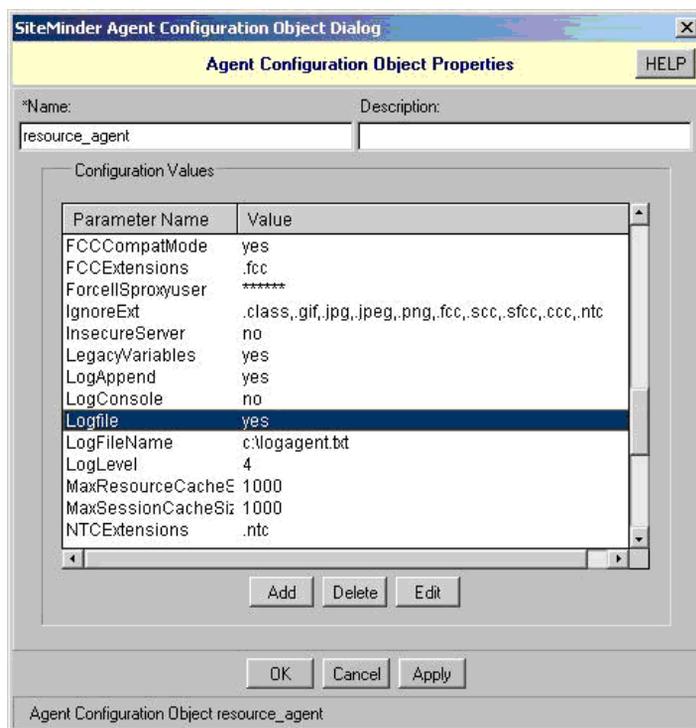
5. Select Append and then enter the log file name as shown in the following screen display:



6. In the SiteMinder Administration Console's Agent Configuration Object dialog, set the value of the Logfile property to Yes to enable SiteMinder WebAgent logs.

7. Enter the log file name (for example: c:\Logagent.txt) and set the log level to 4 (to obtain detailed logs).

The following screen display shows the log file configuration settings in the dialog box:



Refer to the SiteMinder documentation for more information on available logging configuration options.

Configuring SiteMinder for Operation as a COREid Federation Domain

Following SiteMinder “back-end” configuration, you can use the COREid Federation Administration Console to set configuration options for SiteMinder used with COREid Federation at either a source domain, destination domain, or both. COREid Federation Administration Console configuration steps you need to perform are the following:

For COREid Federation source domains:

- Configure a COREid Federation IdMBridge for SiteMinder.

- Configure COREid Federation login to SiteMinder, choosing from Basic, Form, or External Credential authentication methods, and add an assertion profile
- Configure an Assertion Profile, including settings such as issuer, subject name, subject format, and user attribute for subject.
- Configure a Destination Domain and enter details such as domain name, domain host name, SSL and client certificate authentication ports, etc.

For COREid Federation destination domains:

- Configure a COREid Federation IdMBridge for SiteMinder.
- Configure Destination Mappings, including settings such as mapping name, searchbase and person object class, and assertion attributes.
- Configure a Source Domain and enter details such as domain name, domain host name, SSL and client certificate authentication ports, etc.

For more information on source domain configuration using the COREid Federation Administration Console, see “Configuring Source Domains” on page 95. For more information on destination domain configuration, see “Configuring Destination Domains” on page 169.

Following COREid Federation configuration from the Administration Console, you can then go back to your SiteMinder console and check that your SiteMinder installation is configured correctly. The following section provides verification steps for both source and destination domains to confirm your COREid Federation SiteMinder configuration is set correctly after performing COREid Federation Administration Console configuration.

SiteMinder Verification Steps Following COREid Federation Configuration

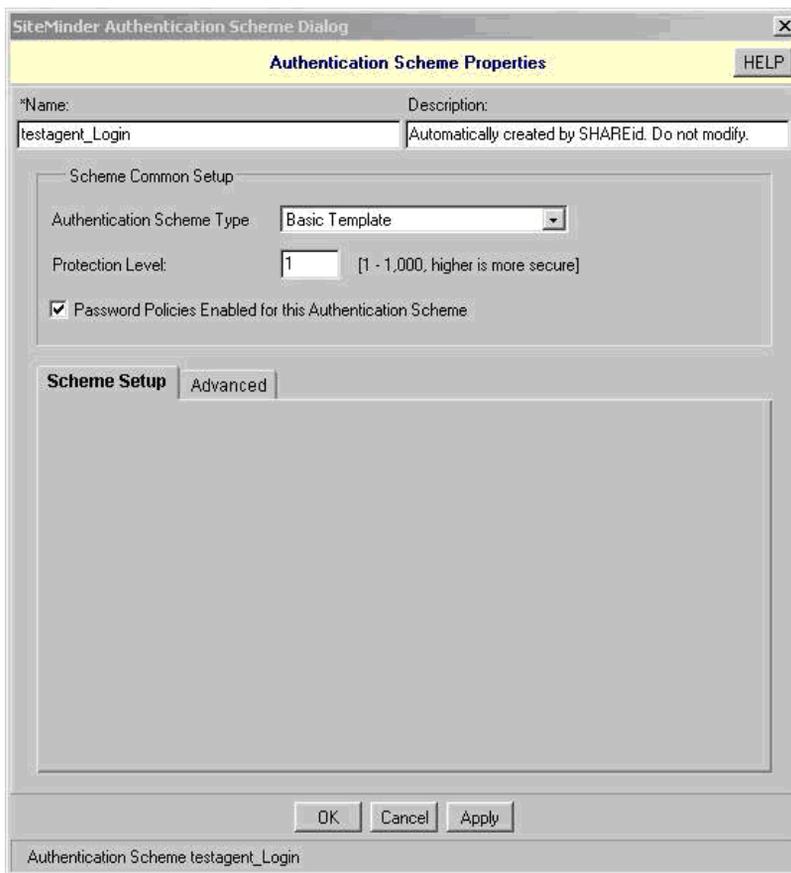
During COREid Federation configuration of a source or destination domain using SiteMinder, the COREid Federation Server creates a number of different policy objects on the SiteMinder Policy Server. So, to verify successful configuration, you can check for the creation of these objects to confirm your SiteMinder configuration was completed successfully. The following sections describe verification steps you can perform for configuration of SiteMinder for either source or destination domains.

SiteMinder Configuration Verification for Source Domains

The following steps assume you created a “testagent” WebAgent Identifier prior to using the COREid Federation Administration Console to configure a source domain. (See “Create a SiteMinder WebAgent Identity” on page 305):

1. From the SiteMinder Administration Console, open the Authentication Scheme Dialog box to check for the creation of a testagent_login authentication scheme and view its properties.

The following dialog box shows the properties you should see for the testagent_login authentication scheme:

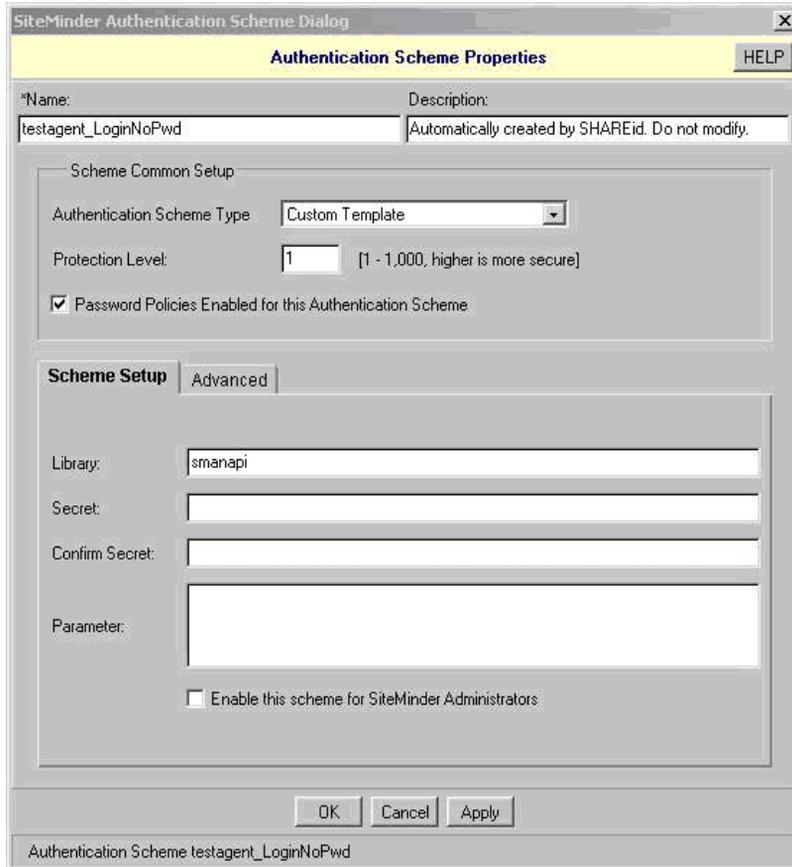


The following fields should be set:

- Name: Agent Name_Login (for example: testagent_login)
- Description: Automatically created by COREid Federation. Do not modify.
- Authentication Scheme Type: Basic

- Protection Level: 1
2. Check for the creation of a testagent_loginNoPwd authentication scheme and view its properties.

The following dialog box shows the properties you should see for the testagent_loginNoPwd authentication scheme:



The following fields should be set:

- Name: Agent Name_LoginNoPwd (for example: testagent_LoginNoPwd)
- Description: Automatically created by COREid Federation. Do not modify.
- Authentication Scheme Type: Custom Template
- Protection Level: 1
- Library: smanapi

3. From the SiteMinder Administration Console, open the SiteMinder Realm Dialog box to view properties for the testagent_login authentication scheme.

The following dialog box shows the properties you should see for the testagent_login authentication scheme realm:

The screenshot shows the 'SiteMinder Realm Dialog' window with the 'Realm Properties' tab selected. The dialog contains the following fields and controls:

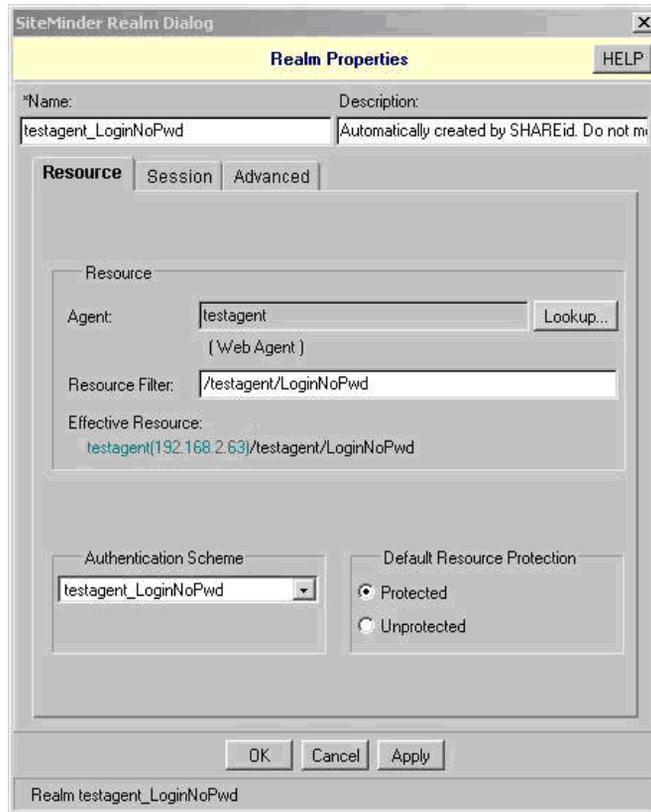
- Name:** testagent_login
- Description:** Automatically created by SHARId. Do not m...
- Resource:** (Selected tab)
- Agent:** testagent (Web Agent) with a 'Lookup...' button.
- Resource Filter:** /testagent/Login
- Effective Resource:** testagent(192.168.2.63)/testagent/Login
- Authentication Scheme:** testagent_login (dropdown menu)
- Default Resource Protection:** Protected (radio button selected), Unprotected (radio button unselected)
- Buttons:** OK, Cancel, Apply
- Footer:** Realm testagent_login

The following fields should be set:

- Name: Agent Name_Login (for example: testagent_login)
- Description: Automatically created by COREid Federation. Do not modify.
- Agent: testagent
- Resource Filter: /testagent/Login
- Authentication Scheme: testagent_login
- Default Resource Protection: Protected

4. Similarly, open the SiteMinder Realm Dialog box to view properties for the testagent_login_noPwd authentication scheme.

The following dialog box shows the properties you should see for the testagent_login_noPwd authentication scheme realm:

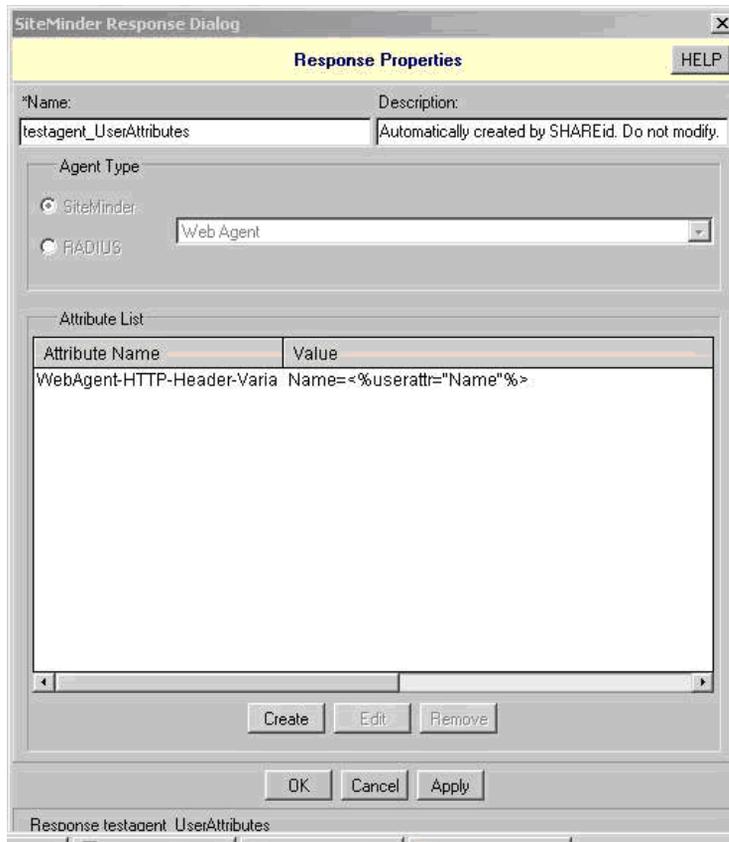


The following fields should be set:

- Name: Agent Name_LoginNoPwd (for example: testagent_LoginNoPwd)
- Description: Automatically created by COREid Federation. Do not modify.
- Agent: testagent
- Resource Filter: /testagent/LoginNoPwd
- Authentication Scheme: testagent_LoginNoPwd
- Default Resource Protection: Protected

5. From the SiteMinder Administration Console, open the SiteMinder Response Dialog box to view properties for the testagent_UserAttributes response.

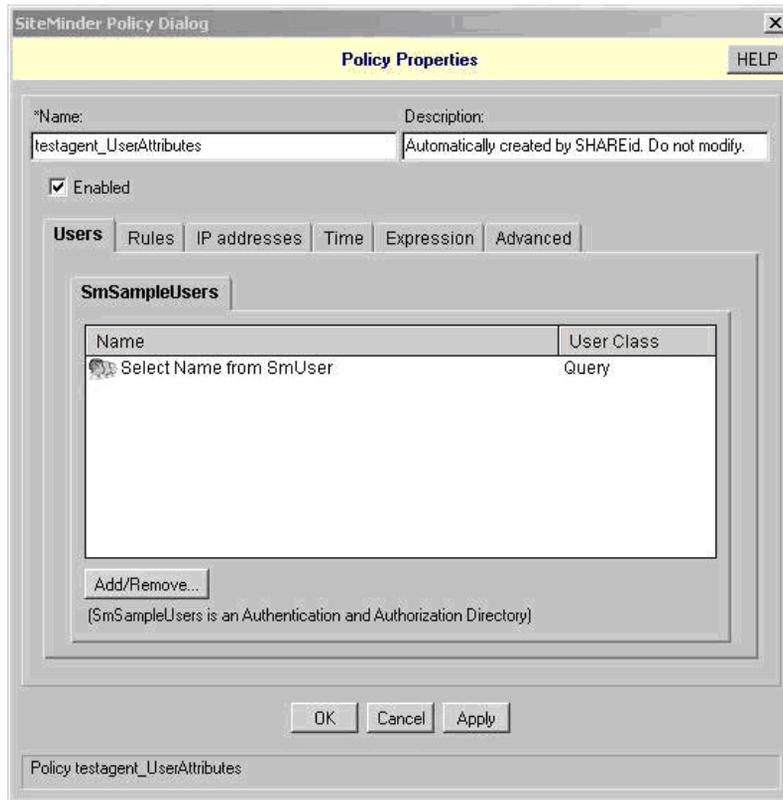
The following dialog box shows the properties you should see for this response:



The following fields should be set:

- Name: Agent Name_UserAttributes (for example: testagent_UserAttributes)
- Description: Automatically created by COREid Federation. Do not modify.
- Agent Type: Web Agent
- Attribute Name: WebAgent-Header-Variable
- Value: Should be the value that is specified in Attribute Mapping. For example, if 'Name' is the attribute being used, the value will be Name=<%userattr="Name"%>

The following dialog shows the filter that needs to be defined on the user tab on the testagent_UserAttributes page:



The following fields should be set:

- Name: Agent Name_UserAttributes (for example: testagent_UserAttributes)
- Description: Automatically created by COREid Federation. Do not modify
- 'Enabled' should be checked.
- Users tab: Select <Attribute used in Assertion Mapping> from <User Directory Name>. for example, Select Name from SmUser

On the Rules tab, the following fields should be set:

- Rule: Agent Name_Login (testagent_Login)
- Realm: Agent Name_Login (testagent_Login)
- Response: Agent Name_UserAttributes (testagent_UserAttributes)

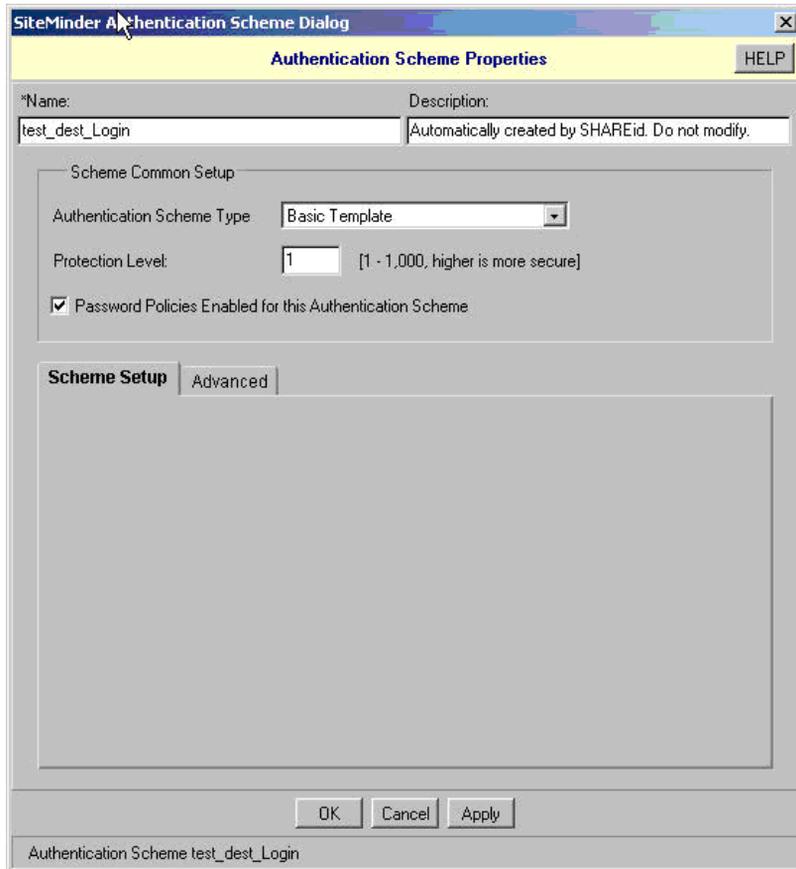
SiteMinder Configuration Verification for Destination Domains

The following steps assume you created a “test_dest_Login” WebAgent Identifier prior to using the COREid Federation Administration Console to configure a source domain. (See “Create a SiteMinder WebAgent Identity” on page 305).

Note: The following steps show verification of SiteMinder configuration of a user directory repository provided by an ODBC database (for example, Microsoft Access or SQL Server), so queries are shown in some of the screen displays are configured accordingly.

1. From the SiteMinder Administration Console, open the Authentication Scheme Dialog box to check for the creation of the test_dest_login authentication scheme and view its properties.

The following dialog box shows the properties you should see for the test_dest_login authentication scheme:



The following fields should be set:

- Name: Agent Name_Login (for example: test_dest_Login)
 - Description: Automatically created by COREid Federation. Do not modify.
 - Authentication Scheme Type: Basic Template
 - Protection Level: 1
 - Password policies enabled for this authentication scheme: Selected
2. Check for the creation of a test_dest_loginNoPwd authentication scheme and view its properties.

The following dialog box shows the properties you should see for the test_dest_loginNoPwd authentication scheme:

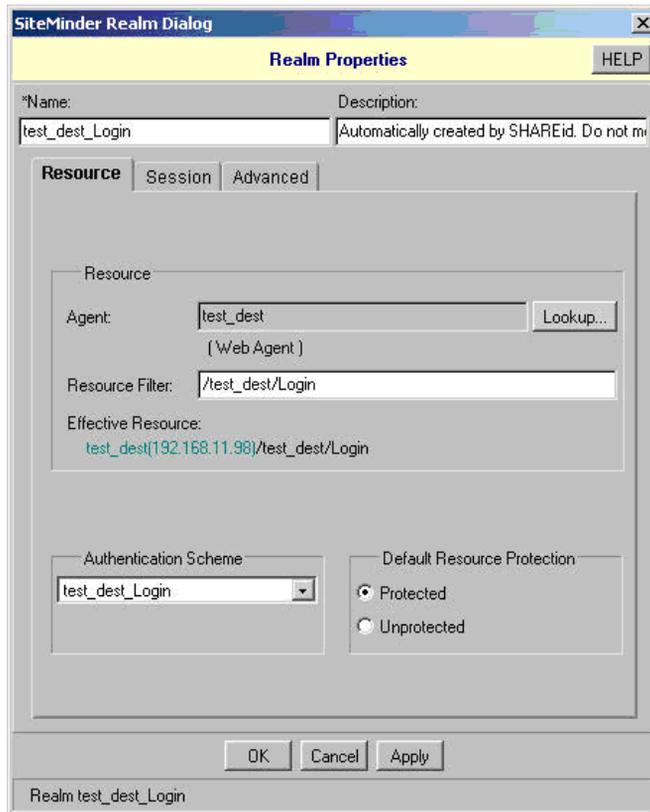
The screenshot shows the 'Authentication Scheme Properties' dialog box for the 'test_dest_LoginNoPwd' scheme. The 'Scheme Common Setup' section is active, showing 'Authentication Scheme Type' set to 'Custom Template' and 'Protection Level' set to '5'. The 'Password Policies Enabled' checkbox is unchecked. The 'Scheme Setup' section is also visible, with 'Library' set to 'smanapi' and other fields empty. The 'Enable this scheme for SiteMinder Administrators' checkbox is unchecked. The dialog has 'OK', 'Cancel', and 'Apply' buttons at the bottom.

The following fields should be set:

- Name: Agent Name_LoginNoPwd (for example: test_dest_LoginNoPwd)
- Description: Automatically created by COREid Federation. Do not modify.
- Authentication Scheme Type: Custom Template
- Protection Level: 5

- Password policies enabled for this authentication scheme: Not Selected
 - Library: smanapi
3. From the SiteMinder Administration Console, open the SiteMinder Realm Dialog box to view properties for the test_dest_login authentication scheme.

The following dialog box shows the properties you should see for the test_dest_login authentication scheme realm:

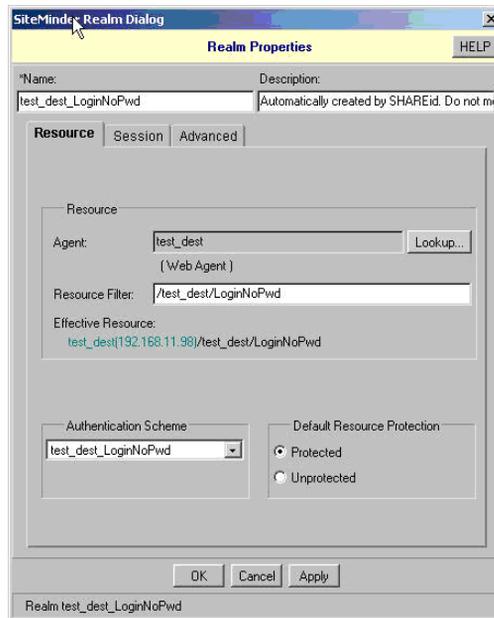


The following fields should be set:

- Name: Agent Name_Login (for example: test_dest_Login)
- Description: Automatically created by COREid Federation. Do not modify.
- Agent: test_dest
- Resource Filter: /test_dest/Login
- Authentication Scheme: test_dest_Login
- Default Resource Protection: Protected

4. Similarly, open the SiteMinder Realm Dialog box to view properties for the test_dest_login_noPwd authentication scheme.

The following dialog box shows the properties you should see for the test_dest_login_noPwd authentication scheme realm:



The following fields should be set:

- Name: Agent Name_LoginNoPwd (for example: test_dest_LoginNoPwd)
- Description: Automatically created by COREid Federation. Do not modify.
- Agent: test_dest
- Resource Filter: /test_dest/LoginNoPwd
- Authentication Scheme: test_dest_LoginNoPwd
- Default Resource Protection: Protected

- From the SiteMinder Administration Console, open the SiteMinder Response Dialog box to view properties for the test_dest_UserAttributes response.

The following dialog box shows the properties you should see for this response:

The screenshot shows the 'SiteMinder Response Dialog' window with the 'Response Properties' tab selected. The dialog contains the following fields and controls:

- *Name:** test_dest_UserAttributes
- Description:** Automatically created by SHAREid. Do not modify.
- Agent Type:** SiteMinder (selected), Web Agent (dropdown menu)
- Attribute List:** A table with the following content:

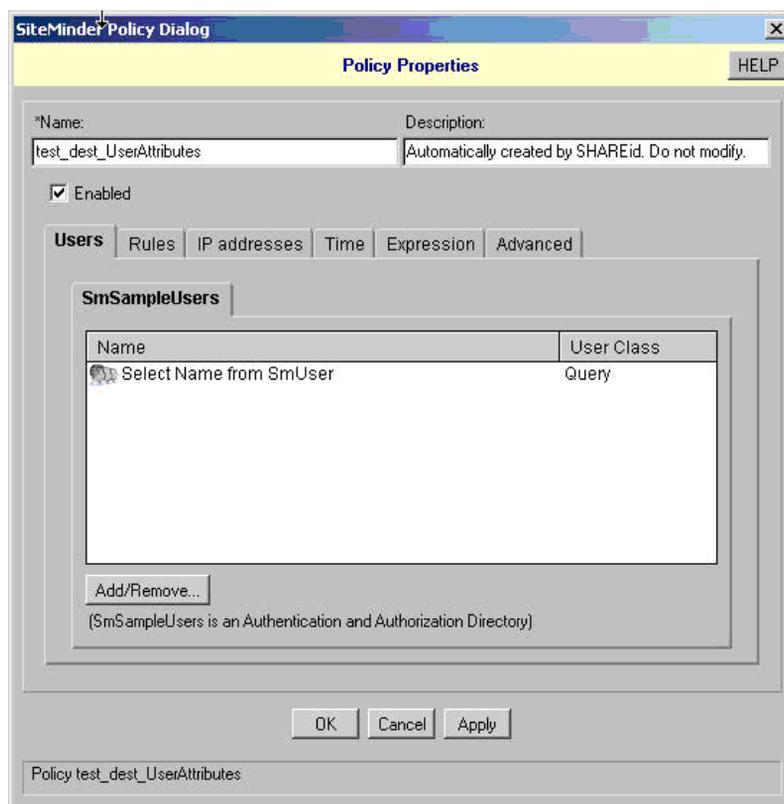
Attribute Name	Value
WebAgent-HTTP-Header-Varia	Name=<%userattr="Name"%>
- Buttons:** Create, Edit, Remove, OK, Cancel, Apply

At the bottom of the dialog, it says 'Response test_dest_UserAttributes'.

The following fields should be set:

- Name: Agent Name_UserAttributes (for example: test_dest_UserAttributes)
- Description: Automatically created by COREid Federation. Do not modify.
- Agent Type: Web Agent
- Attribute Name: WebAgent-Header-Variable
- Value: Should be the value that is specified in Attribute Mapping. For example, if 'Name' is the attribute being used, the value will be: Name=<%userattr="Name"%>

The following dialog shows the filter that needs to be defined on the user tab on the test_dest_UserAttributes page:



The following fields should be set:

- Name: Agent Name_UserAttributes (for example: test_dest_UserAttributes)
- Description: Automatically created by COREid Federation. Do not modify
- 'Enabled' should be checked.
- Users tab: Select <Attribute used in Assertion Mapping> from <User Directory Name>, for example, Select Name from SmUser

On the Rules tab, the following fields should be set:

- Rule: Agent Name_Login (test_dest_Login)
- Realm: Agent Name_Login (test_dest_Login)
- Response: Agent Name_UserAttributes (test_dest_UserAttributes)

C Troubleshooting and Optimizing COREid Federation Performance

This appendix provides information on troubleshooting possible COREid Federation issues and problems and also provides general methods and tips for optimizing COREid Federation performance.

Topics covered in this appendix are the following:

- “Troubleshooting COREid Federation” on page 328
- “Optimizing COREid Federation Performance” on page 334

Also check the Release Notes which provides up-to-the minute details about the latest release of COREid Federation.

Troubleshooting COREid Federation

When troubleshooting COREid Federation issues, it is helpful to examine the log files the COREid Federation Server creates to look for specific errors or indications as to why particular operations are failing. Log files from the Tomcat Server running with COREid Federation are located in the following location:

```
SHAREid_Install_Dir/logs/localhost_log.date.txt
```

To look for startup errors, you can look at stdout and stderr logs. When the COREid Federation server is run as a service, stdout and stderr are written to the following log file:

```
SHAREid_Install_Dir/logs/catalina.out
```

To create a dump of request headers and cookies, go to the following file:

```
SHAREid_Install_Dir/conf/server.xml
```

where *SHAREid_Install_Dir* is the directory where COREid Federation is installed. In this file, uncomment the following line:

```
<Valve classname="org.apache.catalina.valves.RequestDumperValve"/>
```

Possible Issues or Problems

Problem—COREid Federation hangs during login.

Solution—This may be caused if you configure the COREid Federation connection URL to use SSL on a non-SSL port, for example:

```
ldaps://directoryhost.example.com:398
```

instead of

```
ldap://directoryhost.example.com:389
```

To correct this problem, you need to correct the connection URL from the Administration Console and restart the COREid Federation Server.

Problem—A Not Found error occurs when trying to access the COREid Federation Administration Console through the COREid Federation proxy.

Solution—Access to the COREid Federation Administration Console is blocked by the COREid Federation proxy by default for security reasons. It can be unblocked by modifying the ProxyPass and ProxyPassReverse directives in httpd.conf and ssl.conf files of the proxy server. Note that access to the COREid Federation Administration Console through the proxy is not certified by Oracle.

Problem—After installing the COREid Federation proxy on Linux or Solaris, you start the proxy using `Proxy_Install_Dir/bin/SHAREidProxy` start, and get the following error:

```
Syntax error on line 115 of /opt/oblix/SHAREid-Proxy/conf/ssl.conf: SSLCertificateFile: file '/opt/oblix/SHAREid-Proxy/conf/ssl.cert/server.crt' does not exist or is empty
```

Solution—You have not yet created the certificate for use with SSL. See “Configuring Certificates for the Proxy” on page 270 for details. If you do not plan to use SSL, comment out this line in `httpd.conf`:

```
include conf/ssl.conf
```

Problem—After re-installing COREid Federation, the host identifiers and policies are incorrect.

Solution—Delete all the COREid Federation policies and schemes and restart the COREid Federation server.

Problem—Cannot start the COREid Federation Administration Console or COREid Federation server.

Solution—If you do not supply a fully qualified host name during installation, COREid Federation will not issue an error. However, not supplying a fully qualified host name may result in errors on startup. You may need to re-install COREid Federation.

Problem—Destination resource is not protected by a COREid Federation policy.

Solution—Check the definitions of your host IDs in the COREid Access System. Add the required host ID if it is missing, and restart the COREid Federation server.

Problem—Error: “Bad Gateway”. The proxy server received an invalid response from an upstream server. You may have started the COREid Federation proxy server but not the COREid Federation server, and then attempted to access COREid Federation.

Solution—Start the COREid Federation server.

Problem—Error: "A local user session could not be created for the assertion" may appear. This error appears if the destination site is not able to perform a mapping to a local user.

Solution—The most common reason for a mapping failure is that there is no user at the destination site who has an attribute value that matches the attribute specified in the assertion mapping. To remedy this, add a user profile that has the correct attribute value. There are two other possible reasons for mapping failures: the access server may not be able to connect to the user directory, or the authentication scheme for the protected resource may have an invalid `ObMappingFilter` parameter for its `credential_mapping` plugin.

Problem—SSO not working when the Destination (Target URL) user session times out. This problem occurs when using the attribute sharing profile to access resources protected by COREid and the COREid WebGate cookie is not set to match the COREid Federation destination domain cookie setting. A user enters valid credentials in a web login form and is successfully transferred to the Target URL. However, after the user session times out, if the user refreshes the page or attempts to re-access the information, the user is asked to re-authenticate.

Solution—In the Access System Console, go to the WebGate configuration page for the WebGate serving protected resources, and set the Primary HTTP Cookie Domain to the same value as the cookie domain defined on the COREid Federation Login configuration page.

Problem—Error:

```
verify error:num=19:self signed certificate in
certificate chain.
```

Solution—In *Proxy_Install_Dir/conf/ssl.conf*, set the SSLVerify directive to optional_no_ca.

Problem—Error during transfer, “An assertion cannot be generated for the logged in user.”

Solution—Look in the source site’s log file. Possible causes are that the assertion profile is incorrect or an attribute you need is missing.

Problem—Errors: One or more of the following errors are displayed:

```
10 Jan 2004 17:17:05 - ERROR - [http8113-Processor3] -
RESPONDER: ERROR User directory entry for CN=oblix
Admin,CN=Users,DC=c840-2003,DC=cameron,DC=oblix,DC=net
does not have the SubjectName attribute name of attribute.
10 Jan 2004 17:17:05 - ERROR - [http8113-Processor3] -
TRANSFER: ERROR An assertion could not be generated for
the logged in user
"Entry doesn't have a (name of attribute) attribute" is
displayed.
```

Solution—Flush the Access Server cache.

Problem: Exception “broken pipe”. In client-certificate mode, you get an error message in the log file with the words “broken pipe”. The error message is also accompanied by an exception. The CA for the certificate from the destination COREid Federation server may not be known to the source proxy server.

Solution: Make sure that the CA for the certificate from the destination COREid Federation server is specified in the ca-bundle.crt file of the source proxy server.

Problem: Exception “java.net.ConnectException: Connection refused”. The exception appears when stopping the COREid Federation server that is already stopped.

Solution: This behavior does not affect COREid Federation functionality in any way. The exception simply occurs because you tried to stop a COREid Federation server that is already stopped.

Problem—Proxy server configuration requires verification.

Solution—You can test the proxy SSL ports to determine if they are working properly:

1. Run the following command:

```
openssl s_client -connect host:port
```

where *host* and *port* are the machine name and port of the proxy SSL ports.

This command displays information about the host, including the certificate being used.

2. Look for the following response:

The response:

```
verify return:1
```

indicates success. The response:

```
verify return:0
```

indicates an error.

Problem—The server does not come up.

Solution—A probable cause could be a configuration error in the certificate store.

To check the configuration on Windows

1. If you started the server as a Windows service, check the log file in:
`SHAREid_Install_Dir/logs/stderr.log`
2. If you started the server by selecting Start COREid Federation server from the Program Menu, start it from the command prompt instead:
 - a) Open a command prompt window.
 - b) Go to `SHAREid_Install_Dir/bin`.
 - c) Run:

```
catalina start
```
3. You should see the exception shown in Figure 5 in the command window.

To check the configuration on Linux or Solaris

1. Open the log file *SHAREid_Install_Dir/logs/stderr.log*
2. If you see the following exception, verify if the keystore path and password used for SSL which is configured in *SHAREid_Install_Dir/conf/server.xml* is correct.

Note that the key password and store password cannot be different for the keystore used for SSL.

Figure 5 Keystore Exception

```
[ERROR] Http11Protocol - -Error initializing endpoint <java.io.IOException:
Keystore was tampered with, or password was incorrect>java.io.IOException:
Keystore was tampered with, or password was incorrect
at sun.security.provider.JavaKeyStore.engineLoad(JavaKeyStore.java:737)
at java.security.KeyStore.load(KeyStore.java:608)
at org.apache.tomcat.util.net.jsse.JSSESocketFactory.getStore
(JSSESocketFactory.java:295)
at org.apache.tomcat.util.net.jsse.JSSESocketFactory.getKeyStore
(JSSESocketFactory.java:259)
at org.apache.tomcat.util.net.jsse.JSSE13SocketFactory.init
(JSSE13SocketFactory.java:129)
at org.apache.tomcat.util.net.jsse.JSSESocketFactory.createSocket
(JSSESocketFactory.java:127)
at org.apache.tomcat.util.net.PoolTcpEndpoint.initEndpoint
(PoolTcpEndpoint.java:281)
at org.apache.coyote.http11.Http11Protocol.init(Http11Protocol.java:184)
at org.apache.coyote.tomcat4.CoyoteConnector.initialize
(CoyoteConnector.java:1173)
at org.apache.catalina.core.StandardService.initialize
(StandardService.java:579)
at org.apache.catalina.core.StandardServer.initialize(StandardServer.java:2246)
at org.apache.catalina.startup.CatalinaService.load(CatalinaService.java:236)
at org.apache.catalina.startup.CatalinaService.load(CatalinaService.java:258)
at java.lang.reflect.Method.invoke(Native Method)
at org.apache.catalina.startup.BootstrapService.init(BootstrapService.java:231)
at org.apache.catalina.startup.BootstrapService.main(BootstrapService.java:297)
```

Problem—The HTTP port works (for example, <http://host.company.com:8101/shareid>) but the HTTPS port (for example, <https://host.company.com:8113/shareid>) receives a “Cannot find server” error. An exception “Keystore was tampered with, or password was incorrect” appears in one of the following places:

- If using COREid Federation as a Windows service, the Tomcat stderr log in *SHAREid_Install_Dir/logs/stderr.log*

- If running COREid Federation from a command prompt, the runtime window.

Solution—If the Tomcat server.xml configuration file does not have the correct password for its keystore, Tomcat will still start up, but its HTTPS ports will not be initialized. Be sure that the keystore password for server.xml matches the keystore password configuration in the COREid Federation Administration Console.

Problem—The following page title appears in the Setup Wizard where IdMBridge setup is expected:

Step 1 of 4 Destination Setup

Solution—Refresh the window.

Problem—Trying to access a site that is protected by the COREid Access Server results in a 404 browser error.

Solution—Be sure that you have an authorization expression in the policy that protects the resource you are trying to access.

Problem—When pointing to the ObSAMLTransferService on the source site you attempt to log in, but get a Tomcat page that says “HTTP Status 401 - This request requires HTTP authentication.”

Solution—Your user ID may be invalid.

Problem—You want to use different passwords for the certificate store and signing key.

Solution—This can be done manually by editing the fields KeystorePassword and SigningKeyPassword in shareid-config.xml. Enter the values in plain text. When the COREid Federation server is restarted, it will write back the passwords with encrypted values in the configuration file.

Problem—You need to retrieve the port numbers that were chosen during installation.

Solution—Do the following:

Finding COREid Federation Server ports

1. Open server.xml in the /conf directory.
2. Search for the strings that contain “<Connector.”
3. If the connector string does not have a CoyoteServerSocketFactory statement, the port listed on this connector is the open port.

The port number is located in the portNumber attribute of this element.

4. If the connector string contains a CoyoteServerSocketFactory statement with the statement ClientAuth=False, the port listed on this connector is the SSL port.

5. If the connector string contains a `CoyoteServerSocketFactory` statement with the statement `ClientAuth=True`, the port listed on this connector is the client certificate port.

Finding COREid Federation proxy ports

1. Open the COREid Federation proxy `httpd.conf` file in the `/conf` folder.
2. Search for “Listen”. The directive `Listen portNumber` provides the number of the open port.
3. In the listen port, be sure that it has a value of `Open`.
4. Open the `ssl.conf` file.

In this file, the first `Listen portNumber` directive provides the SSL port. The second port is the client certificate port. If you see the string `##REPLACE_SSL_PROXY_CLIENT_CERT_PORT`, the client certificate port has not been set up.

Optimizing COREid Federation Performance

The overall throughput and performance of COREid Federation in your network can vary widely based on a number of factors. These factors can range from the types of transactions your system is processing (e.g., using Artifact versus POST, or using certificates) to the effect of using individual components or elements in your system that are involved in processing a transaction (e.g., Firewalls, Proxies, LDAP Servers, and Identity Management Systems).

As an example, executing transactions using the SAML POST profile is generally faster than using the Artifact Profile, based on requiring fewer round trips between the source and destination domains. In addition, the LDAP directories, RDBMS, or Identity Management Systems you use can effect transaction processing times. Similarly, introducing firewalls and proxy servers and using SSL and client certificates for login and user authentication adds additional steps in the process. So, there are often trade-offs between the benefits various operations provide, and the effects they have on overall performance.

In designing a COREid Federation system to meet your requirements, you may first want to consult with your Oracle sales or support representative to help you with your COREid Federation planning and designing an overall COREid Federation system architecture. For example, in planning your COREid Federation installation, it may be helpful for you to estimate the expected number of users in your network, the type, number, and rate of transactions (COREid Federation requests and transfers) you anticipate you will need to process, and the transaction

rates or benchmarks that you expect COREid Federation to meet. Following that, you can more appropriately choose the network, security, protocols, transaction profiles, and platform you want to use in your COREid Federation installation and also more accurately determine the specific hardware, number and type of CPU, memory, and storage you'll need for your system.

In environments where you need to handle larger loads and need greater throughput, or you have specific requirements to provide continuous site operation, you can also consider scaling and load balancing multiple COREid Federation servers. You can then use one of any number of commercial load balancing solutions available to load balance COREid Federation Servers, provide backup, and failover protection of your site. See “Setting up COREid Federation Load Balancing and Failover” on page 215.

Tunable Parameters

The following information is for system administrators who are experienced with turning system performance.

RDBMS Parameters

Parameters you can specify in the configuration of databases both for the RDBMS IdMBridge and the COREid Federation Assertion Store database, if used, provide options to control database connection pool settings. If you have expertise with the particular database used with these COREid Federation features, you can optimize these settings for your particular environment. See “Configuring an RDBMS IdMBridge” on page 124 “Setting up a Common Assertion Store Database” on page 216, for more information.

JAVA Parameters

The following are Java Virtual Machine (JVM) -X parameters that you may wish to tune. This information assumes that you are using HotSpot VM for JDK 1.3.0.9.

Parameter	Description
-Xms<size>	Sets the initial Java heap size. The default is 128 MB.
-Xmx<size>	Sets the maximum Java heap size. The default is 512 MB.
-Xss<size>	Sets the Java thread stack size.

The listed java parameters are available in the <SHAREid InstallDir>/bin/catalina.bat file on Windows platforms and catalina.sh on Linux or Solaris platforms.

References

The following online publications may be of use when assessing system performance:

- <http://jakarta.apache.org/tomcat/faq/memory.html>
Discusses memory issues that you may encounter with the Tomcat servlet that COREid Federation uses.
- <http://jakarta.apache.org/tomcat/articles/performance.pdf>
Discusses strategies for enhancing Tomcat performance.

Index

A

- Administration Console
 - destination domain configuration 170, 177
 - source domain configuration 96, 112
- Artifact profile 33
 - authentication and certificates 260
 - basic over SSL authentication 264
 - certificate overview 246, 249
 - client certificate configuration 264
 - Receiver service 34
 - Requester component 33
 - Responder service 33
 - security requirements 71
 - Transfer service 33
- assertion
 - attribute statement 28, 30
 - attribute statement, example of 30
 - authentication statement 28
 - authentication statement, example of 29
 - authorization statement 28, 31
 - authorization statement, example of 31
 - conditions element 28
 - description of 24
 - elements and attributes 65
 - issuer attribute 28, 204
 - issuer statement 28
 - mapping attributes 182
 - mapping, process overview 66
 - mapping, task overview 67
 - mappings for attributes 66
 - namespace 28
 - optional attributes 69
 - sent from unauthorized domains 198
 - setting up common data store 216
 - using attribute sharing profile 221
- assertion mapping 179
- assertion profile 98, 160
 - choosing Artifact or POST 70
 - planning and creating 64
- assertion store database 216
- attribute mapping
 - SmartWalls 206
- attribute sharing profile 221
 - identify provider configuration 221
 - illustration 42
- attribute statement 28
 - example of 30
- attributes and elements 65
- audience 13
- audits and logs 277

- authenticating users 20
- authentication
 - Artifact basic over SSL 264
 - basic 161, 188
 - destination domain requester 188
 - for Artifact profile 260
 - requester 161
 - X.509 161, 188
- authentication scheme level 181, 182
- authentication statement 28
- authorization queries 150, 159, 192
- authorization statement 28, 31

B

- basic authentication 161
- basic login dialog 103, 135

C

- Cannot start COREid Federation error 329
- certificate
 - Artifact and POST overview 249
 - Artifact profiles 246
 - configuration for COREid FederationProxy 270
 - configuring 243
 - configuring for Artifact profile 264
 - configuring for LDAP 274
 - configuring for POST profile 254
 - COREid Federation default 247
 - keystore location 248
 - POST profiles 244
- certificate store 137, 244
- configuration
 - advanced topics 197
 - assertion mapping 179
 - assertion store database 216
 - audits and logging 277
 - certificate store 137
 - COREid Federation load balancing and failover 215
 - COREid Federation secure connections 244
 - COREid Federation steps 89
 - COREid IdMBridge 178
 - creating local identities 198
 - customizing error messages 198
 - destination domains 169, 172
 - destination domains, task overview 173
 - domains 147
 - IdMBridge 113, 178

- keys and certificates 243, 251
- keytool command 248
- LDAP IdMBridge 121
- MyDomain 148, 184
- password and cert encryption 196
- RDBMS IdMBridge 124
- redirection user login 198
- restricting user access 198
- setting the active IdMBridge 114
- SiteMinder 291, 301, 312
- SiteMinder IdMBridge 129
- source domain 95, 98
- source domain, task overview 98
- transport security connection type 118
- using X.509 attribute sharing profile 221
- contact information 15
- conventions 15
- COREid
 - documentation 14
 - IdMBridge configuration 115, 178
 - IdMBridge, transport security 59
 - policies 69
 - single sign-on cookie 63
 - user data repository 178
 - workflow 67
- COREid Federation
 - audits and logs 277
 - basics 17
 - before installing 80
 - certificate store 244
 - components 33
 - configuration steps 89
 - configuring secure connections 244
 - default certificate 247
 - deploying 91
 - features 19
 - installation 80, 81, 83
 - introduction 17
 - load balancing and failover 215
 - logging in on Linux or Solaris 87
 - logging in on Windows 87
 - login process 35
 - performance tuning 335
 - planning 45
 - ports 82, 157
 - purpose 18
 - Receiver service 34
 - Requester component 33
 - Responder service 33
 - SiteMinder configuration for domain 301, 312
 - SiteMinder IdMBridge 291
 - source authentication planning 61
 - starting on Linux or Solaris 87
 - starting on Windows 87
 - support for SAML 23
 - supported platforms 47
 - system requirements 47
 - Transfer service 33

- troubleshooting 327
- tunable parameters 335
- uninstalling 92
- Web services 33
- COREid Federation ports 191
- COREid Federation Proxy
 - certificate configuration 270
 - installation 81, 85
- cross-domain security
 - illustration of 18

D

- debug logs 279
- deploying COREid Federation 91
- destination domain
 - about configuring 172
 - accepting authorization queries 185
 - assertion mapping configuration 179
 - configuration 169, 177
 - configuration prerequisites 173
 - configuration task overview 173
 - configuration, using the setup wizard 175
 - configuring for source 98
 - definition 23
 - hostname 191
 - methods for configuration 174
 - password and cert encryption 196
 - supported profiles 186
- direct login 61
- documentation 14
- domain
 - configuring 147
 - verifying a user's domain 20

E

- error logs 279
- error messages
 - configuring 198
 - customizing 198
- error redirection 34, 150, 186
- error reporting 210
 - ObSAMLError servlet and template 211
- Error URL 186
- errors
 - at startup 328
 - error redirection scenario 213
 - redirecting errors back to the source domain 210
- external credentials 103, 127, 135, 220

H

- header variable
 - single sign-on 61

- host identifiers
 - problems with 329
- hostname 157
 - destination domain 191

I

- idle timeout 135
- IdMBridge 98
 - configuration 113, 178
 - COREid configuration 115
 - LDAP configuration 121
 - LDAP planning 52
 - RDBMS configuration 124
 - setting active 114
 - SiteMinder configuration 129
 - SiteMinder requirements 292
- information logs 279
- installing
 - COREid Federation 80, 81, 83
 - COREid Federation Proxy 81, 85
- introduction to COREid Federation 17
- Issuer attribute 28, 185, 204
- Issuer statement 204

J

- Java Virtual Machine 335

K

- keys
 - configuring 243, 282
 - maintaining 251
- keystore
 - changing location and passwords 273
 - command 248

L

- LDAP
 - certificates 274
 - filters 53
 - IdMBridge planning 52
- load balancing and failover 215
- local domain, see MyDomain 98
- logging in
 - COREid Federation on Linux or Solaris 87
 - COREid Federation on Windows 87
- login
 - basic 135
 - configuration 98, 133
 - customizing the login form 63, 198
 - direct 61
 - external credentials 103, 135

- presenting a login page 208
- process for COREid Federation 35
- redirecting a user for login 61, 198
- single sign-on 61
- using SmartMarks 201
- using the Artifact profile 37, 294, 295
- using the Attribute Sharing profile 38
- web form 135
- login form
 - customizing 63, 198
- logs and audit recording 277

M

- mapping
 - assertion 66, 179
 - assertion attributes 66
- MyDomain
 - configuration 148, 184
 - supported protocols 150, 185

N

- namespace
 - for an assertion 28, 69
- NetPoint, see COREid

O

- OASIS
 - description 23
- ObSAMLError Servlet 211
- optimizing performance 327, 334, 335
- Oracle contact information 15

P

- password
 - changing for keystore 273
- password and cert encryption 196
- performance tuning 327, 334, 335
- platforms for COREid Federation 47
- policies
 - created in COREid 69
 - problems with 329
- ports
 - COREid Federation 82, 157
- POST profile
 - and the Receiver service 34
 - and the Transfer service 33
 - certificate overview 244, 249
 - configuring certificates 254
 - login process illustration 35
 - security requirements 71
- profiles

- advantages and disadvantages 70
- assertion 160
- supported 151, 186
- protocols
 - supported for MyDomain 150, 185

R

- Receiver service
 - illustration 35
- Receiver URL 161, 188
- redirecting errors 213
- redirection 61
 - of unauthorized users 198
- Requester component 33
 - illustration 37
- Resource not protected error 329
- Responder service 33, 153
 - illustration 37, 299

S

- SAML
 - assertions 24
 - COREid Federation support 23
 - definition 23
 - how it works 23
 - transport protocol 33
- search base
 - LDAP filters 53
 - top directory node 182
- security
 - Artifact profile requirements 71
 - overview 281
 - POST requirements 71
- server does not start 331
- session timeout 135
- setup wizard 99, 174, 175
- single sign-on 61
 - based on a COREid cookie 63
 - based on a header variable 61
 - based on an LDAP cookie 63
- SiteMinder
 - components used with COREid Federation 294
 - configuration for COREid Federation 291, 301, 312
 - COREid Federation configuration, illustration of 293
 - destination process flow, illustration 299
 - requirements for use with COREid Federation 292
 - SDK 302
 - source domain, illustration 296
- SmartMaps 20
 - invoking IdentityXML 208
 - mapping user identities 208
 - using login page 208
- SmartMarks 20, 61
 - authentication scheme 200

- configuring as a destination domain 203
- configuring as a source domain 202
- handling unauthenticated users 199
- how it works 201
- login process 201
- transfer form login 201

- SmartWalls 20
 - attribute mapping 206
 - configuring by structuring the directory tree 207
 - configuring via an LDAP user attribute 207
 - restricting users to local domain 204

- SOAP
 - definition 33
 - description of 33
- source domain
 - configuration from the Admin Console 112, 177
 - configuration task overview 98
 - configuration, using the setup wizard 99
 - configuring 95, 98
 - definition 23
 - login configuration 133
 - methods for configuration 99
- starting
 - COREid Federation on Linux or Solaris 87
 - COREid Federation on Windows 87
- startup errors 328
- Subject field
 - in an attribute statement 30
 - in an authentication statement 29
- supported platforms 47
- supported protocols 150, 185
- system requirements 47

T

- timeouts
 - idle 135
 - session 135
- Tomcat logs 328
- top directory node 182
- Transfer form login 34, 201, 203
- Transfer query string 151
- Transfer Service 33
- Transfer service 33, 151, 159, 186, 193
 - illustration 35, 37
- Transfer service query string 151, 160, 186, 193
- Transfer URL 186
- transport security 59
 - configuration 118
- troubleshooting
 - COREid Federation 327
- typographical conventions 15

U

- uninstalling COREid Federation 92

- user access
 - restricting to one domain 198
- user identities
 - mapping using SmartMaps 208
- user redirection
 - configuring for a destination 203
- users
 - creating identities for source users 198
 - unauthenticated, handling 199

W

- Web browser profiles 32, 70
- Web login form 103, 126, 135, 220

X

- X.509 attribute sharing 221

