



Achieving Interoperability between Active Directory Federation Services and Shibboleth

A White Paper by Oxford Computer Group

Oxford Computer Group
Published: February, 2007

www.oxfordcomputergroup.com

Abstract

Active Directory Federation Services (ADFS) and Shibboleth are two federation technologies that allow web browser users in one organization to access web-based applications in another. Shibboleth is open source software developed by Internet2, a US-based advanced networking consortium, and ADFS is a component of Microsoft's Windows Server 2003 R2 system.

Both Microsoft and Internet2 have announced that interoperability between their two technologies is possible, but we found little documented evidence that this had been achieved outside of a development environment.

This paper describes how we set up and tested that interoperability. The paper was developed by Oxford Computer Group with the kind support of Simon Veale from Microsoft Limited.

Oxford Computer Group

Oxford Computer Group (OCG) is an IT Service company who specialize in Identity and Access (IdA) Solutions and Training. With operations in North America, the UK and Germany, OCG have an enviable repository of expertise, solution components and training courses.

OCG have deployed 100+ enterprise wide IdA solutions and trained 2500+ people on Microsoft IdA technologies. In addition OCG ADFS training, and our expertise with ADFS and other identity and federation technologies (including Shibboleth integration) will position you to take early advantage of this growing area.

For more information <http://www.oxfordcomputergroup.com/microsoftIdA> or Email: info@oxfordcomputergroup.com

Contents

Introduction	4
Active Directory Federation Services	4
Shibboleth	6
Overcoming the Language Barrier	8
Claims, Assertions and Attributes	8
Federation Account Server (IdP)	8
Federation Resource Server (SP)	8
Trust Policy (Metadata)	8
Organizational Claims (resolver.xml)	9
Account Stores (resolver.xml)	9
Federation Partners	9
Claims Mapping (Attribute Acceptance Policy & Attribute Release Policy)	9
Outgoing Claims	9
Incoming Claims	9
Home Realm Discovery (Where are you from (WAYF)?)	10
Proof of Concept	10
Participants	10
Goals	10
Non Goals	10
Environment	11
ADFS FS-A	11
ADFS FS-R	11
Shibboleth IdP	11
Shibboleth SP	11
Reference Material	11
Setting up the ADFS FS-A to work with the Shibboleth SP	12
Trust Policy	12
Federation Service URI	13
Federation service endpoint URL	13
Token signing certificate	13
Organizational Claims	14
Account Stores	14
Defining the Resource Partner	14
Outgoing Claims Mapping	15
Setting up the Shibboleth SP to work with the ADFS FS-A	16
Shibboleth SP main configuration (shibboleth.xml)	16
Universal Extensions	16
Request Mapping	16
Session Initiator	17
Applications	17
Assertion Consumer Service	17
Metadata Provider	18
Certificates	18

Information required by the FS-A	18
Configuring the Apache Web Server	18
ssl.conf	18
shib.conf	19
Web Application	19
printenv.pl	19
Certificates	19
Defining the Account Partner	20
Attribute Acceptance Policy (AAP.xml)	22
Setting up the Shibboleth IdP to work with the ADFS FS-R	23
Installing the Shibboleth ADFS Extensions	23
Extracting the files	23
Building the ADFS Extensions	25
Main IdP Configuration (idp.xml)	25
Relying Party	25
Name Mapping	25
Credentials	26
Protocol Handler for ADFS	26
Metadata Provider	26
Partner Metadata (trex-metadata.xml)	26
Attribute Extraction (resolver.xml)	28
Testing Attribute Resolution	29
Attribute Release Policy (arp.site.xml)	29
Certificates	30
Setting up the ADFS FS-R to work with the Shibboleth IdP	31
Trust Policy	31
Federation Service URI	32
Federation service endpoint URL	32
Token signing certificate	32
Defining the Account Partner	33
Token Verification Certificate	33
Incoming Claims Mapping	33
Summary	34
Related Links	34
Appendices	34
Appendix 1 – ADFS FS-A Trust Policy	35
Appendix 2 - ADFS FS-R Trust Policy	36
Appendix 3 – Shibboleth IdP Configuration Files	38
idp.xml	38
resolver.xml	43
arps-site.xml	45
Appendix 4 – Shibboleth SP Configuration Files	47
shibboleth.xml	47
AAP.xml	57
adatum-metadata.xml	69

Introduction

Federated identity and access management builds on the Web Services technology wave, which describes the technology and business arrangements necessary for richly connecting users, applications, and systems within and across organizational boundaries, using the Internet and its associated standard communication mechanisms. Participants in federated systems may use different technologies with different security approaches and programming models, yet they can still integrate their businesses without substantial custom integration. In this federated system, each organization continues to manage its own identities, but is capable of securely sharing and accepting identities and credentials from other organizations. The goal of federated identity is to allow businesses and partners that trust each other in the real world to mirror that trust in their digital systems.

A number of federation technologies have been developed that are designed to exchange identity information – in the form of claims - across organizational boundaries to allow single sign-on and authorization to web-based applications. They provide a secure framework to transmit attributes about a web-browsing individual to local or remote web resources. When a user accesses a resource using federated identity, the user's own home domain can send information about that user to the resource which can then be used to determine appropriate access to the resource.

Active Directory Federation Services and Shibboleth are examples of these federation technologies and this paper documents the proof-of-concept that we ran to prove interoperability between the two.

Active Directory Federation Services

Active Directory™ serves as a primary identity and authentication service in many organizations and by employing Active Directory Federation Services (ADFS), organizations can extend their existing Active Directory infrastructures to provide access to resources that are offered by trusted partners across the Internet. These trusted partners can include external third parties or other departments or subsidiaries in the same organization.

ADFS is tightly integrated with Active Directory; retrieving user attributes from Active Directory (also Active Directory Lightweight Directory Services), and authenticating users against Active Directory. ADFS also uses Windows Integrated Authentication.

ADFS supports distributed authentication and authorization over the Internet and can be integrated into an organization's existing access management solution to translate the terms that are used in the organization into claims that are agreed on as part of a federation. ADFS can create, secure, and verify the claims that move between organizations. It can also audit and monitor the activity between organizations and departments to help ensure secure transactions.

The following are some of the key features of ADFS in Windows Server 2003 R2:

- **Federation and Web SSO.** When an organization uses the Active Directory directory service, it benefits from SSO functionality through Windows-integrated authentication within the organization's security or enterprise boundaries. ADFS extends this functionality to Internet-facing applications, which enables customers, partners, and suppliers to have a similar, streamlined, Web SSO user experience when they access the organization's Web-based applications. Furthermore, federation servers can be deployed in multiple organizations to facilitate business-to-business (B2B) federated transactions between partner organizations.
- **Web Services (WS)-* interoperability.** ADFS provides a federated identity management solution that interoperates with other security products that support the WS-* Web Services Architecture. ADFS does this by employing the federation specification of WS-*, called WS-Federation. The WS-Federation specification makes it possible for environments that do not use the Windows identity model to federate with Windows environments.

- **Extensible architecture.** ADFS provides an extensible architecture that supports the Security Assertion Markup Language (SAML) token type and Kerberos authentication (in the Federated Web SSO with Forest Trust scenario). ADFS can also perform claim mapping, for example, modifying claims using custom business logic as a variable in an access request. Organizations can use this extensibility to modify ADFS to coexist with their current security infrastructure and business policies.

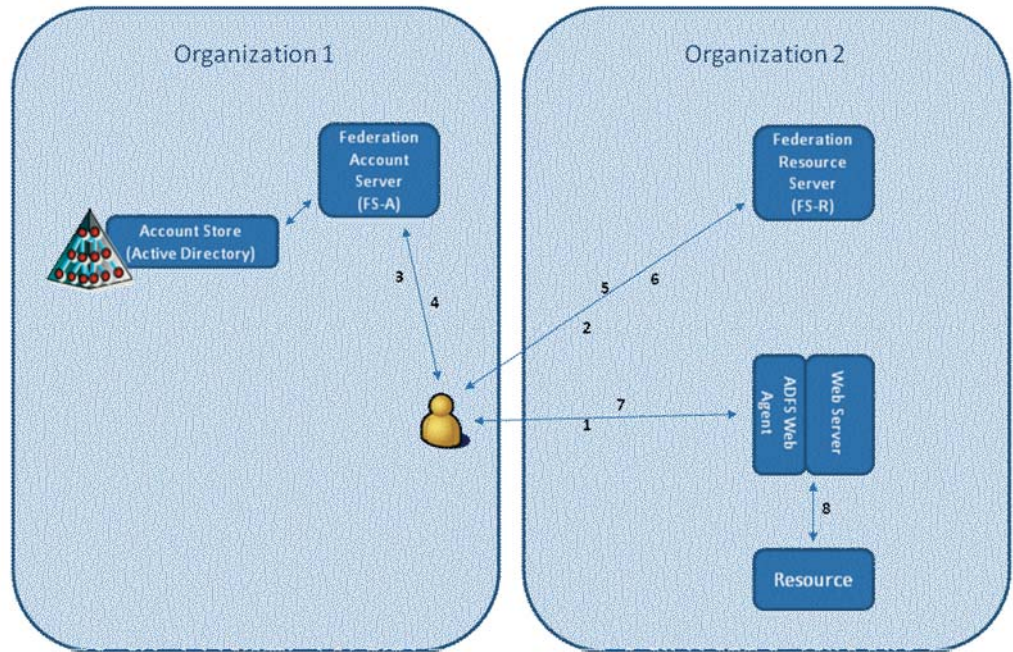


Figure 1 ADFS Schematic Diagram

1. User attempts to access ADFS-protected resource on site application server.
2. User is redirected to the FS-R.
3. FS-R directs the user to the FS-A in their home domain.
4. FS-A uses local credentials to validate user and return claims in a SAML token.
5. FS-R validates the signature on the SAML token.
6. FS-R re-signs the SAML token and passes it back.
- 7, 8. User is directed back – via the ADFS web agent - to the resource. Resource uses the claims for access control and other application-level decisions.

Shibboleth

Shibboleth is a project of Internet2/MACE concerned with the development of architectures, policy structures, practical technologies, and an open source federation implementation to support inter-institutional sharing of web resources subject to access controls. Its primary audience is within the education and research sector.

Internet2 is a U.S. advanced networking consortium led by the education and research community comprising universities, partner organizations, laboratories, government agencies and other institutions of higher learning.

The Shibboleth team consists of Internet2 and a group of campus middleware architects from Internet2 member schools and corporate partners. Organizations collaborating in its development include national and international higher education institutions, their partners, content providers, and government agencies.

Key concepts within Shibboleth include:

- **Federated Administration.** The Identity Provider (origin) campus (home to the browser user) provides attribute assertions about that user to the Service Provider (target) site. A trust fabric exists between campuses, allowing each site to identify the other speaker, and assign a trust level. Identity Provider sites are responsible for authenticating their users, but can use any reliable means to do this.
- **Access Control Based On Attributes.** Access control decisions are made using those assertions. The collection of assertions might include Identity, but many situations will not require this (eg accessing a resource licensed for use by all active members of the campus community, accessing a resource available to students in a particular course).
- **Active Management of Privacy.** The Identity Provider (origin) site, and the browser user, control what information is released to the Service Provider (target). A typical default is merely "member of community". Individuals can manage attribute release via a web-based user interface. Users are no longer at the mercy of the target's privacy policy.
- **Standards Based.** Shibboleth will use OpenSAML for the message and assertion formats, and protocol bindings which is based on Security Assertion Markup Language (SAML) developed by the OASIS Security Services Technical Committee.
- **A Framework for Multiple, Scaleable Trust and Policy Sets (Federations).** Shibboleth uses Federations to specify a set of parties who have agreed to a common set of policies. (A site can be in multiple Federations, though.) This moves the trust framework beyond bi-lateral agreements, while providing flexibility when different situations require different policy sets.
- **A Standard (yet extensible) AttributeValue Vocabulary.** Shibboleth has defined a standard set of attributes; the first set is based on the eduPerson object class that includes widely-used person attributes in higher education.

The Shibboleth software implements the OASIS SAML v1.1 specification, but in December 2005 Internet2 announced that it had developed a new extension of Shibboleth to support Microsoft Windows Server 2003 R2 using WS-Federation, the passive requestor profile, and the passive responder interoperability profile.

The new extension, implemented in Shibboleth versions 1.3c and later, provides interoperability with Microsoft's Active Directory Federation Services (ADFS), allowing sites using ADFS to participate in Shibboleth-based federations and vice versa.

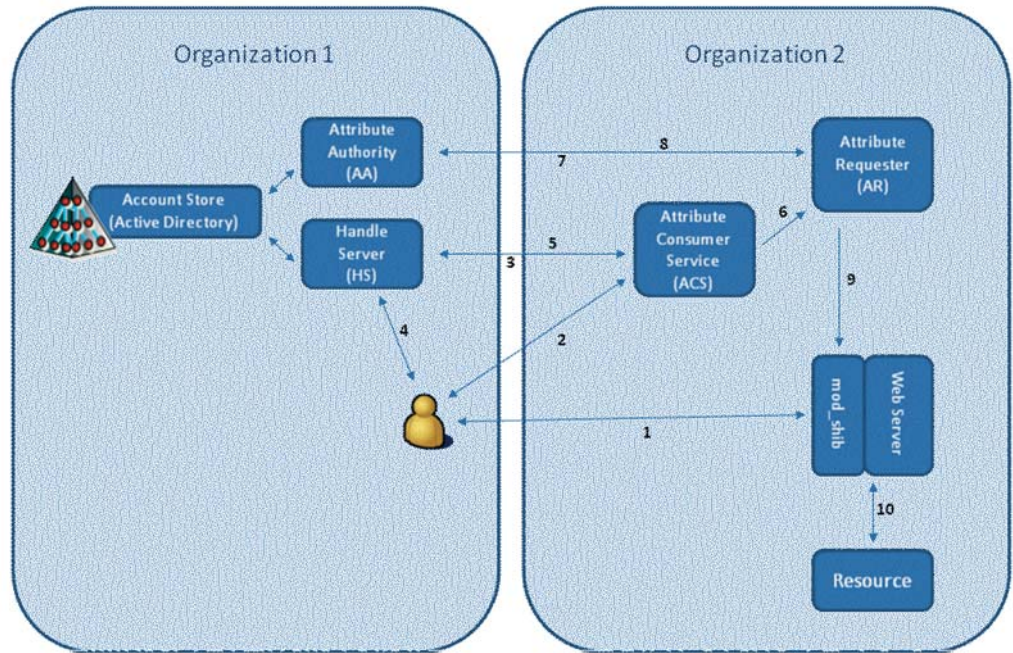


Figure 2 Shibboleth Schematic Diagram

1. User attempts to access Shibboleth-protected resource on SP site application server.
- 2, 3. User is redirected to the Handle Service at their IdP (there is no WAYF server in this scenario).
4. User authenticates at their IdP, using local credentials.
5. Handle service generates unique ID (Handle) and redirects user to Service Provider site's Assertion Consumer Service (ACS). ACS validates the supplied assertion, creates a session, and transfers to Attribute Requestor (AR).
- 6, 7, 8. AR uses the Handle to request attributes from the IdP site's Attribute Authority. The attribute authority responds with attribute assertions subject to attribute release policies.
- 9, 10. User is directed back to resource. Resource uses attributes for access control and other application-level decisions.

Overcoming the Language Barrier

As we can see from the preceding sections, ADFS and Shibboleth are very similar in their approach to federation and address the same problem space. However, as with many cross-platform or cross-technology interoperability scenarios we have something of a language barrier to overcome. A lot of the terminology used in the ADFS world is not the same as that used in the Shibboleth world and someone familiar with either one of the technologies will not necessarily recognize the equivalent components of the other. As a simple example, an ADFS account federation server (FS-A) is the functional equivalent of the Shibboleth Identity Provider (IdP), and the ADFS resource federation server (FS-R) is the equivalent of the Shibboleth Service Provider (SP). As we get into the more detailed configuration of the systems this becomes more of an issue.

This section describes some of the common components in both their ADFS and Shibboleth contexts.

Claims, Assertions and Attributes

These are all more-or-less the same thing and the terms tend to be used interchangeably. An attribute can be thought of as a piece of information that describes something about a user; their name, affiliation etc. That piece of information is passed to a federation partner in the form of a claim. However claims could also include other information, e.g. that a user authenticated at a particular time, or that they used a particular authentication strength.

Federation Account Server (IdP)

Federation servers in the account partner are used to authenticate local user accounts and then issue security tokens that can be used to access Web-based applications that are hosted in resource partners. In addition, federation servers in the account partner issue cookies to users to maintain login status. These cookies enable SSO capabilities so that users do not have to enter credentials each time that they visit different Web-based applications in the resource partners.

In ADFS the account partner is known as the federation account server (FS-A). In Shibboleth it is known as the Identity Provider or IdP.

Federation Resource Server (SP)

Federation servers at the resource partner validate the security tokens that are issued by the federation servers at the account partner. Federation servers at the resource partner also issue security tokens that are meant for the Web-based applications in the resource partner. In addition, federation servers in the resource partner issue cookies to the user accounts, which come from the account partner. These cookies enable SSO capabilities so that users do not have to log in again at their federation servers in the account partner when the users attempt to access different Web-based applications at the resource partner.

In ADFS the resource partner is known as the federation resource server (FS-R). In Shibboleth it is known as the service provider (SP).

Trust Policy (Metadata)

Trust policies – expressed as XML metadata in Shibboleth - are what the federation partners use to find each other and establish secure communications. As a minimum they contain the unique names of the federation partners, the internet addresses to which requests should be sent, and the certificates that should be used to validate the SAML tokens that are exchanged.

An ADFS server defines itself to its federation partners through a trust policy which is set up and viewed in the ADFS administration console. It can be exported to an XML file for easy import into another ADFS server. Each ADFS server can act as both an FS-A and an FS-R and the trust policy contains the details for both. Federation partners will automatically select the correct options when they import the trust policy XML file during federation partner configuration.

A Shibboleth implementation defines itself to its federation partners through metadata in metadata.xml files which can be loaded into another Shibboleth system. Typically a Shibboleth system is either an IdP OR an SP.

Organizational Claims (resolver.xml)

Organizational claims are the superset of claims that an organization wishes to pass to/receive from its federation partners. A specific federation partnership may only require a subset of the available organizational claims.

In ADFS, organizational claims are defined within an organization's trust policy in the ADFS administration console. They are defined as one of two types; group or custom.

In Shibboleth the superset of claims that may be released by the system is declared in the XML file resolver.xml. All claims in Shibboleth are analogous to the custom claim type in ADFS.

Account Stores (resolver.xml)

Neither ADFS or Shibboleth stores information about users itself. They both rely on separate identity stores to provide valid identity information.

ADFS can retrieve identity information from Active Directory or Active Directory Light Weight Directory Services (previously known as ADAM) in order to populate claims. Account stores are defined under the organization's trust policy and define mappings between each claim and the directory source of the information required to populate it. Typically this would be an attribute or group(s).

In Shibboleth the mapping of claims back to an identity store is handled by the XML file resolver.xml. It defines the type of store, along with appropriate connection information as well as the attribute/claim mappings. Typically it will use an LDAP directory or a database as the identity store. For this proof of concept we used a simple MySQL database.

Federation Partners

An ADFS server defines its federation partners through the ADFS administration console. Federation partners are defined as either Account Partners or Resource Partners.

Shibboleth defines its federation partners through metadata.xml files. They are either Identity Providers (IdP) or Service Providers (SP). Each partner may be defined in a separate XML file, or the entire federation may be combined into a single XML file. This makes it relatively straightforward to implement a new Shibboleth system with full knowledge of the entire federation that it is joining.

Claims Mapping (Attribute Acceptance Policy & Attribute Release Policy)

Attributes names used internally within an institution may be different to those exchanged over the wire between federation partners. A mapping process can be configured to map the internal attribute names with the names agreed between the federation partners.

Outgoing Claims

The ADFS FS-A sets up outgoing claims as part of the Resource Partner definition in the ADFS administration console. It is known as outgoing claims mapping. The Shibboleth IdP uses its Attribute Release Policy as defined in the arps.site.xml configuration file.

Incoming Claims

The ADFS FS-R sets up incoming claims as part of the Account Partner definition in the ADFS administration console. It is known as incoming claims mapping. The Shibboleth SP uses its Attribute Acceptance Policy as defined in the AAP.xml configuration file.

Home Realm Discovery (Where are you from (WAYF)?)

Shibboleth supports, as part of its architecture, the concept of a centralized discovery service called the “Where are you from?” or WAYF server. When a user attempts to access a Shibboleth-protected resource, they can be directed to the WAYF server which will ask them where they are from (their home domain) and then redirect them to the Identity Provider for that domain.

ADFS doesn't have an equivalent component, but relies instead on each federation resource server (FS-R) performing home realm discovery itself, based on the federation account servers (FS-A) that it knows about. I.e. the bilateral agreements that have been set up with its federation partners.

This needn't look any different from the web users' perspective but will rely on a common implementation of an ADFS home realm discovery web page across the federation.

Proof of Concept

Participants

The proof of concept was run with the invaluable help of the following:

- Simon Veale & Rob Jones Microsoft Limited
- Dr Simon MacLeish of the Library Projects Team at the London School of Economics & Political Science
- Leslie Want of the County Borough of Neath, Port Talbot
- Steve Mitchell of Oxford Computer Group, an IT service company who specialize in Identity and Access Management
- JISC – the Joint Information Systems Committee

Goals

The goals of the proof of concept were primarily to prove that ADFS and Shibboleth can interoperate successfully, and to document how to achieve it.

The first scenario we set out to prove was that an ADFS account federation server could interoperate with Shibboleth service providers. This was seen as being the most likely real-world implementation scenario where institutions with Microsoft-based infrastructures wish to access existing resources protected by Shibboleth.

Secondary to this – but only just - was to prove the reverse scenario where a Shibboleth identity provider would offer its users access to resources in a Microsoft infrastructure, for example SharePoint, behind an ADFS federation resource server.

Our third goal was to look at the attributes (claims) that could be passed. There are specific attributes mandated by the UK Access Management Federation and we wanted to check that these could be passed in the correct formats in the mixed ADFS/Shibboleth environment. For example the SHIB_TARGETEDID attribute.

Non Goals

It was not a goal of the proof of concept to look at the initial installation and deployment of ADFS and Shibboleth. We made the assumption that all of the functional components were installed and working, and all we were looking at was getting them to interoperate.

The exception to this was the ADFS extensions for Shibboleth. We included this because we felt that it would be unlikely that it would already be installed in existing Shibboleth federations and should therefore be considered as part of the configuration of Shibboleth in this scenario.

It was also not a goal to look at real-world applications, but just to prove that user authentication happened before the application was invoked, and that the correct information – in the form of claims – was passed across the wire. How any application subsequently consumes those claims was out of scope for this exercise.

An additional non-goal was that we weren't going to demonstrate interoperability with any implementation of the Shibboleth profile apart from the reference software developed by Internet2.

Environment

The proof of concept was conducted in a virtualized environment consisting of 4 separate virtual PCs hosting the ADFS FS-A and FS-R, and the Shibboleth IdP and SP.

We decided to install Shibboleth on a Unix/Linux platform rather than Windows servers because we felt that this would more accurately reflect typical environments today. This added to our challenges in terms of having to become familiar with the operating system, its filesystem and tools, as well as Shibboleth itself.

We selected Fedora Core 5 as the Linux platform for the Shibboleth components because it was the easiest to build, having packages already built and available for download. Other Linux platforms appeared at the time to be a good deal more complex to build. This is a situation that is still evolving; see <http://shibboleth.internet2.edu/latest.html> for up to date information.

The following servers were built.

ADFS FS-A

- Windows Server 2003 R2
- Domain = Adatum.com
- Identity store = Active Directory

ADFS FS-R

- Windows Server 2003 R2
- Domain = TreyResearch.com
- ADFS web agent
- Test application - the Trey Research component ordering system used in Microsoft's ADFS demonstration systems

Shibboleth IdP

- Fedora Core 5
- Java 1.5

Note: The Java version is very important. Apache's XML Security package doesn't work with JDK 1.4 without making some changes to override some of the classes in the JDK.

- Shibboleth 1.3f
- Shibboleth ADFS extension module
- MySQL
- Domain = Adatum.com
- Identity store = a MySQL database

Shibboleth SP

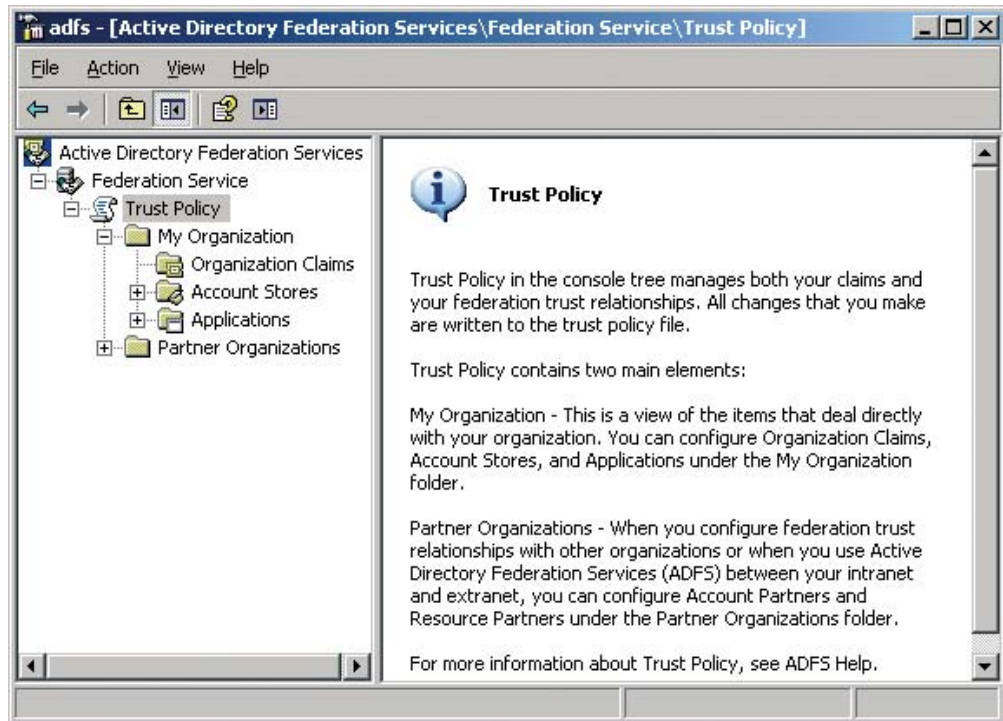
- Fedora Core 5
- Shibboleth 1.3f
- Domain = Contoso.com
- Test application - a printenv.pl perl script that displays the processes environment variables - which include the attributes passed by the FS-A

Reference Material

Our main points of reference for building Shibboleth and configuring to work with ADFS were the Internet2 site at <http://shibboleth.internet2.edu> and the Shibboleth Wiki at <https://authdev.it.ohio-state.edu/twiki/bin/view/Shibboleth/WebHome>. For ADFS we used the ADFS Design Guide and ADFS Deployment Guide.

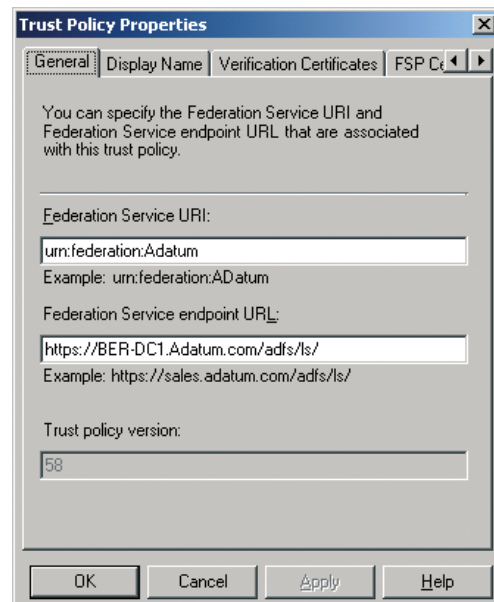
Setting up the ADFS FS-A to work with the Shibboleth SP

The ADFS FS-A defines itself, and its own configuration, and its resource partners in its trust policy. Trust Policy is managed using the ADFS console.



Trust Policy

The properties of the trust policy (right click on **Trust Policy** and select **Properties**) describe some of the key elements of the FS-As configuration, including information required by resource partners.



Federation Service URI

This is a value used by partner organizations and applications to uniquely identify a federation partner.

```
urn:federation:Adatum
```

Federation service endpoint URL

This is the address used by partner organizations and applications to send requests and responses.

```
https://BER-DC1.Adatum.com/adfs/ls/
```

Token signing certificate

The FS-A uses a token signing certificate to sign the SAML tokens it passes to the SP. This has to be included in the partner metadata XML file (adatum-metadata.xml) on the Shibboleth side so that incoming SAML tokens can be verified by the SP. It can be extracted from the exported trust policy XML file (see appendices) and pasted into the Shibboleth metadata.

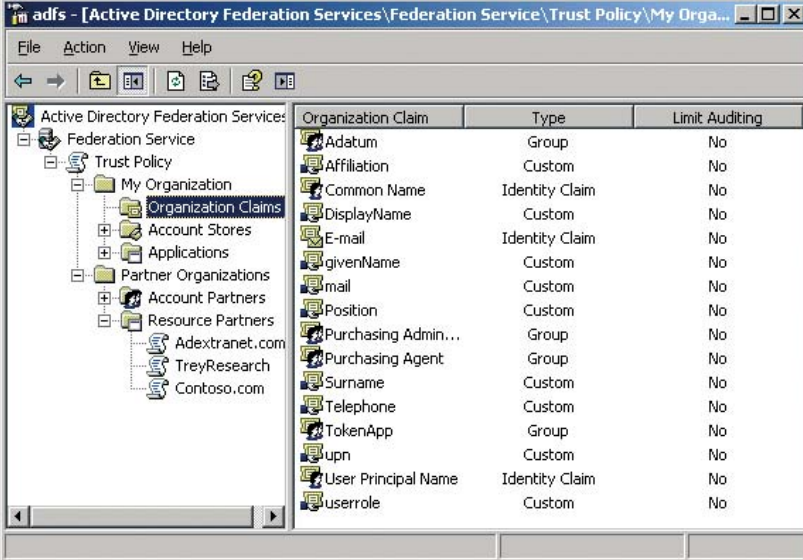
```
<X509Certificate>MIICyDCCAbSgAwIBAgIQsVv1sV0V36RHicKd5gWLLjAJBgUrDgMCHQUAMCQ
xIjAgBgNVBAMTGUZlZGVyYXRpb24gU2VydMvYIEJFUi1EQzEwHhcNMDYwOTA1MjE1MTE5WhcNMDc
wOTA2MDM1MTE5WjAkMSIwIAYDVQQDExlGZWRLcmF0aW9uIFNlcnZlciBCRVItREMxMIIBIjANBgk
qhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAx0wsqoDvMhtBsqu3Mi9+vv8PuyknyoyWxvAbQ6axY
P4N+oYyugyx/7rEky/nr6u/cZct1XZ5UWfKrEx8RW6Re4eoY7d9ua1JbleJYmx97LXEZKNc1TrTA
1DMY2J0yeQPFq0UQOYLcDDhpIBi8PJ48gmsMtKjI0/lrxXGsjs8f97gzEsA970372ZnYkzi9rqT4
C42Soz5SaRwFlz4glDa99QBkNKgmnLjWUn1sjgaCQUdwI794dWIt22h1kvziMyKvqD8/ygPclPgM
2RQH5hLaU9UFbnqQ+jx36Kr2raIq/2Yp9QLFoEEQfe4bUgKUtR+sgocvgyllUgAZuYTOyPwIDAQA
BMAkGBSsOAwIdBQADggEBAMD15++UrKv3SIEohqe7SEd/nh3j1Rc15hBx9W4dAhicZ+uokVYzpK2
L4F8dQYEEYT/RR2UCbChBFzZjJWaXik9SkBvEWKU6vCeRnBc5faNaJGPKSXwd25TtxIM9LV4misB
OIifaXuFhOpIngkXYWRaOQETS7hQRfpoGeB/UXyWjICfnaDLJdTUIWJmlytPmtCfFN0e/avjx0p
POoc3CXN2XX+XNwd9eDX4LEn+XjKKooAbsaYl/dRS4INcIZx4hPpImvucuPxXAMqc177JBzOLIPq
KCEr/ZtIUyZHEwWpxn7mn4qI9gLubpz5bokkgJa4V141jXK7nuexmB7K0FDU=</X509Certifica
te>
```

The trust policy can be exported into an XML file for easy import into an ADFS resource partner. The trust policy for this FS-A has been included in the appendices. However the format isn't out-of-the-box compatible with the format of the partner metadata required by Shibboleth, although it should be possible to write a transformation to accomplish this.

In the meantime, the above information from the trust policy is required from the FS-A to set up the correct metadata (adatum-metadata.xml) in the Shibboleth SP.

Organizational Claims

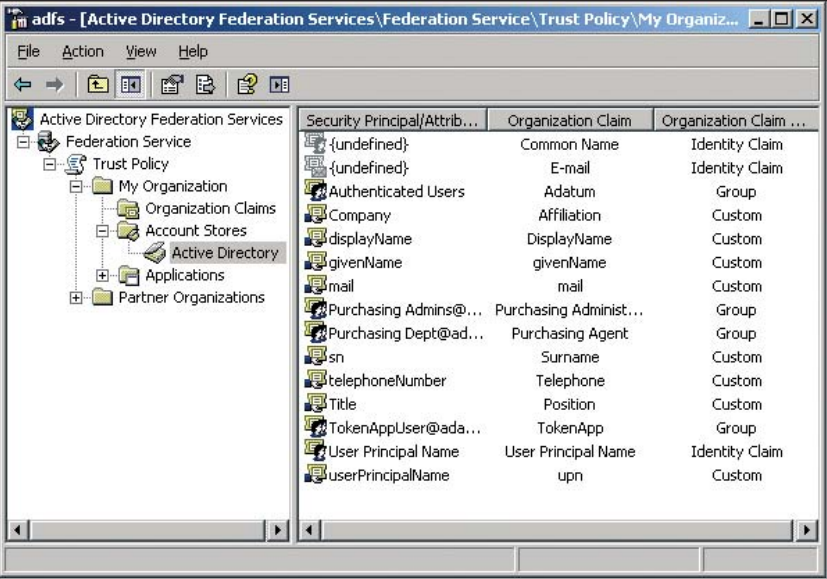
Organizational claims are the superset of claims that this FS-A can pass to resource partners. Each resource partner definition will reference an agreed subset of the organizational claims. This FS-A has three resource partners and organizational claims are shown below.



Organization Claim	Type	Limit Auditing
Adatum	Group	No
Affiliation	Custom	No
Common Name	Identity Claim	No
DisplayName	Custom	No
E-mail	Identity Claim	No
givenName	Custom	No
mail	Custom	No
Position	Custom	No
Purchasing Admin...	Group	No
Purchasing Agent	Group	No
Surname	Custom	No
Telephone	Custom	No
TokenApp	Group	No
upn	Custom	No
User Principal Name	Identity Claim	No
Userrole	Custom	No

Account Stores

Organizational claims are populated from account stores. For this PoC the account store is Adatum's Active Directory. Mapping is set up to populate organizational claims from groups or attributes in Active Directory.

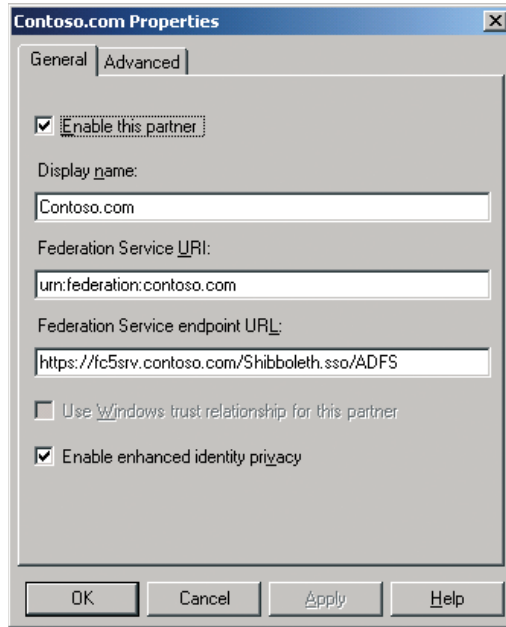


Security Principal/Attrib...	Organization Claim	Organization Claim ...
{undefined}	Common Name	Identity Claim
{undefined}	E-mail	Identity Claim
Authenticated Users	Adatum	Group
Company	Affiliation	Custom
displayName	DisplayName	Custom
givenName	givenName	Custom
mail	mail	Custom
Purchasing Admins@...	Purchasing Administ...	Group
Purchasing Dept@ad...	Purchasing Agent	Group
sn	Surname	Custom
telephoneNumber	Telephone	Custom
Title	Position	Custom
TokenAppUser@ada...	TokenApp	Group
User Principal Name	User Principal Name	Identity Claim
UserPrincipalName	upn	Custom

Defining the Resource Partner

Resource partners are defined under Partner Organizations within the Trust Policy using the “Add Resource Partner Wizard”.

The Shibboleth service provider is configured in exactly the same way as an ADFS resource partner in the ADFS administration console. The information entered below in the partner properties was supplied by the service provider, and comes from the partner's own Shibboleth metadata.



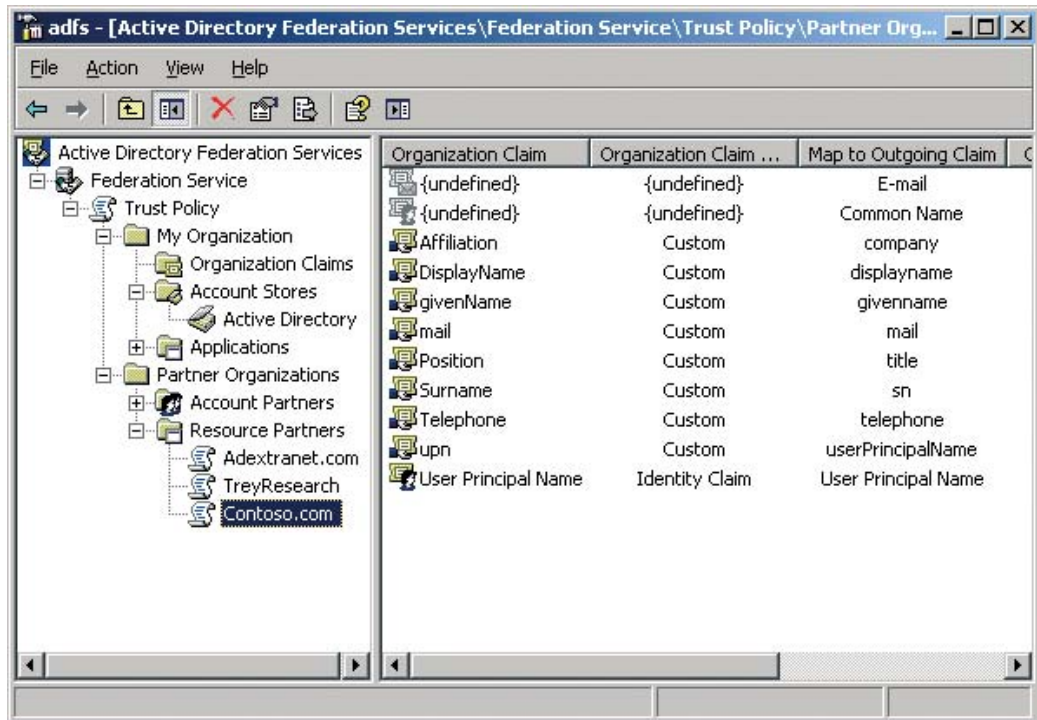
Note: The UK Access Management Federation's requirements state that a hashed and scoped identity should be passed into the SHIB_TARGETEDID attribute. These are of the form...

b8Z63K9kyiLzRPLMTsDoC58H/sY=@adat.um.com

This can be achieved by selecting "Enable enhanced identity privacy" (above), and mapping the UPN identity claim through to the SHIB_TARGETEDID on the service provider.

Outgoing Claims Mapping

The agreed set of claims is added to the resource partner definition, and the claim names are mapped to the names agreed with the resource partner.



Setting up the Shibboleth SP to work with the ADFS FS-A

The 1.3f release of Shibboleth includes an extension library, `adfs.so`, that provides protocol support for interoperating with ADFS. This is installed by default but Shibboleth must then be configured to enable interoperation with the ADFS FS-A.

The Shibboleth service provider is configured through a variety of XML files. On the Fedora Core 5 systems these are found in the `/etc/shibboleth` directory.

Shibboleth SP main configuration (`shibboleth.xml`)

The following modifications were required in the `shibboleth.xml` file. The complete `shibboleth.xml` is listed in the appendices, with these sections highlighted.

Universal Extensions

These extensions are loaded by all Shibboleth-aware processes. Ensure that the library path for ADFS (highlighted below) is added.

```
<Extensions>
  <Library path="/usr/libexec/xmlproviders.so" fatal="true"/>
    <Library path="/usr/libexec/adfs.so" fatal="true"/>
</Extensions>
```

Request Mapping

The following section causes all sessions referencing web pages in `/secure` on the web server – on the default ports - to be authenticated by Shibboleth.

```
<RequestMapProvider
type="edu.internet2.middleware.shibboleth.sp.provider.NativeRequestMapProvider">
  <RequestMap applicationId="default">
    <!--
      This requires a session for documents in /secure on the
      containing host with http and https on the default ports. Note that
      the name and port in the <Host> elements MUST match Apache's
      ServerName and Port directives or the IIS Site name in the <ISAPI>
      element below. You should also be sure that Apache's UseCanonicalName
      setting is On
    --> <Host name="fc5srv.contoso.com">
      <Path name="secure" authType="shibboleth"
        requireSession="true"/>
    </Host>
  </RequestMap>
</RequestMapProvider>
```

See the corresponding changes in the Apache configuration below, where `secure` is mapped to the actual files on disk.

Session Initiator

Here the URL for the Shibboleth WAYF service is replaced by the federation service endpoint of the FS-A. This URL has come from the FS-A's trust policy. In this example we've set this as the default for session initiation (`isDefault="true"`). In an existing Shibboleth federation a specific session initiator will probably be required for the ADFS partner(s).

```
<SessionInitiator isDefault="true" id="testshib"
Location="https://fc5srv.contoso.com/Shibboleth.sso/ADFS"

  Binding="urn:mace:shibboleth:sp:1.3:SessionInit"

  wayfURL="https://BER-DC1.Adatum.com/adfs/ls/"

  wayfBinding="http://schemas.xmlsoap.org/ws/2003/07/secext"/>
```

Applications

The RequestMap element contains host and path elements. Shibboleth matches request URLs against these elements in order to determine how to process the request. Attributes on the RequestMap, Host, and Path elements specify whether to require an authenticated session, and how to locate the associated Application element and settings.

```
<RequestMap applicationId="default">
  <Host name="fc5srv.contoso.com">
    <Path name="secure" authType="shibboleth"
      requireSession="true"/>
  </Host>
</RequestMap>
```

The Applications sections are where most of Shibboleth's SAML bits are defined. We defined an Application to be our `printenv.pl` script. For convenience we set this as the default application.

The applicationID in the requestMap above is then referenced as an application below.

```
<Applications id="default" providerId="urn:federation:contoso.com"
  homeURL="https://fc5srv.contoso.com"
  xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
  xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata">
```

The providerId is referenced later in the partner metadata.

Assertion Consumer Service

An assertion consumer service is SAML terminology for a protocol endpoint at a service provider. It is equivalent to the ADFS federation server endpoint URL. The binding parameter indicates that this session is using the WS-Federation protocol. The following should be added to the existing AssertionConsumerServices definition – making sure that the index parameter is unique.

```
<md:AssertionConsumerService Location="/ADFS" index="4"
  Binding="http://schemas.xmlsoap.org/ws/2003/07/secext"
  ResponseLocation="/"/>
```

This assertion consumer services is also referenced in the metadata of the FS-A (`adatum-metadata.xml`)

Metadata Provider

Here we added a pointer to the file containing the metadata for our federation partner.

```
<MetadataProvider
type="edu.internet2.middleware.shibboleth.metadata.provider.XMLMetadata"
    uri="/etc/shibboleth/adatum-metadata.xml"/>
```

Certificates

Here we added the paths of the files containing the X.509 key and certificate to be used by the SP.

```
<CredentialsProvider
type="edu.internet2.middleware.shibboleth.common.Credentials">
  <Credentials xmlns="urn:mace:shibboleth:credentials:1.0">
    <FileResolver Id="defcreds">
      <Key>
        <Path>/etc/shibboleth/sp.key</Path>
      </Key>
      <Certificate>
        <Path>/etc/shibboleth/sp.crt</Path>
      </Certificate>
    </FileResolver>
  </Credentials>
</CredentialsProvider>
```

Information required by the FS-A

Information about the SP must be passed to the FS-A so they can set up the other end of the communication.

In this case that is:

providerID	urn:federation:contoso.com
AssertionConsumer Service location	https://fc5srv.contoso.com/Shibboleth.sso/ADFS

Configuring the Apache Web Server

This is not really ADFS-specific configuration; something similar would be required for any application on a Shibboleth SP. It is included here to show how we told our web server that browser sessions with this address/application have to be passed to Shibboleth for authentication.

ssl.conf

We added the following lines to the end of the /etc/httpd/conf.d/ssl.conf file to map the /secure web address to the cgi-bin directory on the server, and to require Shibboleth authentication.

```
Alias /secure /var/www/cgi-bin

<Location /secure>
  AuthType shibboleth
  ShibRequireSession On
  require valid-user
  Options ExecCGI
</Location>
```

shib.conf

In the /etc/httpd/conf.d directory there is a Shibboleth configuration file, shib.conf that Apache will automatically load on startup.

Web Application**printenv.pl**

Our test application printenv.pl was located in the cgi-bin directory. The full URL used to reach it was:

```
https://fc5srv.contoso.com/secure/printenv.pl

#!/usr/bin/perl -w

# quick app to see how environment variables are being set

print "Content-type: text/html\n\n";
print "<html xmlns=\"http://www.w3.org/1999/xhtml\" xml:lang=\"en\"
lang=\"en\"><head><title>Shib-enabled environment
variables</title></head><body>\n<table><tr><td><b>Key</b></td><td><b>Value</
b></td></tr>";

my @sortedkeys = sort (keys(%ENV));

foreach $key (@sortedkeys){
print "<tr><td bgcolor=\"#FFFFCC\">$key</td><td
bgcolor=\"#FFCCCC\">${ENV}{$key}</td></tr>\n";

}print "</table></body></html>";
exit 0;
```

Certificates

We generated a key and certificate for the Shibboleth SP using openssl. The resulting files were saved in the /etc/shibboleth directory along with the other Shibboleth configuration files.

```
# openssl genrsa -out sp.key 2048
# openssl req -new -key sp.key -x509 -days 3650 -out sp.crt
```

These are referenced as a credentials provider in the shibboleth.xml file.

Note: We used self-signed certificates for this proof-of-concept which would not be recommended for production use.

Defining the Account Partner

The account partner is defined in a metadata.xml file. The partner metadata file can have any name – in our PoC it was adatum-metadata.xml. It may contain information pertaining to one partner or many – even the entire federation. The filename must be declared in the shibboleth.xml file as a metadata provider. Partner metadata contains the information supplied by the federation partner(s) to facilitate communication. The information provided by the ADFS FS-A (Adatum) is shown highlighted in the metadata file below.

```
<EntitiesDescriptor
  xmlns="urn:oasis:names:tc:SAML:2.0:metadata"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:shibmd="urn:mace:shibboleth:metadata:1.0"
  xsi:schemaLocation="urn:oasis:names:tc:SAML:2.0:metadata
/usr/share/xml/shibboleth/saml-schema-metadata-2.0.xsd
urn:mace:shibboleth:metadata:1.0 @-PKGXMLDIR-@/shibboleth-metadata-1.0.xsd
http://www.w3.org/2000/09/xmldsig# @-PKGXMLDIR-@/xmldsig-core-schema.xsd"
  Name="urn:federation:Adatum.com"
  validUntil="2010-01-01T00:00:00Z">
<!--
  Identity provider
-->
  <EntityDescriptor entityID="urn:federation:Adatum">
    <IDPSSODescriptor
protocolSupportEnumeration="http://schemas.xmlsoap.org/ws/2003/07/secext">
      <KeyDescriptor use="signing">
        <ds:KeyInfo>
          <ds:X509Data>
            <ds:X509Certificate>
MIICyDCCAbSgAwIBAgIQsVv1sV0V36RHicKd5gWLLjAJBgUrDgMCHQUAMCQxiJAg
BgNVBAMTGUZlZGVyYXRpb24GU2VydMvYIEJFUilEQzEwHhcNMDYwOTA1MjE1MTE5
WhcNMDcwOTA2MDM1MTE5WjAkMSIwIAYDVQQDExlGZWRLcmF0aW9uIFNlcnZlciBC
RVItREMxMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAx0wsqoDvMhtB
sqdvu3Mi9+vv8PuyknyoyWxvAbQ6axYP4N+oJyYugyx/7rEky/nr6u/cZct1XZ5UW
fKrEx8RW6Re4eoY7d9ualJbleJYmx97LXEZKNC1TrTALDMY2J0yeQPFq0UQOYLcD
DhpIBi8PJ48gmsQtKjI0/lrxXGsj8f97gzEsA970372ZnYkzi9rqT4C42Soz5Sa
RwFlz4glDa99QBkNKgmLjWUn1sjgaCQUdWI794dWit22hlkvziMyKvqD8/ygPc1
PgM2RQH5hLaU9UFbnqQ+jx36Kr2raIg/2Yp9QLFoEEQfe4bUgKUTr+sogocvgy1lU
gAZuYTOyPwIDAQABMAKGBSsOAwIdBQADggEBAMD15++UrKv3SIEohqe7SEd/nh3j
1Rc15hBx9W4dAHicZ+uokVYzpk2L4F8dQYEEYT/RR2UCbChBFZzjJWaXiK9SkBvE
WKU6vCeRnBc5faNaJGPKSXwd25TtXIM9LV4misBOIifaXuFhOpLngkXYWRAOQEtS
P7hQRfpoGeB/UXyWjICfnaDLJdTUIWJmlytPmtCfFN0e/avjx0pPOoc3CXN2XX+X
Nwd9eDX4LEn+XjKKooAbsay1/dRS4INcIzX4hPpImvucuPxXAMqc177JBzOLIPqK
CEr/ZtIUyZHEwWpxn7mn4qI9gLubpz5bokkgJa4V141jXK7nuexmB7K0FDU=
            </ds:X509Certificate>
          </ds:X509Data>
        </ds:KeyInfo>
      </KeyDescriptor>
    </IDPSSODescriptor>
  </EntityDescriptor>
</EntitiesDescriptor>
```

```

        </ds:X509Data>
    </ds:KeyInfo>
</KeyDescriptor>
    <SingleSignOnService
Binding="http://schemas.xmlsoap.org/ws/2003/07/secext"
        Location="https://BER-DC1.Adatum.com/adfs/ls/" />
    </IDPSSODescriptor>
<AttributeAuthorityDescriptor
protocolSupportEnumeration="http://schemas.xmlsoap.org/ws/2003/07/secext">
    <Extensions>
        <shib:Scope
xmlns:shib="urn:mace:shibboleth:metadata:1.0">adatum.com</shib:Scope>
    </Extensions>
    <!-- The certificate has to be repeated here (or a different
one specified if necessary). -->
    <KeyDescriptor use="signing">
        <ds:KeyInfo>
            <ds:X509Data>
                <ds:X509Certificate>
MIICyDCCAbSgAwIBAgIQsVv1sV0V36RHicKd5gWLLjAJBgUrDgMCHQUAMCQxIjAg
BgNVBAMTGUZlZGVyYXRpb24gU2VydMvyIEJFUilEQzEwHhcNMDYwOTA1MjE1MTE5
WhcNMDcwOTA2MDM1MTE5WjAkMSIwIAYDVQQDExlGZWRLcmF0aW9uIFNlcnZlciBC
RVItREMxMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAx0wsqoDvMhtB
sqdvu3Mi9+vv8PuyknyoyWxvAbQ6axYP4N+oYyugyx/7rEky/nr6u/cZct1XZ5UW
fKrEx8RW6Re4eoY7d9ualJbleJYmx97LXEZKNC1TrTALDMY2J0yeQPFq0UQOYLcd
DhpIBi8PJ48gmsQtKjI0/lrxXGsjs8f97gzEsA970372ZnYkzi9rqT4C42Soz5Sa
RwFlz4g1Da99QBkNKgmnLjWUn1sjgaCQUdwI794dWit22hlkvziMyKvqD8/ygPcl
PgM2RQH5hLaU9UFbnqQ+jx36Kr2raIg/2Yp9QLFoEEQfe4bUgKUtr+sgocvgyllU
gAZuYTOyPwIDAQAABMAkGBSsOAwIdBQADggEBAMD15++UrKv3SIEohqe7SEd/nh3j
1Rc15hBx9W4dAHicZ+uokVYzpk2L4F8dQYEEYT/RR2UCbChBFZzjJWaXik9SkBvE
WKU6vCeRnBc5faNajGPKSXwd25TtxIM9LV4misBOIifaXuFhOplngkXYWRaOQets
P7hQRfpoGeB/UXyWjICfnaDLJdTUIWJmlytPmtCfFN0e/avjx0pPOoc3CXN2XX+X
Nwd9eDX4LEn+XjKKooAbsaY1/dRS4INcIZx4hPpImvucuPxXAMqc177JBzOLIPqK
CEr/ZtIUyZHEwWpxn7mn4qI9gLubpz5bokkgJa4V141jXK7nuexmB7K0FDU=
                </ds:X509Certificate>
            </ds:X509Data>
        </ds:KeyInfo>
    </KeyDescriptor>
    <AttributeService
Binding="http://schemas.xmlsoap.org/ws/2003/07/secext"
        Location="https://BER-DC1.Adatum.com/adfs/ls/" />
    <NameIDFormat>urn:mace:shibboleth:1.0:nameIdentifier</NameIDFormat>
</AttributeAuthorityDescriptor>

```

```

</EntityDescriptor>
</EntitiesDescriptor>

```

Attribute Acceptance Policy (AAP.xml)

An AAP is a policy defining rules for the processing of SAML attributes by a Shibboleth service provider. These rules include whether or not to accept the attribute's values.

Here are some of the attributes we set up. Note the inclusion of the namespace as part of the attribute name. Also note the mapping of the UPN claim to the SHIB_TARGETEDID attribute as required by the UK Access Management Federation (this was set to have "enhanced privacy" by the FS-A).

```

<AttributeRule Name="http://schemas.xmlsoap.org/claims/UPN" Header="Shib-TargetedID" Alias="targeted_id">
  <AnySite>
    <AnyValue />
  </AnySite>
</AttributeRule>

<AttributeRule Name="userPrincipalName"
  Namespace="http://schemas.xmlsoap.org/claims" Header="REMOTE_USER"
  Alias="user">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="givenname"
  Namespace="http://schemas.xmlsoap.org/claims" Header="Shib-InetOrgPerson-givenName" CaseSensitive="false">
  <AnySite>
    <AnyValue />
  </AnySite>
</AttributeRule>

<AttributeRule Name="sn" Namespace="http://schemas.xmlsoap.org/claims"
  Header="Shib-Person-surname" CaseSensitive="false">
  <AnySite>
    <AnyValue />
  </AnySite>
</AttributeRule>

<AttributeRule Name="mail" Namespace="http://schemas.xmlsoap.org/claims"
  Header="Shib-InetOrgPerson-mail" CaseSensitive="false">
  <AnySite>
    <AnyValue />
  </AnySite>
</AttributeRule>

```

Setting up the Shibboleth IdP to work with the ADFS FS-R

Installing the Shibboleth ADFS Extensions

The ADFS extensions package, `shib.ADFS.extension-0.8.tar`, can be downloaded from:

<http://shibboleth.internet2.edu/downloads/extensions/>

Extracting the files

It's best to extract the files somewhere where we can safely build them and install a war file into the tomcat5 installation. We used `/opt`, but use an installation base you can refer back to.

```
[root@ber-dc1 opt]# pwd
/opt
[root@ber-dc1 opt]# ls -al
total 14472
drwxr-xr-x  2 root root    4096 Dec  5 09:32 .
drwxr-xr-x 23 root root    4096 Dec  5 08:58 ..
-rw-r--r--  1 root root   6439 Dec  5 09:32 shib.ADFS.extension-0.8.tar
-rw-r--r--  1 root root 14771027 Dec  5 09:32 shibboleth-idp-1.3.tar.gz
[root@ber-dc1 opt]# tar -xvf /opt/shib.ADFS.extension-0.8.tar
shib.ADFS.extension-0.8/
shib.ADFS.extension-0.8/build.properties
shib.ADFS.extension-0.8/src/
shib.ADFS.extension-0.8/src/edu/
shib.ADFS.extension-0.8/src/edu/internet2/
shib.ADFS.extension-0.8/src/edu/internet2/middleware/
shib.ADFS.extension-0.8/src/edu/internet2/middleware/shibboleth/
shib.ADFS.extension-0.8/src/edu/internet2/middleware/shibboleth/common/
shib.ADFS.extension-
0.8/src/edu/internet2/middleware/shibboleth/common/provider/
shib.ADFS.extension-
0.8/src/edu/internet2/middleware/shibboleth/common/provider/UPNNameIdentifier
Mapping.java
shib.ADFS.extension-0.8/src/edu/internet2/middleware/shibboleth/idp/
shib.ADFS.extension-0.8/src/edu/internet2/middleware/shibboleth/idp/provider/
shib.ADFS.extension-
0.8/src/edu/internet2/middleware/shibboleth/idp/provider/ADFS_SSOHandler.java
shib.ADFS.extension-0.8/lib/
```

```
[root@ber-dc1 opt]# ls -al
total 14480
drwxr-xr-x  4 root root    4096 Dec  5 09:39 .
drwxr-xr-x 23 root root    4096 Dec  5 08:58 ..
drwxr-x---  4 1140  502    4096 Feb 14  2006 shib.ADFS.extension-0.8
-rw-r--r--  1 root root   6439 Dec  5 09:32 shib.ADFS.extension-0.8.tar
drwxrwx--- 14  501  501    4096 Dec  5 09:38 shibboleth-idp-1.3c-install
-rw-r--r--  1 root root 14771027 Dec  5 09:32 shibboleth-idp-1.3.tar.gz
[root@ber-dc1 opt]#
```

Building the ADFS Extensions

We need to add these files to the shibboleth-idp-1.3c-install directory so they can be built as part of the IdP ant¹ process.

Note: We found a bug in this. The build properties for the ADFS extensions is incomplete and builds an empty ADFS.jar file. Follow the process below to get around this. It copies two ADFS extension Java classes into a subdirectory of the main Shibboleth source directory where the ant process will find them.

```
[root@ber-dc1 opt]# cd shibboleth-idp-1.3c-
install/src/edu/internet2/middleware/shibboleth/common/provider/
[root@ber-dc1 provider]# cp /opt/shib.ADFS.extension-
0.8/src/edu/internet2/middleware/shibboleth/common/provider/UPNNameIdentifierM
apping.java .
[root@ber-dc1 provider]# cp /opt/shib.ADFS.extension-
0.8/src/edu/internet2/middleware/shibboleth/idp/provider/ADFS_SSOHandler.java.
[root@ber-dc1 provider]# ls -al
total 108
drwxrwx--- 2  501  501    4096 Dec  5 09:47 .
drwxrwx--- 3  501  501    4096 Oct 11  2005 ..
-rw-r--r--  1 root root   19176 Dec  5 09:47 ADFS_SSOHandler.java
-rw-rw----  1  501  501    3225 May 20  2005 AQHNameIdentifierMapping.java
-rw-rw----  1  501  501    2939 May 20  2005 BaseNameIdentifierMapping.java
-rw-rw----  1  501  501    4705 May 20  2005 BasicTrust.java
-rw-rw----  1  501  501   18130 Aug  8  2005 CryptoShibHandle.java
-rw-rw----  1  501  501    2771 Jun 22  2005 PrincipalNameIdentifier.java
-rw-rw----  1  501  501    7534 Jun 22  2005 SharedMemoryShibHandle.java
-rw-rw----  1  501  501   18452 Jun 20  2005 ShibbolethTrust.java
-rw-r--r--  1 root root    3906 Dec  5 09:47 UPNNameIdentifierMapping.java
-rw-rw----  1  501  501    5305 Jun 22  2005
X509SubjectNameNameIdentifierMapping.java
[root@ber-dc1 provider]#
```

This has been logged on Shibboleth Bugzilla as bug number 591.

We also logged bug number 592 regarding a missing adfs.jsp file which is used to report errors from the ADFS server. For now it can be downloaded from the repository at

<http://viewvc.internet2.edu/viewvc.py/shibboleth/java/webApplication/adfs.jsp?view=markup>

Main IdP Configuration (idp.xml)

The Shibboleth identity provider is configured through a variety of XML files. On the Fedora Core 5 systems these are found in the `/usr/local/shibboleth-idp/etc` directory.

We added the following sections into the `idp.xml` file.

Relying Party

```
<RelyingParty name="urn:federation:TreyResearch"
  providerId="urn:federation:Adatum"
  signingCredential="example_cred">
  <NameID nameMapping="shm" />
</RelyingParty>
```

Name Mapping

```
<NameMapping
  xmlns="urn:mace:shibboleth:namemapper:1.0"
  id="shm"
  format="http://schemas.xmlsoap.org/claims/UPN"
  class="edu.internet2.middleware.shibboleth.common.provider.UPNName
  IdentifierMapping"
  handleTTL="28800" scope="Adatum.com" />
```

Credentials

This matches back to the relying party signingCredential above.

```
<Credentials xmlns="urn:mace:shibboleth:credentials:1.0">
  <FileResolver Id="example_cred">
    <Key>
      <Path>file:/etc/httpd/certs/server.key</Path>
    </Key>
    <Certificate>
      <Path>file:/etc/httpd/certs/server.crt</Path>
      <CAPath>file:/etc/httpd/certs/server.crt</CAPath>
    </Certificate>
  </FileResolver>
</Credentials>
```



```

lN1zAgFkAgECMB8GA1UdIwQYMBaAFES1I57gqhP4jNKVRJpnFe0KWSLKMIBGwYDVR0fBIIBEjCC
AQ4wggEkoIIBBqCCAQKGGb1sZGFwOi8vL0NOPVRYZXlyZXNlYXJjaCUyMENBLENOPVBBU1lEQzEs
Q049Q0RQLENOPVB1YmXpYyUyMEtleSUyMFNlcnZpY2VzLENOPVnlcnZpY2VzLENOPUNvbmZpZ3Vy
YXRpb24sREM9VHJleXJlc2VhcmNoLERDPWNvbT9jZXJ0aWZpY2F0ZVJldm9jYXRpb25MaXN0P2Jh
c2U/b2JqZWNOQ2xhc3M9Y1JMRG1zdHJpYnV0aW9uUG9pbnsGQGH0dHA6Ly9wYXItZGMxLnRyZXly
ZXNlYXJjaC5jb20vQ2VydEVucm9sbC9UcmV5cmVzZWZyY2g1MjBDQS5jcmwwggExBggrBgEFBQcB
AQSCASmWggEfMIG1BggrBgEFBQcwAoaBqGxkYXA6Ly8vQ049VHJleXJlc2VhcmNoJTITwQ0EsQ049
QU1BLENOPVB1YmXpYyUyMEtleSUyMFNlcnZpY2VzLENOPVnlcnZpY2VzLENOPUNvbmZpZ3VyYXRp
b24sREM9VHJleXJlc2VhcmNoLERDPWNvbT9jQU1cnRpbZmljYXRlP2Jhc2U/b2JqZWNOQ2xhc3M9
Y2VydGlmawNhdGlvbkF1dGhvcml0eTB1BggrBgEFBQcwAoZZaHR0cDovL3Bhci1kYzEudHJleXJl
c2VhcmNoLmNvbS9DZXJ0RW5yb2xsL1BBU1lEQzEuVHJleXJlc2VhcmNoLmNvbV9UcmV5cmVzZWZy
Y2g1MjBDQS5jcnQwEwYDVR0lBAwwCgYIKwYBBQUHAWAwEwGwYJKwYBBAGCNxUKBA4wDDAKBggrBgEF
BQcDATANBgkqhkiG9w0BAQUFAAOCAQEAhPb+nmip0xQEx2pZmHwaHGdClppM58qYIZtQQo4n1qE
j6RtcZT6bcJZKJkSLhWuVE/EWI3ivID0zkeVhRqN6C0/pVci2h70va9m4x/3mxPlr3TAELPV5ih3
TomVw2xWs/A3Qovu2E3hg/feFotWDLnqk2ydr4HzIn5BcguPZW+Dti37oiUL2yIUWlUps6ikfiV6
C9PWLlIBdoTm3rXdkSlJxAsu/GfevzHahMzoRG8WzIHonp5e+QkJ9AUEzDSpUfjQp1KMTBAPLK1A
DvMEe6qMNGdao9K/FQteE1sn7ztJgS7k6vRYRWR9foVvjd++d1Xjd8Jgw/BdQgobxov6ag==

</ds:X509Certificate>

    </ds:X509Data>

    </ds:KeyInfo>

    </KeyDescriptor>

    <!-- This tells IdPs that you support only the Shib handle
format. -->

<NameIDFormat>urn:mace:shibboleth:1.0:nameIdentifier</NameIDFormat>

    <!--

    This tells IdPs where and how to send authentication
assertions. Mostly

    the SP will tell the IdP what location to use in its request,
but this is how the IdP validates the location and also
figures out which SAML profile to use. There are six listed to
accomodate common testing scenarios used by C++ and Java SP
installations. At deployment time, only the actual endpoints
to be used are needed.

    -->

    <AssertionConsumerService index="1" isDefault="true"
Binding="http://schemas.xmlsoap.org/ws/2003/07/secext"
    Location="https://PAR-DC1.treyresearch.com/adfs/ls/" />
</SPSSODescriptor>

    <!-- This is just information about the entity in human terms. -->

    <Organization>

        <OrganizationName xml:lang="en">Trey
Research</OrganizationName>

```

```

<OrganizationDisplayName xml:lang="en">Trey
Research</OrganizationDisplayName>
    <OrganizationURL
xml:lang="en">http://www.treyresearch.com/</OrganizationURL>
    </Organization>
    <ContactPerson contactType="technical">
        <SurName>Technical Support</SurName>
        <EmailAddress>support@treyrresearch.com</EmailAddress>
    </ContactPerson>
</EntityDescriptor>
</EntitiesDescriptor>

```

Attribute Extraction (resolver.xml)

Like ADFS, Shibboleth doesn't store any attributes about users itself but relies on external data stores to supply the information. Attributes are pulled from data sources using data connectors. The IdP queries the data source for attribute information. Once this information has been pulled in, it is transformed into SAML attributes for transport to the SP. At this point it can also be transformed, filtered, composed, split, and more.

The names for attributes in back-end data stores are decoupled from the expression of attributes on the wire, allowing for arbitrary local naming in the same way as ADFS's outgoing claims mapping. In Shibboleth this mapping is performed within *resolver.xml*. The attributes are then made available to the service provider through the **attribute release policy** files.

For the proof-of-concept we used a MySQL database as the user attribute store, with a test user called *Robert*.

Database connectivity was set up in *resolver.xml* as follows:

```

<JDBCDataConnector id="db1"
    dbURL="jdbc:mysql://localhost/test?user=root&password=Pa$$w0rd"
    dbDriver="com.mysql.jdbc.Driver"
    maxActive="10"
    maxIdle="5">
    <Query>select * from adatumuser where userid = ?</Query>
</JDBCDataConnector>

```

Attribute extraction was set up in *resolver.xml* in the following form – once for each required attribute. Note the value for the namespace.

```

<SimpleAttributeDefinition id="userid" sourceName="userid"
namespace="http://schemas.xmlsoap.org/claims">
    <DataConnectorDependency requires="db1"/>
</SimpleAttributeDefinition>

```

Testing Attribute Resolution

A command-line tool called `resolvertest` that is bundled with Shibboleth can be used to test attribute resolution for a given deployment. The tool can be found in the `IDP_HOME/bin` directory after installing the IdPs. Input it takes `resolver.xml` and the name of a user. It outputs the resulting SAML `<Attribute />` elements, allowing administrators to view the results of attribute resolution without having to continually reload the application.

Prior to executing this application be sure that the `IDP_HOME` environment variable points to your IdP installation.

```
# export IDP_HOME=/usr/local/shibboleth-idp
# $IDP_HOME/bin/resolvertest --user=robert \
    --resolverxml=file:///usr/local/shibboleth-idp/etc/resolver.xml
```

This does not filter the resulting attributes through the applicable attribute release policies so, although it shows the attributes generated by the resolver for a particular user, it does not necessarily reflect what will be released to a particular service provider.

It is possible to run the command with different parameters if you want to test the release policies too. E.g.

```
#IDP_HOME/bin/resolvertest --user=robert \ --
requester=urn:federation:TreyResearch-bilateral\
--responder=urn:federation:Adatum \
--idpXml=file:///usr/local/shibboleth-idp/etc/idp.xml
```

Attribute Release Policy (`arp.site.xml`)

An attribute release policy specifies which attributes are released to which service providers, and even which values for attributes may be permitted or denied. The attribute release policy is a collection of attribute release rules.

Once we had defined all our attributes in `resolver.xml`, we defined a policy to release them to the FS-R in `arp.site.xml`; one `<Rule>` for each attribute. Here are two examples.

```
<Rule>
  <Target>
    <AnyTarget/>
  </Target>
  <Attribute name="userid">
    <AnyValue release="permit"/>
  </Attribute>
</Rule>
<Rule>
  <Target>
    <AnyTarget/>
  </Target>
  <Attribute name="sn">
```

```
<AnyValue release="permit"/>
</Attribute>
</Rule>
```

Certificates

We generated and saved an X.509 key and certificate for the IdP using **openssl**. The resulting files were saved in the `/etc/httpd/certs` directory

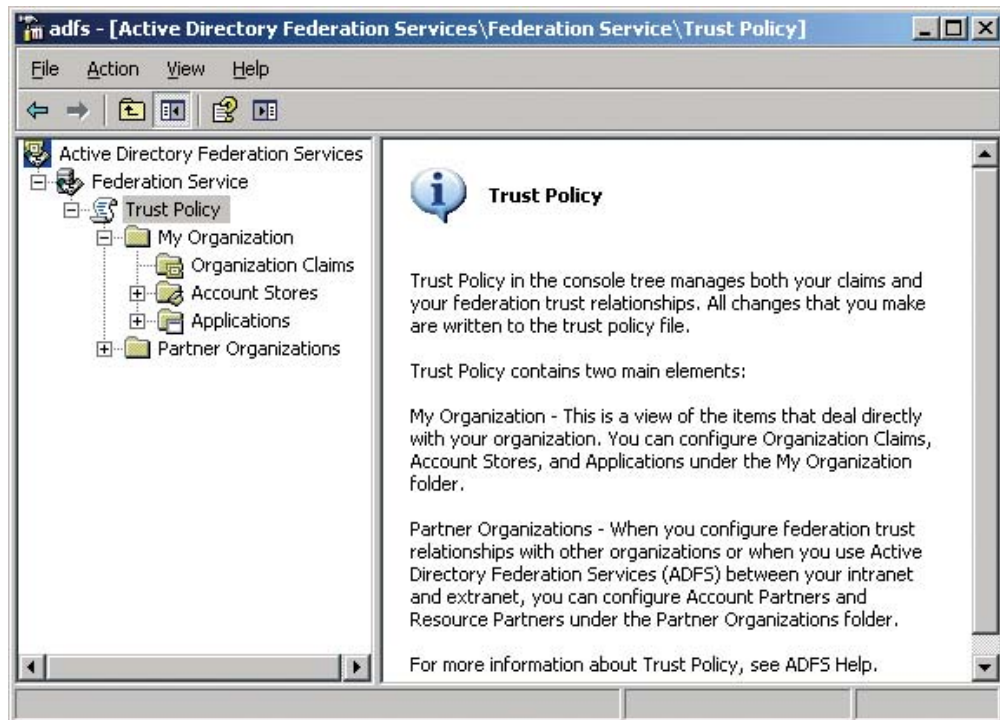
```
# openssl genrsa -out server.key 2048
# openssl req -new -key server.key -x509 -days 365 -out server.crt
```

These are referenced as a credential provider in `idp.xml`.

Note: We used self-signed certificates for this proof-of-concept which would not be recommended for production use.

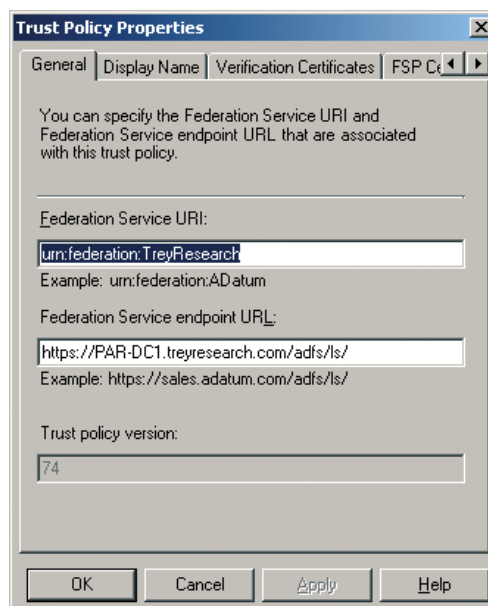
Setting up the ADFS FS-R to work with the Shibboleth IdP

The ADFS FS-R defines itself, and its own configuration, and its account partners in its trust policy. Trust Policy is managed using the ADFS console.



Trust Policy

The properties of the trust policy (right click on Trust Policy and select Properties) describe some of the key elements of the FS-Rs configuration, including information required by account partners.



Federation Service URI

This is a value used by partner organizations and applications to uniquely identify a federation partner.

```
urn:federation:TreyResearch
```

Federation service endpoint URL

This is the address used by partner organizations and applications to send requests and responses.

```
https://PAR-DC1.treyresearch.com/adfs/ls/
```

Token signing certificate

The FS-R uses a token signing certificate to sign the SAML tokens it passes to the web agent. This has to be included in the partner metadata XML file (*adatum-metadata.xml*) on the Shibboleth side so that incoming SAML tokens can be verified by the SP. It can be extracted from the exported trust policy XML file (see appendices) and pasted into the Shibboleth metadata.

```
<X509Certificate>
MIIFiDCCBHCgAwIBAgIKYRSVaAAAAAACDANBgkqhkiG9w0BAQUFADBMRMwEQYKCZImiZPyLQG
GRYDY29tMRwwGgYKCZImiZPyLQBGRYMVHJleXJlc2VhcmNoMRgwFgYDVQQDEw9UcmV5cmVzZW
FyY2ggQ0EwHhcNMDUxMjIyMDkxMTA5WhcNMTEwOTE5MDA5ODAxWjAjaMSEwHwYDVQQDE
xhQQVItREMxLlRyZXNlYXJjaC5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAKN
TDWuqRUPREaWEXohDOcGSfJFuS3HcppqiyHsc40x1136Yt5U1jSk9YTBhXOAMNuQM11LufB2h/ESrW5hCIsatUEY
vbY0qOz6bbByX2876dh5Iq3/d5pp2nFJlxa9qznoJThwigi80n7FPWa55REfMbn2aRbN08sEBW5/
yGwDAGMBAAGjggMWMIIIDEjALBgNVHQ8EBAMCBaAwHQYDVR0OBBYEFdK+lQH+5R5ckT041T
odtuwZ3x2nMD0GCSsGAQQBgjcVbWQwMC4GJisGAQQBgjcVCIA+oTCHwfZjh/2bLtiwSI04s
xSBGYGAVDiGln1zAgFkAgECMB8GA1UdIwQYMBaAFES1I57gqhP4jNKVRJpnFe0KWSLKM
IIBGwYDVR0fBIIBEjCCAQ4wggEKOIIBBqCCAQKGGb1sZGFwOi8vL0NOPVRYZXLyZXNlYX
JjaCUyMENBLENOPVBBU1IEQzEsQ049Q0RQLENOPVB1YmXpYyUyMEtleSUyMFNlcnZpY2
VzLENOPVN1cnZpY2VzLENOPUNvbmZpZ3VyYXRpb24sREM9VHJleXJlc2VhcmNoLERD
PWNvbT9jZXJ0aWZpY2F0ZVZlZm9jYXRpb25MaXN0P2Jhc2U/b2JqZWNOQ2xhc3M9Y
1JMRG1zdHJpYnV0aW9uUG9pbmSGQGh0dHA6Ly9wYXItZGMxLnRyZXNlYXJjaC5jb20vQ
2VydEVucm9sbC9UcmV5cmVzZWFiY2g1MjBDQS5jcmwwGgExBggrBgEFBQcBAQSCASm
wggEfMIG1BggrBgEFBQcwAoaBqGxkYXA6Ly8vQ049VHJleXJlc2VhcmNoJTlWQ0EsQ04
9QU1BLENOPVB1YmXpYyUyMEtleSUyMFNlcnZpY2VzLENOPVN1cnZpY2VzLENOPUNvbm
ZpZ3VyYXRpb24sREM9VHJleXJlc2VhcmNoLERDpWNvbT9jQU1cnRpb25MaXN0P2Jhc2
U/b2JqZWNOQ2xhc3M9Y2VydGlmawNhdGlvbWk1dGhvcml0eTB1BggrBgEFBQcwAoaZ
aHR0cDovL3Bhcil1kYzEudHJleXJlc2VhcmNoLmNvbS9DZXJ0RW5yb2xsL1BBU1IEQzE
uVHJleXJlc2VhcmNoLmNvbV9UcmV5cmVzZWFiY2g1MjBDQS5jcnQwEwYDVR0lBAwwCg
YIKwYBBQUHAWGwYJKwYBBAGCNxUKBA4wDDAKBggrBgEFBQcDATAANBgkqhkiG9w0BAQU
FAAOCAQEAhPb+nmip0xQEx2pZmHwaHGdClppM58qYIZtQQo4n1qEj6RtcZT7bcJZK
JkSLhWuVE/EWI3ivID0zkeVhRqN6C0/pVci2h70va9m4x/3mxPlr3TAEPLV5ih3IomV
w2xWs/A3Qovu2E3hg/feFotWDLnqk2ydr4HzIn5BcguPZW+Dti37oiUL2yIUWlUps6ik
fiV6C9PWL1IBdoTm3rXdkS1JxAsu/GfevzHahMzoRG8WzIHonp5e+QkJ9AUEzDSpUfj
QplKMTBAPLK1ADvME6qMNGdao9K/FQteE1sn7ztJgS7k6vRYRWR9foVvjD++d1Xjd8J
gw/BdQgobxov6ag==</X509Certificate>
```

The trust policy can be exported into an XML file for easy import into an ADFS account partner. The **trust policy for this FS-R** has been included in the appendices. However the format isn't out-of-the-box compatible with the format of the partner metadata required by Shibboleth. It should be possible to write a transformation to accomplish this.

In the meantime, the above information from the trust policy is required from the FS-A to set up the correct metadata (*adatum-metadata.xml*) in the Shibboleth SP.

Defining the Account Partner

Account partners are defined under Partner Organizations within the Trust Policy using the “Add Account Partner Wizard”.

The Shibboleth IdP is configured in exactly the same way as an ADFS account partner in the ADFS administration console. The information entered below in the partner properties was **supplied by the identity provider**, and comes from the partner’s own Shibboleth metadata.

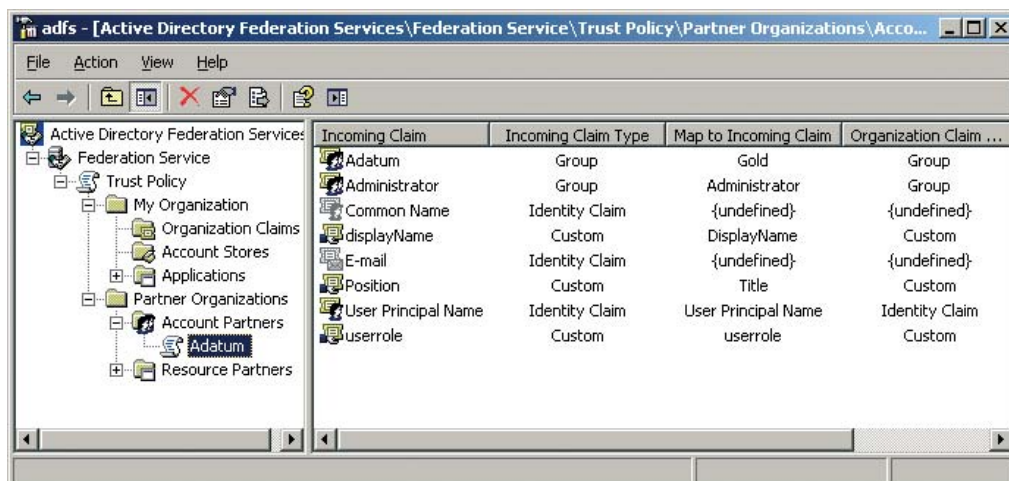
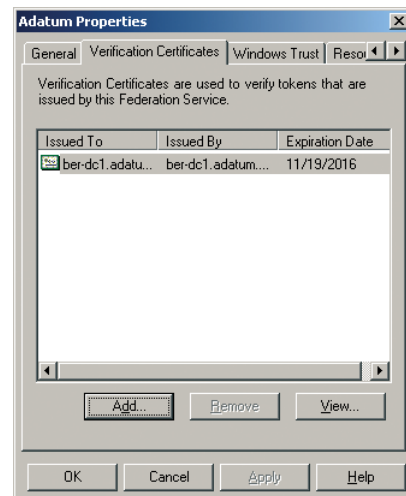
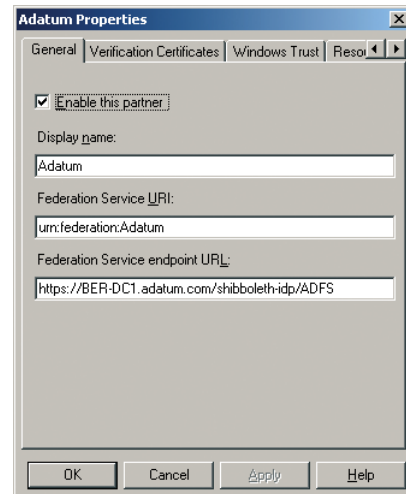
Token Verification Certificate

The token signing certificate must first be obtained from the Shibboleth IdP. This can be achieved through FTP, or using a memory stick. **The token signing certificate** was created as part of the IdP installation and in our implementation was saved in */etc/httpd/certs* on the IdP server.

The token signing certificate from the Shibboleth IdP must be associated with the account partner defined in ADFS. Right click on the account partner (in our case Adatum) and select **Properties**. Select the **Verification Certificates** tab and click **Add**.

Incoming Claims Mapping

The agreed set of claims is added to the account partner definition, and the claim names are mapped to the names agreed with the account partner.



Summary

The goals of the proof of concept were all achieved. We successfully demonstrated interoperability between:

- An ADFS FS-A and a Shibboleth SP
- A Shibboleth IdP and an ADFS FS-R

We also were able to successfully “hide” personal identifying information – in the form of the hashed and scoped SHIB_TARGETEDID attribute - by the use of ADFS “enhanced privacy” options.

The main challenge was one of understanding how the component parts of one system related to their equivalents in the other and this is why we have included a “**Language Barrier**” chapter in this document. In particular those of us with ADFS experience, who are used to configuring a federated environment through the use of wizards and GUI tools, had to adjust to performing similar tasks by making changes to multiple XML files. Inevitably this resulted in accidental mis-configurations through inconsistently applied changes (for example, where the content of one file contains references to another) as well as through simple typographical errors.

However once we had overcome this hurdle, it became clear that achieving interoperability is not hard to do; in essence it is just a matter of exchanging a common set of configuration parameters which are then entered into the appropriate system.

The Shibboleth systems need to have the ADFS extensions installed and configured in order to enable the WS-Federation protocol end points, but once this is done then federation partners are set up using the familiar (to Shibboleth folks) metadata XML files. It is quite straightforward to build these from information provided in the ADFS trust policy export XML files.

From the standpoint of the ADFS servers, there is no difference in configuration of a Shibboleth federation partner to configuration of another ADFS federation partner.

It should also be noted here that we were working in an isolated environment and dealing with a limited usage scenario. This means that some of the Shibboleth configuration examples in this document, whilst sufficient for the PoC would not necessarily be optimal in a production deployment. For example we’ve used default rather than partner-specific *sessionInitiators*.

Related Links

See the following resources for further information:

- Oxford Computer Group <http://www.oxfordcomputergroup.com/microsoftida>
- Internet2 Shibboleth Site at <http://shibboleth.internet2.com>
- UK Access Management Federation at <http://www.ukfederation.org.uk/>
- Microsoft ADFS page at <http://www.microsoft.com/windowsserver2003/techinfo/overview/adfsoverview.msp>

For the latest information about Windows Server 2003, see the **Windows Server 2003 Web site** at <http://www.microsoft.com/windowsserver2003>.

Appendices

The following appendices contain the configuration files as they exist on the servers, with no attempt made to tidy them up. They are just as they were copied from the VPCs concerned.

Appendix I – ADFS FS-A Trust Policy

```

<fed:federationTrustAgreement
xmlns:fed="http://schemas.xmlsoap.org/ws/2004/01/Federation/TrustAgreement">
  <fed:accountRealmFriendlyName>adatum.com</fed:accountRealmFriendlyName>
  <fed:accountRealmURI>
    <wsp:EndpointReference
xmlns:wsp="http://schemas.xmlsoap.org/ws/2003/03/addressing">
      <wsa:Address URI="urn:federation:Adatum"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing" />
    </wsp:EndpointReference>
  </fed:accountRealmURI>
  <fed:accountRealmSTSURL>
    <wsp:EndpointReference
xmlns:wsp="http://schemas.xmlsoap.org/ws/2003/03/addressing">
      <wsa:Address URI="https://BER-DC1.Adatum.com/adfs/ls/"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing" />
    </wsp:EndpointReference>
  </fed:accountRealmSTSURL>

<fed:resourceRealmFriendlyName>adatum.com</fed:resourceRealmFriendlyName>
  <fed:resourceRealmURI>
    <wsp:EndpointReference
xmlns:wsp="http://schemas.xmlsoap.org/ws/2003/03/addressing">
      <wsa:Address URI="urn:federation:Adatum"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing" />
    </wsp:EndpointReference>
  </fed:resourceRealmURI>
  <fed:resourceRealmSTSURL>
    <wsp:EndpointReference
xmlns:wsp="http://schemas.xmlsoap.org/ws/2003/03/addressing">
      <wsp:Address URI="https://BER-DC1.Adatum.com/adfs/ls/"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing" />
    </wsp:EndpointReference>
  </fed:resourceRealmSTSURL>
  <fed:accountRealmCertChain>
    <dsig:KeyInfo xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
      <X509Data>

```

```

<X509Certificate>MIICyDCCAbSgAwIBAgIQsVv1sV0V36RHicKd5gWLLjAJBgUrDgMCHQUAMCQx
IjAgBgNVBAMTGUZlZGVyYXRpb24gU2VydjVIEFJFUi1EQzEwHhcNMDYwOTA1MjE1MTE5WncNMDcwO
TA2MDM1MTE5WjAkMSIwIAYDVQQDExlGZWRLcmF0aW9uIFNlcnZlciBCRVItREMxMIIBIjANBgkqhki
g9w0BAQEFAAOCAQ8AMIIBCgKCAQEAx0wsqoDvMhtBsquv3Mi9+vv8PuyknyoyWxvAbQ6axYP4N+
oJYugyx/7rEky/nr6u/cZct1XZ5UWfKrEx8RW6Re4eoY7d9ua1JbleJYmx97LXEZKNC1TrTALDMY2
J0yeQPFq0UQOYLcDDhpIBi8PJ48gmsQtKjI0/lrxXGsjs8f97gzEsA970372ZnYkzi9rqT4C42Soz
5SaRwFlz4glDa99QBkNKgmnLjWUn1sjgaCQUdWI794dWIt22hlkvziMyKvqD8/ygPclPgM2RQH5hL
aU9UFbnqQ+jx36Kr2raIg/2Yp9QLFoEEQfe4bUgKUtR+sgocvgyllUgAZuYTOyPwIDAQABMAkGBSs
OAwIdBQADggEBAMD15++UrKv3SIEohqe7SEd/nh3j1Rc15hBx9W4dAHicZ+uokVYzpK2L4F8dQYEE
YT/RR2UCbChBFZzjJWaXik9SkBvEWKU6vCeRnBc5faNajGPKSXwd25TTxIM9LV4misBOIifaXuFhO
plngkXYWRaOQEtSP7hQRfpoGeb/UXyWjICfnaDLJdTUIWJmlytPmtCfFN0e/avjx0pPOoc3CXN2XX
+XNwd9eDX4LEn+XjKKooAbsaYl/dRS4INcIZx4hPpImvucuPxXAMqc177JBzOLIPqKCEr/ZtIUyZH
EwWpxn7mn4qI9gLubpz5bokkgJa4V141ljXK7nuexmB7K0FDU=</X509Certificate>

  </X509Data>

  </dsig:KeyInfo>

</fed:accountRealmCertChain>

<fed:allowedClaims IsWindowsRealm="false">

</fed:allowedClaims>

  <fed:tokenType>urn:oasis:names:tc:SAML:1.0:assertion</fed:tokenType>

</fed:federationTrustAgreement>

```

Appendix 2 - ADFS FS-R Trust Policy

```

<fed:federationTrustAgreement
xmlns:fed="http://schemas.xmlsoap.org/ws/2004/01/Federation/TrustAgreement">

  <fed:accountRealmFriendlyName>Treyresearch.com</fed:accountRealmFriendlyName>

  <fed:accountRealmURI>

    <wsp:EndpointReference
xmlns:wsp="http://schemas.xmlsoap.org/ws/2003/03/addressing">

      <wsa:Address URI="urn:federation:TreyResearch"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing" />

    </wsp:EndpointReference>

  </fed:accountRealmURI>

  <fed:accountRealmSTSURL>

    <wsp:EndpointReference
xmlns:wsp="http://schemas.xmlsoap.org/ws/2003/03/addressing">

      <wsa:Address URI="https://PAR-DC1.treyresearch.com/adfs/ls/"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing" />

    </wsp:EndpointReference>

  </fed:accountRealmSTSURL>

  <fed:resourceRealmFriendlyName>Treyresearch.com</fed:resourceRealmFriendlyName>
</fed:federationTrustAgreement>

```

```

<fed:resourceRealmURI>
  <wsp:EndpointReference
xmlns:wsp="http://schemas.xmlsoap.org/ws/2003/03/addressing">
    <wsa:Address URI="urn:federation:TreyResearch"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing" />
  </wsp:EndpointReference>
</fed:resourceRealmURI>
<fed:resourceRealmSTSURL>
  <wsp:EndpointReference
xmlns:wsp="http://schemas.xmlsoap.org/ws/2003/03/addressing">
    <wsp:Address URI="https://PAR-DC1.treyresearch.com/adfs/ls/"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing" />
  </wsp:EndpointReference>
</fed:resourceRealmSTSURL>
<fed:accountRealmCertChain>
  <dsig:KeyInfo xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
    <X509Data>

```

```

<X509Certificate>MIIFiDCCBHCgAwIBAgIKYRsVaAAAAAACDANBgkqhkiG9w0BAQUFADBMMRM
wEQYKCZImiZPyLQG0BGRYDY29tMRwwGgYKCZImiZPyLQG0BGRYDYMVHJleXJlc2VhcmNoMRgwFgYDVQQ
DEw9UcmV5cmVzZWZyY2ggQ0EwHhcNMDUxMjIyMDkxMTA5WhcNMTAwOTE5MDAyODAxWjA5MSEwHwY
DVQ0EExhQ0VlRExhQ0VlRExhQ0VlRExhQ0VlRExhQ0VlRExhQ0VlRExhQ0VlRExhQ0VlRExhQ0VlRE
BAKNTDWeuqRUPREaWEXohDOcGSfJFuS3HcppqiyHsc40x1136Yt5U1jSk9YTBhXOAMNuQM11LufB
2h/ESrW5hCIsatUEYvbyOqOz6bbByX2876dh5Ig3/d5pp2nFJlxa9qnzoJThwigi80n7FPWa55RE
fMbn2aRbn08sEBW5/yGwDagMBAAGjggMwMIIDEjALBgNVHQ8EBAMCBaAwHQYDVR0OBBYEFdK+lQH
+5R5ckT041TodtuwZ3x2nMD0GCSsGAQQBgjcVBwQwMC4GJisGAQQBgjcVCIA+oTCHwfZjh/2bLti
wSIO4sxSBGYGAvDiG1N1zAgFkAgECMB8GA1UdIwQYMBaAFES1I57gqhP4jNKVRJpnFe0KWSLKMII
BGwYDVR0fBIIBEjCCAQ4wggEkoIIBBqCCAQKGGb1sZGFwOi8vL0NOPVRYZlYXNlYXJjaCUyMEN
BLENOPVBBUi1EQzEsQ049Q0RQLENOPVB1YmXpYyUyMEtleSUyMFNlcnZpY2VzLENOPVNlcnZpY2V
zLENOPUNvbmZpZ3VyYXRpb24sREM9VHJleXJlc2VhcmNoLERDPWNvbT9jZXJ0awZpY2F0ZVJldm9
jYXRpb25MaXN0P2Jhc2U/b2JqZWN0Q2xhc3M9Y1JMRGlzdHJpYnV0aw9uUG9pbnsGQGH0dHA6Ly9
wYXItZGMxLnRyZXlYXNlYXJjaCUyMENBLENOPVB1YmXpYyUyMEtleSUyMFNlcnZpY2VzLENOPVNlcnZpY2V
zLENOPUNvbmZpZ3VyYXRpb24sREM9VHJleXJlc2VhcmNoLERDPWNvbT9jQUNlcnRzZmljYXRlP2Jhc2U
/b2JqZWN0Q2xhc3M9Y2VydGlmawNhdG1vbKf1dGhvcml0eTB1BggrBgEFBQcwoAaBQgXkYXA6Ly8vQ049VHJleXJlc2V
hcmNoJTlWQ0EsQ049Q0U1BLENOPVB1YmXpYyUyMEtleSUyMFNlcnZpY2VzLENOPVNlcnZpY2VzLEN
OPUNvbmZpZ3VyYXRpb24sREM9VHJleXJlc2VhcmNoLERDPWNvbT9jQUNlcnRzZmljYXRlP2Jhc2U
/b2JqZWN0Q2xhc3M9Y2VydGlmawNhdG1vbKf1dGhvcml0eTB1BggrBgEFBQcwoAaBQgXkYXA6Ly8vQ049VHJleXJlc2V
hcmNoLmNvbS9DZXJ0RW5yb2xsL1BBU1lEQzEuVHJleXJlc2VhcmNoLmN
vbV9UcmV5cmVzZWZyY2ggQ0EwHhcNMDUxMjIyMDkxMTA5WhcNMTAwOTE5MDAyODAxWjA5MSEwHwY
KBA4wDDAKBggrBgEFBQcDATANBgkqhkiG9w0BAQUFAAOCAQEAPb+nmip0xQEx2pZmHwaHGdClp
pM58qYIZtQ0o4n1qEj6RtcZT6bcJZKJkSLhWuVE/EWI3ivID0zkeVhRqN6C0/pVci2h70va9m4x/
3mxP1r3TAE1PV5ih3IomVw2xWs/A3Qovu2E3hg/feFotWDLnqk2ydr4HzIn5BcguPZW+Dti37oiU
L2yIUWlUps6ikfiV6C9PWL1IBdoTm3rXdkSlJxAsu/GfevzHahMzoRG8WzIHonp5e+QkJ9AUEzDS
pUfjQplKMTBAPLk1ADvMEe6qMNGdao9K/FQteE1sn7ztJgS7k6vRyRWR9foVjvd++d1Xjd8Jgw/B
dQgobxov6ag==</X509Certificate>

```

```

<X509Certificate>MIIEmDCCA4CgAwIBAgIQMRm3ve0mcK9Mwa2i09yYHTANBgkqhkiG9w0BAQU
FADBMMRMwEQYKCZImiZPyLQG0BGRYDY29tMRwwGgYKCZImiZPyLQG0BGRYDYMVHJleXJlc2VhcmNoMRg

```

```
wFgYDVQQDEw9UcmV5cmVzZWZlY2ggQ0EwHhcNMDUwOTE5MDAxODM5WhcNMTAwOTE5MDAyODAxWjB
NMRMwEQYKZImiZPyLQGBGRYDY29tMRwwGgYKZImiZPyLQGBGRYMHJleXJlc2VhcmNoMRgWfGy
DVQQDEw9UcmV5cmVzZWZlY2ggQ0EwgGgEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDn3Xl
pWDFbMRQvewzI/C65NYbESZEoVBuqWHGkGxDPWEEM6YIXSRjOWY2IBX1sdLsHSA87ha2TcTkD3zn
7HqVgcpBCiEjos+4cLkpDJBmUM4XzcPqUK1osDxxKz5xSAsNcVcWDOlm7PfkRnZE1ZdB+Vffq0v
QX7ziuxSb/n8QrtGE1nulitRhqcoYlhzcg/TA1pfKfrK0IOKcRkGe/ZxTidipZ0DLmrATUi7xxB0
sj21tue6upzUsViJSase3E5sZUxlqYs5T698W9ZTKi2Zb21D9w6UsDpukLb+9kz436h0RVUgT7XB
B9sARgpsVhyx7VfcY/a66UysVKHK2pU/VAgMBAAGjggFyMIIBbjALBgNVHQ8EBAMCAYYwDwYDVR0
TAQH/BAUwAwEB/zAdBgNVHQ4EFgQUSzUjnuCqE/im0pVEmmcV7QpZIsowggEbBgNVHR8EEgESMII
BDjCCAQgggEGoIIBAoaBvWxkYXA6Ly8vQ049VHJleXJlc2VhcmNoJTIwQ0EsQ049UEFSLURDMSx
DTj1DRFASQ049UHvibG1jJTIwS2V5JTIwU2VydmljZXMsQ049U2VydmljZXMsQ049Q29uZmlndXJ
hdGlvbixEQz1UcmV5cmVzZWZlY2gsREM9Y29tP2N1cnRpZmljYXRlUmV2b2NhdGlvbkxpc3Q/YmF
zZT9vYmplY3RDbGFzc3l1UkxEXXN0cmliXDRpb25Qb2ludlZAAHR0cDovL3Bhcik1kYzEudHJleXJ
lc2VhcmNoLmNvbS9DZXJ0RW5yb2xsL1RyZXlyZXNlYXJjaCUyMENBLmNybDAQBGRBgEEAYI3FQE
EAwIBADANBgkqhkiG9w0BAQUFAAOCAQEAS8CusuV2HSPkfomXNYqu5MMSUoFsbXTB/bGGJP5dPGF
ndJ51UtyolyyqlsZkC8Z5VowgGtgTiWPykZuEkoHDYtuPrhtoYEAi56CveJCUbEQ1kYoYiyw4rEG
8vPwFQ//O5kvPJVGM/vB2tuvat1FwbrcSD08Ls6wHq6c4Dxk1NOF5CJuVsgk7AtqIwou4aK14ArqGE
fg7R5DxVjtFBie5WAPiewJY6QsLJ4Wx6mMAa3WmSDMLQgEQ2//1Gu922C6k6hnmfBirgWv/DEDkP
gPmYqrB2oV498XS2wx2b55G0TID/sVe8Kd1FcSgUJEgmAm+cuzg7P1lBMZ7ui8eBmMd3VOg==/X
509Certificate>
</X509Data>
</dsig:KeyInfo>
</fed:accountRealmCertChain>
<fed:allowedClaims IsWindowsRealm="false">
</fed:allowedClaims>
<fed:tokenType>urn:oasis:names:tc:SAML:1.0:assertion</fed:tokenType>
</fed:federationTrustAgreement>
```

Appendix 3 – Shibboleth IdP Configuration Files

This appendix contains listings of the Shibboleth identity provider configuration files, just as they came off the servers, but with areas of special interest - that are discussed earlier in this document - highlighted.

idp.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Shibboleth Identity Provider configuration -->

<IdPConfig
xmlns="urn:mace:shibboleth:idp:config:1.0"
xmlns:cred="urn:mace:shibboleth:credentials:1.0"
xmlns:name="urn:mace:shibboleth:namemapper:1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:mace:shibboleth:idp:config:1.0
../schemas/shibboleth-idpconfig-1.0.xsd"
AAUrl="https://ber-dc1.adatum.com:8443/shibboleth-idp/AA"
```

```

resolverConfig="file:/usr/local/shibboleth-idp/etc/resolver.jdbc.xml"
    defaultRelyingParty="urn:federation:TreyResearch"
    providerId="urn:federation:Adatum">

    <!-- This section contains configuration options that apply only to
a site or group of sites

        The signing Credential attribute value here needs to match
the name of some credential defined
        below -->
    <RelyingParty name="urn:federation:TreyResearch"
        providerId="urn:federation:Adatum"
        signingCredential="example_cred">
        <NameID nameMapping="shm"/>
    </RelyingParty>

    <!-- This section contains configuration options that apply only to a site
or group of sites

        This would normally be adjusted when a new federation or bilateral
trust relationship is established -->
    <RelyingParty name="urn:mace:shibboleth:examples"
signingCredential="example_cred"> <!-- (signingCredential) must correspond
to a <Credential/> element below -->
        <NameID nameMapping="shm"/> <!-- (nameMapping) must correspond to a
<NameMapping/> element below -->
    </RelyingParty>

    <!-- InQueue example (the schemaHack is needed for 1.1/1.2 SPs)-->
    <!--
    <RelyingParty name="urn:mace:inqueue" signingCredential="inqueue_cred"
        schemaHack="true">
        <NameID nameMapping="shm"/>
    </RelyingParty> -->

    <!-- Configuration for the attribute release policy engine

        For most configurations this won't need adjustment -->
    <ReleasePolicyEngine>
        <ArpRepository
implementation="edu.internet2.middleware.shibboleth.aa.arp.provider.FileSystemArpRepository">

```

```

        <Path>file:/usr/local/shibboleth-idp/etc/arps/</Path>
    </ArpRepository>
</ReleasePolicyEngine>

    <!-- Logging Configuration

        The defaults work fine in this section, but it is sometimes helpful
to use "DEBUG" as the level for the <ErrorLog/> when trying to diagnose
problems -->

    <Logging>

        <ErrorLog level="DEBUG" location="file:/usr/local/shibboleth-
idp/logs/shib-error.log" />

        <TransactionLog level="DEBUG" location="file:/usr/local/shibboleth-
idp/logs/shib-access.log" />

    </Logging>

    <!-- Uncomment the configuration section below and comment out the one
above if you would like to manually configure log4j -->

    <!--
<Logging>

    <Log4JConfig location="file:///tmp/log4j.properties" />
</Logging> -->

    <!-- This configuration section determines how Shibboleth maps between
SAML Subjects and local principals.

        The default mapping uses shibboleth handles, but other formats can be
added.

        The mappings listed here are only active when they are referenced
within a <RelyingParty/> element above -->

    <NameMapping

        xmlns="urn:mace:shibboleth:namemapper:1.0"

        id="shm"

        format="urn:mace:shibboleth:1.0:nameIdentifier"

        type="SharedMemoryShibHandle"

        handleTTL="28800"/>

    <NameMapping

        xmlns="urn:mace:shibboleth:namemapper:1.0"

        id="shm"

        format="http://schemas.xmlsoap.org/claims/UPN"

class="edu.internet2.middleware.shibboleth.common.provider.UPNNameIdentifier
Mapping"

```

```

        handleTTL="28800" scope="Adatum.com"/>

    <!-- Determines how SAML artifacts are stored and retrieved
        The (sourceLocation) attribute must be specified when using type 2
        artifacts -->
    <ArtifactMapper
        implementation="edu.internet2.middleware.shibboleth.artifact.provider.Memory
        ArtifactMapper" />

    <!-- This configuration section determines the keys/certs to be used when
        signing SAML assertions -->

    <!-- The credentials listed here are used when referenced within
    <RelyingParty/> elements above -->

    <Credentials xmlns="urn:mace:shibboleth:credentials:1.0">

        <!-- InQueue example (Deployments would need to generate an InQueue-
        compatible certificate) -->

            <FileResolver Id="example_cred">

                <Key>

<Path>file:/etc/httpd/certs/server.key</Path>

                </Key>

                <Certificate>

<Path>file:/etc/httpd/certs/server.crt</Path>

<CAPath>file:/etc/httpd/certs/server.crt</CAPath>

                </Certificate>

            </FileResolver>

        </Credentials>

    <!-- Protocol handlers specify what type of requests the IdP can respond
    to. The default set listed here should work

        for most configurations. Modifications to this section may require
        modifications to the deployment descriptor -->

    <ProtocolHandler
        implementation="edu.internet2.middleware.shibboleth.idp.provider.ShibbolethV
        1SSOHandler">

        <Location>https://[^:/>+(:443|80))?.*/shibboleth-idp/SSO</Location>
    <!-- regex works when using default protocol ports -->

    </ProtocolHandler>

    <ProtocolHandler

```

```

implementation="edu.internet2.middleware.shibboleth.idp.provider.SAMLv1_AttributeQueryHandler">
    <Location>.+:8443/shibboleth-idp/AA</Location>
</ProtocolHandler>

<ProtocolHandler
implementation="edu.internet2.middleware.shibboleth.idp.provider.SAMLv1_ArtifactQueryHandler">
    <Location>.+:8443/shibboleth-idp/Artifact</Location>
</ProtocolHandler>

<ProtocolHandler
implementation="edu.internet2.middleware.shibboleth.idp.provider.Shibboleth_StatusHandler">
    <Location>https://[^:/]+(:443)?/shibboleth-idp/Status</Location>
</ProtocolHandler>

<ProtocolHandler
implementation="edu.internet2.middleware.shibboleth.idp.provider.ADFS_SSOHandler">
    <Location>https://[^:/]+(:443)?/shibboleth-idp/ADFS</Location>
</ProtocolHandler>

<!-- This section configures the loading of SAML2 metadata, which contains
information about system entities and how to authenticate them. The
metadatatool utility can be used to keep federation metadata files in synch.

Metadata can also be placed directly within this these elements. -->
<MetadataProvider
type="edu.internet2.middleware.shibboleth.metadata.provider.XMLMetadata"
    uri="file:/usr/local/shibboleth-idp/etc/trey-metadata.xml"/>

<!-- InQueue example (Deployments would need to get updated InQueue
metadata) -->

<!--
<MetadataProvider
type="edu.internet2.middleware.shibboleth.metadata.provider.XMLMetadata"
    uri="file:/usr/local/shibboleth-idp/etc/IQ-metadata.xml"/> -
-></IdPConfig>

```

resolver.xml

```
<AttributeResolver xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:mace:shibboleth:resolver:1.0"
xsi:schemaLocation="urn:mace:shibboleth:resolver:1.0 shibboleth-resolver-
1.0.xsd">

  <!-- Simple example JDBC Connector setup with a query based on principal
name -->

  <SimpleAttributeDefinition id="userid" sourceName="userid"
namespace="http://schemas.xmlsoap.org/claims">
    <DataConnectorDependency requires="db1"/>
  </SimpleAttributeDefinition>

  <SimpleAttributeDefinition id="sn" sourceName="sn"
namespace="http://schemas.xmlsoap.org/claims">
    <DataConnectorDependency requires="db1"/>
  </SimpleAttributeDefinition>

  <SimpleAttributeDefinition id="givenname" sourceName="givenname"
namespace="http://schemas.xmlsoap.org/claims">
    <DataConnectorDependency requires="db1"/>
  </SimpleAttributeDefinition>

  <SimpleAttributeDefinition id="cn" sourceName="cn"
namespace="http://schemas.xmlsoap.org/claims">
    <DataConnectorDependency requires="db1"/>
  </SimpleAttributeDefinition>

  <SimpleAttributeDefinition id="userrole" sourceName="userrole"
namespace="http://schemas.xmlsoap.org/claims">
    <DataConnectorDependency requires="db1"/>
  </SimpleAttributeDefinition>

  <SimpleAttributeDefinition id="partner" sourceName="partner"
namespace="http://schemas.xmlsoap.org/claims">
    <DataConnectorDependency requires="db1"/>
  </SimpleAttributeDefinition>
</AttributeResolver>
```

```

<SimpleAttributeDefinition id="mail" sourceName="mail"
namespace="http://schemas.xmlsoap.org/claims">
    <DataConnectorDependency requires="db1"/>
</SimpleAttributeDefinition>

<JDBCDataConnector id="db1"
    dbURL="jdbc:mysql://localhost/test?user=root&password=Pa$$w0rd"
    dbDriver="com.mysql.jdbc.Driver"
    maxActive="10"
    maxIdle="5">
    <Query>select * from adatumuser where userid = ?</Query>
</JDBCDataConnector>

<!-- A more complicated example, where a query is constructed based on
dependencies on another Data Connector and an Attribute Definition -->

<!-- <SimpleAttributeDefinition id="urn:mace:dir:attribute-
def:eduPersonAffiliation">
    <DataConnectorDependency requires="echo"/>
</SimpleAttributeDefinition>

<SimpleAttributeDefinition id="urn:x-mace:shibboleth:date">
    <DataConnectorDependency requires="db2"/>
</SimpleAttributeDefinition>

<JDBCDataConnector id="db2"
    dbURL="jdbc:postgresql://test.example.edu/test?user=postgres&password=te
st"
    dbDriver="org.postgresql.Driver"
    maxActive="10"
    maxIdle="5">
    <DataConnectorDependency requires="echo"/>
    <AttributeDependency requires="urn:mace:dir:attribute-
def:eduPersonEntitlement"/>
    <Query>select date from foo where principalName = ? and entitlement =
?</Query>

```

```

<StatementCreator
class="edu.internet2.middleware.shibboleth.aa.attrresolv.provider.Dependency
StatementCreator">
    <Parameter type="String"
attributeName="eduPersonPrincipalName" connectorId="echo"
nullMissing="false"></Parameter>
    <Parameter type="String"
attributeName="urn:mace:dir:attribute-def:eduPersonEntitlement"
nullMissing="false"></Parameter>
    </StatementCreator>
</JDBCDataConnector>

<CustomDataConnector id="echo"
class="edu.internet2.middleware.shibboleth.aa.attrresolv.provider.SampleConn
ector"/>
-->
</AttributeResolver>

```

arps-site.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<AttributeReleasePolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="urn:mace:shibboleth:arp:1.0"
xsi:schemaLocation="urn:mace:shibboleth:arp:1.0 shibboleth-arp-1.0.xsd" >
    <Description>Simplest possible ARP.</Description>
    <Rule>
        <Target>
            <AnyTarget/>
        </Target>
        <Attribute name="userid">
            <AnyValue release="permit"/>
        </Attribute>
    </Rule>
    <Rule>
        <Target>
            <AnyTarget/>
        </Target>
        <Attribute name="sn">
            <AnyValue release="permit"/>
        </Attribute>
    </Rule>

```

```
<Rule>
  <Target>
    <AnyTarget/>
  </Target>
  <Attribute name="givenname">
    <AnyValue release="permit"/>
  </Attribute>
</Rule>
<Rule>
  <Target>
    <AnyTarget/>
  </Target>
  <Attribute name="cn">
    <AnyValue release="permit"/>
  </Attribute>
</Rule>
<Rule>
  <Target>
    <AnyTarget/>
  </Target>
  <Attribute name="userrole">
    <AnyValue release="permit"/>
  </Attribute>
</Rule>
<Rule>
  <Target>
    <AnyTarget/>
  </Target>
  <Attribute name="partner">
    <AnyValue release="permit"/>
  </Attribute>
</Rule>
<Rule>
  <Target>
    <AnyTarget/>
  </Target>
```

```

    <Attribute name="mail">
        <AnyValue release="permit"/>
    </Attribute>
</Rule>
</AttributeReleasePolicy>

```

Appendix 4 – Shibboleth SP Configuration Files

This appendix contains listings of the Shibboleth service provider configuration files, just as they came off the servers, but with areas of special interest (that are discussed earlier in this document) highlighted.

shibboleth.xml

```

<SPConfig xmlns="urn:mace:shibboleth:target:config:1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:mace:shibboleth:target:config:1.0
  /usr/share/xml/shibboleth/shibboleth-targetconfig-1.0.xsd"
  logger="/etc/shibboleth/shibboleth.logger" clockSkew="180">

  <!-- These extensions are "universal", loaded by all Shibboleth-aware
  processes. -->
  <Extensions>
    <Library path="/usr/libexec/xmlproviders.so" fatal="true"/>
    <!--
      Added by Simon
    -->
    <Library path="/usr/libexec/adfs.so" fatal="true"/>
  </Extensions>

  <!-- The Global section pertains to shared Shibboleth processes like the
  shibd daemon. -->
  <Global logger="/etc/shibboleth/shibd.logger">

    <!--
    <Extensions>
      <Library path="/usr/libexec/shib-mysql-ccache.so"
  fatal="false"/>
    </Extensions>
    -->

    <!-- Only one listener can be defined. -->

```

```

<UnixListener address="/var/run/shib-shar.sock"/>

<!-- <TCPListener address="127.0.0.1" port="12345" acl="127.0.0.1"/>
-->

<!--
See Wiki for details:

    cacheTimeout - how long before expired sessions are purged
    from the cache

    AATimeout - how long to wait for an AA to respond

    AAConnectTimeout - how long to wait while connecting to an AA

    defaultLifetime - if attributes come back without guidance,
    how long should they last?

    strictValidity - if we have expired attrs, and can't get new
    ones, keep using them?

    propagateErrors - suppress errors while getting attrs or let
    user see them?

    retryInterval - if propagateErrors is false and query fails,
    how long to wait before trying again

    Only one session cache can be defined.

-->

    <MemorySessionCache cleanupInterval="300" cacheTimeout="3600"
    AATimeout="30" AAConnectTimeout="15"

        defaultLifetime="1800" retryInterval="300"
    strictValidity="false" propagateErrors="false"/>

    <!--

    <MySQLSessionCache cleanupInterval="300" cacheTimeout="3600"
    AATimeout="30" AAConnectTimeout="15"

        defaultLifetime="1800" retryInterval="300"
    strictValidity="false" propagateErrors="false"

        mysqlTimeout="14400" storeAttributes="false">

    <Argument>#{x2D};#{x2D};language=/usr/share/english</Argument>

        <Argument>#{x2D};#{x2D};datadir=/usr/data</Argument>

    </MySQLSessionCache>

-->

<!-- Default replay cache is in-memory. -->

<!--

<MySQLReplayCache>

<Argument>#{x2D};#{x2D};language=/usr/share/english</Argument>

```

```

<Argument>&#x2D;&#x2D;datadir=/usr/data</Argument>
    </MySQLReplayCache>
    -->
</Global>

<!-- The Local section pertains to resource-serving processes (often
process pools) like web servers. -->

<Local logger="/etc/shibboleth/native.logger" localRelayState="true">
    <!--
        To customize behavior, map hostnames and path components to
applicationId and other settings.

        See: https://authdev.it.ohio-
state.edu/twiki/bin/view/Shibboleth/RequestMap

    -->
    <RequestMapProvider
type="edu.internet2.middleware.shibboleth.sp.provider.NativeRequestMapProvid
er">
        <RequestMap applicationId="default">
            <!--
                This requires a session for documents in /secure on
the containing host with http and https on the default
ports. Note that the name and port in the <Host>
elements MUST match

                Apache's ServerName and Port directives or the IIS
Site name in the <ISAPI> element below. You should
also be sure that Apache's UseCanonicalName setting is
On

            -->
            <Host name="fc5srv.contoso.com">
                <Path name="secure" authType="shibboleth"
requireSession="true"/>
            </Host>

            <!-- Example shows the vhost "sp-admin.example.org"
assigned to a separate <Application> -->
            <!--
                <Host name="sp-admin.example.org"
applicationId="admin" redirectToSSL="443">
                    <Path name="secure" authType="shibboleth"
requireSession="true"/>
                </Host>

```

```

-->
    </RequestMap>
</RequestMapProvider>

<Implementation>
    <ISAPI normalizeRequest="true">
        <!--
            Maps IIS Instance ID values to the host
            scheme/name/port/sslport. The name is required so that the proper <Host> in
            the request map above is found without having to cover
            every possible DNS/IP combination the user might
            enter. The port and scheme can usually be omitted, so
            the HTTP request's port and scheme will be used.

            <Alias> elements can specify alternate permissible
            client-specified server names.

            If a client request uses such a name, normalized
            redirects will use it, but the request map processing
            is still based on the default name attribute for the
            site. This reduces duplicate data entry in the request
            map for every legal hostname a site might permit. In
            the example below, only sp.example.org
            needs a <Host> element in the map, but
            spalias.example.org could be used by a client and
            those requests will map to sp.example.org for
            configuration settings.
        -->

        <Site id="1" name="sp.example.org">
            <Alias>spalias.example.org</Alias>
        </Site>

    </ISAPI>
</Implementation>
</Local>

<!--
The Applications section is where most of Shibboleth's SAML bits are
defined.

Resource requests are mapped in the Local section into an applicationId
that points into to this section.
-->

```

```

<Applications id="default" providerId="urn:federation:contoso.com"
  homeURL="https://fc5srv.contoso.com/cgi-bin/printenv.pl"
  xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
  xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata">

  <!--

  Controls session lifetimes, address checks, cookie handling, and the
  protocol handlers.

  You MUST supply an effectively unique handlerURL value for each of
  your applications.

  The value can be a relative path, a URL with no hostname
  (https:///path) or a full URL.

  The system can compute a relative value based on the virtual host.
  Using handlerSSL="true"

  will force the protocol to be https. You should also add a
  cookieProps setting of "; path=/;secure" in that case. Note that while
  we default checkAddress to "false", this has a negative impact on the
  security of the SP. Certain attacks are a bit easier with this
  disabled. The consistentAddress property is even more critical, and
  should rarely be disabled. It will only trip if a client uses a
  different source address at the SP after the cookie is issued.
  Allowing that means many scripting attacks against applications can
  result in theft and impersonation using the Shibboleth session.

  -->

  <Sessions lifetime="7200" timeout="3600" checkAddress="false"
  consistentAddress="true"

    handlerURL="/Shibboleth.sso" handlerSSL="false"
  idpHistory="true" idpHistoryDays="7">

  <!--

  SessionInitiators handle session requests and relay them to a WAYF or
  directly to an IdP, if possible. Automatic session setup will use the
  default or first element (or requireSessionWith can specify a
  specific id to use). Lazy sessions can be started with any initiator
  by redirecting to it. The only Binding supported is the
  "urn:mace:shibboleth:sp:1.3:SessionInit" lazy session profile using
  query string parameters:

  * target the resource to direct back to later (or homeURL will be
  used)

  * acsIndex optional index of an ACS to use on the way back in

  * providerId optional direct invocation of a specific IdP

  -->

```

```

        <!-- This default example directs users to a specific IdP's
SSO service. -->
        <!--
        <SessionInitiator isDefault="true" id="example"
Location="/WAYF/idp.example.org"
                Binding="urn:mace:shibboleth:sp:1.3:SessionInit"
                wayfURL="https://idp.example.org/shibboleth-idp/SSO"
wayfBinding="urn:mace:shibboleth:1.0:profiles:AuthnRequest"/>
        -->
        <!--
                Added by Simon - to replace the above section
        -->
        <SessionInitiator isDefault="true" id="testshib"
Location="https://fc5srv.contoso.com/Shibboleth.sso/ADFS"
                Binding="urn:mace:shibboleth:sp:1.3:SessionInit"
                wayfURL="https://BER-DC1.Adatum.com/adfs/ls/"
wayfBinding="http://schemas.xmlsoap.org/ws/2003/07/secext"/>

        <!-- This example directs users to a specific federation's
WAYF service. -->
        <SessionInitiator id="IQ" Location="/WAYF/InQueue"
                Binding="urn:mace:shibboleth:sp:1.3:SessionInit"
                wayfURL="https://wayf.internet2.edu/InQueue/WAYF"
wayfBinding="urn:mace:shibboleth:1.0:profiles:AuthnRequest"/>

        <!--
                md:AssertionConsumerService elements replace the old shireURL
                function with an explicit handler for particular profiles,
                such as SAML 1.1 POST or Artifact.

                The isDefault and index attributes are used when sessions are
                initiated to determine how to tell the IdP where and how to
                return the response.
        -->
        <md:AssertionConsumerService Location="/SAML/POST"
isDefault="true" index="1"
                Binding="urn:oasis:names:tc:SAML:1.0:profiles:browser-
post"/>
        <md:AssertionConsumerService Location="/SAML/Artifact"
index="2"

```

```

        Binding="urn:oasis:names:tc:SAML:1.0:profiles:artifact-01"/>
    <!--
        Added by Simon
    -->
    <md:AssertionConsumerService Location="/ADFS" index="4"
        Binding="http://schemas.xmlsoap.org/ws/2003/07/secext"
        ResponseLocation="/" />

    <!--
        md:SingleLogoutService elements are mostly a placeholder for
        2.0, but a simple
        cookie-clearing option with a ResponseLocation or a return URL
        parameter is
        supported via the "urn:mace:shibboleth:sp:1.3:Logout" Binding
        value.
    -->
    <md:SingleLogoutService Location="/Logout"
        Binding="urn:mace:shibboleth:sp:1.3:Logout" />

</Sessions>

<!--
    You should customize these pages! You can add attributes with values
    that can be plugged into your templates. You can remove the access
    attribute to cause the module to return a standard 403 Forbidden
    error code if authorization fails, and then customize that condition
    using your web server.
-->
<Errors session="/etc/shibboleth/sessionError.html"
    metadata="/etc/shibboleth/metadataError.html"
    rm="/etc/shibboleth/rmError.html"
    access="/etc/shibboleth/accessError.html"
    ssl="/etc/shibboleth/sslError.html"
    supportContact="root@localhost"
    logoLocation="/shibboleth-sp/logo.jpg"
    styleSheet="/shibboleth-sp/main.css" />

<!-- Indicates what credentials to use when communicating -->
<CredentialUse TLS="defcreds" Signing="defcreds">

```

```

        <!-- RelyingParty elements can customize credentials for
specific IdPs/sets. -->
        <!--
        <RelyingParty Name="urn:mace:inqueue" TLS="inqueuecreds"
Signing="inqueuecreds"/>
        -->
    </CredentialUse>

    <!-- Use designators to request specific attributes or none to ask
for all -->
    <!--
    <saml:AttributeDesignator AttributeName="urn:mace:dir:attribute-
def:eduPersonScopedAffiliation"
AttributeNamespace="urn:mace:shibboleth:1.0:attributeNamespace:uri"/>
    -->

    <!-- AAP can be inline or in a separate file -->
    <AAPProvider
type="edu.internet2.middleware.shibboleth.aap.provider.XMLAAP"
uri="/etc/shibboleth/AAP.xml"/>

    <!-- Operational config consists of metadata and trust providers. Can
be external or inline. -->

    <!-- Dummy metadata for private testing, delete for production
deployments. -->
    <MetadataProvider
type="edu.internet2.middleware.shibboleth.metadata.provider.XMLMetadata"
uri="/etc/shibboleth/example-metadata.xml"/>
    <!--
    Added by Simon
    -->
    <MetadataProvider
type="edu.internet2.middleware.shibboleth.metadata.provider.XMLMetadata"
uri="/etc/shibboleth/adatum-metadata.xml"/>

    <!-- InQueue pilot federation, delete for production deployments. -->
    <MetadataProvider
type="edu.internet2.middleware.shibboleth.metadata.provider.XMLMetadata"

```

```

        uri="/etc/shibboleth/IQ-metadata.xml"/>

        <!-- The standard trust provider supports SAMLv2 metadata with path
validation extensions. -->

        <TrustProvider
type="edu.internet2.middleware.shibboleth.common.provider.ShibbolethTrust"/>

        <!--

Zero or more SAML Audience condition matches (mainly for Shib 1.1
compatibility).

If you get "policy mismatch errors, you probably need to supply
metadata about your SP to the IdP if it's running 1.2. Adding an
element here is only a partial fix.

-->

<saml:Audience>urn:mace:inqueue</saml:Audience>

        <!--

You can customize behavior of specific applications here. The default
elements inside the outer <Applications> element generally have to be
overridden in an all or nothing fashion.

That is, if you supply a <Sessions> or <Errors> override, you MUST
include all attributes you want to apply, as they will not be
inherited. Similarly, if you specify an element such as

<MetadataProvider>, it is not additive with the defaults, but
replaces them.

Note that each application must have a handlerURL that maps uniquely
to it and no other application in the <RequestMap>. Otherwise no
sessions will reach the application.

If each application lives on its own vhost, then a single handler at
"/Shibboleth.sso" is sufficient, since the hostname will distinguish
the application.

The example below shows a special application that requires use of
SSL when establishing sessions, restricts the session cookie to SSL,
and inherits most other behavior except that it requests only EPPN
from the IdP instead of asking for all attributes. Note that it will
inherit all of the handler endpoints defined for the default
application.

-->

        <!--

        <Application id="admin">

```

```

        <Sessions lifetime="7200" timeout="3600" checkAddress="true"
consistentAddress="true"
            handlerURL="/Shibboleth.sso" handlerSSL="true"
cookieProps="; path=/; secure"/>
        <saml:AttributeDesignator
AttributeName="urn:mace:dir:attribute-def:eduPersonPrincipalName"
AttributeNamespace="urn:mace:shibboleth:1.0:attributeNamespace:uri"/>
    </Application>
-->

```

```
</Applications>
```

```
<!-- Define all the private keys and certificates here that you reference
from <CredentialUse>. -->
```

```

<CredentialsProvider
type="edu.internet2.middleware.shibboleth.common.Credentials">
    <Credentials xmlns="urn:mace:shibboleth:credentials:1.0">
        <FileResolver Id="defcreds">
            <Key>
                <Path>/etc/shibboleth/sp.key</Path>
            </Key>
            <Certificate>
                <Path>/etc/shibboleth/sp.crt</Path>
            </Certificate>
        </FileResolver>

```

```
<!--
```

Mostly you can define a single keypair above, but you can define and name a second

keypair to be used only in specific cases and then specify when to use it inside a

```
<CredentialUse> element.
```

```
-->
```

```
<!--
```

```

<FileResolver Id="inqueuecreds">
    <Key>
        <Path>/etc/shibboleth/inqueue.key</Path>
    </Key>

```

```

        <Certificate>
            <Path>/etc/shibboleth/inqueue.crt</Path>
        </Certificate>
    </FileResolver>
    -->
    </Credentials>
</CredentialsProvider>

<!-- Specialized attribute handling for cases with complex syntax. -->
<AttributeFactory AttributeName="urn:oid:1.3.6.1.4.1.5923.1.1.1.10"
type="edu.internet2.middleware.shibboleth.common.provider.TargetedIDFactory"
/>

</SPConfig>

```

AAP.xml

```

<AttributeAcceptancePolicy xmlns="urn:mace:shibboleth:1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:mace:shibboleth:1.0
/usr/share/xml/shibboleth/shibboleth.xsd">

```

<!--

An AAP is a set of AttributeRule elements, each one referencing a specific attribute by URI. All attributes that should be visible to an application running at the target should be listed, or they will be filtered out.

The Header and Alias attributes map an attribute to an HTTP header and to an htaccess rule name respectively. Without Header, the attribute will only be obtainable from the exported SAML assertion in raw XML.

Scoped attributes can also be filtered on Scope via rules in the asserting identity provider's metadata. Finally, a note on naming. The attributes in this file are mostly drawn from the set documented here:

<http://middleware.internet2.edu/urn-mace/urn-mace-dir-attribute-def.html>

The actual naming convention most of them follow is NOT to be used for any subsequent attributes bound to SAML, and you are NOT free to just make up names using it, because the urn:mace:dir namespace tree is controlled. For help and advice on defining new attributes, refer to:

```

https://authdev.it.ohio-
state.edu/twiki/bin/view/Shibboleth/AttributeNameing
-->

<!-- First some useful eduPerson attributes that many sites might use. -->

<AttributeRule Name="urn:mace:dir:attribute-
def:eduPersonScopedAffiliation" Scoped="true" CaseSensitive="false"
Header="Shib-EP-Affiliation" Alias="affiliation">

    <!-- Filtering rule to limit values to eduPerson-defined enumeration.
-->

    <AnySite>
        <Value>MEMBER</Value>
        <Value>FACULTY</Value>
        <Value>STUDENT</Value>
        <Value>STAFF</Value>
        <Value>ALUM</Value>
        <Value>AFFILIATE</Value>
        <Value>EMPLOYEE</Value>
    </AnySite>

    <!-- Example of Scope rule to override site metadata. -->
    <SiteRule Name="urn:mace:inqueue:shibdev.edu">
        <Scope Accept="false">shibdev.edu</Scope>
        <Scope Type="regexp">^.\.shibdev\.edu$</Scope>
    </SiteRule>
</AttributeRule>

<!--
This attribute is provided mostly to ease testing because an IdP out of
the box only sends the unscoped version. It has little use because it
lacks the context needed to work in a multi-domain scenario and is a
subset of the scoped version anyway.
-->

<AttributeRule Name="urn:mace:dir:attribute-def:eduPersonAffiliation"
CaseSensitive="false" Header="Shib-EP-UnscopedAffiliation" Alias="unscoped-
affiliation">

```

```

<AnySite>
    <Value>MEMBER</Value>
    <Value>FACULTY</Value>
    <Value>STUDENT</Value>
    <Value>STAFF</Value>
    <Value>ALUM</Value>
    <Value>AFFILIATE</Value>
    <Value>EMPLOYEE</Value>
</AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:eduPersonPrincipalName"
Scoped="true" Header="REMOTE_USER" Alias="user">
    <!-- Basic rule to pass through any value. -->
    <AnySite>
        <Value Type="regexp">^[^@]+$</Value>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:eduPersonEntitlement"
Header="Shib-EP-Entitlement" Alias="entitlement">
    <!-- Entitlements tend to be filtered per-site. -->

    <!--
Optional site rule that applies to any site
<AnySite>
    <Value>urn:mace:example.edu:exampleEntitlement</Value>
</AnySite>
-->

    <!-- Specific rules for an origin site, these are just
development/sample sites. -->
    <SiteRule Name="urn:mace:inqueue:example.edu">
        <Value Type="regexp">^urn:mace:..+$</Value>
    </SiteRule>
    <SiteRule Name="urn:mace:inqueue:shibdev.edu">
        <Value Type="regexp">^urn:mace:..+$</Value>

```

```

        </SiteRule>
    </AttributeRule>

    <!-- A persistent id attribute that supports personalized anonymous
    access. -->

    <!-- First, the deprecated version: -->
    <AttributeRule Name="urn:mace:dir:attribute-def:eduPersonTargetedID"
    Scoped="true" Header="Shib-TargetedID" Alias="targeted_id">
        <AnySite>
            <AnyValue/>
        </AnySite>
    </AttributeRule>

    <!-- Second, the new version (note the OID-style name): -->
    <AttributeRule Name="urn:oid:1.3.6.1.4.1.5923.1.1.1.10" Header="Shib-
    TargetedID" Alias="targeted_id">
        <AnySite>
            <AnyValue/>
        </AnySite>
    </AttributeRule>

    <!-- Some more eduPerson attributes, uncomment these to use them... -->
    <!--
    <AttributeRule Name="urn:mace:dir:attribute-def:eduPersonNickname">
        <AnySite>
            <AnyValue/>
        </AnySite>
    </AttributeRule>

    <AttributeRule Name="urn:mace:dir:attribute-
    def:eduPersonPrimaryAffiliation" CaseSensitive="false" Header="Shib-EP-
    PrimaryAffiliation">
        <AnySite>
            <Value>MEMBER</Value>
            <Value>FACULTY</Value>
            <Value>STUDENT</Value>

```

```

        <Value>STAFF</Value>
        <Value>ALUM</Value>
        <Value>AFFILIATE</Value>
        <Value>EMPLOYEE</Value>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:eduPersonPrimaryOrgUnitDN"
Header="Shib-EP-PrimaryOrgUnitDN">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:eduPersonOrgUnitDN"
Header="Shib-EP-OrgUnitDN">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:eduPersonOrgDN"
Header="Shib-EP-OrgDN">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

-->
<!--Examples of common LDAP-based attributes, uncomment to use these... --
>
<!--

<AttributeRule Name="urn:mace:dir:attribute-def:cn" Header="Shib-Person-
commonName">
    <AnySite>
        <AnyValue/>
    </AnySite>

```

```
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:sn" Header="Shib-Person-
surname">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:mail" Header="Shib-
InetOrgPerson-mail">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:telephoneNumber"
Header="Shib-Person-telephoneNumber">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:title" Header="Shib-
OrgPerson-title">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:initials" Header="Shib-
InetOrgPerson-initials">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:description" Header="Shib-
```

```
Person-description">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:carLicense" Header="Shib-
InetOrgPerson-carLicense">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:departmentNumber"
Header="Shib-InetOrgPerson-deptNum">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:displayName" Header="Shib-
InetOrgPerson-displayName">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:employeeNumber"
Header="Shib-InetOrgPerson-employeeNum">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>
```

```
<AttributeRule Name="urn:mace:dir:attribute-def:employeeType" Header="Shib-
InetOrgPerson-employeeType">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:preferredLanguage"
Header="Shib-InetOrgPerson-prefLang">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:manager" Header="Shib-
InetOrgPerson-manager">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:roomNumber" Header="Shib-
InetOrgPerson-roomNum">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:seeAlso" Header="Shib-
OrgPerson-seeAlso">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:facsimileTelephoneNumber"
Header="Shib-OrgPerson-fax">
    <AnySite>
```

```
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:street" Header="Shib-
OrgPerson-street">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:postOfficeBox"
Header="Shib-OrgPerson-POBox">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:postalCode" Header="Shib-
OrgPerson-postalCode">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:st" Header="Shib-
OrgPerson-state">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:givenName" Header="Shib-
InetOrgPerson-givenName">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>
```

```

<AttributeRule Name="urn:mace:dir:attribute-def:l" Header="Shib-OrgPerson-
locality">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:businessCategory"
Header="Shib-InetOrgPerson-businessCat">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-def:ou" Header="Shib-
OrgPerson-orgUnit">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

<AttributeRule Name="urn:mace:dir:attribute-
def:physicalDeliveryOfficeName" Header="Shib-OrgPerson-OfficeName">
  <AnySite>
    <AnyValue/>
  </AnySite>
</AttributeRule>

-->

<!--
  Added by Simon
-->

<AttributeRule Name="Group" Namespace="http://schemas.xmlsoap.org/claims"
CaseSensitive="true"

  Header="Shib-Group" Alias="shib-group">
  <AnySite>

```

```

        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="http://schemas.xmlsoap.org/claims/UPN" Header="Shib-
TargetedID" Alias="targeted_id">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="userPrincipalName"
Namespace="http://schemas.xmlsoap.org/claims" Header="REMOTE_USER"
Alias="user">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="company"
Namespace="http://schemas.xmlsoap.org/claims" Scoped="true"
CaseSensitive="false" Header="Shib-EP-Affiliation" Alias="affiliation">
    <!-- Filtering rule to limit values to eduPerson-defined enumeration.
-->
    <AnySite>
        <Value>MEMBER</Value>
        <Value>FACULTY</Value>
        <Value>STUDENT</Value>
        <Value>STAFF</Value>
        <Value>ALUM</Value>
        <Value>AFFILIATE</Value>
        <Value>EMPLOYEE</Value>
    </AnySite>
</AttributeRule>

<AttributeRule Name="givenname"
Namespace="http://schemas.xmlsoap.org/claims" Header="Shib-InetOrgPerson-
givenName" CaseSensitive="false">
    <AnySite>

```

```
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="sn" Namespace="http://schemas.xmlsoap.org/claims"
Header="Shib-Person-surname" CaseSensitive="false">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="mail" Namespace="http://schemas.xmlsoap.org/claims"
Header="Shib-InetOrgPerson-mail" CaseSensitive="false">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="telephone"
Namespace="http://schemas.xmlsoap.org/claims" Header="Shib-Person-
telephoneNumber" CaseSensitive="false">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="title" Namespace="http://schemas.xmlsoap.org/claims"
Header="Shib-OrgPerson-title" CaseSensitive="false">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>

<AttributeRule Name="displayname"
Namespace="http://schemas.xmlsoap.org/claims" Header="Shib-InetOrgPerson-
displayName" CaseSensitive="false">
    <AnySite>
        <AnyValue/>
    </AnySite>
</AttributeRule>
```

```

<!--
  End of stuff added by Simon
-->
</AttributeAcceptancePolicy>

```

adatum-metadata.xml

```

<EntitiesDescriptor
  xmlns="urn:oasis:names:tc:SAML:2.0:metadata"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:shibmd="urn:mace:shibboleth:metadata:1.0"
  xsi:schemaLocation="urn:oasis:names:tc:SAML:2.0:metadata
/usr/share/xml/shibboleth/saml-schema-metadata-2.0.xsd
urn:mace:shibboleth:metadata:1.0 @-PKGXMLDIR-@/shibboleth-metadata-1.0.xsd
http://www.w3.org/2000/09/xmldsig# @-PKGXMLDIR-@/xmldsig-core-schema.xsd"
  Name="urn:federation:contoso.com"
  validUntil="2010-01-01T00:00:00Z">

  <!--

  This is a starter set of metadata for testing Shibboleth. It shows
  a pair of example entities, one an IdP and one an SP. Each party
  requires metadata from its opposite in order to interact with it.
  Thus, your metadata describes you, and your partner(s)' metadata
  is fed into your configuration.

  The software components do not configure themselves using metadata
  (e.g. the IdP does not configure itself using IdP metadata). Instead,
  metadata about SPs is fed into IdPs and metadata about IdPs is fed into
  SPs. Other metadata is ignored, so the software does not look for
  conflicts between its own configuration and the metadata that might
  be present about itself. Metadata is instead maintained based on the
  external details of your configuration.

  -->

  <!--

  Added by Simon

  -->

  <!--

  Identity provider

  -->

```

```

<EntityDescriptor entityID="urn:federation:Adatum">

  <IDPSSODescriptor
protocolSupportEnumeration="http://schemas.xmlsoap.org/ws/2003/07/secext">

    <KeyDescriptor use="signing">

      <ds:KeyInfo>

        <ds:X509Data>

          <ds:X509Certificate>

MIICyDCCAbSgAwIBAgIQsVv1sV0V36RHicKd5gWLLjAJBgUrDgMCHQUAMCQxIjAg
BgNVBAMTGUZlZGVyYXRpb24gU2VydmVyIEJFUilEQzEwHhcNMDYwOTA1MjE1MTE5
WhcNMDcwOTA2MDM1MTE5WjAkMSIwIAYDVQQDExlGZWRLcmF0aW9uIFNlcnZlciBC
RVItREMxMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAx0wsqoDvMhtB
sqdvu3Mi9+vv8PuyknyoyWxvAbQ6axYP4N+ojYugyx/7rEky/nr6u/cZct1XZ5UW
fKrEx8RW6Re4eoY7d9ualJbleJYmx97LXEZKNc1TrTALDMY2J0yeQPFq0UQOYLcD
DhpIBi8PJ48gmsQtKjI0/lrxXGsjs8f97gzEsA970372ZnYkzi9rqT4C42Soz5Sa
RwFlz4glDa99QBkNKgmnLjWUn1sjgaCQUdwI794dWit22hlkvziMyKvqD8/ygPcl
PgM2RQH5hLaU9UFbnqQ+jx36Kr2raIg/2Yp9QLFoEEQfe4bUgKUtR+sgocvgyllU
gAZuYTOyPwIDAQAAMakGBSsOAwIdBQADggEBAMD15++UrKv3SIEohqe7SEd/nh3j
1Rcl5hBx9W4dAHicZ+uokVYzPK2L4F8dQYEEYT/RR2UCbChBFZzjJWaXi9SkBvE
WKU6vCeRnBc5faNajGPKSXwd25TtxIM9LV4misBOIifaXuFhOplngkXYWRaOQets
P7hQRfpoGeB/UXyWjICfnaDLJdTUIWJmlytPmtCfFN0e/avjx0pPOoc3CXN2XX+X
Nwd9eDX4LEn+XjKKooAbsaYl/dRS4INcIZx4hPpImvucuPxXAMqc177JBzOLIPqK
CER/ZtIUyZHEwWpxn7mn4qI9gLubpz5bokkgJa4V141jXK7nuexmB7K0FDU=

          </ds:X509Certificate>

        </ds:X509Data>

      </ds:KeyInfo>

    </KeyDescriptor>

    <SingleSignOnService
Binding="http://schemas.xmlsoap.org/ws/2003/07/secext"
Location="https://BER-DC1.Adatum.com/adfs/ls/" />

  </IDPSSODescriptor>

  <AttributeAuthorityDescriptor
protocolSupportEnumeration="http://schemas.xmlsoap.org/ws/2003/07/secext">

    <Extensions>

```

```

        <shib:Scope
xmlns:shib="urn:mace:shibboleth:metadata:1.0">adatum.com</shib:Scope>

        </Extensions>

        <!-- The certificate has to be repeated here (or a different
one specified if necessary). -->

        <KeyDescriptor use="signing">

                <ds:KeyInfo>

                        <ds:X509Data>

                                <ds:X509Certificate>

MIICyDCCAbSgAwIBAgIQsVv1sV0V36RHicKd5gWLLjAJBgUrDgMCHQUAMCQxIjAg
BgNVBAMTGUZlZGVyYXRpb24gU2VydmVyIEJFUilEQzEwHhcNMDYwOTA1MjE1MTE5
WhcNMDcwOTA2MDM1MTE5WjAkMSIwIAYDVQQDExlGZWRLcmF0aW9uIFNlcnZlciBC
RVItREMxMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAx0wsqoDvMhtB
sqdvu3Mi9+vv8PuyknyoyWxvAbQ6axYP4N+ojYugyx/7rEky/nr6u/cZct1XZ5UW
fKrEx8RW6Re4eoY7d9ualJbleJYmx97LXEZKNc1TrTALDMY2J0yeQPFq0UQOYLcD
DhpIBi8PJ48gmsQtKjI0/lrxXGsjs8f97gzEsA970372ZnYkzi9rqT4C42Soz5Sa
RwFlz4glDa99QBkNKgmnLjWUn1sjgaCQUdwI794dWit22hlkvziMyKvqD8/ygPcl
PgM2RQH5hLaU9UFbnqQ+jx36Kr2raIg/2Yp9QLFoEEQfe4bUgKUtR+sgocvgyllU
gAZuYTOyPwIDAQAAMakGBSs0AwIdBQADggEBAMD15++UrKv3SIEohqe7SEd/nh3j
1Rc15hBx9W4dAHicZ+uokVYzpk2L4F8dQYEEYT/RR2UCbChBFZzjJWaXik9SkBvE
WKU6vCeRnBc5faNajGPKSXwd25TtxIM9LV4misBOIifaXuFhOplngkXYWRaOQEtS
P7hQRfpoGeB/UXyWjICfnaDLJdTUIWJmlytPmtCfFN0e/avjx0pPOoc3CXN2XX+X
Nwd9eDX4LEn+XjKKooAbsaYl/dRS4INcIZx4hPpImvucuPxXAMqc177JBzOLIPqK
CEr/ZtIUyZHEwWpxn7mn4qI9gLubpz5bokkgJa4Vl41jXK7nuexmB7K0FDU=

                                </ds:X509Certificate>

                        </ds:X509Data>

                </ds:KeyInfo>

        </KeyDescriptor>

        <AttributeService
Binding="http://schemas.xmlsoap.org/ws/2003/07/secext"

                Location="https://BER-DC1.Adatum.com/adfs/ls/">

```

```

<NameIDFormat>urn:mace:shibboleth:1.0:nameIdentifier</NameIDFormat>
  </AttributeAuthorityDescriptor>
<!--
  <SPSSODescriptor
protocolSupportEnumeration="http://schemas.xmlsoap.org/ws/2003/07/secext">
  <AssertionConsumerService
Binding="http://schemas.xmlsoap.org/ws/2003/07/secext"
  index="1"
  Location="https://BER-DC1.Adatum.com/adfs/ls/" />
</SPSSODescriptor>
-->

</EntityDescriptor>

```



Oxford Computer Group Deutschland
 Winterlestr. 10b
 D-85435 Erding, Deutschland
Tel: +49 8122 892089-0
Fax: +49 8122 892089-99
Email: info@oxfordcomputergroup.com
www.oxfordcomputergroup.de

Oxford Computer Group UK
 Bignell Park Barns, Chesterton
 Oxfordshire OX26 1TD, UK
Tel: +44 (0)8456 584425
Fax: +44 (0)8456 584426
Email: info@oxfordcomputergroup.com
www.oxfordcomputergroup.com

Oxford Computer Group Canada
 146 Codrington Street
 Barrie, Ontario
 L4M 1S1
Tel: +1 705 737 4457
Email: info@oxfordcomputergroup.com
www.oxfordcomputergroup.com

Oxford Computer Group USA
 111 Avenue C Suite 104
 Snohomish WA 98290
Tel: +1 360 862 1617
Fax: +(877) 862-1617
Email: info@oxfordcomputergroup.com
www.oxfordcomputergroup.com

About Oxford Computer Group

Oxford Computer Group (OCG) specialises in identity and access management with operations in North America and Europe, OCG have an enviable repository of MIIS 2003 expertise and knowledge. Services include: strategic and functional consulting; system integration; solution development; and identity training.

Microsoft selected OCG to produce and deliver the first training courses on MIIS 2003. We have delivered over 100 enterprise identity solutions based on MIIS and have trained over 2000 people on MIIS – including some of Microsofts own consultants.

To discuss your Identity and Access Management requirements, please contact us.