Introduction to Shibboleth

UF IT/CNS/Open Systems Group

February 23, 2012

Eli Ben-Shoshan (ebs@ufl.edu)
Martin Smith (smithmb@ufl.edu)

Goals

Goals for this session:

- Understand general concepts behind Shibboleth
- Learn how to install and configure software
- Learn how to protect your content

Requirements

You should have the following ready for this class:

- A test/dev machine at your office
- Access to your test/dev machine
- Capability to install software on test/dev machine
- Willingness to have your test/dev machine go down for a bit

Definitions

- Security Assertion Markup Language (SAML)
  An XML-based open standard for exchanging authentication and authorization data between security domains
- Shibboleth Service Provider (SP)
  You, your application, or the SP software that you install and maintain on your webserver (a consumer of assertions).
- Shibboleth Identity Provider (IdP)
  The central authentication server. The IdP authenticates the user and vends attributes about the user (a producer of assertions).

Definitions (continued)

- Service Endpoint
  A set of URLs on the SP and IdP that are used to transfer SAML documents.
- Metadata
  A document that names all of the service endpoints.

A service provider relies on an identity provider to identify a principal. At the principal's request, the identity provider passes a SAML assertion to the service provider. On the basis of this assertion, the service provider makes an access control decision.

Definitions (continued)

- Uniform Resource Identifier (URI)
  A string of characters used to identify a name or a resource
- Uniform Resource Name (URN)
  A URI that uses the urn scheme and does not imply availability of the identified resource
- Entity Identifier (entityID)
  A universal resource name (URN) that identifies your SP

Definitions (continued)

- All entityID's for UF take the following form:
  - `urn:edu:ufl:prod:XXXXX` for production
  - `urn:edu:ufl:test:XXXXX` for test
  - `urn:edu:ufl:dev:XXXXX` for development
- It also common to see URL based entity IDs:
  `https://idp_name.example.edu/idp`

Shibboleth software on your SP

The Shibboleth software that runs on your SP is setup as follows:

- **Shibboleth module** that runs in your webserver (IIS/Apache) that maps URIs to requests and talks to Shibboleth daemon
- **Shibboleth daemon** that does all the heavy lifting, decrypts SAML, extracts attributes

Software Install

Official directions are here:
http://www.it.ufl.edu/identity/shibboleth/technical.html

Some repos for popular distros exist:
https://spaces.internet2.edu/display/SHIB2/NativeSPLinuxRPMInstall

The directions are similar between Windows/IIS and Unix/Apache.

Install the software - Windows

See http://www.it.ufl.edu/identity/shibboleth/technicalIIS.html.

- Download the latest MSI installer from this page for your platform and install it, then reboot
- Please do not change any defaults offered by the installer unless absolutely necessary
- Verify that the installer correctly created an ISAPI filter on your site and configured the Shibboleth daemon as a Windows service

Install the software RHEL

See http://www.it.ufl.edu/identity/shibboleth/technicalapache.html.

- Download and install the RPMs from this page for your platform, or use your favorite package manager
- Edit Apache config to load the shibboleth module and set UseCanonicalName
- Restart Apache and start the Shibboleth daemon

Configuring Shibboleth Daemon

All configuration for daemon is in the `shibboleth2.xml` file. Get the template from the Open Systems site:
http://open-systems.ufl.edu/shibboleth

Place the file in the correct location:

Windows -

```
C:\opt\shibbolethsp\etc\shibboleth\shibboleth2.xml
```

Unix -

```
/etc/shibboleth/shibboleth2.xml
```

Configuring Shibboleth Daemon (continued)

Update shibboleth2.xml template, replacing variables:

- _HOSTNAME_ - fully qualified domain of your site
- _URN_ - entityID assigned to you by Bridges IAM Admin

For Windows you also have

- _SITEID_ - IIS "Site Identifier" for this website

Configuring Shibboleth Daemon (continued)

Remove the sp-cert.pem and sp-key.pem from the Shibboleth configuration directory for your platform

Windows -

```
C:\opt\shibbolethsp\etc\shibboleth
```

Unix -

```
/etc/shibboleth
```

Configure Shibboleth Daemon (continued)

Generate the key and certificate:

Windows - `keygen.bat -h _HOSTNAME_ -e _URN_`

Unix - `keygen.sh -h _HOSTNAME_ -e _URN_`

Configure Shibboleth Daemon

Rename the generated files:

`sp-cert.pem` should be renamed to `_HOSTNAME_.cert`

`sp-key.pem` should be renamed to `_HOSTNAME_.key`

Now, **restart** the shibboleth daemon.

Checking your install

If all went well, then you should have a shibboleth daemon running and the webserver should respond with your SP's metadata at this URL:

`http:// _HOSTNAME_ /Shibboleth.sso/Metadata`

Check your install

Review your metadata:

- Make sure the **entityID is correct** for this SP
- Make sure there is **at least one** of these services defined:
  - AssertionConsumerService
  - ManageNameIDService
  - SingleLogoutService

Service provider completed

Congratulations! Your SP is now configured.

Submit your Metadata for inclusion in the IdP using https://open-systems.ufl.edu/shibmeta.

Until this happens your will get an error message on your SP:

Error Message: SAML 2 SSO profile is not configured for relying party urn:edu:ufl:XXXX:YYYYY

Protecting Content

Ways to accomplish content protection:

- Modify shibboleth2.xml
- Modify .htaccess (Apache only)
- Application logic

Protecting Content (shibboleth2.xml)

This can be used for both IIS and Apache, but this is **the only way to protect content in IIS**.

- Add a `Path` element to the `Host` element
- Add a `AccessControl` element to `Path` element
- Add a `Rule` element to the `AccessControl` element

Protecting Content, Simple (shibboleth2.xml)

```
<RequestMapper>
<RequestMap>
<Host name="example.com">
<Path name="secure"
   requireSession="true" authType="shibboleth">
<AccessControl>
<Rule require="primary-affiliation">S</Rule>
</AccessControl>
</Path>
</Host>
</RequestMap>
</RequestMapper>
```

Protecting Content, Complex (shibboleth2.xml)

```
<RequestMapper>
<RequestMap>
<Host name="example.com"
    requireSession="true" authType="shibboleth">
<Path name="secure">
<AccessControl>
<OR>
<Rule require="primary-affiliation">S</Rule>
<Rule require="primary-affiliation">F</Rule>
</OR>
</AccessControl>
</Path>
</Host>
</RequestMap>
</RequestMapper>
```

Protecting Content (.htaccess)

Much easier to use and maintain.

If you are using Apache, use this method.

Protecting Content (.htaccess)

Simple Example

```
AuthType Shibboleth
ShibRequireSession On
Require valid-user
```

Protecting Content (.htaccess)

Complex Example

```
AuthType Shibboleth
ShibRequireSession On
Require primary-affliation ~ S|F
```

Protecting Content (Application logic)

Use the application programming language of your choice. Create applications that are controlled by the existence of webserver environment variables.

Questions?

Thank you.

Also, please see these resources:
http://open-systems.ufl.edu/shibboleth/resources