SP Hands-on Session

Installing and Configuring a Shibboleth 2 Service Provider

SWITCH

Credits and General Information

- Slides were originally created by Scott Cantor, Internet 2 Developer of the Shibboleth Service Provider
- Course material is adapted for use in SWITCHaai
- Course material is published online
- If you see this ⌨ on a slide, hands-on work is required
- URLs at bottom right point to pages with more details
- On slides with Detour separate presentations focus on special topic

Main Goals of Hands-On Session

- Install and configure a Shibboleth Service Provider 2
- Register it with the AAI Test federation
- Know how and where to configure things
- Learn how to protect static web pages
- Understand how attributes can be used in web applications

Essential OS Commands for Linux

DOS Command	Linux Command
`dir`	`ls -l`
`cd <directory>`	`cd <directory>`
`mkdir` or `md <directory>`	`mkdir <directory>`
`rmdir` or `rd <directory>`	`rmdir <directory>`
`chdir`	`pwd`
`del` or `erase <file>`	`rm <file>`
`copy` and `xcopy <file>`	`cp` and `cp -R <file>`
`find` or `findstr <file>`	`grep <string> <file>`
`comp <file1> <file2>`	`diff <file1> <file2>`
`edit <file>`	`nano` or `vim` or `emacs <file>`
`ping <host>`	`ping <host>`
`reboot`	`reboot`

Tips and Tricks for Hands-On Session

- The password usually is "password"
- Lines starting with `$` are commands to be executed
  - Replace # with a number (your participation number during the training)
- Command should be executed as root user
  - Happens automatically if Terminal is opened or if text editor is used
- Character `\` is line break symbol,
  which allows to break a line when typed
- Watch out for invalid XML/configuration errors
  - Consult Debugging SP Handout for hints to resolve problems

More Tips and Tricks for Hands-On Session

- Restart the Shibboleth daemon shibd after every change
  - shibd automatically reloads config but only restarts "reveal" errors
  - Alternatively, look at the log file for errors
- Delete session cookies after changes (or restart browser)
  - Should not be necessary but is safer for testing
- SSH access to connect to your VM (only with VirtualBox)

  ```
  $ ssh -p 2222 sp-admin@127.0.0.1
  ```

  The password is 'password', user is in sudoers list
  Useful for `$ tail -f /var/log/shibboleth/shibd.log`
- On the VM you will find a web page with useful bookmarks
  In your web browser open: `https://sp#.example.org/`

Test Users on AAI Demo Home Organisation

- Username: `g.utente` Password: `password`

```
Givenname surname: Giovanni Utente
Affiliation:       faculty;member
Entitlements:      http://example.org/res/99999
                   http://publisher-xy.com/e-journals
```

- Username: `p.etudiant` Password: `password`

```
Givenname surname: Pière Edudiant
Affiliation:       student;member
Entitlements:      urn:mace:dir:entitlement:common-lib-terms
                   http://www.example.org/aai/agreement-2011
```

- Username: `h.mitarbeiter` Password: `password`

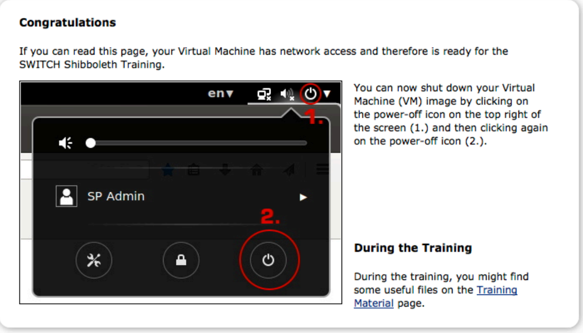
```
Givenname surname: Hans Mitarbeiter
Affiliation:       staff;student;member
Entitlements:      urn:mace:dir:entitlement:common-lib-terms
                   http://www.example.org/vip
```

VM Operating System Environment

- Ubuntu 14.04 LTS, Virtual Box/VMWare VDK image
- User: "**sp-admin**" / Password: "**password**" (in sudoers list)
- Apache 2 on ports 80 (http) and 443 (https)
- Self-signed SSL web server certificate
- AuthConfig added to /cgi-bin and /html for .htaccess
- Hostnames:
  - `sp#.example.org`
  - `altsp#.example.org` (alternative hostname)

Boot up the image

1. Open "SWITCH-Shibboleth-Training.vbox" image in Virtual Box
2. Start the virtual machine (VM)
3. After login, Firefox will open automatically.
   Ensure that it displays this page:



Congratulations

If you can read this page, your Virtual Machine has network access and therefore is ready for the SWITCH Shibboleth Training.

You can now shut down your Virtual Machine (VM) image by clicking on the power-off icon on the top right of the screen (1.) and then clicking again on the power-off icon (2.).

During the Training

During the training, you might find some useful files on the Training Material page.

If you don't see this message, contact an assistant.

VM Setup

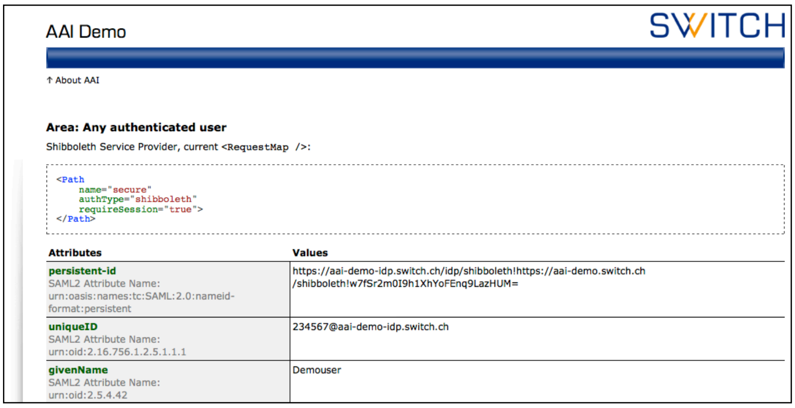
4. Open a Terminal
5. Enter "password" to become root user.
6. Then execute:

```
$ setupVM
```

7. Enter your participation number from the name tag

VM will then reboot automatically after a few seconds.

Do the AAI Demo as a Quick Test



AAI Demo

Area: Any authenticated user

Shibboleth Service Provider, current <RequestMap />:

```
<Path
    name="secure"
    authType="shibboleth"
    requireSession="true">
</Path>
```

Attributes	Values
persistent-id SAML2 Attribute Name: urn:oasis:names:tc:SAML:2.0:nameid-format:persistent	https://aai-demo-idp.switch.ch/idp/shibboleth!https://aai-demo.switch.ch/shibboleth!w7fSr2m0I9h1XhYoFEnq9LazHUM=
uniqueID SAML2 Attribute Name: urn:oid:2.16.756.1.2.5.1.1.1	234567@aai-demo-idp.switch.ch
givenName SAML2 Attribute Name: urn:oid:2.5.4.42	Demouser

1. In Firefox, open aai-demo.switch.ch
2. Click on "Any authenticated user"
3. Select the "AAI Demo Home Organisation"
4. Log in using a test user (e.g. "g.utente" "password")

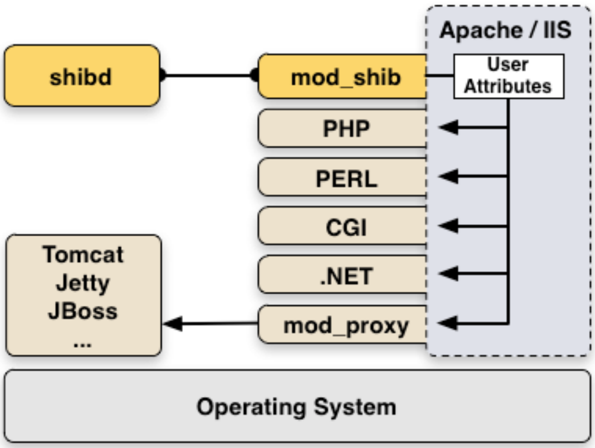
SP Overview and Installation

Goals:

1. Terminology and SP Overview
2. Installation
3. Configuration
4. Quick Sanity Check

Shibboleth SP: Daemon & mod_shib

- Runs on: Linux, Solaris, Windows, Mac OS X, FreeBSD, …
- Protects web applications
- shibd processes attributes
- Can authorize users with
  - Apache directives
  - Shibboleth XML Access rule
- Provides attributes to applications



Terminology

- **Service Provider (SP)**
  Consumes SAML assertions, protects web applications
- **Identity Provider (IdP)**
  Asserts digital identities using SAML
- **Discovery Service/WAYF (DS/WAYF)**
  Lets user choose Identity Provider/home organisation
- **shibd** (Shibboleth daemon)
  SP service/daemon for maintaining state
- **Session**
  Security context and cached data for a logged-in user
- **Session Initiator**
  Part of SP that controls how SSO requests are started

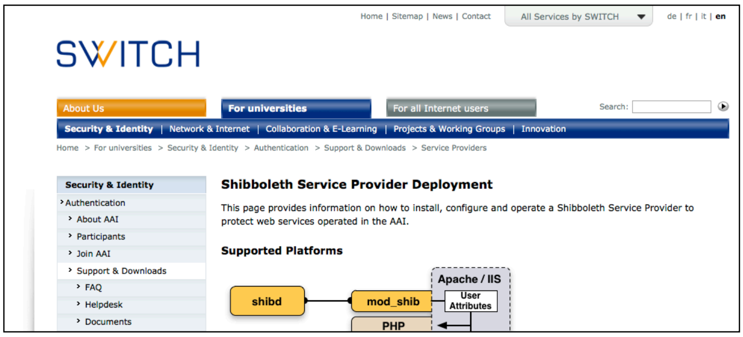
SWITCHaai Deployment Guides

How to install Shibboleth?

- Instructions SWITCHaai and AAI Test federations:
  https://www.switch.ch/aai/guides/sp/
  - Separation between installation and configuration.
  - Instructions for all major operating systems
  - SWITCHaai guides are custom-tailored and easier!
- General instructions on Shibboleth Wiki:
  https://wiki.shibboleth.net/confluence/display/SHIB2/Installation
  - Comprehensive documentation on most features
  - Not targeted for a specific federation

Deployment Guides

- In Firefox (on the VM) open: http://www.switch.ch/aai/
- Find the page with the Service Provider Deployment
  - Or find the link on the bookmarks page


Home | Sitemap | News | Contact
All Services by SWITCH
de | fr | it | en
SWITCH
About Us
For universities
For all Internet users
Search:
Security & Identity | Network & Internet | Collaboration & E-Learning | Projects & Working Groups | Innovation
Home > For universities > Security & Identity > Authentication > Support & Downloads > Service Providers
Security & Identity
Authentication
About AAI
Participants
Join AAI
Support & Downloads
FAQ
Helpdesk
Documents
Apache / IIS
shibd
mod_shib
User Attributes
PHP

Shibboleth Service Provider Deployment

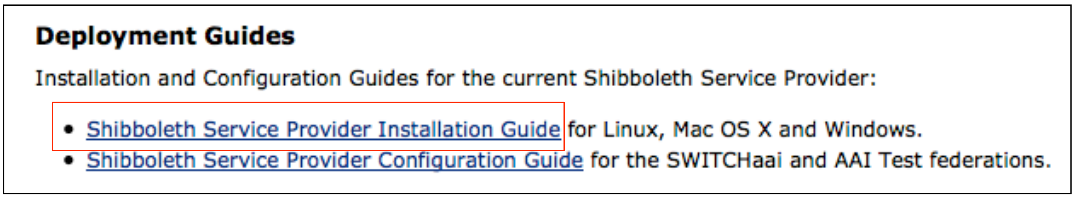
This page provides information on how to install, configure and operate a Shibboleth Service Provider to protect web services operated in the AAI.

Supported Platforms

- Deployment Guide is split into:
  - **Installation Guide:** Custom tailored for all major operating systems
  - **Configuration Guide:** Independent from OS (except Windows)

Service Provider Installation

- Start with the "Installation Guide"



Deployment Guides

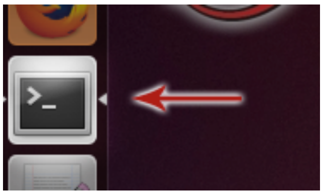
Installation and Configuration Guides for the current Shibboleth Service Provider:

- Shibboleth Service Provider Installation Guide for Linux, Mac OS X and Windows.
- Shibboleth Service Provider Configuration Guide for the SWITCHaai and AAI Test federations.

- In section 1. "Introduction" select "Ubuntu" as operating system
  - Guide will adapt itself automatically depending on selected OS
- Proceed with sections 3 – 5 in the guide
  - Section 2 can be skipped (`sudo` and `curl` are already installed)
  - You must open a terminal window for these steps
  - Provide `sp#.example.org` for the mod_shib Test in section 5

Installation

- Open a Terminal window
  - Click on this icon in the launch bar at the left



- Proceed with sections 3 – 5
  - Provide `sp#.example.org` for the mod_shib Test in section 5
  - When running `sudo shibd -t` there will be some (expected) errors
    Just ensure that the `Overall configuration is loadable`
- **Service Provider is now installed but not configured yet!**

Installation for Other Operating Systems

- On Debian/Ubuntu (by Debian/Ubuntu):
  ```
  $ apt-get install libapache2-mod-shib2
  ```
- On Mac OS X with MacPort (by Shibboleth team):
  ```
   $ port install shibboleth
  ```
- On Redhat/Suse/OpenSuse/CentOS (by Shibboleth team):
  ```
  $ yum install shibboleth
  ```
- On Windows with MSI packet (by Shibboleth team)
- Manual compilation not very difficult either
  - But more difficult to maintain efficiently

https://wiki.shibboleth.net/confluence/display/SHIB2/Installation

Service Provider Binaries and Paths

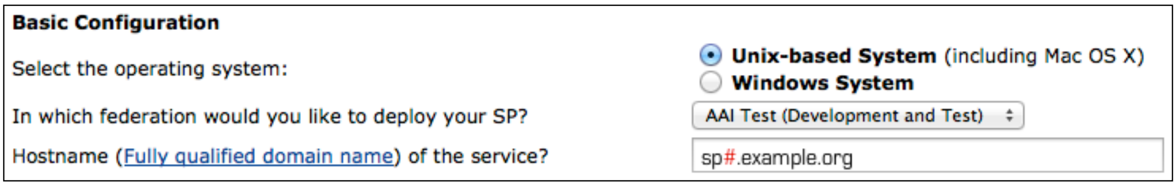
- **Shibboleth Daemon binary**
  Linux/Unix: `/usr/sbin/shibd`
  Win: `C:\opt\shibboleth-sp\sbin\shibd.exe`
- **Shibboleth main configuration file**
  Linux/Unix: `/etc/shibboleth/shibboleth2.xml`
  Win: `C:\opt\shibboleth-sp\etc\shibboleth\shibboleth2.xml`
- **Shibboleth Libraries/Modules/Extensions**
  Linux/Unix: `/usr/lib/shibboleth/*.so`
  Win: `C:\opt\shibboleth-sp\lib\*.so`

Important directories

- `/etc/shibboleth/`
  - Master and supporting configuration files
  - Locally maintained metadata files
  - HTML templates (to customize the look & feel of service)
  - Logging configuration files (*.logger)
  - Credentials (certificates and private keys)
- `/var/run/shibboleth/` and `/var/cache/shibboleth/`
  - UNIX socket
  - remote metadata backups
- `/var/log/shibboleth/`
  - shibd.log and transaction.log files
- `/var/log/apache2/` or `/var/log/shibboleth/apache2/`
  - `native.log` (is written by mod_shib web server module)

Configuration

- Continue with Configuration Guide
  - "Configuration Guide for new installations" at bottom of installation guide



Basic Configuration

Select the operating system:	● **Unix-based System** (including Mac OS X) ○ **Windows System**
In which federation would you like to deploy your SP?	AAI Test (Development and Test)
Hostname (Fully qualified domain name) of the service?	sp#.example.org

- In Setup Profile
  - Select "AAI Test Federation"
  - Provide `sp#.example.org` as host name
  - Don't change the other values which were updated automatically
  - Click on "Update configuration guide with above Data"

EntityID of an SP

- Every SP needs a unique identifier: The **entityID**

- Where is entityID used?
  - In transmitted messages, local configuration, metadata
  - IdP log files, configuration, filtering policies

- EntityID should be: Unique, locally scoped, representative and unchanging

- Convention: Include FQDN of your service
  - Guide automatically sets the following entityID:
    `https://sp#.example.org/shibboleth`

X.509 Certificates

Purpose and usage of certificates in SAML

Please consult the table of contents to find this presentation in your hand-outs.

Generate X.509 Key/Certificate

- Script to generate certificate and private key:
  `/usr/sbin/shib-keygen` or `/etc/shibboleth/keygen.sh`
  - Runs automatically during installation on some OS

- Proceed with section 4 in the guide about the X.509 certificates
  - Generates an X.509 certificate according to SWITCHaai requirements
  - Results in `sp-cert.pem` and `sp-key.pem` in /etc/shibboleth
  - Have a look at the PEM encoded certificate with

    ```
    $ less /etc/shibboleth/sp-cert.pem
    ```
  - To see content of certificate, execute:

    ```
    $ openssl x509 -text -in /etc/shibboleth/sp-cert.pem
    ```

Install and Test Configuration Files

- **Continue with section 5, the 'Shibboleth Configuration'**
- This downloads:
  - Shibboleth main configuration (shibboleth2.xml)
  - Attribute map configuration (attribute-map.xml)
  - Attribute policy configuration (attribute-policy.xml)
  - SWITCHaai Root CA certificate to verify metadata signature

- Configuration files are custom-tailored for your
  Service Provider based on values in 'Setup Profile'

- **Run 'Configuration Tests' in section 6 of the guide**
  - Most important: Check Shibboleth configuration with: `$ shibd -t`

Sanity Checks

- Start processes:

```
$ /etc/init.d/shibd restart or $ service shibd restart
$ /etc/init.d/apache2 restart or $ service apache2 restart
```

- Check shibd status (XML should be returned on success):

```
$ curl -k --interface lo \
  https://sp#.example.org/Shibboleth.sso/Status
```

- Access session handler from your browser:

```
https://sp#.example.org/Shibboleth.sso/Session
```

  After certificate warning, you get "A valid Session was not found" error

- See how a Shibboleth error looks like (you get an exception):

```
https://sp#.example.org/Shibboleth.sso/Foobar
```

Bootstrapping the SP

Goals:

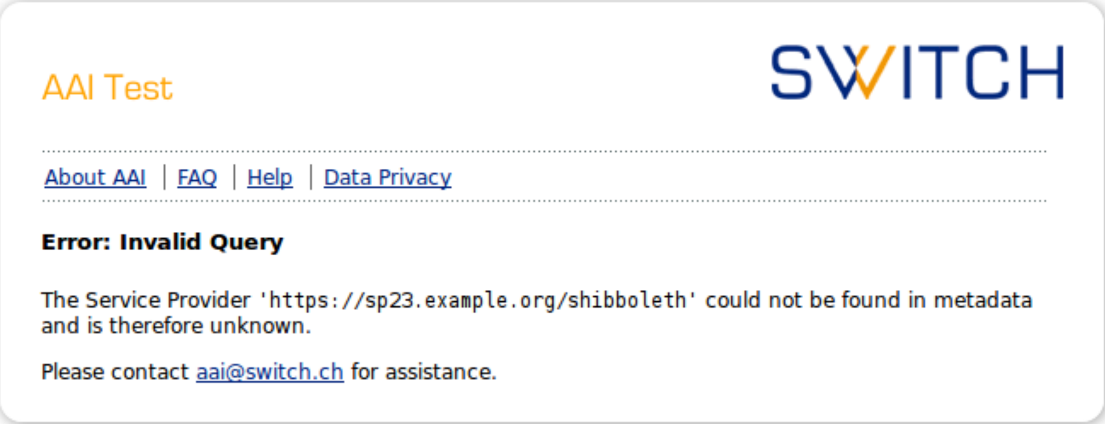
1. First attempt to login on Service Provider
2. Learn about Metadata
3. Learn about the AAI Resource Registry
4. Register Service Provider for AAI Test federation

Configuration Files

- **Service Provider is now installed and configured**
  - Let's see if authentication with AAI already works
- **Open in Firefox:**
  `https://sp#.example.org/Shibboleth.sso/Login`
  - `/Shibboleth.sso/Login` is the default login initiator.
    It can be used to start the AAI login process
  - Shibboleth will redirect you to the Discovery Service/WAYF
  - You will see an error message

Service Provider Not Yet in Metadata

Discovery Service (WAYF) and Identity Provider don't "know" your Service Provider yet because they don't have metadata about it.



Error: Invalid Query

The Service Provider '`https://sp23.example.org/shibboleth`' could not be found in metadata and is therefore unknown.

Please contact aai@switch.ch for assistance.

(Federation) Metadata

- SAML Metadata is an XML document
- Typically is provided by a federation operator (e.g. SWITCH)
- Contains descriptions of all SPs and IdPs:
  - **entityID:** The unique identifier of the entity
  - **Supported protocols:** E.g. SAML1, SAML2
  - **X.509 certificates:** Contain the public key of a key pair
  - **Endpoint URLs:** What URLs to query or send messages to
  - **Descriptive** information: E.g. Display name, description, logos
  - **Contact information:** e.g. for support
  - **Registration information:** Who registered this entity when

(Federation) Metadata

1. Have a look at it by opening (`view` or gedit) the file:
   `/var/cache/shibboleth/metadata.aaitest.xml`
   Look for entityID:
   `https://aai-demo-idp.switch.ch/idp/shibboleth`

2. Now also have a look at your SP's metadata by opening:
   `https://sp#.example.org/Shibboleth.sso/Metadata`
   SP can generate (technical) SAML metadata about itself

3. To get your Service Provider into the AAI Test metadata, it has to be registered with the AAI Resource Registry

Resource Registry

Purpose of the SWITCHaai Resource Registry and how to use it

Please consult the table of contents to find this presentation in your hand-outs.

Registration

1. Continue with section 7 "Register Service Provider"
   - Use your regular AAI account to log in to the Resource Registry
   - If you don't have an AAI account:
     1. Click on "Login for AAI Test Federation" button
     2. Select the "AAI Demo Home Organisation"
     3. To authenticate use Username: `sp-training-admin`
        Password: `password`

2. Click on the "Resources" tab
3. Then on "Add a Resource Description"

The description of your training SP will be deleted some time after the training.

Resource Description I

The Shibboleth wizard can download an SP's metadata

- But not if the SP is behind NAT or firewall
- Therefore, you have to provide metadata manually

1. Open in Firefox:
   `https://sp#.example.org/Shibboleth.sso/Metadata`
   - This should open the XML file in the gedit text editor
2. Copy all XML and paste it into the text area on the Resource Registry above the button "Run Shibboleth 2.x wizard"
3. Then hit the button "Run Shibboleth 2.x wizard"
   - "Descriptive Information", "Service Locations" and "Certificates" should now be green (=completed)

Resource Description II

4. Click on "Basic Resource Information":
   - Choose "aai-demo-idp.switch.ch" as Home Organisation.
   - Add a name and description for your Service Provider
     E.g. "Demo SP #", "SP used for improving my Shib knowledge"
   - Finally, click on "Save and Continue"
5. Click on "Contacts"
   - Complete the fields with your email address and contact info.
   - Finally, click on "Save and Continue"
6. Click on "Requested Attributes"
   - Add as required attributes: Targeted ID/Persistent ID, Surname, Given Name, E-Mail, Affiliation, Entitlement and PrincipalName
   - Finally, click on "Save and Continue"

Resource Description III

7. Click on "Intended Audience":
   - Click on "Set all to included".
   - Finally, click on "Save and Continue"
   - You now have completed the Resource Registry
8. Add a comment that you register this Resource Description in the context of the SP training.
9. Click on "Submit for Approval"
   - You should then see the Resource Registration Authority (RRA) administrators who have to approve your Resource Description
   - The RRA admins might ask you for the certificate fingerprint to prove that you generated the certificate
   - To get the certificate's SHA1 fingerprint, run:

```
$ openssl x509 -fingerprint -in /etc/shibboleth/sp-cert.pem
```

Test Access to Service Provider

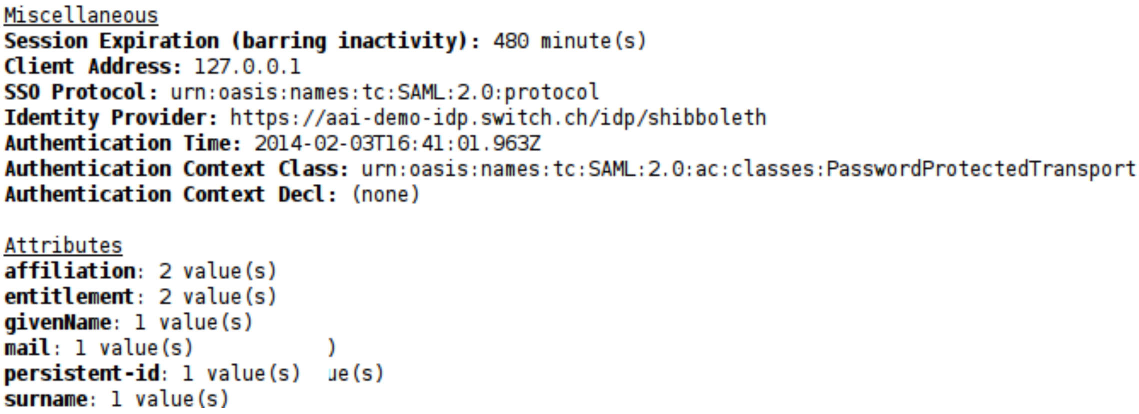
- Once the Resource Description is approved, you become admin
- SP's metadata is included in federation's metadata:
  - In real life, it can up to two hours after the approval of the Resource Description until metadata has propagated to all Identity Providers
  - During the training event, metadata propagation is max. 5 minutes

Test Access

1. In Firefox, open the URL:
   `https://sp#.example.org/Shibboleth.sso/Login`
2. Select the "AAI Demo Home Organisation" on the WAYF
   - If you instead get an error, wait a few minutes more
3. Use "g.utente" and "password" as login name and password

Service Provider Deployment Complete

- When you are back on the Shibboleth Session handler (`/Shibboleth.sso/Session`) and see the following ...



```
Miscellaneous
Session Expiration (barring inactivity): 480 minute(s)
Client Address: 127.0.0.1
SSO Protocol: urn:oasis:names:tc:SAML:2.0:protocol
Identity Provider: https://aai-demo-idp.switch.ch/idp/shibboleth
Authentication Time: 2014-02-03T16:41:01.963Z
Authentication Context Class: urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
Authentication Context Decl: (none)

Attributes
affiliation: 2 value(s)
entitlement: 2 value(s)
givenName: 1 value(s)
mail: 1 value(s)           )
persistent-id: 1 value(s)  ue(s)
surname: 1 value(s)
```

... then you successfully deployed your Service Provider ☺

If all fails: Use Catch-Up Configuration

- Download the SP Catch-Up configuration from the training page:
  `https://www.switch.ch/aai/docs/training/`

- Extract Shibboleth SP configuration files with:
  ```
  $ tar xvzf shib-sp-catchup.tgz  -C / --overwrite
  ```

- From now on use **#**=1 as participation number,
  `sp1.example.org`

Basic Configuration

Goals:

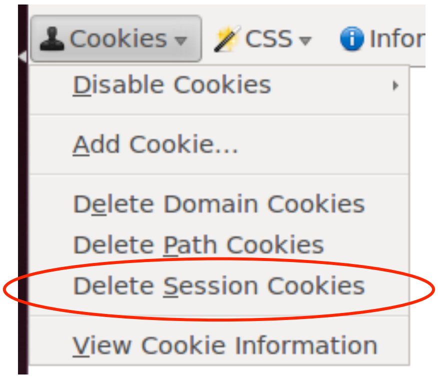
1. Local Logout
2. Understand purpose and structure of SP configuration files
3. Increase log level to DEBUG
4. Make a few simple configuration changes

Logging Out

- To logout locally from the SP and kill your session:
  `https://sp#.example.org/Shibboleth.sso/Logout`
  But this won't delete your session on the IdP!

- **Close the browser and restart it again!**
  Still the easiest and safest method for most web browsers

- Or delete all your session cookies
  For testing and development purposes use the "Firefox Web Developer" extension (installed on VM).



Configuration Files in /etc/shibboleth

- **`shibboleth2.xml` – main configuration file**
- `attribute-map.xml` – attribute handling
- `attribute-policy.xml` – attribute filtering settings
- `*.logger` – logging configuration
- `*Error.html` –HTML templates for error messages
- `localLogout.html` – SP-only logout template
- `globalLogout.html` – single logout template

Recommendation:
Adapt *.html files for production configuration to match the look and feel of the protected application improves user experience.

Shibboleth2.xml Structure

Since Shibboleth 2.4: Simplified configuration but old format still accepted

`<SPConfig>`	Document root element

Outer elements of the shibboleth.xml configuration file

`<OutOfProcess> / <InProcess>`	(Optional) Log settings, extensions
`<UnixListener> / <TCPListener>`	(Optional) Communication shibd/mod_shib
`<StorageService>`	(Optional) Where session information is stored
`<SessionCache>`	(Optional) Session timeouts and cleanup intervals
`<ReplayCache>`	(Optional) Where replay cache is stored
`<ArtifactMap>`	(Optional) Timeout of artifact messages
`<RequestMapper>`	(Optional) Session initiation and access control
`<ApplicationDefaults>`	**Contains the most important settings of SP**
`<SecurityPolicyProvider>`	Define various security options
`<ProtocolProviders>`	Defines supported protocols (SAML, ADFS, ...)

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPConfigurationElements

ApplicationDefaults Structure

You are most likely to modify <ApplicationDefaults>:

`<Sessions>`	Defines handlers and how sessions are initiated and managed. Contains `<SSO>`, `<Logout>`, `<Handler>`
`<Errors>`	Used to display error messages. E.g. logo, email and CSS
`<RelyingParty>`	(optional) To modify settings for certain IdPs/federations
`<MetadataProvider>`	Defines the metadata to be used by the SP
`<AttributeExtractor>`	Attribute map file to use
`<AttributeResolver>`	Attribute resolver file to use
`<AttributeFilter>`	Attribute filter file to use
`<CredentialResolver>`	Defines certificate and private key to be use
`<ApplicationOverride>`	**(Optional) Can override any of the above for certain applications**

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPConfigurationElements

File Editing Commands for Terminal Editor

Editor	nano	vim
Open file	`$ nano <file>`	`$ vim <file>`
Save file	<ctrl>-o	<esc>, :w
Save and exit	<ctrl>-x	<esc>, :wq
Search string	<ctrl>-w, **string**	<esc>, /**string**
Go to line number	<ctrl>--, **number**	<esc>, **number**, <shift>-G

gedit is the recommended text editor. Is started as root user. Its icon is in the launch bar on the left side of the desktop.

Debugging SP Problems on Linux

1. Make sure the edited XML config file is valid and correct XML with:
   ```
   $ xmlwf /etc/shibboleth/shibboleth2.xml
   $ sudo shibd -t  or
   $ sudo shibd -tc /etc/shibboleth/shibboleth2.xml
   ```
2. Stop Shibboleth daemon with:
   ```
   $ /etc/init.d/shibd stop
   ```
3. Increase log verbosity of shibd by seting log level to DEBUG in:
   `/etc/shibboleth/shibd.logger`
4. Have a look at log file and search ERROR or CRIT messages in:
   ```
   $ tail -f /var/log/shibboleth/shibd.log
   ```
5. Start Shibboleth daemon again with:
   ```
   $ /etc/init.d/shibd start
   ```
6. If you fixed an error, also restart Apache with:
   ```
   $ /etc/init.d/apache2 restart
   ```
   Don't forget to set log level back to `INFO` for a production service

Debugging SP Problems on Windows

1. Make sure the edited XML config file is valid XML
   by opening in Firefox the Shibboleth configuration file:
   `C:\opt\shibboleth-sp\etc\shibboleth\shibboleth2.xml`
   Firefox checks if XML file is well-formed
2. Check Shibboleth configuration file:
   ```
   $ C:\opt\shibboleth-sp\sbin\shibd.exe –check
   ```
3. Stop “Shibboleth 2 Daemon” in Windows Services
4. Increase log verbosity of shibd by setting log level to DEBUG in
   `C:\opt\shibboleth-sp\etc\shibboleth\shibd.logger`
5. Have a look at log file and search for ERROR and CRIT messages in:
   `C:\opt\shibboleth-sp\var\log\shibboleth\shibd.log`
6. Start “Shibboleth 2 Daemon” in Windows “Services” again
7. If the error is fixed, also restart Apache or IIS in Windows Services
   Don't forget to set log level back to `INFO` for a production service

Logging

- Your number one friend in case of problems
- `shibd.log` and `transaction.log` written by shibd, `native.log` written by mod_shib
- `*.logger` files contain predefined settings for output locations and a default logging level (`INFO`) along with useful categories to raise to `DEBUG`

Increase Log Level to DEBUG

- Raise categories:

```
$ vim /etc/shibboleth/shibd.logger

Line 2:
log4j.rootCategory=DEBUG, shibd_log, warn_log

Line 16:
# tracing of SAML messages and security policies
log4j.category.OpenSAML.MessageDecoder=DEBUG
log4j.category.OpenSAML.MessageEncoder=DEBUG
log4j.category.OpenSAML.SecurityPolicyRule=DEBUG
```

Make Session Handler Show Values

- For debugging purposes, it helps seeing the attribute values on `/Shibboleth.sso/Session`

- Open the `/etc/shibboleth/shibboleth2.xml`

```
Line 53:
<!-- Session diagnostic service. -->
<Handler type="Session"
         showAttributeValues="true"
         Location="/Session"/>
```

Apply Configuration Changes

- To follow the shibd log file in real time and examine what happens during a login use tail (not available on Windows):

```
$ tail -f /var/log/shibboleth/shibd.log
```

- `shibd` reloads configuration when `shibboleth2.xml` is changed, but generally it is still better to restart shibd:

```
$ /etc/init.d/shibd restart
```

  Shibboleth detects invalid configurations if it reloads them automatically.
  In this case it will continue to use the last valid configuration it still has in memory.
  This behavior hides errors that will only be discovered at the next restart!

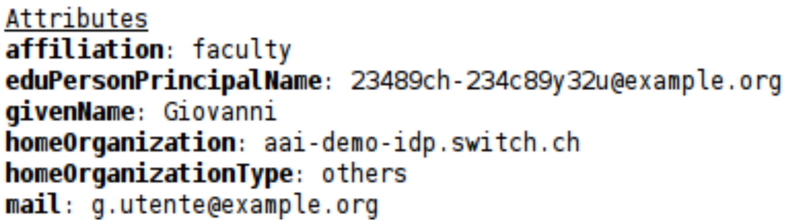
Check Changes

Login again with:
`https://sp#.example.org/Shibboleth.sso/Login`

You should see the encrypted and decrypted XML assertion received by SP

```
DEBUG Shibboleth.SSO.SAML2 [1]: decrypted Assertion: <saml2:Assertion
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"
ID="_efc943c04c742ae96d15e19e95afba68"
IssueInstant="2014-02-10T14:59:29.841Z" Versi
on="2.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">[...]
```

And the Session Handler should now display also the attribute values:



```
Attributes
affiliation: faculty
eduPersonPrincipalName: 23489ch-234c89y32u@example.org
givenName: Giovanni
homeOrganization: aai-demo-idp.switch.ch
homeOrganizationType: others
mail: g.utente@example.org
```

SP Metadata Features

- Metadata describes the other components (IdPs) that the Service Provider can communicate with
- **Four primary methods built-in:**
  - **Local metadata file (you download/edit it manually)**
  - **Downloaded remotely from URL (periodic refresh, local backup)**
  - Dynamic resolution of entityID (=URL), hardly used
  - "Null" source that disables security ("OpenID" model), hardly used
- Security comes from metadata filtering, either by you or the SP:
  - Signature verification
  - Expiration dates
  - White and blacklists

Signature Verification

- Have a look at the configuration :

```
$ vim /etc/shibboleth/shibboleth2.xml
```

Change the certificate of MetadataProvider signature verification:

Line 72:

```
<MetadataProvider type="XML" […] >
  <MetadataFilter type="Signature" [...]
     <TrustEngine type="StaticPKIX"
        certificate="sp-cert.pem"
        [...]
</MetadataProvider>
```

- Then go to next slide…

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPMetadataFilter

Signature Verification Continued

Run `$ shibd -tc /etc/shibboleth/shibboleth2.xml`
Output should look like:

```
ERROR OpenSSL : path validation failure at depth(2): self signed
certificate in certificate chain
ERROR OpenSSL : path validation failure at depth(2): self signed
certificate in certificate chain
WARN OpenSAML.MetadataFilter.Signature : filtering out group at root of
instance after failed signature check: TrustEngine unable to verify
signature.
CRIT Shibboleth.Application : error initializing MetadataProvider:
SignatureMetadataFilter unable to verify signature at root of metadata
instance.
overall configuration is loadable, check console for non-fatal problems
```

Metadata could not be loaded because it was signed with a different key (we "broke" the setup). So, let’s get the right key…

Signature Verification Corrected again

- To correct the metadata signature validation again :

```
$ vim /etc/shibboleth/shibboleth2.xml
```

Change the certificate of MetadataProvider signature verification:

Line 72:

```
<MetadataProvider type="XML" […] >
  <MetadataFilter type="Signature" [...]
     <TrustEngine type="StaticPKIX"
        certificate="SWITCHaaiRootCA.crt.pem"
        [...]
</MetadataProvider>
```

- Then restart shibd and try logging in again to check if metadata is accepted again

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPMetadataFilter

Skip Discovery Service

- Discovery Service/WAYF can be skipped if service has only users from one organisation.

```
$ vim /etc/shibboleth/shibboleth2.xml
```

Change the SSO element to look like below to send
users directly to the AAI Demo Home Organisation for authentication:

Line 34:

```
<SSO
    entityID="https://aai-demo-idp.switch.ch/idp/shibboleth">
SAML2
</SSO>
```

- Then again access `/Shibboleth.sso/Login`
  You should now directly be sent to login page of Demo IdP

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPMetadataFilter

Discovery Service Options

What Discovery Service implementations are there? What are the advantages and considerations?

Please consult the table of contents to find this presentation in your hand-outs.

Attribute Handling

Goals:

1. Learn how attributes are mapped and filtered
2. See how attributes can be used as identifiers
3. Add an attribute mapping and filtering rule

Attribute Mappings

- SAML attributes from any source are "extracted" using the configuration rules in attribute map file in: `/etc/shibboleth/attribute-map.xml`
- Each element is a rule for decoding a SAML attribute and assigning it a local `id` which becomes its mapped variable name
- Attributes can have one or more `id` and multiple attributes can be mapped to the same `id`
- The `id` is also used as header name in the webserver for this attribute.

Dissecting an Advanced Attribute Rule

Line 139 (attribute-map.xml):

```
<Attribute
   name="urn:mace:dir:attribute-def:eduPersonScopedAffiliation"
   id="scoped-affiliation" >
    <AttributeDecoder xsi:type="ScopedAttributeDecoder"
     caseSensitive="false"/>
</Attribute>
```

- `name`
  SAML attribute name or NameID format to map from
- `id`
  The primary "id" to map into, also used in web server environment
- `AttributeDecoder xsi:type`
  Decoder plugin to use (defaults to simple/string)
- `caseSensitive`
  How to compare values at runtime (defaults to true)

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPAttributeExtractor

Adding Attribute Mappings

- Add eduPersonPrincipal name SAML 2 attribute mappings:

```
$ vim /etc/shibboleth/attribute-map.xml

Line 18:
 <Attribute
  name="urn:oid:1.3.6.1.4.1.5923.1.1.1.6"
  id="eduPersonPrincipalName" />
```

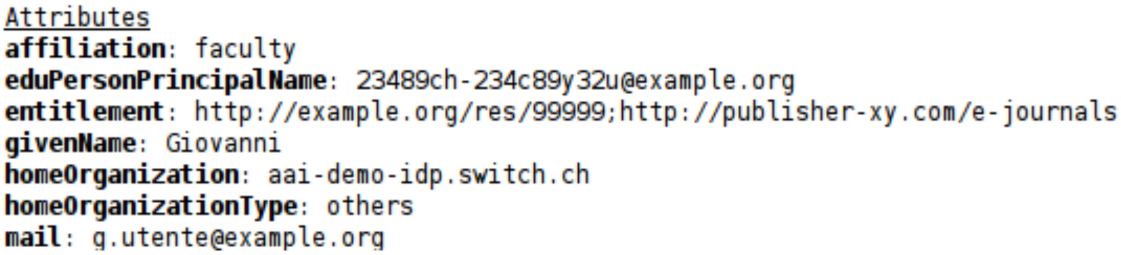
- After saving, changes take effect immediately but NOT for any existing sessions

- Therefore, restart your browser (or delete your session cookies) and continue on next slide …

Testing Added Attribute Mapping

- Then access `/Shibboleth.sso/Login` and log in again.

- After that, check the Shibboleth Session Handler
  You should now also see the `eduPersonPrincipalName`



```
Attributes
affiliation: faculty
eduPersonPrincipalName: 23489ch-234c89y32u@example.org
entitlement: http://example.org/res/99999;http://publisher-xy.com/e-journals
givenName: Giovanni
homeOrganization: aai-demo-idp.switch.ch
homeOrganizationType: others
mail: g.utente@example.org
```

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPHandler

Side note: Aliases

- The `attribute-map.xml` supported attribute aliases like:

```
<Attribute id="Shib-EP-Affiliation"
  name="urn:mace:dir:attribute-def:eduPersonAffiliation"
  aliases="affiliation aff affil" />
```

- Allowed using aliases in access control rules like:

```
require affiliation staff
require Shib-EP-Affiliation staff
```

- Aliases are deprecated since 2.5
- Recommend to use only offical LDAP names in the future (e.g. `surname, givenName, mail`)

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPAttributeExtractor

Attribute Filtering

- Answers the "who can say what" question on behalf of an application
- Service Provider can make sure that only allowed attributes and values are made available to application
- Some examples:
  - constraining the possible values or value ranges of an attribute (e.g. eduPersonAffiliation, telephoneNumber, ...)
  - limiting the scopes/domains an IdP can speak for (e.g. university x cannot assert faculty@university-z.edu)
  - limiting custom attributes to particular sources

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPAttributeFilter

Default Filter Policy

- As default, **attributes are filtered out unless there is a rule!**
- Shared rule for legal affiliation values
- Shared rule for scoped attributes
- Generic policy applying those rules and letting all other attributes through
- Check `/var/log/shibboleth/shibd.log` for signs of filtering in case of problems with attributes not being available.
  You would find something like "`no values left, removing attribute (#attribute name#)`"

Add a Source-Based Filtering Rule

- Add a rule to limit acceptance of "`surname`" to a single IdP:

```
$ vim /etc/shibboleth/attribute-policy.xml
```

Add surname mapping **and** comment out catch-all rule at bottom :

```
Line 145:
<afp:AttributeRule attributeID="surname">
  <afp:PermitValueRule
    xsi:type="AttributeIssuerString"
    value="https://aai-demo-idp.switch.ch/idp/shibboleth"/>
</afp:AttributeRule>
<!--
<afp:AttributeRule attributeID="*">
     <afp:PermitValueRule xsi:type="ANY"/>
 </afp:AttributeRule>
-->
```

After login: **`givenName`** is filtered out but **`surname`** is not due to rule

https://wiki.shibboleth.net/confluence/display/SHIB2/IdPAddAttributeFilter

Add Catch-all Rule Again

- Add a rule to limit acceptance of "`surname`" to a single IdP:

```
$ vim /etc/shibboleth/attribute-policy.xml
```

Line 145:

```
<afp:AttributeRule attributeID=”surname">
  <afp:PermitValueRule xsi:type="AttributeIssuerString"
   value="https://non.existing.example.org/idp/shibboleth"/>
</afp:AttributeRule>
```

Uncomment catch-all rule at bottom:

```
<afp:AttributeRule attributeID="*">
              <afp:PermitValueRule xsi:type="ANY"/>
 </afp:AttributeRule>
```

Then login again: **`surname`** is now filtered out but other attributes aren't.

Because a specific rule exists, the catch-all rule does not apply anymore!

Remove Specific Rule

- Remove rule for (non-) acceptance of `surname`:

```
$ vim /etc/shibboleth/attribute-policy.xml
```

Delete rule for `surname` again.

- Save file and access `/Shibboleth.sso/Login` again
- Now you should see the `surname` attribute again

Interfederation Attributes and Checking

Excursion about user attributes available via AAI.

Please consult the table of contents to find this presentation in your hand-outs.

Session Initiation

Goals:

1. Learn how to initiate a Shibboleth session
2. Understand their advantages/disadvantages
3. Know where to require a session, what to protect

Content Protection and Session initiation

- Before access control (will be covered later on) can occur, a Shibboleth session must be initiated
- Session Initiation and content protection go hand in hand
- Requiring a session means the user has to authenticate
- Only authenticated users can access protected content

Content Protection Settings

Protect hosts, directories, files or queries

- **Apache**
  .htaccess (dynamic) or httpd.conf (static)
- **Apache / IIS / other**
  <RequestMap> in shibboleth2.xml
  Requires Shibboleth to know exact hostname
  Very powerful and flexible thanks to boolean/regex operations
- Try accessing `https://sp#.example.org/secure/`
  You should get access because the directory is not protected (yet)
  `/secure/` used to be protected by default in older Shibboleth distributions

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPAccessControl

Content Protection with .htaccess File

- Let’s protect the directory by requiring a Shibboleth session:

```
$ vim /var/www/secure/.htaccess
```

```
AuthType shibboleth
require shibboleth
ShibRequestSetting requireSession true
```

Synonym for the last line (used in Shibboleth 1.3, deprecated):

```
ShibRequireSession On
```

Rules could also be in static httpd configuration file directly, see
`/etc/apache2/conf.d/shib.conf` (default rule for `/secure/`)

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPApacheConfig

Session Initiation and Content Settings

- **forceAuthn** `(ShibRequestSetting forceAuthn true)`
  - Disable Single-Sign on and force a re-authentication
- **isPassive** `(ShibRequestSetting isPassive true)`
  - Check whether a user has an SSO session and if he has, automatically create a session on SP without any user interaction
- Use a specific IdP to use for authentication
- Requesting types of authentication
  - E.g enforce X.509 user certificate authentication
- Custom error handling pages to use
- Redirection-based error handling
  - In case of an error, redirect user to custom error web page with error message/type as GET arguments

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPContentSettings

Test Content Protection Rule

- Clear session and then access the protected URL again: `https://sp#.example.org/secure`
- Authentication is enforced and access should be granted
- Currently, all authenticated users get access
- Content protection to limit access only to specific users will be covered later

Content Protection with RequestMap

- `mod_shib` provides request URL to `shibd` to process it
  Therefore, `shibd` can enforce access control as well
  This is required for IIS web servers!
- First ensure that requests for `/other-secure/` are handled by `shibd` without setting any specific session requirements:

```
$ vim /var/www/other-secure/.htaccess
```

```
AuthType shibboleth
require shibboleth
```

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPRequestMapper

How to Add a RequestMap

- Open the Shibboleth configuration:

  `$ vim /etc/shibboleth/shibboleth2.xml`

  Before `<ApplicationDefaults>` insert a `<RequestMap>`:

  Line 7:

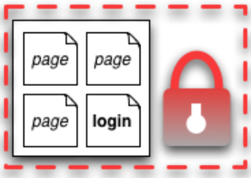
```
<RequestMapper type="Native">
  <RequestMap applicationId="default">
    <Host name="sp#.example.org">
      <Path name="other-secure"
       authType="shibboleth" requireSession="true"/>
    </Host>
  </RequestMap>
</RequestMapper>
```

- Clearing session and then accessing `/other-secure/` now, one also is forced to authenticate

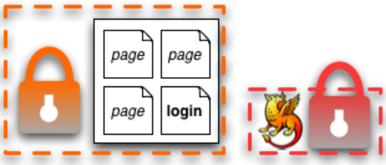
https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPRequestMapper

Where to Require a Shibboleth Session

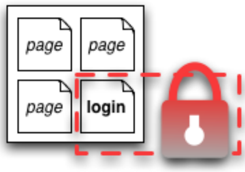
- **Whole application with "required" Shibboleth session**
  - Easiest way to protect a set of documents
  - No other authentication methods possible
  - Needs special config to preserve HTTP POST requests



- **Whole application with "lazy" Shibboleth session**
  - Also allows for other authentication methods
  - Authorization can only be done in application



- **Only page that sets up application session**
  - Well-suited for dual login
  - Application can control session time-out
  - **Generally the most popular solution (many Shibboleth-enabled applications use this)**



Protect a Simple Web Application

- Access `https://sp#.example.org/cgi-bin/attribute-viewer`
  Simple CGI script as a sample application that show attributes.
- Lets protect that script with Shibboleth by requiring a session:

```
$ vim /usr/lib/cgi-bin/.htaccess
```

```
AuthType shibboleth
ShibRequestSetting requireSession true
require shibboleth
```

This will require a session for all requests to `/cgi-bin/` and make attributes available to application in environment.

- Try again to access script with a browser:

  Script should enforce authentication and show attributes

Make Script "see" Shibboleth Session

- What if we wanted to grant access also to non-authenticated users but use attributes if somebody is authenticated?
- Use Shibboleth (lazy) session:

```
$ vim /usr/lib/cgi-bin/.htaccess
```

```
AuthType shibboleth
require shibboleth
```

This will not require a session but make attributes available to application in environment if somebody has a session.

- Try again with a browser:

  `https://sp#.example.org/cgi-bin/attribute-viewer`

  Unauthenticated access still possible. No attributes are shown yet.

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPEnableApplication

REMOTE_USER

- Special single-valued variable that all web applications should support for container-managed authentication of a unique user.

- Any attribute, once extracted/mapped, can be copied to REMOTE_USER

- Multiple attributes can be examined in order of preference, but only the first value will be used.

Changing REMOTE_USER

- In case your application needs to have a remote user for authentication, you just could make shibboleth put an attribute (e.g. "mail") as REMOTE_USER:

```
/etc/shibboleth/shibboleth2.xml

Line 12 in <ApplicationDefaults>:
REMOTE_USER="mail eppn persistent-id targeted-id"
```

- If mail attribute is available, it will be put into REMOTE_USER

- Attribute `mail` has precedence over `persistent-id` in this case
- This allows very easy "shibbolization" of some web applications

How To Initiate a (Lazy) Session

- Close your browser, and access the attribute-viewer again,

  ```
  https://sp#.example.org/cgi-bin/attribute-viewer
  ```

- Then click on one of the buttons and login at Test IdP
  You should be sent to IdP or WAYF and attribute-viewer should display attributes after successful authentication

- Have a look at the HTML source and what it does:

  ```
  https://sp#.example.org/cgi-bin/attribute-viewer
  ```

- Script initiates Shibboleth session by sending user to:

  ```
  /Shibboleth.sso/Login?target=/cgi-bin/attribute-viewer
  &entityID=https://aai-demo-idp.switch.ch/idp/shibboleth
  ```

Try to Initiate a Session Yourself

- Try to construct a Session Initiation URL yourself by using these parameters to see the result: e.g. try supplying the IdP:

  ```
  https://sp#.example.org/Shibboleth.sso/Login?
    target=https://sp#.example.org/cgi-bin/attribute-viewer&
    entityID=https://aai-demo-idp.switch.ch/idp/shibboleth
  ```

- This way, a session using a specific IdP can be initiated directly with a link, e.g. on a portal web page.

- This allows creating "login links" to skip the WAYF/Discovery Service

- It also allows overriding certain content settings

Session Creation Parameters

- Key Parameters
  - `target` (defaults to homeURL or "/")
  - `entityID` (specific IdP to use or WAYF/DS if not present)

- Most parameters can be set at three places.
  In order of precedence:
  - In query string parameter of a URL to handler
  - a content setting (.htaccess or RequestMap)
  - <SessionInitiator> element

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPSessionCreationParameters

Lazy Sessions Summary

- Won’t enforce a Shibboleth session but use it if it is available
  - If valid **session exists**
    - then process it as usual (put attributes in server environment, etc.),
    but if a **session does NOT exist** or is invalid,
    - ignore it and pass on control to application

- Three common cases:
  - Public and private access to the same resources
  - Separation of application and SP session
  - Dual login (use Shibboleth and some other authentication method)

Using Lazy Sessions

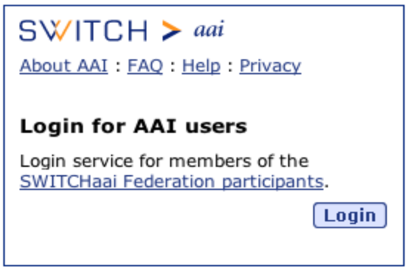
- In place of an API to "doLogin", the SP uses redirects:
  `https://testsp1.example.org/Shibboleth.sso/Login`

- When your application wants a login to happen, redirect the browser to a SessionInitiator (`/Shibboleth.sso/Login` by convention) with any parameters you want to supply

Some Concerns Regarding Dual Login

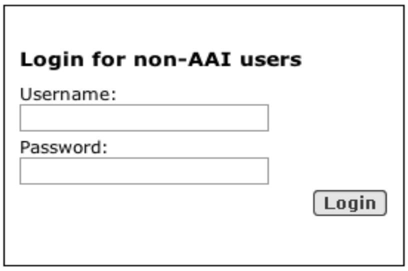
- Can be a viable option in case application must also be accessed by non-Shibboleth users

- Generally not recommended due to issues with:
  - **Usability:** Difficult to teach the users how to authenticate
  - **Security:** Shibboleth users shouldn't enter their password in the login form for the non-Shibboleth users…



Login for AAI users

Login service for members of the SWITCHaai Federation participants.



Login for non-AAI users

Username:

Password:

Access Control

Goals:

1. Create some simple access control rules
2. Get an overview about the three ways to authorize users
3. Understand their advantages/disadvantages

Access Control

- Integrated in Service Provider via an AccessControl API built into the request processing flow
- Two implementations are provided by the SP:
  - .htaccess "require" rule processing
  - XML-based policy syntax attached to content via RequestMap
- Third option: Integrate access control into web application

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPAccessControl

Access Control Mechanisms

	1.a httpd.conf	1.b .htaccess	2. XML AccessControl *	3. Application Access Control
⊕	▪ Easy to configure ▪ Can also protect locations or virtual files ▪ URL regex	▪ Dynamic ▪ Easy to configure	▪ Platform independent ▪ Powerful boolean rules ▪ URL Regex ▪ Dynamic	▪ Very flexible and powerful with arbitrarily complex rules ▪ URL Regex Support
⊖	▪ Only works for Apache ▪ Not dynamic ▪ Very limited rules	▪ Only works for Apache ▪ Only usable with "real" files and directories	▪ XML editing ▪ Configuration error can prevent SP from restarting	▪ You have to implement it yourself ▪ You have to maintain it yourself

* Configured in RequestMap or referenced by an .htaccess file

1. Apache httpd.conf or .htaccess Files

- Work almost like known Apache "require" rules
  E.g `Require shib-attr affiliation staff`
  or `Require shib-attr mail user1@idp1.com user2@idp2.org`
- Special rules:
  - `shibboleth` (no authorization)
  - `valid-user` (require a session, but NOT identity)
  - `user` (REMOTE_USER as usual)
  - `authnContextClassRef`, `authnContextDeclRef`
- Default is boolean "OR", use `ShibRequireAll` for AND rule
- Regular expressions supported using special syntax:

```
Require shib-attr mail ~ [exp]
Require shib-attr mail ~ ^.*@(it|faculty).example.org$
```

1. Example .htaccess File

- Require a user to be a staff member:

```
$ vim /var/www/staff-only/.htaccess
```

```
AuthType shibboleth
ShibRequestSetting requireSession true
Require shib-attr affiliation staff
```

Then access: `https://sp#.example.org/staff-only/`
with `h.mitarbeiter`/`password`. Access should be granted.

- Then try the same again with `p.etudiant`/`password`
  Access should be denied

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPApacheConfig

1. More Advanced .htaccess File

- Require a user to be a student or to have an entitlement:

```
$ vim /var/www/students-only/.htaccess
```

```
AuthType shibboleth
ShibRequestSetting requireSession 1
Require shib-attr affiliation student
Require shib-attr entitlement ~ .*example\.org.*
```

Then access : `https://sp#.example.org/students-only/`
with `p.etudiant`/`password` . Access should be granted.

- Then try the same with `h.mitarbeiter`/`password`
  Access should be granted too because this staff member has entitlement!

2. XML Access Control

- Can be used for access control independent from web server and operating system
- XML Access control rules can be embedded inside RequestMap or be dynamically loaded from external file
- Boolean operators (AND,OR,NOT) can be used
- .htaccess files can reference to XMLAccessControl files
  Allows outsourcing access control rules to non-root users

2. XML Access Control Example

- Require an entitlement or specific users (same as before):

```
$ vim /etc/shibboleth/shibboleth2.xml
```

```
Line 8:
<RequestMapper type="Native">
  <RequestMap applicationId="default">
    <Host name="sp#.example.org">
      <Path name="other-secure" authType="shibboleth" requireSession="true" />
        <Path name="cgi-bin" authType="shibboleth" requireSession="true">
          <AccessControl>
            <OR>
              <RuleRegex require="entitlement">^.*agreement.*$ </RuleRegex>
              <Rule require="affiliation">student</Rule>
            </OR>
          </AccessControl>
        </Path>
     </Host>
  </RequestMap>
</RequestMapper>
```

- Access `https://sp#.example.org/cgi-bin/attribute-viewer`
  Once with `h.mitarbeiter` (access denied) and `p.etudiant` (access granted)

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPXMLAccessControl

3. Application Managed Access Control

- Application can access and use Shibboleth attributes by reading them from the web server environment
- Attributes then can be used for authentication/access control/authorization

PHP:

```
if ($_SERVER['affiliation'] == 'staff;member')
    { grantAccess() }
```

Perl:

```
if ($ENV{'affiliation'} == 'staff;member')
    { &grantAccess() }
```

Java:

```
if (request.getHeader("affiliation").equals("staff;member") )
   { grantAccess() }
```

SWITCHtoolbox and Group Management Tool

Excursion about using the Group Management Tool (GMT) or the SWITCHtoolbox.

Please consult the table of contents to find this presentation in your hand-outs.

Using the SWITCHaai Embedded WAYF

Goals:

1. Add the Embedded WAYF to a HTML web page
2. Configure Embedded WAFY
3. Add the Guest Login IdP to the Embedded WAYF
4. Configure discovery for a single IdP

How to Add Embedded WAYF

- Open in Web Browser (login with your AAI Account): https://rr.aai.switch.ch/gen_embedding_code.php
  Then type `sp#.example.org` and select your SP
- Or without AAI Login: https://wayf-test.switch.ch/SWITCHaai/WAYF/embedded-wayf.js/snippet.html
- Copy the whole HTML snippet
- Then open /var/www/index.html in

  ```
  $ vim /var/www/index.html
  ```

  and paste the snippet after line 24

  ```
  <!-- EMBEDDED-WAYF-START -->
  <script type="text/javascript"><!--

  // To use this JavaScript, please access:
  […]
  ```

Configure Embedded WAYF

- Adapt essential settings of Embedded WAYF

```
$ vim /var/www/index.html
```

Edit the following mandatory settings of the Embedded WAYF

```
// EntityID of the Service Provider that protects this Resource
[…]
var wayf_sp_entityID = "https://sp#.example.org/shibboleth";

// Shibboleth Service Provider handler URL
[…]
var wayf_sp_handlerURL = "https://sp#.example.org/Shibboleth.sso";

// URL on this resource that the user shall be
[…]
var wayf_return_url = "https://sp#.example.org/cgi-bin/attribute-viewer";
```

Test the Embedded WAYF

- Access the URL `https://sp#.example.org/`
- Select the Test Identity Provider in the drop-down list
- Authenticate with demostudent/demo
  You should see access to the attribute-viewer

- Go back to `https://sp#.example.org/`
  Note how the Embedded WAYF now looks different

- Set `var wayf_auto_redirect_if_logged_in = true;`
  and open again `https://sp#.example.org/`

Configure discovery for a single IdP

Configure your SP to use only a specific IdP (demo-idp or guest-login) and skip the Discovery Service/WAYF:

```
<SSO entityID="https://aai-demo-idp.switch.ch/idp/shibboleth">
              SAML2
</SSO>
```

or

```
<SSO entityID="https://aai.guest-login.ch/idp/shibboleth">
              SAML2
</SSO>
```

Test the configuration:

```
$ shibd -tc /etc/shibboleth/shibboleth2.xml
```

Logout

What is possible and what are the limitations of local and global logout.

Please consult the table of contents to find this presentation in your hand-outs.

Service Provider Virtualization

How to protect multiple applications with one physical Service Provider and how to have one Shibboleth application distributed across multiple physical hosts.

Please consult the table of contents to find this presentation in your hand-outs.

Service Provider Handlers

Goals:

1. Understand the idea of a handler
2. Get an overview about the different types of handlers
3. Know how to configure them if necessary

SP Handlers

- **"Virtual" applications inside the SP with API access:**
  - SessionInitiator (requests)
    Start Shibboleth sesion: `/Shibboleth.sso/Login`
  - AssertionConsumerService (incoming SSO)
    Receives SAML assertions: `/Shibboleth.sso/SAML/POST`
  - LogoutInitiator (SP signout)
    Log out from SP: `/Shibboleth.sso/Logout`
  - SingleLogoutService (incoming SLO)
  - ManageNameIDService (advanced SAML)
  - ArtifactResolutionService (advanced SAML)
  - Generic (diagnostics, other useful features)
    - Returns session information: `/Shibboleth.sso/Session`
    - Returns detailed SP status: `/Shibboleth.sso/Status`
    - Returns SP metadata: `/Shibboleth.sso/Metadata`

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPHandler

SP Handlers

- The URL of a handler = handlerURL + the Location of the handler.
  E.g. for a virtual host testsp.example.org with handlerURL of "/Shibboleth.sso", a handler with a Location of `/Login` will be `https://sp#.example.org/Shibboleth.sso/Login`

- Handlers aren't always SSL-only, but usually should be Recommended to set handlerSSL="true" in shibboleth2.xml

- Metadata basically consists of entityID, keys and handlers

- Handlers are never "protected" by the SP
  But sometimes by IP address (e.g. with `acl="127.0.0.1"`)

Error Pages and Customization

How to improve the user experience by customizing error pages.

Please consult the table of contents to find this presentation in your hand-outs.

Shibboleth-aware Applications

Some examples of applications that already support Shibboleth out of the box.

Please consult the table of contents to find this presentation in your hand-outs.