Presentation Overview

1. Why adapting web applications to an AAI?
2. Do I have to care about authentication?
3. The three ways of authorization
4. How to adapt existing web applications
5. Session management and WAYF integration
AAI - Key to rule access them all

AAI = Authentication and Authorization Infrastructure
Without AAI

- Tedious user registration at all resources
- Unreliable and outdated user data at resources
- Different login processes
- Many different passwords
- Many resources not protected due to difficulties
- Often IP-based authorization
- Costly implementation of inter-institutional access
With AAI

- No user registration and user data maintenance at resource needed
- Single login process for the users
- Many new resources available for the users
- Enlarged user communities for resources
- Authorization independent of location
- Efficient implementation of inter-institutional access
Attribute Based Authorization Example

Dermatology Online with Interactive Technology (DOIT)

DOIT: http://www.cyberderm.net

Authorization Rule

- HomeOrg = UniZH | UniBE | UniL
- Affiliation = Student
- StudyBranch = Medicine
- StudyLevel = 20
Key points

AAI makes life easier for everybody

- Users have a single account for all their services
- Authentication is done only at user’s home organization
- User data is maintained only once
- Collaboration between multiple organizations is simplified
Presentation Overview

1. Why adapting web applications to an AAI?
2. Do I have to care about authentication?
3. The three ways of authorization
4. How to adapt existing web applications
5. Session management and WAYF integration
Conventions

The following slides:

• Cover web-based applications only
• Are aimed mostly at e-learning/collaboration applications
• Focus on Shibboleth (the “standard“)

Names

<table>
<thead>
<tr>
<th>Technical</th>
<th>Organizational</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Provider (SP)</td>
<td>= Resource</td>
</tr>
<tr>
<td>Identity Provider (IdP)</td>
<td>= Home Organization</td>
</tr>
<tr>
<td>IdP Discovery Service (WAYF)</td>
<td>= Where Are You From Service</td>
</tr>
</tbody>
</table>
The two ‘A’s in AAI

**Authentication**: Who are you?

**Authorization**: What are you allowed to do?

- **Authentication** is done by the Identity Provider, so you don’t have to care about that.

- **Authorization** has to be done by the resource administrator!

- There are three ways to authorize users…

**More information on:**

http://www.switch.ch/aai/support/serviceproviders/sp-access-rules.html
Authentication

• Is done at the user’s Identity Provider
  – So, you don’t have to care about that

• Authentication has to be examined on resource side
  – Check for presence of a unique attribute
PHP Authentication Example

```php
<?php
    // read AAI uniqueID
    $uniqueid= null;
    // check if set and not empty
    if (isset($_SERVER['HTTP_SHIB_SWISSEP_UNIQUEID']) and !empty($_SERVER['HTTP_SHIB_SWISSEP_UNIQUEID'])) {
        $uniqueid= $_SERVER['HTTP_SHIB_SWISSEP_UNIQUEID'];
        // decode UTF8 to Latin1 (Surname, Given Name, Address, ...)
        $uniqueid= utf8_decode($uniqueid);
        // continue processing...
    } else {
        // Error: attribute is missing!
    }
?>
```

- Read Shibboleth attributes from Apache environment
- Attribute value can be null or empty and is UTF-8 encoded
Java Authentication Example

```java
public class MyHttpServlet extends HttpServlet {
    public void doPost(HttpServletRequest request, HttpServletResponse response)
        throws IOException, ServletException {
        // get the AAI uniqueID
        String uniqueid = request.getHeader("Shib-SwissEP-UniqueID");
        // check not null and not empty
        if (uniqueid != null && !uniqueid.equals("")) {
            // decode UTF8 to Latin1 (Surname, Given Name, Address, ...)
            uniqueid = new String(uniqueid.getBytes("ISO-8859-1"), "UTF-8");
            // continue processing...
        } else {
            // Error: attribute is missing!
            throw new ServletException("Shibboleth HTTP header 'Shib-SwissEP-UniqueID' is missing");
        }
    }
}
```

- Read Shibboleth attributes from HTTP request header
- Attribute value can be null or empty and is UTF-8 encoded
Presentation Overview

1. Why adapting web applications to an AAI?
2. Do I have to care about authentication?
3. The three ways of authorization
4. How to adapt existing web applications
5. Session management and WAYF integration
I. Authorization with Apache

Access rules can be configured in

- Static `httpd.conf` file
- Dynamically processed `.htaccess` files

For the elements `<Files>`, `<Directory>` or `<Location>`

```
AuthType shibboleth
ShibRequireSession On
ShibRequireAll On
require affiliation student
require homeOrganization uzh.ch ethz.ch

Example: .htaccess file
```

```
<Location /secure/admins>
AuthType shibboleth
ShibRequireSession On
require mail ~.*@it.uzh.ch$
</Location>

Example: httpd.conf directive
```
II. Authorization with Shibboleth

Access rules can be configured with boolean rules:

- Inline in `shibboleth.xml` (in RequestMap)
- Linked in external `XMLAccessControl` file

```
<Host name="sp.example.org">
  <Path name="secure" authType="shibboleth" requireSession="true">
    <AccessControl>
      <AND>
        <Rule require="affiliation">student</Rule>
        <OR>
          <Rule require="homeOrganization">ethz.ch</Rule>
          <Rule require="homeOrganization">unizh.ch</Rule>
        </OR>
        <NOT>
          <Rule require="homeOrganization">vho-switchaai.ch</Rule>
        </NOT>
      </AND>
    </AccessControl>
  </Path>
</Host>
```
III. Authorization within Application

Attributes are accessible in web server environment variables:

- **PHP:**
  
  ```
  if ($_SERVER['HTTP_SHIB_EP_AFFILIATION'] == 'staff') {...}
  ```

- **Perl:**
  
  ```
  if ($ENV{'HTTP_SHIB_EP_AFFILIATION'} == 'staff') {...}
  ```

- **Java:**
  
  ```
  if (request.getHeader("Shib-EP-Affiliation").equals("staff") ) {...}
  ```

- **Python:**
  
  ```
  if(environ.get('HTTP_SHIB_EP_AFFILIATION') == "staff") {...}
  ```

… and so on. Works for all CGIs/scripting languages.
Comparison of Authorization Methods

- **Apache Authorization**
  - No complex rules possible
  - `.htaccess` files
    - Dynamic reloading of rules
    - Only files and folder can be protected
  - `httpd.conf` file
    - Can also protect (virtual) locations with regular expressions
    - Apache has to be reloaded after changing rules

- **Shibboleth Authorization**
  - Complex boolean rules can be used
  - Dynamic reloading of rules
  - Locations can be protected (with 2.0 even with regular expressions)

- **Web Application Authorization**
  - Completely flexible and dynamic rules
  - Code has to be adapted and maintained
Users with common attributes

Access Rule

```
HomeOrg = IdP X| IdP Y| IdP Z
Affiliation = Student
StudyBranch = Medicine
```
Specific users without common attributes

How can these users be authorized?
Solution 1: Create common attribute

Add an entitlement attribute for specific users

Access Rule

Require entitlement urn:mace:dir:entitlement:common-lib-terms

Easy solution for a difficult problem

Additional work for user directory administrator
Difficult to efficiently manage many entitlement values

Only IdP admin can manage access
Solution 2.a: Use uniqueIDs or email

1. Get unique IDs or AAI email addresses of users.
2. Create access rules like:

   Access Rule
   
   **require uniqueID 465@idpx.ch 234@idpy.ch [...]**
   **require email hans.muster@idpx.ch pierre.m@idpz.ch [...]**

   Straight-forward solution

   SP administrator must know unique ID/Email address
   Difficult to efficiently manage for many users/apps
   Only SP admin can manage access
Solution 2.b: Group Management Tool

[Diagram showing group management system with connections and user roles]
Facts about the Group Management Tool

- Open Source, software (BSD license)
- Light-weight and easy to install PHP application
- Human readable text files to store group data

Features
- Manage multiple groups for multiple applications
- Three user/admin role classes with different privileges
- Transfer privileges to other users
- Invite new users to join group via email
- User can request to join a group (self-registration)
- Generate authorization files (Apache .htaccess)
- PHP, PERL, JAVA API for use on remote hosts

http://www.switch.ch/aai/gmt
Presentation Overview

1. Why adapting web applications to an AAI?
2. Do I have to care about authentication?
3. The three ways of authorization
4. How to adapt existing web applications
5. Session management and WAYF integration
Web Application Integration with Shibboleth

• New application
  – Static content (protect documents, files, ...)
    Easy to protect with Shibboleth
    Just require a session for the content and set authorization rules
  – Dynamic application where you are the developer
    Quite easy too because you can design in with AAI in mind
    Has some similarities with Case 2

• Existing dynamic application (with a user session)
  – Case 1: Without user management
    ■ Applications that perform authentication with BASIC AUTH
    ■ Only needs REMOTE_USER set by Shibboleth
  – Case 2: With user management
    ■ Needs generally more customizations
    ■ Examples are Moodle, ILIAS, OpenCMS, BSCW, ...
Case 1: Without user management

- These applications mostly use HTTP BASIC AUTH
- Authorization can be done with Shibboleth means:
  1. Protect the whole application with Shibboleth
  2. Let Shibboleth set the REMOTE_USER header:

AAP.xml

```xml
ATTRIBUTE_RULE
  Name="urn:mace:dir:attribute-def:eduPersonUnscopedPrincipalName"
  Header="REMOTE_USER" Alias="user">
  <AnySite>
    <AnyValue />
  </AnySite>
</ATTRIBUTE_RULE>
```

Use principal, mail or another unique ID as REMOTE_USER
# Case 2: Already Shibbolethized applications

## Shibboleth® Enabled Applications and Services

Main Shibboleth Site: [shibboleth.internet2.edu](http://shibboleth.internet2.edu)

For corrections and additions to this table please contact shib-help@internet2.edu. On this page you will find:

<table>
<thead>
<tr>
<th>Information Providers:</th>
<th>Learning Management Systems:</th>
<th>Other Systems:</th>
</tr>
</thead>
<tbody>
<tr>
<td>American Chemical Society</td>
<td>Blackboard</td>
<td>Bedington.org</td>
</tr>
<tr>
<td>ArSTOR</td>
<td>Ilias</td>
<td>Condor</td>
</tr>
<tr>
<td>Atopen</td>
<td>Moodle</td>
<td>Confluence Wiki</td>
</tr>
<tr>
<td>CSA</td>
<td>OLAT</td>
<td>Darwin Streaming Server</td>
</tr>
<tr>
<td>Digitalbrain PLC</td>
<td>SakaI</td>
<td>DSpace</td>
</tr>
<tr>
<td>EBSCO Publishing</td>
<td>WebAssiun</td>
<td>eAcademy</td>
</tr>
<tr>
<td>Enlarged Science Research Institute</td>
<td></td>
<td>eNet</td>
</tr>
<tr>
<td>Exhibits</td>
<td></td>
<td>GridSphere</td>
</tr>
<tr>
<td>JSTOR</td>
<td></td>
<td>GridShib</td>
</tr>
<tr>
<td>The Literary Encyclopedia</td>
<td></td>
<td>Higher Markets</td>
</tr>
<tr>
<td>NSDL</td>
<td></td>
<td>Horde</td>
</tr>
<tr>
<td>OCLC</td>
<td></td>
<td>Hupnet</td>
</tr>
<tr>
<td>Ovid Technologies Inc.</td>
<td></td>
<td>JISCmail</td>
</tr>
<tr>
<td>Project MUSE</td>
<td></td>
<td>LionShore</td>
</tr>
<tr>
<td>Proquest Information and Learning Services</td>
<td></td>
<td>Media Wiki</td>
</tr>
<tr>
<td>Serials Solutions</td>
<td></td>
<td>MyProxy</td>
</tr>
<tr>
<td>SCIRPT</td>
<td></td>
<td>Nsper</td>
</tr>
<tr>
<td>Thomson Gale</td>
<td></td>
<td>PHEAA</td>
</tr>
<tr>
<td>Thomson IST/Scientific</td>
<td></td>
<td>Sharepoint® from Microsoft</td>
</tr>
<tr>
<td>Useful Utilities - EZproxy</td>
<td></td>
<td>SWPA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symplicity</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Turnitin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TWiki</td>
</tr>
</tbody>
</table>

[https://wiki.internet2.edu/confluence/display/seas/Home](https://wiki.internet2.edu/confluence/display/seas/Home)
Case 2: Before you start, ask yourself…

• Feasibility
  – Is the application open source?
  – Can one get the source code for development (with NDA)?
  – Is there a usable API?

• Approach
  – Authorization? Auto-Enrollment? Auto-deletion?
  – Dual login or Shibboleth only?
  – Do you have users without an AAI account? How you handle this?

• Sustainability
  – Can modifications be integrated in the official source tree?
  – What happens after version 1.0 and future versions in general?
  – Can implementation be made part of official source tree?
Case 2: The ideal solution

• Modular and least invasive design
  – Shibboleth just as another authentication method. However, many e-learning people like to have only Shibboleth authentication

• Attribute mapping and transformation included

• First time users are automatically registered

• Option to automatically update user data upon login

• Option to lock user data when it comes from Shibboleth

• WAYF service integrated in application

• Helpful and easy configuration guide

Nice to have:

• Dynamic auto-enrolment based on attributes
Case 2: Solution for closed-source with API

What to do if you cannot get your hands on source code?

http://aai-portal.sourceforge.net/
Case 2: AAIportal Functionalities

- Course management
- Subscription management
  - Waiting list
  - Automatic subscription
  - Password subscription
- User management
- Interactive mode
- Transparent mode
  - Login URL: https://aaiportal.example.ch/user/aai/login?rid=234.ADFASFASDF

Available Adapters
- Blackboard CE/Vista Courses
- Blackboard CE/Vista Institution
- WebCT Vista 3 and 4 Courses
- WebCT Vista 3 and 4 Institution
- WebCT CE 4 and 6 Courses
- WebCT CE 4 and 6 Institution
- VITELS Gridlab Courses
- AdLearn Courses

Online Demo: https://demo.aaiportal.switch.ch/
Case 2: General Integration Approach

1. Get the source code/API documentation
2. Learn how the application is used and how it works
   - Especially pay attention to the login and logout mechanisms
3. Learn how the application is programmed/structured
   - Find functions and code that is used to authenticate and set up the session
4. Implement the solution of choice
   - This is often the most straightforward part of the job
5. Thoroughly test the implementation
   - Let experienced users/administrators test the implementation for some time
6. Contribute your patches to the official source tree
   - You may have to adapt your code to meet the main developers coding style
   - Getting the code in the source tree may take quite a while
7. Maintain your change
   - Make sure you get a CVS account
Case 2: Common Problems and Solutions

- **Login name vs Username vs Screen name**
  - **Problem:** Maybe there is no username/login name attribute available in AAI but often a screen name (e.g. for forum posts) is needed
  - **Solution:** Generate screen name from attributes or ask user for one

- **Password that is not available as an attribute**
  - **Problem:** There is no attribute that contains a password
  - **Solution:** Generate a random (secure) password. Won’t be used anyway

- **Integrated HTTP-like services like WebDAV**
  - **Problem:** Is not (natively) Shibboleth compatible
  - **Solution:** Provide way for user to set password for that service (bad, but necessary)

- **Federation dependent attributes and values**
  - **Problem:** Attributes names and values differ from federation to federation
  - **Solution:** Provide hook/API to do conversion/transformation
Presentation Overview

1. Why adapting web applications to an AAI?
2. Do I have to care about authentication?
3. The three ways of authorization
4. How to adapt existing web applications
5. Session management and WAYF integration
Case 2: Where require Shibboleth session?

- **Whole application**
  - Easiest way to protect a set of documents
  - No dual login possible
  - Not user friendly because no “home page” before login
  - Problems with lost HTTP POST requests

- **Whole application with “lazy” session**
  - Well-suited for dual login
  - Lazy sessions are more complicated
  - Authorization can only be done in application
  - Problems with lost HTTP POST requests

- **Only (login) page that sets up application session**
  - Well-suited for dual login
  - Application sets up session on login page and controls it
  - No problems with HTTP POST after Shibboleth session timeout
  - Generally the best solution
Case 2: Some experiences

- Integrating an application with Shibboleth is much easier and less time consuming for Open Source programs!
  - Convince them of the added benefit and promise to support solution
  - Follow code guidelines and ask for a CVS/SVN account

- Commercial developers are more difficult to work:
  - “Shibboleth supported” on website, but is it really?
  - Only interested to integrate this feature if you pay
  - Even if you offer to develop feature for them:
    - Difficult to get the source code at all (and only with NDA)
    - Will the code be an add-on or officially integrated?
    - Who will support and maintain this code?
    - Don’t expect too much help…
Case 2: Integrated WAYF

WAYF Look&Feel can be adapted to application by providing WAYF directly on Home/Login page.

+ User feels more “at home”
  Minimizes click effort by users

- Addition coding effort
  SSO suffers a bit (no central WAYF - no redirection)

Integrating WAYF service in web application enhances ease-of use
Case 2: Examples of integrated WAYFs

Case 2: Available WAYF Implementations

• Several WAYF implementations available, e.g.
  • WAYF from Internet 2, Java
    ▪ Comes with Shibboleth IdP
    ▪ Uses same metadata format as Shibboleth
    ▪ There will be a separate package for Shibboleth 2
  
• SWITCH WAYF, PHP
  ▪ Enhanced ease-of use for the user
  ▪ Light-weight implementation of a WAYF service
  ▪ Multilingual (Currently en, fr, de, it)
  ▪ OpenSource (BSD License)

http://www.switch.ch/aai/support/tools/wayf.html
Questions?

Lukas Haemmerle
lukas.haemmerle@switch.ch