Lecture no. 3

Shibboleth SP: advanced configuration
Outline

• Virtual hosts (credits to Marco Ferrante)
• Access control
• Forced re-authentication
• Discovery Service (credits to Marco Malavolti)
Virtual hosts

• We will configure three different virtual hosts:
  • sp1.local, sp2.local e sp3.local
• We will verify that the SP won’t work for endpoints not registered in the IdP
• We will show the most common solutions
• Most frequent use case: delegate the management of the virtual host to Apache, keeping the configuration of the SP as simple as possible
Setup

```
sudo su -
cd /home/testusers/CORSO_IDEM/3_SESSIONE
./update_stato_3.0.sh
```

- **Host names**
  - `hosts`: sp1.local sp2.local sp3.local

- **Virtual hosts**
  - `sites-enabled/*.conf`:
    - `ServerName` sp1.local – `DocumentRoot` /var/www/html
    - `ServerName` sp2.local – `DocumentRoot` /var/www-sp2.local/html
    - `ServerName` sp3.local – `DocumentRoot` /var/www-sp3.local/html

- **entityId**
  - `shibboleth2.xml`: `<ApplicationDefaults entityId=https://sp1.local/shibbolethVerifica`
sp3.local

• Go to https://sp3.local
• Try to access the Intranet area
• Authenticate yourself (if session has expired)
• We get the following IdP error:
  • No peer endpoint available to which to send SAML response
• Let’s check the SAMLRequest
SAMLRequest

• Go to https://sp3.local
• Run Firebug, select Net and then Persist
• Try to access the Intranet area
• Inside Firebug click on GET SSO?SAMLRequest=...
• Open the parameters’ tab and copy the content of SAMLRequest in the clipboard
• Go to https://sp1.local/tools/decoder.php
• Paste the SAMLRequest in the form and click on Decode
SAMLRequest: elements

• AssertionConsumerServiceURL:
  • https://sp3.local/Shibboleth.sso/SAML2/POST

• Destination:
  • https://idp-corsos.iirccs.garr.it/idp/profile/SAML2/Redirect/SSO

• Issuer (match entityId):
  • https://sp1.local/shibboleth
AssertionConsumerService in Metadata

• Check what AssertionConsumerService Location(s) were present in the metadata sent to the IdP:

[...]
<md:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="https://sp1.local/Shibboleth.sso/SAML2/POST" index="1"/>
<md:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POSTSimpleSign"
Location="https://sp1.local/Shibboleth.sso/SAML2/POST-SimpleSign" index="2"/>
<md:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Artifact"
Location="https://sp1.local/Shibboleth.sso/SAML2/Artifact" index="3"/>
<md:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:2.0:bindings:PAOS"
Location="https://sp1.local/Shibboleth.sso/SAML2/ECP" index="4"/>
<md:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:1.0:profiles:browser-post"
Location="https://sp1.local/Shibboleth.sso/SAML/POST" index="5"/>
<md:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:1.0:profiles:artifact-01"
Location="https://sp1.local/Shibboleth.sso/SAML/Artifact" index="6"/>
[...]
Solutions

1. Configure all possible virtual hosts or domain names of the website and export different sets of metadata to be sent to the IdP with the correct values of AssertionConsumerService Location

   1. See https://wiki.cam.ac.uk/raven/Virtual_hosting_issues_with_Shibboleth

2. Export one single metadata file adding the correct values of AssertionConsumerService Location to enable all necessary endpoints
One Metadata, many endpoints

• Download [https://sp1.local/Shibboleth.sso/Metadata](https://sp1.local/Shibboleth.sso/Metadata)

• Add the AssertionConsumerService Location
  • [https://sp2.local/Shibboleth.sso/SAML2/POST](https://sp2.local/Shibboleth.sso/SAML2/POST)

```xml
<md:AssertionConsumerService
  Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
  Location="https://sp2.local/Shibboleth.sso/SAML2/POST"
  index="7"/>
```

• The value of index must be next to those already present

• Only holds for HTTP POST Binding!

• **Send the metadata file to the IdP managers for registration**
Access control

• Integrated from SP side:
  • “require” in apache2.conf (static) or .htaccess (dynamic)
  • XML rules linked to the contents via RequestMap in shibboleth2.xml (static) or in an ACL file (dynamic)

• Web application side
### Access control: mechanisms compared (*)

<table>
<thead>
<tr>
<th></th>
<th>apache2.conf</th>
<th>.htaccess</th>
<th>Shibboleth SP XML</th>
<th>web application</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>+</strong></td>
<td>• Easy to configure</td>
<td>• Dynamic</td>
<td>• Independent from web server</td>
<td>• Flexible and with no limits for rules</td>
</tr>
<tr>
<td></td>
<td>• Protects location</td>
<td>• Easy to configure</td>
<td>• True boolean logic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• URL Regex</td>
<td></td>
<td>• Dynamic (with external ACL files)</td>
<td></td>
</tr>
<tr>
<td><strong>−</strong></td>
<td>• Works only with Apache Static</td>
<td>• Works only with Apache</td>
<td>• XML!</td>
<td>• Implementation and operation to be done by the developer</td>
</tr>
<tr>
<td></td>
<td>• Limited rules with Apache 2.2</td>
<td>• Can be used only with files and directories</td>
<td>• Configuration errors block all the SP</td>
<td></td>
</tr>
</tbody>
</table>

Apache2.conf and .htaccess

• Special rules:
  • shibboleth (activate the module)
  • shib-session (requires an active session) [deprecated valid-user]
  • shib-user value (matches with REMOTE_USER) [deprecated user]

• Rules are evaluated in OR, unless otherwise indicated (<RequireAny>, <RequireAll>)

• Regular expressions, e.g.:

  Require shib-attr mail ~ ^.*@^(noc|adm).example.org$
Apache2.conf and .htaccess: affiliation

• Let’s create a simple rule based on affiliation(*)
• **After** <Location /intranet>, insert:

```html
<Location /intranet/affiliation_staff.html>
  AuthType shibboleth
  ShibRequestSetting requireSession true
  Require shib-attr affiliation staff@irccs.garr.it
</Location>
```

(*) Apache 2.4
Custom page for Error 401 - Unauthorised

- In `service_provider.conf` add:
  - `ErrorDocument 401 /401.html`

- Restart Apache2:
  - `service apache2 restart`
Apache2.conf and .htaccess – complex rules

In `<RequireAny>`, `<RequireAll>` and “!” we can use truly boolean logic(*):

```html
<Location /intranet/boolean.html>
AuthType shibboleth
ShibRequestSetting requireSession true
<RequireAny>
<RequireAll>
Require shib-attr affiliation staff@irccs.garr.it
Require shib-attr mail ~ .*@uni.*\.it$
</RequireAll>
<RequireAll>
Require shib-attr affiliation !staff@irccs.garr.it
Require shib-attr mail ~ \.it$
</RequireAll>
</RequireAny>
</Location>
```

(*) Apache 2.4
Check

• If examples don’t work, do:

```bash
cd /home/testuser/CORSO_IDEM/3_SESSIONE
./update_stato_3.1.sh
```
Access control in the Shibboleth SP

• Independent of the web server (IIS, FastCGI)
• XML access rules can be specified in RequestMap (shibboleth2.xml) or dynamically loaded from ACL files
• Boolean operators (AND, OR, NOT)
• Regex with <RuleRegex>
• The rules can be taken from the .htaccess file to allow non-root users to modify them
SP XML Access Control: Apache

• Enable the i canonical names; in the section `<VirtualHost..>` of `service_provider.conf`, let’s add:

  ServerName sp1.local
  UseCanonicalName On

• Delete all `<Location>` directives present and add only one `<Location />` (*) in `service-provider.conf`:

  `<Location />`
  
  AuthType shibboleth
  Require shibboleth

  `</Location>`

(*) In practice, we completely delegate access control to Shibboleth
SP XML Access Control: Shibboleth

• In shibboleth2.xml, before <ApplicationDefaults[...], add:

```xml
<RequestMapper type="Native">
    <RequestMap>
        <Host name="sp1.local">
            <Path name="intranet" authType="shibboleth« requireSession="true">
                <AccessControl>
                    <OR>
                        <Rule require="affiliation">staff@irccs.garr.it</Rule>
                        <Rule require="affiliation">student@irccs.garr.it</Rule>
                    </OR>
                </AccessControl>
            </Path>
        </Host>
    </RequestMap>
</RequestMapper>
```
Check

• If examples don’t work, do:

```bash
cd /home/testuser/CORSO_IDEM/3_SESSIONE
./update_stato_3.2.sh
```
Forced re-authentication

• Breaks the Single Sign On (SSO)
• Uses federated authentication anyway
• Can be implemented in Apache-Location (but only for whole virtual hosts)
• Can be implemented in Shibboleth per single SPs or per application through ApplicationOverride (quite complex)
forceAuthn: example from Apache side

- Modify the protected location:

```xml
<Location /intranet>
  AuthType shibboleth
  ShibRequestSetting forceAuthn true
  ShibRequestSetting requireSession true
  Require shib-session
</Location>
```
Any questions?