# Shibboleth SP: configurazione avanzata

Virtual hosts (credits Marco Ferrante)
Controllo dell'accesso
Riautenticazione forzata
Discovery Service (Marco Malavolti)

Davide Vaghetti– Università di Pisa – CTS IDEM – Roma, 30/09/2014

# Virtual hosts

- Configureremo tre distinti virtual host: sp1.local, sp2.local e sp3.local
- Verificheremo che il SP non può funzionare per endpoint non registrati presso l'IdP
- Mostreremo le soluzioni più comuni
- Use case più frequente: delegare la gestione dei virtual host ad apache e tenere la configurazione del SP più semplice

# Setup

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

- **Nomi hosts**
  *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/shibboleth

# sp3.local

- Aprite l'URL https://sp3.local
- Provate ad accedere all'area Intranet
- Autenticatevi (se la sessione non è più attiva)
- Errore dell'IdP:

```
No peer endpoint available to which to
send SAML response
```

- Verifichiamo la SAMLRequest

# SAMLRequest

- Aprite https://sp3.local
- Avviate Firebug, selezionate Net e poi Mantieni (Persist)
- Provate ad accedere all'area Intranet
- Da Firebug cliccate GET SSO?SAMLRequest=…
- Aprite la scheda dei parametri e copiate il contenuto di SAMLRequest
- Aprite https://sp1.local/tools/decoder.php
- Incollate la SAMLRequest nel form e cliccate su Decodifica

# SAMLRequest: elementi

- AssertionConsumerServiceURL:
  https://sp3.local/Shibboleth.sso/SAML2/POST
- Destination:
  https://idp-corso.irccs.garr.it/idp/profile/SAML2/Redirect/SSO
- Issuer (match entityId):
  https://sp1.local/shibboleth

# AssertionConsumerService in Metadata

Verifichiamo quali AssertionConsumerService Location erano presenti nei metadata inviati all'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-POST-
SimpleSign" 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"/>
[…]
```

# Soluzioni

1. Configuriamo tutti i possibili virtual hosts o nomi di dominio con cui vogliamo che sia raggiungibile il sito web ed esportiamo **diversi** set di metadata da inviare all'IdP con i corretti valori di AssertionConsumerService Location - Rif. https://wiki.cam.ac.uk/raven/Virtual_hosting_issues_with_Shibboleth
2. Esportiamo un **unico** file di metadata aggiungendo i corretti valori di AssertionConsumerService Location per abilitare tutti gli endpoint necessari.

# Un solo Metadata, tanti endpoint

- Scaricare https://sp1.local/Shibboleth.sso/Metadata
- Aggiungere la AssertionConsumerService Location https://sp2.local/Shibboleth.sso/SAML2/POST

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

- Il valore di index deve essere consecutivo rispetto a quelli già presenti
- Vale solo per HTTP POST Binding!
- **Inviare il file di metadata ai gestori dell'Idp per la registrazione**

# Controllo dell’accesso

- Direttamente integrato lato SP:
  - “require” in apache2.conf (statico) o .htaccess (dinamico)
  - regole XML collegate ai contenuti via RequestMap in shibboleth2.xml (statico) o in file di ACL (dinamico)
- Lato web application

# Controllo dell'accesso: meccanismi a confronto*

| | apache2.conf | .htaccess | Shibboleth SP XML | web application |
|---|---|---|---|---|
| + | • Facile da configurare<br>• Protegge location<br>• URL Regex | • Dinamico<br>• Facile da configurare | • Indipendente dal server web<br>• Vera logica booleana<br>• Dinamico (con file di ACL esterni) | • Flessibile e senza limiti di regole |
| - | • Funziona solo con Apache<br>• Statico<br>• Regole limitate con Apache 2.2 | • Funziona solo con Apache<br>• Può essere usato solo con file e directory | • XML!<br>• Errori di configurazione bloccano tutto il SP | • Implementazione e manutenzione totalmente a carico dello sviluppatore |

* Vedi https://www.switch.ch/aai/support/presentations/sp-training-2014/SP-Hands-On.pdf

# Apache2.conf e .htaccess

- Regole speciali:
  - shibboleth (attiva il modulo)
  - shib-session (richiede una sessione attiva)
    [deprecated valid-user]
  - shib-user valore (match su REMOTE_USER)
    [deprecated user]
- Le regole sono valutate in OR se non altrimenti specificato (<RequireAny>, <RequireAll>)
- Espressioni regolari, es:

```
Require shib-attr mail ~ ^.*@(noc|adm).example.org$
```

# Apache2.conf e .htaccess: affiliation

- Creiamo una semplice regola  basata su affiliation*, **dopo** la <Location /intranet>, inseriamo:

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

* Apache 2.4

# Custom page per errori 401 - Unauthorized

- in *service_provider.conf* aggiungiamo:

```
ErrorDocument 401 /401.html
```

- Riavviamo Apache2:

```
service apache2 restart
```

# Apache2.conf e .htaccess: regole complesse

Con <RequireAny>, <RequireAll> e “!” possiamo usare vera logica booleana*:

```
<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

# Verifica

- Se non riuscite a far funzionare gli esempi, eseguite:

```
cd /home/testuser/CORSO_IDEM/3_SESSIONE
./update_stato_3.1.sh
```

# Controllo dell'accesso in Shibboleth SP

- Indipendete dal server web (IIS, FastCGI)
- Le regole di accesso XML possono essere specificate in RequestMap (shibboleth2.xml) o caricate dinamicamente da file di ACL
- Operatori booleani (AND, OR, NOT)
- Regex con <RuleRegex>
- Le regole possono essere richiamate da file .htaccess per consentire la modifica a utenti non-root

# Controllo dell'accesso SP XML: Apache

- Abilitiamo i canonical names, nella sezione <VirtualHost..> di *service_provider.conf* aggiungiamo:

```
ServerName sp1.local
UseCanonicalName On
```

- Eliminiamo tutte le direttive <Location> presenti e inseriamo un'unica <Location />* in *service-provider.conf*:

```
<Location />
    AuthType shibboleth
    Require shibboleth
</Location>
```

* in pratica deleghiamo completamente il controllo dell'accesso a shibboleth

# Controllo dell'accesso SP XML: Shibboleth

In *shibboleth2.*xml, prima della sezione <ApplicationDefaults[…], aggiungiamo:

```
<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>
```

# Verifica

- Se non riuscite a far funzionare gli esempi, eseguite:

```
cd /home/testuser/CORSO_IDEM/3_SESSIONE
./update_stato_3.2.sh
```

# Riautenticazione forzata

- **Rompe il SSO**
- Sfrutta comunque l'autenticazione federata
- Implementabile in Apache-Location (ma solo per interi virtual host)
- Implementabile in Shibboleth per singoli SP o per **application** tramite ApplicationOverride (molto complesso)

# forceAuthn: esempio lato Apache

- Modificare la location protetta:

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

# Grazie