Shib2IdpRHELQuickStart

Ideális esetben az alábbi lépéseket végigjárva működő IdP-t kaphatunk RHEL, vagy ezzel rokonrendszereken.

Előkészületek

Tűzfal

- Be kell majd engedni a 443-as és a 8443-as portokat

Shibboleth IdP letöltése

```
cd /tmp
wget http://software.niif.hu/maven2/edu/internet2/middleware/shibboleth-identityprovider/2.2.1-slo10/shibboleth-identityprovider-2.2.1-slo10-bin.tar.gz
tar xzf shibboleth-identityprovider-2.2.1-slo10-bin.tar.gz
```

Telepíteni is fogjuk később, de előbb beállítjuk a környezetet. A kicsomagolt állományból is kell majd ezt-azt másolni, ezért vettük előre a folyamatot.

Tomcat

- Telepítsünk Tomcat 6-ot
  cd /etc/yum.repos.d wget 'http://www.jpackage.org/jpackage50.repo' yum update rpm -Uvh 'http://plone.lucidsolutions.co.nz/linux/centos/images/jpackage-utils-compat-el5-0.0.1-1.noarch.rpm' yum install tomcat6 tomcat6-webapps tomcat6-admin-webapps
- Be kell másolni a letöltött Shibboleth pakkban található endorsed library-ket a tomcatnek
  mkdir /usr/share/tomcat6/endorsed cp /tmp/shibboleth-identityprovider-2.2.1-slo10/endorsed/*.jar /usr/share/tomcat6/endorsed/

A `/etc/tomcat6/tomcat6.conf` állományba tegyük be az alábbi sort:
JAVA_ENDORSED_DIRS="/usr/share/tomcat6/endorsed"

- A leendő shibboleth idp webalkalmazás paramétereit is adjuk meg

cd /etc/tomcat6/Catalina/localhost vim idp.xml

A fájl tartalma pedig a következő legyen (úgy tervezzük, hogy a /usr/local/shibboleth-idp alá telepítünk mindjárt)

```
<Context
       docBase="/usr/local/shibboleth-idp/war/idp.war"
       privileged="true"
       antiResourceLocking="false"
       antiJARLocking="false"
       unpackWAR="false" />
```

Apache

A webszervernek meg kell mondani, hogy egyfelől hallgasson a 8443-as porton is, másfelől, hogy a /idp-re érkező kéréseket proxyzza tovább a tomcat felé

SSO URL (443-as port)

Be kell állítani a virtuális hosztot, amelyhez az IdP-t rendeltük. Először a 443-as portot konfiguráljuk. A 443-as porthoz tartozó tanúsítványok nem azonosak a 8443-as porthoz tartozóéval.

```
 <VirtualHost _default_:443>
   ServerName aai.example.org:443
   SSLEngine On
   SSLCertificateFile /etc/ssl/certs/aai.example.org.crt
   SSLCertificateKeyFile /etc/ssl/private/aai.example.org.key
   SSLCertificateChainFile /etc/ssl/certs/aai.example.org.crt
   ProxyRequests Off
   <Proxy ajp://localhost:8009>
     Allow from all
   </Proxy>
   ProxyPass /idp ajp://localhost:8009/idp retry=5
 </VirtualHost>
```

Ezen a porton valamilyen széles körben ismert tanúsítványt kell használni, mivel a felhasználók böngészőjének ismerniük kell(ene) a kibocsátót.

AA ill. Artifact (8443-as port)

Ezen keresztül az SP és az IdP közvetlenül kommunikálnak egymással. Ide arra a tanúsítványra van szükség, amely a föderációs metadatában szerepel - az aláírója nem érdekes.

A csatorna felépítésekor az IdP és az SP is autentikálja magát. Az SP autentikációját az Apache végzi, ami nem végez kibocsátó-ellenőrzést (`optional_no_ca`). Ez utóbbit az IdP alkalmazás végzi el, ezért nagyon fontos, hogy a kliens tanúsítványát az Apache továbbadja az alkalmazásnak (`ExportCertData`).

```
<VirtualHost _default_:8443>
  ServerName aai.example.org:8443
  SSLEngine On
  SSLCipherSuite ALL:!ADH:!EXPORT56:!EXPORT40:RC4+RSA:!SSLv2:+HIGH:+MEDIUM:+LOW:+EXP
  SSLCertificateFile /usr/local/shibboleth-idp/credentials/idp.crt
  SSLCertificateKeyFile /usr/local/shibboleth-idp/credentials/idp.key
  SSLVerifyDepth 10
  SSLVerifyClient optional_no_ca
  SSLOptions -StdEnvVars +ExportCertData
  ProxyRequests Off
  <Proxy ajp://localhost:8009>
    Allow from all
  </Proxy>
  ProxyPass /idp ajp://localhost:8009/idp retry=5
</VirtualHost>
```

A virtuális hoszt engedélyezése után be kell tölteni az `ssl` és `proxy_ajp` modulokat, majd újra kell indítani az apache-ot.

Telepítés

```
cd /tmp/shibboleth-identityprovider-2.2.1-slo10
./install.sh
```

Utómunkálatok

Jogosultságok beállítása

Engedjük meg, hogy a tomcat írja a log ill. a metadata könyvtárat

```
chown -R tomcat:tomcat /usr/local/shibboleth-idp/logs /usr/local/shibboleth-idp/metadata
```

Naplófájlok rotálása

Az alapértelmezett logging.xml nem törli a régi állományokat, ezért ezek egy idő után megtöltik a diszket.

Erre a korrekt megoldás az (lenne), ha a Logback alrendszert utasítjuk, hogy az N (a példában 90) napnál régebbi fájlokat rotálja ki. Ehhez a `logging.xml`-ben adjuk meg a `maxHistory` paramétert az *összes* `rollingPolic`y-nál, valahogy így:

```
 <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
   <FileNamePattern>/usr/local/shibboleth-idp/logs/idp-access-%d{yyyy-MM-
dd}.log</FileNamePattern>
   <maxHistory>4</maxHistory>
 </rollingPolicy>
```

Sajnos azonban jelenleg a logback csak egy állományt töröl, a régi file-okat megtartja (pl. akkor is, ha több, mint egy napig nem futott az IdP. Amíg ez nincs megoldva, addig kerülő megoldás lehet cron-ból törölni a régi file-okat

```
sudo crontab -u tomcat -e


MAILTO=mail@example.com
#m h  dom mon dow   command
52 18 *   *   *     find /var/log/shibboleth-idp/ -mtime +90 -delete
```

Ellenőrzés

Ahhoz, hogy kiderítsük, működik-e (ill. fut-e :)) az IdP webalkalmazásunk, ahhoz böngészőben hívjuk meg az alábbi urlt: https://idp.example.org/idp/profile/Status, amennyiben az oldalon egy ok-t látunk, akkor az alkalmazásunk fut, és elkezdhetjük beállítani az attribútumok feloldását és kiadását.

Konfiguráció

Ha idáig rendben vagyunk, nyergeljünk át erre a szócikkre

Változat #1
document-uploader hozta létre 2025-08-07 12:04:48 CEST
document-uploader frissítette 2025-08-07 12:04:48 CEST