Install a Shibboleth v3 IdP on Ubuntu Linux (version 14.04 LTS)

Author : Pascal Panneels, Belnet - R&E Federation

Versions :

- 1.0 (27/10/2015) : initial release, format and content inspired by :
  - Tuakiri's IdP v3 document (cfr. https://tuakiri.ac.nz/confluence/display/Tuakiri/Installing+a+Shibboleth+3.x+IdP)
  - SWITCH's Shibboleth Identity Provider (IdP) 3 Installation Guide (cfr. https://www.switch.ch/aai/guides/idp/installation/)
- 1.1 (07/12/2015) : fixes some problems with URLs given in the document (thanks to Steve Colin from HECondorcet)

Foreword

This document explains how to install an identity provider (further referred as *IdP*) based on Shibboleth major version 3. As the middleware has been completely rewritten, Shibboleth's documents referring to version 2 may have becomed obsolete.

There are 2 methods to install the version 3, either as an automatic upgrade on a version 2 instance, or as a vanilla installation. It is the last option that will be explained here.

Automatic upgrade may work but, the obtained setup will not be compatible with all the new features of Shibboleth version 3 (as the clustering of IdP for example, the users consents of attributes, etc). You can obtained it working by tweaking the generated configuration files, but it is, IMHO, rather difficult without a very good knowledge and understanding of Shibboleth 3.

So, we recommend to proceed to a fresh setup such as described in this document.

Table of content

0. Prerequisites

We suppose that the following requirements are met :

- ***Hardware*** : the machine would have a minimum of 2 GB of RAM and 20 GB of HDD space ; it may be a physical of virtual machine.
- ***Operating System*** : properly installed Ubuntu Linux server version 14.04 LTS or above ; other Linux OS would work but are not covered by this document, but you may find essential to keep you on track with the used distribution in your institution.
- ***Network*** :
  - a static IP public address (v4/v6) ;
  - an associated domain name in the format of *idp.yourdomain.be* ;
  - adapted firewall rules to permit trafic flows of TCP ports 80, 443 and 8443 ;
  - no proxy between the machine and the Net (it may result in SSL failures) ;
  - NTP synchronized ; may use *ntp.belnet.be* as time server.
- ***Public X509 server certificate*** : issued for the allocated server name ; it can be obtained via our DCS service for example ;
- ***Access to a LDAP directory*** : access to the personal accounts of your insitution to be used by the IdP ; you need to have following :
  - LDAP server IP/hostname and port number (if not default) ;
  - search base ;
  - bind DN account for generic read queries ;
  - bind password for this account
- you are working under the ***root account*** of your system ; if not, open a terminal and issue following command :

```
sudo su –
```

1. Installation of required softwares

1. Apache, Java, Tomcat : (used to make Shibboleth's base works)

```
apt-get install apache2 openjdk-7-jdk tomcat7
```

2. Install MySQL server : (ie : used to store the *users consents* when visiting sites)

```
apt-get install mysql-client mysql-server
```

(Other DB -like postgresql- may be installed but we'll only cover MySQL)

The notion of *user's consents* will be explained later.

2. Basic Shibboleth IdP installation

2.1. Rationale

To clarify, we will define some shell variables containing useful shortcuts ; to proceed, create a file named : ***/etc/profile.d/shib.sh*** containing following content :

```
IDP_VERSION="3.1.2"
SHIB_HOME=/opt/shibboleth-idp
SHIB_INST_HOME=/root/shibboleth-identity-provider-$IDP_VERSION  IDP_HOME=/opt/shibboleth-idp
JAVA_HOME=/usr

export SHIB_HOME IDP_HOME JAVA_HOME SHIB_INST_HOME IDP_VERSION
```

You should adapt the *IDP_VERSION* according to the latest release you will install of course.

Make the file executable and launch it in your terminal :

```
chmod +x /etc/profile.d/shib.sh
/etc/profiled.d/shib.sh
```

2.2. Install Shibboleth middleware itself

- Check the latest version of Shibboleth on http://www.shibboleth.net/downloads/identity-provider/

- Prepare the installation and get the latest version of the IdP :

```
cd /root
wget http://www.shibboleth.net/downloads/identity-provider/latest/shibboleth-identity-provider-
${IDP_VERSION}.tar.gz
tar xzf shibboleth-identity-provider-${IDP_VERSION}.tar.gz
cd $SHIB_INST_HOME
```

- Launch the installer :

```
sh ./bin/install.sh
```

- Your answers to the installer's questions should be inspired by the following :
  - source directory : confirm the current one (simply press *ENTER*);
  - installation directory : accept **/opt/shibboleth-idp/** ;
  - hostname : the one you've defined ealier in your DNS, such as *idp.yourdomain.be* ;
  - SAML entity ID : accept the proposed one (ie : *https://idp.yourdomain.be/idp/shibboleth*);
  - attribute scope : should be set to *yourdomain.be* ;
  - passphrase to protect the generated keystore : you may leave the default one (*changeit*) ;
    ELABORATE ON THIS PART BEFORE PUBLISHING !!
- After the run, the web application will be installed in ***/opt/shibboleth-idp/war/idp.war*** .
- **SOMETHING TO SAY OVER GENERATED CERTIFICATES HERE BEFORE PUBLISHING !!**

2.3. Configure Tomcat and deploy the IdP WAR

- Create a file in ***/etc/tomcat7/Catalina/localhost/idp.xml*** containing following :

```
<Context docBase="/opt/shibboleth-idp/war/idp.war"
   unpackWAR="false"
   swallowOutput="true">
 <Manager pathname="" />
</Context>
```

- Define the correct connectors in ***/etc/tomcat7/server.xml*** :

  Add following AJP one :

```
<Connector port="8009" address="127.0.0.1" protocol="AJP/1.3" enableLookups="false" tomcatAuthentication="false" />
```

  Comment the ones for port 8080 and 8443 by enclosing them in *<!-- ... -->*.

- Tweak Tomcat settings for memory usage to use something like 1 GB of RAM by editting ***/etc/defaults/tomcat7*** :

```
JAVA_OPTS="-server -Djava.security.egd=file:/dev/./urandom -Xms768m -Xmx1024m"
```

- Install a missing required Java library for Shibboleth v3 to properly work :

```
apt-get install libjstl1.1-java
```

2.4. Configure Apache

The configuration of Apache requires following :

- ***Listen on port 443, 8443***
  It is, in principle, already OK as long as you have enabled mod_ssl in Apache ; if not, enter following command :

```
a2enmod mod_ssl
```

- ***Setup 2 virtual hosts for your IdP*** ;
  You may use following configuration as a template and put in a file such as ***/etc/apache2/site-available/idp.conf*** :

```
<VirtualHost *:443>
 ServerName idp.YOURDOMAIN.be
 ServerAdmin admin@YOURDOMAIN.be
 CustomLog /var/log/apache2/idp.YOURDOMAIN.be.access.log combined
 ErrorLog /var/log/apache2/idp.YOURDOMAIN.be.error.log

 SSLEngine On
 SSLCipherSuite HIGH:MEDIUM:!aNULL:!kRSA:!MD5:!RC4
```

```
  SSLProtocol all -SSLv2 -SSLv3
  SSLCertificateKeyFile /etc/ssl/private/idp.YOURDOMAIN.be.key
  SSLCertificateFile /etc/ssl/certs/idp.YOURDOMAIN.be.crt
  SSLCertificateChainFile /etc/ssl/certs/DigiCertCA.crt

  ProxyPass /idp ajp://localhost:8009/idp retry=5
  <Proxy ajp://localhost:8009>
    Require all granted
  </Proxy>
</VirtualHost>
```

You may create a similar configuration for ***VirtualHost :8443***. The access to port 8443 (=used by *SOAP* is necessary if you use the *Single Logout* feature.

In the configuration, we suppose that you've obtained a certificate from our current provider (DigiCert) ; you'd modify the *SSLCertificateChainFile* parameter according to your provider if it is a different one of course.

You'd replace the *ServerName* with the one that fits yours and also the word ***YOURDOMAIN*** with yours of course.

Don't forget to enable your new website :

```
a2ensite idp
a2ensite idp8443
```

- Restart Apache to make the changes effective :

```
service apache2 restart
```

2.5. Setup the metadata and the metadata service

In order to make the federation working, we need to describe the new IdP related informations and publish it to our Federation's pals. It is done using a XML formatted file called a *metadata* file. The publication is done by Belnet after uploading the metadata file to the Federation Metadata manager website. We will glue all the received metadata files together in the global federation metadata file that will be validated and signed by Belnet in order for all participants to be able to trust the verified signature of the data.

- ***IdP metadata file***
  The file is/will be located in ***$SHIB_HOME/metadata/idp-metadata.xml***
  Here is an example :

```
<EntityDescriptor  xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:shibmd="urn:mace:shibboleth:metadata:1.0" xmlns:xml="http://www.w3.org/XML/1998/namespace"
xmlns:mdui="urn:oasis:names:tc:SAML:metadata:ui" xmlns:mdrpi="urn:oasis:names:tc:SAML:metadata:rpi"
entityID="https://idp.YOURDOMAIN.be/idp/shibboleth">

  <Extensions>
    <mdrpi:RegistrationInfo registrationAuthority="http://federation.belnet.be/" registrationInstant="2012-03-27T12:00:00Z">
      <mdrpi:RegistrationPolicy xml:lang="en">http://federation.belnet.be/files/Belnet-metadata-registration-practice-
statement.txt</mdrpi:RegistrationPolicy>
    </mdrpi:RegistrationInfo>
  </Extensions>
```

```
    <IDPSSODescriptor protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol
   urn:oasis:names:tc:SAML:1.1:protocol urn:mace:shibboleth:1.0">

  <Extensions>
    <shibmd:Scope regexp="false">YOURDOMAIN.be</shibmd:Scope>
    <mdui:UIInfo xmlns:mdui="urn:oasis:names:tc:SAML:metadata:ui">
      <mdui:DisplayName xml:lang="en">YOUR_INSTITUTION_NAME</mdui:DisplayName>
      <mdui:Description xml:lang="en">INSTITUTION IS A DOING RESEARCH IN ...</mdui:Description>
      <mdui:Logo height="16" width="16">https://www.YOURDMAIN.be/images/smallINSTITUTIONlogo.png</mdui:Logo>
      <mdui:Logo height="75" width="153">https://www.YOURDOMAIN.be/images/INSTITUTIONlogo.png</mdui:Logo>
            </mdui:UIInfo>
            <mdui:DiscoHints xmlns:mdui="urn:oasis:names:tc:SAML:metadata:ui">
                <mdui:IPHint>YOURIPV4RANGE/24</mdui:IPHint>
                <mdui:IPHint>YOURIPV6RANGE/48</mdui:IPHint>
                <mdui:DomainHint>YOURDOMAIN.be</mdui:DomainHint>
                <mdui:GeolocationHint>geo:50.825312,4.365471</mdui:GeolocationHint>
            </mdui:DiscoHints>
  </Extensions>

  <KeyDescriptor use="signing">
    <ds:KeyInfo>
      <ds:X509Data>
        <ds:X509Certificate>
...
>>PASTE HERE YOUR CERTIFICATE<<
...
        </ds:X509Certificate>
      </ds:X509Data>
    </ds:KeyInfo>
  </KeyDescriptor>


  <ArtifactResolutionService Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
Location="https://idp.YOURDOMAIN.be:8443/idp/profile/SAML1/SOAP/ArtifactResolution" index="1"/>
  <ArtifactResolutionService Binding="urn:oasis:names:tc:SAML:2.0:bindings:SOAP"
Location="https://idp.YOURDOMAIN.be:8443/idp/profile/SAML2/SOAP/ArtifactResolution" index="2"/>


  <SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"
Location="https://idp.YOURDOMAIN.be/idp/profile/SAML2/Redirect/SLO"/>
  <SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://idp.
YOURDOMAIN.be/idp/profile/SAML2/POST/SLO"/>
  <SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST-SimpleSign" Location="https://idp.
YOURDOMAIN.be/idp/profile/SAML2/POST-SimpleSign/SLO"/>
  <SingleLogoutService Binding="urn:oasis:names:tc:SAML:2.0:bindings:SOAP"
Location="https://idp.YOURDOMAIN.be:8443/idp/profile/SAML2/SOAP/SLO"/>


  <NameIDFormat>urn:mace:shibboleth:1.0:nameIdentifier</NameIDFormat>
  <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat>

  <SingleSignOnService Binding="urn:mace:shibboleth:1.0:profiles:AuthnRequest" Location="https://idp.
YOURDOMAIN.be/idp/profile/Shibboleth/SSO"/>
  <SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://idp.
YOURDOMAIN.be/idp/profile/SAML2/POST/SSO"/>
  <SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST-SimpleSign" Location="https://idp.
YOURDOMAIN.be/idp/profile/SAML2/POST-SimpleSign/SSO"/>
  <SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="https://idp.
YOURDOMAIN.be/idp/profile/SAML2/Redirect/SSO"/>

 </IDPSSODescriptor>


 <AttributeAuthorityDescriptor protocolSupportEnumeration="urn:oasis:names:tc:SAML:1.1:protocol">

  <Extensions>
    <shibmd:Scope regexp="false">YOURDOMAIN.be</shibmd:Scope>
  </Extensions>

  <KeyDescriptor use="signing">
    <ds:KeyInfo>
      <ds:X509Data>
        <ds:X509Certificate>
...
```

```
      >>PASTE HERE YOUR CERTIFICATE<<
      ...
          </ds:X509Certificate>
        </ds:X509Data>
       </ds:KeyInfo>
      </KeyDescriptor>

     <AttributeService Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding" Location="https://idp.
YOURDOMAIN.be:8443/idp/profile/SAML1/SOAP/AttributeQuery"/>
     <!-- <AttributeService Binding="urn:oasis:names:tc:SAML:2.0:bindings:SOAP" Location="https://idp.
YOURDOMAIN.be:8443/idp/profile/SAML2/SOAP/AttributeQuery"/> -->

   </AttributeAuthorityDescriptor>

   <Organization>
      <OrganizationName xml:lang="en"
xmlns:xml="http://www.w3.org/XML/1998/namespace">YOURORGANISATION</OrganizationName>
      <OrganizationDisplayName xml:lang="en" xmlns:xml="http://www.w3.org/XML/1998/namespace">YOURORGANISATION
IdP</OrganizationDisplayName>
      <OrganizationURL xml:lang="en" xmlns:xml="http://www.w3.org/XML/1998/namespace">http://www.
YOURDOMAIN.be/</OrganizationURL>
     </Organization>
   <ContactPerson contactType="technical">
      <GivenName> YOURORGANISATION Technical Staff</GivenName>
      <SurName>YOURORGANISATION</SurName>
      <EmailAddress>mailto:admin@ YOURDOMAIN.be</EmailAddress>
      <TelephoneNumber>+32-1-11111111</TelephoneNumber>
   </ContactPerson>

</EntityDescriptor>
```

You may download this file from http://federation.belnet.be/shib3/doc/metadata-example.xml

You'd replace all the parts in **RED** by appropriate values for your own institution.
The part in **GREEN** is mandatory to be published in eduGAIN's federation ; it is worth to leave it.

- ***Upload your metadata to our metadata's manager website***

  Go to following URL :

  ```
  https://federation.belnet.be/re/md-mgmt/
  ```

  and follow the « ***upload your metadata's*** » instructions.

- ***Configure the metadata service on your IdP***

  There are several ways to organize the needed information, but for sanity we've decided to setup it as following :

  ```
  wget http://federation.belnet.be/shib3/doc/metadata-example.xml
  ```

  - rename this file to ***$SHIB_HOME/conf/metadata-example.xml***
  - edit the file ***$SHIB_HOME/conf/services.xml***, locate section *shibboleth.MetadataResolverResources* and replace it by following lines :

```
<util:list id="shibboleth.MetadataResolverResources">
  <value>%{idp.home}/conf/metadata-example.xml</value>
```

```
    <value>%{idp.home}/system/conf/metadata-providers-system.xml</value>
  </util:list>
```

- the setup requires that you have Belnet Federation's certificate to validate the signature of the published metadata ; to get it, issue following command :

```
wget https://federation.belnet.be/newcertificate.federation.belnet.be.pem -O /opt/shibboleth-idp/conf/credentials/certificate.federation.belnet.be.crt
```

2.6. Configure the LDAP Authentication service

There are different ways to setup the authentication mechanism in Shibboleth. We have choosen here to explain the way LDAP is working.

LDAP will be used for both *authentication* and *attributes resolving*.

Of course, some adaptations will be done according your own settings.

To setup LDAP authentication, simply edit the ***$SHIB_HOME/conf/ldap.properties*** file and change parameters in **RED** :

```
idp.authn.LDAP.authenticator        = bindSearchAuthenticator
idp.authn.LDAP.ldapURL              = ldap://YOURLDAPSERVER:389
idp.authn.LDAP.useStartTLS          = false
idp.authn.LDAP.useSSL               = false
idp.authn.LDAP.returnAttributes     = uid,cn,mail
idp.authn.LDAP.baseDN               = dc=YOURTLD,dc=be
idp.authn.LDAP.subtreeSearch        = true
idp.authn.LDAP.userFilter           = (uid={user})
idp.authn.LDAP.bindDN               = cn=YOURCN,dc=YOURTLD,dc=be
idp.authn.LDAP.bindDNCredential     = YOURCREDENTIAL
idp.authn.LDAP.dnFormat             = uid=%s,dc=YOURTLD,dc=be
idp.authn.LDAP.sslConfig            = jvmTrust
```

2.7. Link the Attribute Resolver to the LDAP server

Edit the file ***$SHIB_HOME/conf/attribute-resolver-ldap.xml*** and setup following :

- We need to define a ***data connector to your LDAP server*** : add following in the file :

```
  <resolver:DataConnector id="myLDAP" xsi:type="dc:LDAPDirectory"
    ldapURL="%{idp.attribute.resolver.LDAP.ldapURL}"
    baseDN="%{idp.attribute.resolver.LDAP.baseDN}"
    principal="%{idp.attribute.resolver.LDAP.bindDN}"
    principalCredential="%{idp.attribute.resolver.LDAP.bindDNCredential}">
<!--      useStartTLS="%{idp.attribute.resolver.LDAP.useStartTLS:true}">-->
    <dc:FilterTemplate>
      <![CDATA[
        %{idp.attribute.resolver.LDAP.searchFilter}
      ]]>
    </dc:FilterTemplate>
<!--      <dc:StartTLSTrustCredential id="LDAPtoIdPCredential" xsi:type="sec:X509ResourceBacked">
      <sec:Certificate>%{idp.attribute.resolver.LDAP.trustCertificates}</sec:Certificate>
    </dc:StartTLSTrustCredential> -->
  </resolver:DataConnector>
```

It is now best to copy this configuration snippet to the file *$SHIB_HOME/conf/attribute-resolver.xml* instead.

2.8. Configure the Attribute Resolver – define the attributes

The configuration for the attributes will also be set in ***$SHIB_HOME/conf/attribute-resolver.xml***. You may find some examples of defined attributes in the following files :

```
$SHIB_HOME/conf/attribute-resolver-full.xml
$SHIB_HOME/conf/attribute-resolver-ldap.xml
```

Link existing LDAP attributes

Find below an example of what we use according to some of our own defined data :

```
<?xml version="1.0" encoding="UTF-8"?>
<resolver:AttributeResolver
    xmlns:resolver="urn:mace:shibboleth:2.0:resolver"
    xmlns:pc="urn:mace:shibboleth:2.0:resolver:pc"
    xmlns:ad="urn:mace:shibboleth:2.0:resolver:ad"
    xmlns:dc="urn:mace:shibboleth:2.0:resolver:dc"
    xmlns:enc="urn:mace:shibboleth:2.0:attribute:encoder"
    xmlns:sec="urn:mace:shibboleth:2.0:security"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:mace:shibboleth:2.0:resolver http://shibboleth.net/schema/idp/shibboleth-attribute-resolver.xsd
                        urn:mace:shibboleth:2.0:resolver:pc http://shibboleth.net/schema/idp/shibboleth-attribute-resolver-pc.xsd
                        urn:mace:shibboleth:2.0:resolver:ad http://shibboleth.net/schema/idp/shibboleth-attribute-resolver-ad.xsd
                        urn:mace:shibboleth:2.0:resolver:dc http://shibboleth.net/schema/idp/shibboleth-attribute-resolver-dc.xsd
                        urn:mace:shibboleth:2.0:attribute:encoder http://shibboleth.net/schema/idp/shibboleth-attribute-encoder.xsd
                        urn:mace:shibboleth:2.0:security http://shibboleth.net/schema/idp/shibboleth-security.xsd">

  <!-- Belnet core attributes -->

  <!-- Affiliation (eduPersonAffiliation) -->
  <resolver:AttributeDefinition xsi:type="ad:Mapped" id="eduPersonAffiliation" sourceAttributeID="eduPersonAffiliation">
    <resolver:Dependency ref="myLDAP" />

    <resolver:DisplayName xml:lang="en">Affiliation</resolver:DisplayName>
    <resolver:DisplayName xml:lang="fr">Affiliation</resolver:DisplayName>
    <resolver:DisplayDescription xml:lang="en">Affiliation: Type of affiliation with Home
Organization</resolver:DisplayDescription>
    <resolver:DisplayDescription xml:lang="fr">Type d'affiliation dans l'organisation</resolver:DisplayDescription>

    <resolver:AttributeEncoder xsi:type="enc:SAML1String" name="urn:mace:dir:attribute-def:eduPersonAffiliation" />
    <resolver:AttributeEncoder xsi:type="enc:SAML2String" name="urn:oid:1.3.6.1.4.1.5923.1.1.1.1"
friendlyName="eduPersonAffiliation" />
    <ad:DefaultValue passThru="true"/>
    <ad:ValueMap>
      <ad:ReturnValue>member</ad:ReturnValue>
      <ad:SourceValue>staff|student|faculty|employee</ad:SourceValue>
    </ad:ValueMap>
    <ad:ValueMap>
      <ad:ReturnValue>$1</ad:ReturnValue>
      <ad:SourceValue>(staff|student|faculty|employee)</ad:SourceValue>
    </ad:ValueMap>
  </resolver:AttributeDefinition>

  <!-- Scoped affiliation (eduPersonScopedAffiliation) -->
  <resolver:AttributeDefinition id="eduPersonScopedAffiliation" xsi:type="ad:Scoped"
    scope="%{idp.scope}" sourceAttributeID="eduPersonAffiliation">
```

```
        <resolver:Dependency ref="eduPersonAffiliation" />

    <resolver:DisplayName xml:lang="en">Affiliation</resolver:DisplayName>
    <resolver:DisplayName xml:lang="fr">Affiliation</resolver:DisplayName>
    <resolver:DisplayDescription xml:lang="en">Affiliation: Type of affiliation with Home Organization
</resolver:DisplayDescription>
    <resolver:DisplayDescription xml:lang="fr">Type d'affiliation dans l'organisation</resolver:DisplayDescription>

    <resolver:AttributeEncoder xsi:type="enc:SAML1ScopedString" name="urn:mace:dir:attribute-def:eduPersonScopedAffiliation"
/>
    <resolver:AttributeEncoder xsi:type="enc:SAML2ScopedString" name="urn:oid:1.3.6.1.4.1.5923.1.1.1.9"
friendlyName="eduPersonScopedAffiliation" />
  </resolver:AttributeDefinition>

  <!-- E-mail -->
  <resolver:AttributeDefinition id="email" xsi:type="ad:Simple" sourceAttributeID="mail">
    <resolver:Dependency ref="myLDAP" />
    <resolver:DisplayName xml:lang="en">E-mail</resolver:DisplayName>
    <resolver:DisplayName xml:lang="fr">Email</resolver:DisplayName>
    <resolver:DisplayDescription xml:lang="en">E-Mail: Preferred address for e-mail to be sent to this
person</resolver:DisplayDescription>
    <resolver:DisplayDescription xml:lang="fr">Exemple: jean.martin@example.org</resolver:DisplayDescription>
    <resolver:AttributeEncoder xsi:type="enc:SAML1String" name="urn:mace:dir:attribute-def:mail" />
    <resolver:AttributeEncoder xsi:type="enc:SAML2String" name="urn:oid:0.9.2342.19200300.100.1.3" friendlyName="mail" />
  </resolver:AttributeDefinition>

  <!-- Given name -->
  <resolver:AttributeDefinition id="givenName" xsi:type="ad:Simple" sourceAttributeID="givenName">
    <resolver:Dependency ref="myLDAP" />
    <resolver:DisplayName xml:lang="en">Given name</resolver:DisplayName>
    <resolver:DisplayName xml:lang="fr">Prénom</resolver:DisplayName>
    <resolver:DisplayDescription xml:lang="en">Given name of a person</resolver:DisplayDescription>
    <resolver:DisplayDescription xml:lang="fr">Prénom de l'utilisateur</resolver:DisplayDescription>
    <resolver:AttributeEncoder xsi:type="enc:SAML1String" name="urn:mace:dir:attribute-def:givenName" />
    <resolver:AttributeEncoder xsi:type="enc:SAML2String" name="urn:oid:2.5.4.42" friendlyName="givenName" />
  </resolver:AttributeDefinition>

  <!-- Surname -->
  <resolver:AttributeDefinition id="surname" xsi:type="ad:Simple" sourceAttributeID="sn">
    <resolver:Dependency ref="myLDAP" />
    <resolver:DisplayName xml:lang="en">Surname</resolver:DisplayName>
    <resolver:DisplayName xml:lang="fr">Nom de famille</resolver:DisplayName>
    <resolver:DisplayDescription xml:lang="en">Surname or family name</resolver:DisplayDescription>
    <resolver:DisplayDescription xml:lang="fr">Nom de famille de l'utilisateur.</resolver:DisplayDescription>
    <resolver:AttributeEncoder xsi:type="enc:SAML1String" name="urn:mace:dir:attribute-def:sn" />
    <resolver:AttributeEncoder xsi:type="enc:SAML2String" name="urn:oid:2.5.4.4" friendlyName="sn" />
  </resolver:AttributeDefinition>

  <!-- Targeted ID/Persistent ID -->
  <resolver:AttributeDefinition id="eduPersonTargetedID" xsi:type="ad:SAML2NameID" sourceAttributeID="persistentID"
    nameIdFormat="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent">
    <resolver:Dependency ref="myStoredId" />
    <!--<resolver:Dependency ref="myLDAP" />-->
    <resolver:DisplayName xml:lang="en">Targeted ID</resolver:DisplayName>
    <resolver:DisplayName xml:lang="fr">Targeted ID</resolver:DisplayName>
    <resolver:DisplayDescription xml:lang="en">Targeted ID: A unique identifier for a person, different for each service
provider.</resolver:DisplayDescription>
    <resolver:DisplayDescription xml:lang="fr">Targeted ID: Un identifiant unique de l'utilisateur, différent pour chaque
fournisseur de service.</resolver:DisplayDescription>
    <resolver:AttributeEncoder xsi:type="enc:SAML1XMLObject" name="urn:oid:1.3.6.1.4.1.5923.1.1.1.10" />
    <resolver:AttributeEncoder xsi:type="enc:SAML2XMLObject" name="urn:oid:1.3.6.1.4.1.5923.1.1.1.10"
friendlyName="eduPersonTargetedID" />
  </resolver:AttributeDefinition>

  <!-- Unique ID -->
  <resolver:AttributeDefinition id="belnetEduPersonUniqueID" xsi:type="ad:Simple"
sourceAttributeID="eduPersonPrincipalName">
    <resolver:Dependency ref="myLDAP" />
    <resolver:DisplayName xml:lang="en">Unique ID</resolver:DisplayName>
    <resolver:DisplayName xml:lang="fr">ID unique</resolver:DisplayName>
    <resolver:DisplayDescription xml:lang="en">Unique ID: A unique identifier for a person, mainly for inter-institutional user
identification.</resolver:DisplayDescription>
    <resolver:DisplayDescription xml:lang="fr">Identifiant unique de l'utilisateur au seins de l'AAI.</resolver:DisplayDescription>
```

```
        <resolver:AttributeEncoder xsi:type="enc:SAML1String" name="urn:mace:belnet.be:attribute-
    def:belnetEduPersonUniqueID"/>
    <resolver:AttributeEncoder xsi:type="enc:SAML2String" name="urn:oid:2.16.756.1.2.5.1.1.1"
friendlyName="belnetEduPersonUniqueID"/>
  </resolver:AttributeDefinition>

  <!--
    A "copy" of the Unique ID which is used for NameID generation

    Note that there is no AttributeEncoder on purpose, as otherwise
    the attribute would be released automatically alongside the
    persistent ID (NameID generation takes place after the attribute
    filtering step).
  -->
  <resolver:AttributeDefinition id="%{idp.persistentId.sourceAttribute}" xsi:type="ad:Simple"
sourceAttributeID="belnetEduPersonUniqueID">
    <resolver:Dependency ref="eduPersonPrincipalName" />
  </resolver:AttributeDefinition>

</resolver:AttributeResolver>
```

Define static attributes (optional)

These attributes are used for example to define a common attributes for all the persons in your LDAP, without having to explicitely define it in a LDAP field. If you need some, here is how to do it.

Following example is used to export the *schacHomeOrg* used by our current certificates provider :

```
  <resolver:DataConnector id="staticSchac" xsi:type="dc:Static">
    <dc:Attribute id="schacHomeOrg">
      <dc:Value>belnet.be</dc:Value>
    </dc:Attribute>
  </resolver:DataConnector>
```

And don't forget to add the definition for the *schacHomeOrg* field as well :

```
  <resolver:AttributeDefinition id="schacHomeOrg" xsi:type="ad:Simple" sourceAttributeID="schacHomeOrg">
    <resolver:Dependency ref="staticSchac" />
    <resolver:AttributeEncoder xsi:type="enc:SAML1String" name="urn:oid:1.3.6.1.4.1.25178.1.2.9" />
    <resolver:AttributeEncoder xsi:type="enc:SAML2String" name="urn:oid:1.3.6.1.4.1.25178.1.2.9"
friendlyName="schacHomeOrganization" />
  </resolver:AttributeDefinition>
```

You may, of course, define more than one static attribute (then it should be nicer to rename the id *schacHomeOrg* by something more generic such as *MyStaticAttributes*…

Define eduPersonTargetedID attribute

The attribute is used to uniquely identify a user when visiting a SP, each value is tied to the SP and thus different when a user visits another SP.

The value can be calculated on the fly as a hash (using *ComputeID* connector), or stored in a database (through *StoreID* connector).

We prefere using the StoreID connector and store the value in a database as it :

- allows keeping track of the values issued (sites visited by each user) ;
- makes possible to preserve the values when redeploying the IdP (if the entityID changes for example) ;
- allows to revoke individual values if a particular user wants to discontinue his identity at a particular site.

To proceed follow next rules :

1. Add following attribute definition into the ***$SHIB_INST/conf/attribute-resolver.xml*** :

```
  <resolver:AttributeDefinition xsi:type="ad:SAML2NameID" id="eduPersonTargetedID"
nameIdFormat="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent" sourceAttributeID="computedID">
    <resolver:Dependency ref="StoredIDConnector" />
    <resolver:DisplayName xml:lang="en">Targeted ID (opaque per-service username)</resolver:DisplayName>
    <resolver:AttributeEncoder xsi:type="enc:SAML1XMLObject" name="urn:oid:1.3.6.1.4.1.5923.1.1.1.10" />
    <resolver:AttributeEncoder xsi:type="enc:SAML2XMLObject" name="urn:oid:1.3.6.1.4.1.5923.1.1.1.10"
friendlyName="eduPersonTargetedID" />
  </resolver:AttributeDefinition>
```

2. Add following connector definition (into the same file) :

```
  <resolver:DataConnector id="StoredIDConnector" xsi:type="dc:StoredId" sourceAttributeID="uid" salt="SALT-SALT-SALT"
generatedAttributeID="computedID">
    <resolver:Dependency ref="myLDAP" />
    <dc:ApplicationManagedConnection
      jdbcDriver="com.mysql.jdbc.Driver"
      jdbcURL="jdbc:mysql://localhost/idp_db?autoReconnect=true&amp;sessionVariables=wait_timeout=31536000"
      jdbcUserName="idp_admin"
      jdbcPassword="PASSWORD"
    />
  </resolver:DataConnector>
```

Of course, you need to adapt what is mentioned in **RED** to your own configuration. (see further on how to setup MySQL).

3. Setup MySQL database and the table to store the values :

   - as MySQL has been already installed, create the user to administer it :
     - mysql –u root –p *(← enter the mysql root password when asked)*
     - create user 'idp_admin'@'localhost' identified by '***PASSWORD'*** ;
     - grant all privileges on idp_db.* to 'idp_admin'@'localhost' ;

   - create the database and the needed table :
     - mysql –u idp_admin –p*(← enter the 'PASSWORD' when asked)*
     - create database idp_db ;
     - use idp_db ;
     - Create the table shibpid with the following DDL code (coming from https://wiki.shibboleth.net/confluence/display/SHIB2/StoredIDDataConnectorDDL):

```
CREATE TABLE IF NOT EXISTS shibpid (
  localEntity TEXT NOT NULL,
  peerEntity TEXT NOT NULL,
  principalName VARCHAR(255) NOT NULL DEFAULT '',
  localId VARCHAR(255) NOT NULL,
  persistentId VARCHAR(36) NOT NULL,
  peerProvidedId VARCHAR(255) DEFAULT NULL,
  creationDate timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
 deactivationDate TIMESTAMP NULL DEFAULT NULL,
 KEY persistentId (persistentId),
```

```
  KEY persistentId_2 (persistentId, deactivationDate),
KEY localEntity (localEntity(16), peerEntity(16), localId),
KEY localEntity_2 (localEntity(16), peerEntity(16),
  localId, deactivationDate)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

- Add following bean's definition in ***$SHIB_INST/conf/global.xml*** :

```
  <bean id="shibboleth.JPAStorageService.DataSource"
      class="org.apache.tomcat.jdbc.pool.DataSource"
      destroy-method="close"
      lazy-init="true"
      p:driverClassName="com.mysql.jdbc.Driver"

p:url="jdbc:mysql://localhost:3306/idp_db?autoReconnect=true&amp;sessionVariables=wait_timeout=31536000"
      p:validationQuery="SELECT 1;"
      p:username="IDP_ADMIN"
      p:password="PASSWORD" />
```

- salt value may be generated by following command :

```
openssl rand –base64 42
```

42 is the length of the generated base64 string ; may be adjusted to your own needs.

Simply replace the value ***SALT-SALT-SALT*** by the randomly base64 generated string.

2.9. Configure Attributes Release

- You need to edit the file $SHIB_INST/conf/attribute-filter.xml.
- It is important to have a policy for each SP you will let your IdP connect to, specifying the needed attributes to be release to it.
- The attributes are given using their *friendly* ID as defined in the ***$SHIB_INST/conf/attribute-resolver.xml*** file earlier.
- Documentation of policy may be found on following URL : https://wiki.shibboleth.net/confluence/display/IDP30/AttributeFilterConfiguration
- Find here below an example related to our *FileSender* service :

```
<!-- policy requirement rule that indicates this policy is only used for requests from https://filesender.belnet.be -->

  <afp:AttributeFilterPolicy id="release_to_filesender">
    <afp:PolicyRequirementRule xsi:type="basic:AttributeRequesterString" value="https://filesender.belnet.be"/>

    <afp:AttributeRule attributeID="uid">
      <afp:PermitValueRule xsi:type="basic:ANY" />
    </afp:AttributeRule>

   <afp:AttributeRule attributeID="eduPersonEntitlement">
      <afp:PermitValueRule xsi:type="basic:ANY" />
    </afp:AttributeRule>

    <afp:AttributeRule attributeID="organizationName">
      <afp:PermitValueRule xsi:type="basic:ANY" />
    </afp:AttributeRule>

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

```
    </afp:AttributeRule>

  <afp:AttributeRule attributeID="eduPersonPrincipalName">
      <afp:PermitValueRule xsi:type="basic:ANY" />
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="commonName">
      <afp:PermitValueRule xsi:type="basic:ANY" />
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="mail">
      <afp:PermitValueRule xsi:type="basic:ANY" />
  </afp:AttributeRule>

</afp:AttributeFilterPolicy>
```

3. Register the IdP in the Belnet Federation

You need to post your metadata on our metadata registration website to let us deploy the new global metadata including yours.

3.1. Upload your metadata

- Open your webbrowser and open following website :

  https://federation.belnet.be/re/md-mgmt/

- You'll be automatically redirected to our customers IdP ; you should login in using the credentials you've received after having registered your institution for the Federation service.
- In the menu, select *« upload a metadata »* ;
- Either do a copy-paste of the content of your metadata file or click on the button « browse » to select the file you want to upload ;
- Click the button « Load, check and update ».
- If everything is validated by our online checker, you'll arrive on the next page ; otherwise, the site mentions in **red** the the error it had encountered ; fix it and reproceed with the upload ; you may adjust it directly in the text box on the site if you want ;
- On the last screen, click the checkbox *« I'm authorized to upload for my insitution »* and click the upload button.

  At that moment, the file has been recorded and the federation technical team has received an email to warn that a new metadata has been received.

  An engineer from Belnet will process the new metadata, adjust the federation where it needs to be located and recreate the federation's global metadata. This is not an automatic task, someone will process your request as soon as possible.

4. Start your IdP

- first, make sure all the files under ***$SHIB_INST*** are owned by Tomcat :

```
chown –R tomcat :tomcat $SHIB_INST
```

- Start MySQL, Tomcat and Apache :

```
service mysqld start
service tomcat7 start
service apache2 start
```

To test it, you may :

- check the status of the IdP itself by pointing your webbrowser to following URL :

```
https://idp.yourdomain/idp/status
```

If your IdP is running, you will see a status page with some informations (some infos over your system, for how long your IdP is up, etc.)

It is probably worth to disable this kind of page when going in production as it gives you a lot of information over your hardware architecture, kernel version, java version, Shibboleth version, etc. All the information may be used by malicious hackers to try to exploit or abuse your system.

What you could do to protect your server is :

- edit ***$SHIB_INST/conf/idp.properties*** and setup an access policy in following entry :

```
idp.status.accessPolicy = AccessByIPAddress
```

- edit ***$SHIB_INST/conf/access-control.xml*** and define the policy itself :

```
<util:map id="shibboleth.AccessControlPolicies">
    <entry key="AccessByIPAddress">
      <bean parent="shibboleth.IPRangeAccessControl"
        p:allowedRanges="#{ {'127.0.0.1/32', '::1/128', 'YOURNET/24'" />
    </entry>
</util:map>
```

and adapt to your own preferences.

- connect to a peculiar SP where you may authenticate using your IdP, for example our attributes reflector test site :

```
https://sptest.belnet.be
```

if you succeed connecting, your IdP is ready to go !

- Examine content of the log files of your IdP :

Look into ***$SHIB_INST/logs/idp-process.log*** and check for error if any.

Of course, you would have enabled proper logging in ***$SHIB_INST/conf/logback.xml.***

5. Advanced configuration

Shibboleth v3 comes with a lot of new features and improvements.

5.1. Databases

Among them, there is now the ability to store some informations (like users consents to release attributes for peculiar SP) in a more persistent way using databases.
By default, storage is done using webcookies on client side (valid till the end of the session and then discarded) or in server's memory (lost when service is restarted).

The following describes how to configure a database storage using JPA (Java Persistence API).

As previously mentioned, we assume that you will be using MySQL DB and Tomcat JDBC (Java Data Base Connectivity).

To setup the storage service, follow these steps (some of them were already covered earlier in this document) :

- connect as root to MySQL server :

```
mysql –u root –p
```

- Create a database including a defined admin user :

```
create database idp_db ;
create user 'idp_admin'@'localhost' identified by 'IDP_ADMIN_PASSWORD' ;
grant all privileges on idp_db.* to 'idp_admin'@'localhost' ;
```

you may change the **RED terms** to whatever fits best for your own setup.

- Create a table 'StorageRecords' :

```
CREATE TABLE `StorageRecords` (
 `context` varchar(255) NOT NULL,
 `id` varchar(255) NOT NULL,
 `expires` bigint(20) DEFAULT NULL,
 `value` longtext NOT NULL,
 `version` bigint(20) NOT NULL,
 PRIMARY KEY (`context`,`id`));
```

- Add the following beans in ***$SHIB_INST/conf/global.xml*** :

  - shibboleth.JPAStorageService
  - shibboleth.JPAStorageService.EntityManagerFactory
  - shibboleth.JPAStorageService.JPAVendorAdapter

- shibboleth.JPAStorageService.DataSource

Here are examples of all the beans definitions :

```
<bean id="shibboleth.JPAStorageService"
    class="org.opensaml.storage.impl.JPAStorageService"
    p:cleanupInterval="%{idp.storage.cleanupInterval:PT10M}"
    c:factory-ref="shibboleth.JPAStorageService.entityManagerFactory" />

<bean id="shibboleth.JPAStorageService.entityManagerFactory"
    class="org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean">
  <property name="packagesToScan" value="org.opensaml.storage.impl"/>
  <!-- <property name="dataSource" ref="shibboleth.PostgreSQLDataSource"/> -->
  <property name="dataSource" ref="shibboleth.JPAStorageService.DataSource"/>
  <property name="jpaVendorAdapter" ref="shibboleth.JPAStorageService.JPAVendorAdapter"/>
  <property name="jpaDialect">
    <bean class="org.springframework.orm.jpa.vendor.HibernateJpaDialect" />
  </property>
</bean>

<bean id="shibboleth.JPAStorageService.JPAVendorAdapter"
    class="org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter">
    <property name="database" value="MYSQL" />
</bean>

<bean id="shibboleth.JPAStorageService.DataSource"
    class="org.apache.tomcat.jdbc.pool.DataSource"
    destroy-method="close"
    lazy-init="true"
    p:driverClassName="com.mysql.jdbc.Driver"
    p:url="jdbc:mysql://localhost:3306/idp_db?autoReconnect=true&amp;sessionVariables=wait_timeout=31536000"
    p:validationQuery="SELECT 1;"
    p:username="idp_admin"
    p:password="IDP_ADMIN_PASSWORD" />
```

Don't forget to install the Java driver for MySQL (if not done already) :

```
apt-get install libmysql-java
```

- Edit the ***$SHIB_INST/conf/idp.properties*** file and adjust following parameters to use the shibboleth.JPAStorageService :

```
- idp.session.StorageService=shibboleth.JPAStorageService
- idp.consent.StorageService=shibboleth.JPAStorageService
- idp.replayCache.StorageService = shibboleth.JPAStorageService
- idp.artifact.StorageService = shibboleth.JPAStorageService
```

5.2. Tweaking automatic reload time

The default in ***$SHIB_INST/conf/services.properties*** file makes most services reload their configuration every 15 minutes. This is sufficient for a production server.

However, during setup, it can be easier to lower the time to see more quickly the changes ; for example, to reload the attribute resolver configuration more often when playing with mappings you could set :

```
idp.service.attribute.resolver.checkInterval = PT5S
```

to have a reload every 5 seconds…

Have a look in the file to tweak the settings.

5.3. Configuring Single Logout

Shibboleth IdP supports a minimalist implementation of Single Log Out (SLO) since version 2.4.0, nothing more has been added in v3.

- it is possible to terminate the session at the IdP, so no further SP sessions can be established ;
- it is possible to initiate logout at the SP level where the user has a session established. The SP can send an SLO message to the IdP and terminate the session as well.
- but the IdP will not propagate the SLO to any additional SP.
- by default, the SLO message from the SP is asynchronous and the flow ends at the IdP Logout page.
- the IdP logout page displays the list of SP the user has accessed during his IdP session and inform the user that the only secure way to close all the sessions is to close his webbrowser.
- a synchronous logout process between IdP and SP is also possible, where the IdP sends back a SLO to the SP that will confirm that both SP and IdP sessions have been terminated.

So, the best way to safely terminate a session is, for the user, to close his webbrowser…

But anyway, here are the steps needed to setup SLO on your IdP :
(1)

- Edit ***$SHIB_INST/conf/idp.properties*** and adjust/add following statements :

```
idp.session.trackSPSessions = true
idp.session.secondaryServiceIndex = true
idp.logout.elaboration = true
```

- Enable the JPA storage feature to store session information (if using memory storage, information is lost on service restart and if using the default cookie storage on client side, logout functionality doesn't work), so add following statement :

```
idp.session.StorageService = shibboleth.JPAStorageService
```

If not enabling previous JPA service, you should at least enable the *in server memory* storage to make SLO working :

```
idp.session.StorageService = shibboleth.StorageService
```

- Adjust the duration for storing the SP sessions to match the default ones of the SP :

```
idp.session.defaultSPlifetime = PT8H
idp.session.slop = P1D
```

(2)

- Customize the Logout page (see further for the customization of displayed pages).

(3)

- Register following end-points as Single Logout Service in your IdP's Metadata (don't forget to upload it to the Belnet's Federation Metadata manger website) :

urn:oasis:names:tc:SAML:2.0:bindings:SOAP	https://**idp.yourdomain**:8443/idp/profile/SAML2/SOAP/SLO
urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect	https://**idp.yourdomain**/idp/profile/SAML2/Redirect/SLO
urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST	https://**idp.yourdomain**/idp/profile/SAML2/POST/SLO

Adjust the name of your server accordingly to your own setup.

5.4. Setup the consent module

Attention has been focused on privacy and protection of the delivered attributes in Shibboleth v3.

Even if it is not something obliged, we've found that it was important to activate this functionality as more and more personal attributes are stored or exchanged on web services.

The service in Shibboleth v3 is called the ***consent module***.

When a user logs in for the first time on a SP, the IdP will ask him if he agrees to release the asked list of attributes to the SP. The user may or may not authorize it.

It will be stored in a database for a certain time.

Here are the parameters to edit in ***$SHIB_INST/conf/idp.properties*** :

- check for value changes :

```
idp.consent.compareValue = true
```

- configure server side storage in database :

```
idp.consent.StorageService = shibboleth.JPAStorageService
```

- unlimit the number of stored consent records :

```
idp.consent.maxStoredRecords = -1
```

- following may be let as is or change according to your preferences :

```
o idp.consent.storageRecordLifetime = P1Y
o idp.consent. allowDoNotRemember = true
o idp.consent.allowGlobal = true
o idp.consent.allowPerAttribute = false
```

- you may change the order in which the attribute are rendered on screen to the user ; this is done as a property in $SHIB_INST/conf/idp.properties :

```
idp.consent.attributeOrder = commonName, displayName, ...
```

You may find more informations on Shibboleth's consent module on
https://wiki.shibboleth.net/confluence/display/IDP30/ConsentConfiguration

5.5. DataSealer Key Refreshing

The IdP uses encryption based on AES algorithm to encrypt client-side storage (cookies) using a secret key.

This key needs to be periodically refreshed. The IdP will be configured to keep a number of past keys (default is 30). New information is encrypted with the newer key. Any older information encrypted with older key may still be decrypted as long as the key is still retained.

If you have configured DataBase storage, you may skip this step ; orherwise for client-side cookie storage, it is recommended to proceed to key renewal.

To do so, you may add a cron job like following :

```
14 3 * * * IDP_HOME=/opt/shibboleth-idp JAVA_HOME=/usr /opt/shibboleth-idp/bin/seckeygen.sh --versionfile /opt/shibboleth-idp/credentials/sealer.kver --storefile /opt/shibboleth-idp/credentials/sealer.jks --storepass CHANGEME  --alias secret
```

Adjust **RED word** with the password you have defined in ***$SHIB_INST/conf/credential.properties*** :

```
idp.sealer.password = CHANGEME
```

5.6. Customization and Branding

In order to customize your IdP page, you need to edit some configuration files to adapt it to the look and feel of your institution including your institution's name, your institution's logo, etc.

Previous version of Shibboleth were based on JSP files contained in a WAR file.
While it is still available, Shibboleth v3 relies on another kind of configuration file, and specifically to a template tool called *Velocity*. (see https://velocity.apache.org/engine/releases/velocity-1.5/user-guide.html for references)

One of the advantage of Velocity is that you don't have to rebuild a WAR file each time you modify something in a dynamic page.

The templates files are located in ***$SHIB_INST/views/***.
There are a lot of messages that can configured as well in ***$SHIB_INST/messages/***.

The only elements that still need to be in the WAR file are the images, static HTML, and CSS.

After you have modified images or CSS, don't forget to regenerate the WAR file by executing following command :

```
$SHIB_INST/bin/build.sh
service tomcat7 restart
```