Clustered Shibboleth IDP

18.10.23

Pegasi Knowledge
https://ghost.pegasi.fi/wiki/

Table of Contents

Clustered Shibboleth IDP

Update 22.02.2019: Tomcat 8.5 / Jetty 9.4 and Shibboleth IDP 3.4.3

Documentation for Shibboleth IDP implementation using clustered PostgreSQL BDR database. For successful SQL replication you must run the same commands on all of the nodes excluding the node*external*dsn and node*join*dsn commands which include host information.

This case has been done with Finnish HAKA federation but should be easily adjustable to any federation.

Links

Install documentation

- HAKA Confluence instructions
- HAKA Eduuni instructions
- Official Shibboleth IDP 3 installation instructions
- Switch version of Shibboleth IDP 3 installation instructions

Usable snipplets

- Configuration file summary
- Logging configuration

Software

- CentOS7
- Application server: Tomcat 8.5 (default) or Jetty 9.4
- Apache with mod_ssl
- Shibboleth identity provider 3.4.3
- eDirectory 9.0.3 LDAP -server with federation schema extensions

I originally used Tomcat 7 from from OS repository but now I have to go for Tomcat 8.5 manual installation but it is easy to do when you install Tomcat 7 from repository but leave it disabled. That we we can have the OS environment more prepared for Tomcat use.

System preparation

First complete the steps below to start installing.

System configuration

Check /etc/hosts contains necessary information such as address for localhost or hostname for HA environment. If you define IDP using ldaps connection to a host you must have a matching hostname which is good to define in hosts file.

We need JAVA_HOME so lets do it in profile file /etc/profile.d/shibboleth.sh:

```
#!/bin/sh
export JAVA_HOME=/usr/lib/jvm/jre-1.8.0-openjdk
```

File /etc/resolv.conf must have DNS data :

```
search domain.com
nameserver n.n.n.n
nameserver n.n.n.n
```

File /etc/sysconfig/iptables allows PostgreSQL BDR sync (5432), Shibboleth IDP sealer copy (SSH 22) and LDAP (389,636) from nodes and http(s) (80,443) from all addresses :

```
*filter
:INPUT DROP [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m tcp -m multiport --dports 80,443 -j ACCEPT
-A INPUT -p tcp -m tcp -m multiport --dports 22,389,636,5432 -s n.n.n.n -j
ACCEPT
-A INPUT -p tcp -m tcp -m multiport --dports 22,389,636,5432 -s n.n.n.n -j
ACCEPT
-A INPUT -j DROP
-A FORWARD -j DROP
COMMIT
```

Generate key for root user in primary node :

```
ssh-keygen -b 2048
```

If using a cluster add group and user in secondary nodes for data sealer replication :

```
groupadd -g 20000 sealer_ssh
useradd -c "Data Sealer Replication" -g 20000 -u 20000 sealer_ssh

mkdir ~sealer_ssh/.ssh
touch ~sealer_ssh/.ssh/authorized_keys
chown -R sealer_ssh.sealer_ssh ~sealer_ssh/.ssh
chmod 700 ~sealer_ssh/.ssh
chmod 400 ~sealer_ssh/.ssh/authorized_keys
```

On the secondary nodes edit ~sealer*ssh/.ssh/authorized*keys -file and copy the primary node root user identification along with restriction to prevent misuse :

```
command="scp -v -d -t /opt/shibboleth-idp/credentials/" ssh-rsa <key-data-
here>"
```

Software download

```
yum update
yum install java-1.8.0-openjdk httpd mod_ssl tomcat apr-util apr
```

Download the Shibboleth IDP package. Please see the latest version from Shibboleth IDP downloads.

```
wget
'http://shibboleth.net/downloads/identity-provider/latest/shibboleth-identit
y-provider-3.4.3.tar.gz'
```

We must download and install Tomcat manually since CentOS / RHEL 7 does not provide rpm source for Tomcat 8. But to make life easier we install Tomcat 7 rpm to get users and selinux preset contexts set up.

```
yum install tomcat
systemctl disable tomcat
```

Download latest Tomcat 8.5

```
wget
'http://www.nic.funet.fi/pub/mirrors/apache.org/tomcat/tomcat-8/v8.5.38/bin/
apache-tomcat-8.5.38.tar.gz'
```

Alternatively if you want to use Jetty download it

```
wget
'https://repo1.maven.org/maven2/org/eclipse/jetty/jetty-distribution/9.4.15.
```

```
v20190215/jetty-distribution-9.4.15.v20190215.tar.gz'
```

Tomcat 8.5 installation

Unpack and set directory permissions.

```
cd /opt
tar -xvzf /path/to/tomcat-8/v8.5.38/bin/apache-tomcat-8.5.38.tar.gz
ln -s /opt/apache-tomcat-8.5.38 /opt/tomcat
cd tomcat
chown -R tomcat.tomcat /opt/tomcat
chmod g+rwx conf
chmod g+r conf/*
chmod g+rwx bin
chmod g+r bin/*
```

Now set up a new systemd unit file for Tomcat 8.

```
vim /usr/lib/systemd/system/tomcat8.service
```

Paste following contents:

```
[Unit]
Description=Apache Tomcat Web Application Container
After=syslog.target network.target

[Service]
Type=forking
Environment=JAVA_HOME=/usr/lib/jvm/jre
Environment=CATALINA_PID=/opt/tomcat/temp/tomcat.pid
Environment=CATALINA_HOME=/opt/tomcat
Environment=CATALINA_BASE=/opt/tomcat
Environment='CATALINA_OPTS=-Xms2048M -Xmx4096M -server -XX:+UseG1GC'
Environment='JAVA_OPTS=-Djava.awt.headless=true -
Djava.security.egd=file:/dev/./urandom -Didp.home=/opt/shibboleth-idp'
ExecStart=/opt/tomcat/bin/startup.sh
ExecStop=/bin/kill -15 $MAINPID
User=tomcat
Group=tomcat

[Install]
WantedBy=multi-user.target
```

Set SELinux contexts

```
cd /opt/tomcat
chcon system_u:object_r:bin_t:s0 chcon system_u:object_r:bin_t:s0
/opt/apache-tomcat-8.5.38/bin
chcon -R system_u:object_r:etc_t:s0 conf
chcon -R system_u:object_r:tomcat_exec_t:s0 bin
chcon -R system_u:object_r:lib_t:s0 lib
chcon -R system_u:object_r:tomcat_log_t:s0 logs
chcon -R system_u:object_r:tomcat_cache_t:s0 temp work
chcon -R system_u:object_r:tomcat_var_lib_t:s0 webapps
chcon system_u:object_r:tomcat_unit_file_t:s0
/usr/lib/systemd/system/tomcat8.service
```

Set up selinux module to allow Tomcat 8.5 startup.sh execution from /opt/tomcat/bin.

```
vim tomcat-startup.te
```

Paste contents:

```
module tomcat-startup 1.0;

require {
        type tomcat_exec_t;
        type tomcat_t;
        class dir search;
}

allow tomcat_t tomcat_exec_t:dir search;
```

Compile and install module

```
checkmodule -M -m -o tomcat-startup.mod tomcat-startup.te
semodule_package -o tomcat-startup.pp -m tomcat-startup.mod
semodule -i tomcat-startup.pp
```

Finally allow tomcat to talk with Postgres

```
setsebool -P tomcat_can_network_connect_db on
```

SQL option 1: Clustered / replicated PostgreSQL

Install BDR enabled Postgres if you wish to use SQL replication for high availability. Configuration explained later.

Add postgres user and group if necessary. Dirty workaround to prevent manual work is to install postgresql-server and remove postgresql-server and postgresql-libs afterwards.

Install PostgreSQL BDR and JDBC :

```
yum erase postgresql postgresql-libs
yum install
http://packages.2ndquadrant.com/postgresql-bdr94-2ndquadrant/yum-repo-rpms/p
ostgresql-bdr94-2ndquadrant-redhat-latest.noarch.rpm
yum install postgresql-bdr94-bdr postgresql-jdbc
```

Create environment variables in file /etc/profile.d/postgresql.sh :

```
PGDATA=/var/lib/pgsql/9.4-bdr/data
export PGDATA
export PATH=/usr/pgsql-9.4/bin:$PATH
```

Activate the environment variables or just log out and in.

Init database to create data directory and config files :

```
/usr/pgsql-9.4/bin/postgresql94-setup initdb
```

Edit $PGDATA/postgresql.conf to contain :

```
listen_addresses = '0.0.0.0'
shared_buffers = 128MB
dynamic_shared_memory_type = posix
logging_collector = on
log_directory = 'pg_log'
log_filename = 'postgresql-%a.log'
log_truncate_on_rotation = on
log_rotation_age = 1d
log_rotation_size = 0
log_timezone = 'Europe/Helsinki'
autovacuum_vacuum_scale_factor = 0.002
autovacuum_analyze_scale_factor = 0.001
datestyle = 'iso, dmy'
timezone = 'Europe/Helsinki'
lc_messages = 'fi_FI.UTF-8'
lc_monetary = 'fi_FI.UTF-8'
lc_numeric = 'fi_FI.UTF-8'
lc_time = 'fi_FI.UTF-8'
default_text_search_config = 'pg_catalog.finnish'

shared_preload_libraries = 'bdr'
wal_level = logical
track_commit_timestamp = on
max_connections = 100
max_wal_senders = 10
```

```
max_replication_slots = 10
max_worker_processes = 10
# port = 5598 # change to whatever port you need
# synchronous_commit = on # uncomment this line if you want synchronous
commit
# bdr.synchronous_commit = on # uncomment this line if you want synchronous
commit
```

Edit $PGDATA/pg_hba.conf to contain following (use all nodes IP addresses):

```
#access
local   all             all                                     peer
host    all             all             127.0.0.1/32            md5
host    all             all             ::1/128                 md5
#replication
local   replication     postgres                                trust
host    replication     postgres     127.0.0.1/32               trust
host    replication     postgres     ::1/128                    trust
host    replication     postgres     1.2.5.7/32         trust
host    replication     postgres     1.2.5.6/32         trust
host    all             postgres     1.2.5.6/32         trust
host    all             postgres     1.2.5.7/32         trust
```

Enable and start service :

```
systemctl enable postgresql-9.4
systemctl start postgresql-9.4
```

Check that you have a working connection between database nodes :

```
nc -v n.n.n.n 5432
```

SQL option 2: Distribution PostgreSQL

Do this if you don't wish to use SQL replication for high availability.

```
yum install postgresql-server postgresql-jdbc
postgresql-setup initdb
```

Set /var/lib/pgsql/data/pg_hba.conf to contain following:

```
local   all             all                                     peer
host    all             all             127.0.0.1/32            md5
host    all             all             ::1/128                 md5
```

Update PostgreSQL autovacuum and analyzer settings to ensure correct updates by editing file /var/lib/pgsql/data/postgresql.conf and setting

```
autovacuum_vacuum_scale_factor = 0.002
autovacuum_analyze_scale_factor = 0.001
```

```
systemctl enable postgresql.service
systemctl start postgresql.service
```

Apache configuration

Set up server key and certificate under /etc/pki/tls/ and set SELinux contexts and permissions

```
cp <cert> /etc/pki/tls/certs/
cp <intermediate ca> /etc/pki/tls/certs/
cp key /etc/pki/tls/private/
restorecon -v /etc/pki/tls/certs/*
restorecon -v /etc/pki/tls/private/*
chown root.root /etc/pki/tls/certs/*
chown root.root /etc/pki/tls/private/*
```

Make a virtualhost configuration to /etc/httpd/conf.d/

```
ServerTokens Prod
ServerSignature off

<VirtualHost *:443>
  ServerName idp.domain.com
  ServerAdmin admin@
  CustomLog /var/log/httpd/idp.domain.com.access.log combined
  ErrorLog /var/log/httpd/idp.domain.com.error.log

  SSLEngine On
  SSLCipherSuite HIGH:MEDIUM:!aNULL:!kRSA:!MD5:!RC4
  SSLProtocol all -SSLv2 -SSLv3
  SSLCertificateKeyFile //etc/pki/tls/private/idp-key.pem
  SSLCertificateFile /etc/pki/tls/certs/idp-cert.crt
  SSLCertificateChainFile /etc/pki/tls/certs/chain.crt

  <IfModule headers_module>
     Header set X-Frame-Options DENY
     Header set Strict-Transport-Security "max-age=31536000 ;
includeSubDomains"
  </IfModule>
```

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

Set SELinux context.

```
chcon --reference /etc/httpd/conf.d /etc/httpd/conf.d/shibboleth-idp.conf
```

If ServerTokens and/or ServerSignature is defined in other conf files or httpd.conf we need to edit it to the above values.

Test and enable Apache

```
apachectl configtest
systemctl enable httpd
systemctl start httpd
```

Tomcat configuration

Updated to Tomcat 8.5

We use Tomcat since the environment (SELinux) is already baked for it. Untested Jetty instructions below for your information.

Open /opt/tomcat/conf/server.xml and do the following.

Comment out <Connector port="8080"

> element

```
<!--
    <Connector port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />
-->
```

Add Ajp connector by removing other definitions to port 8009 and adding

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

Disable auto deploy by modifying audoDeploy to false

```
autoDeploy="false"
```

Save and close /opt/tomcat/conf/server.xml and create deployment configuration for idp war.

```
mkdir -p /opt/tomcat/conf/Catalina/localhost
vim /opt/tomcat/conf/Catalina/localhost/idp.xml
```

Paste following into the /opt/tomcat/conf/Catalina/localhost/idp.xml:

```
<Context docBase="/opt/shibboleth-idp/war/idp.war"
         privileged="true"
         antiResourceLocking="false"
         swallowOutput="true">
</Context>
```

Alternative: Jetty configuration

```
tar -xvzf jetty-distribution-9.3.20.v20170531.tar.gz -C /opt/
ln -s /opt/jetty-distribution-9.3.20.v20170531 /opt/jetty
useradd -m jetty
chown -R jetty:jetty /opt/jetty/
ln -s /opt/jetty/bin/jetty.sh /etc/init.d/jetty
ln -s /opt/jetty/logs /var/log/jetty
chkconfig --add jetty
chkconfig --level 345 jetty on
```

Make default definitions to following file

```
vim /etc/default/jetty
```

By adding these

```
JETTY_HOME=/opt/jetty
JETTY_USER=jetty
JETTY_PORT=8080
JETTY_HOST=1.2.5.6
JETTY_LOGS=/var/log/jetty/
```

Database initialization

We do slightly different configurations on multiple node clustered (replicated) environment and single node environment. Database contents are identical but replication needs extra extensions before

content is introduced to the database.

Prepare database for persistent id and attribute release entries.

```
su - postgres
psql <<EOF
CREATE ROLE shibboleth WITH LOGIN;
CREATE DATABASE shibboleth WITH OWNER shibboleth ENCODING 'UTF8' TEMPLATE
template0;
EOF
```

If you are using distribution PostgreSQL without BDR extensions you can skip straight to shibboleth IDP table initialization. If you are using clustering you must add and enable BDR extensions as stated below. You must do this to a freshly created shibboleth database without contents :

```
su - postgres
psql shibboleth <<EOF
CREATE EXTENSION btree_gist;
CREATE EXTENSION bdr;
EOF
```

After you created BDR extensions to all nodes run following command on node1 :

```
psql shibboleth
SELECT bdr.bdr_group_create(
 local_node_name := 'node1',
 node_external_dsn := 'host=n.n.n.n dbname=shibboleth'
 );
SELECT bdr.bdr_node_join_wait_for_ready();
```

After after that run following command on node2 (and similarly for additional nodes) :

```
SELECT bdr.bdr_group_join(
 local_node_name := 'node2',
 node_external_dsn := 'host=n.n.n.n dbname=shibboleth',
 join_using_dsn := 'host=n.n.n.n dbname=shibboleth'
 );
SELECT bdr.bdr_node_join_wait_for_ready();
```

Shibboleth IDP table initialization. **Continue from here if NOT using BDR.**

```
su - postgres
psql <<EOF
\c shibboleth
SET ROLE shibboleth;
CREATE TABLE shibpid (
```

```
    localEntity VARCHAR(1024) NOT NULL,
    peerEntity VARCHAR(1024) NOT NULL,
    principalName VARCHAR(255) NOT NULL,
    localId VARCHAR(255) NOT NULL,
    persistentId VARCHAR(36) NOT NULL,
    peerProvidedId VARCHAR(255) NULL,
    creationDate TIMESTAMP NOT NULL DEFAULT LOCALTIMESTAMP,
    deactivationDate TIMESTAMP NULL DEFAULT NULL,
    PRIMARY KEY (localEntity, peerEntity, persistentId)
);
CREATE INDEX shibpid_getbysourcevalue_index ON shibpid(localEntity,
peerEntity, localId, deactivationDate);
CREATE TABLE storagerecords (
    context VARCHAR(255) NOT NULL,
    id VARCHAR(255) NOT NULL,
    expires BIGINT DEFAULT NULL,
    value TEXT NOT NULL,
    version BIGINT NOT NULL,
    PRIMARY KEY (context, id)
);
CREATE INDEX storagerecords_expires_index ON storagerecords(expires);
EOF
```

Set shibboleth user password on all nodes :

```
sudo -u postgres psql
ALTER USER "shibboleth" WITH PASSWORD 'some_complex_password';
GRANT ALL PRIVILEGES ON DATABASE shibboleth TO shibboleth;
```

Not sure about this one but in case there are problems with SQL connections copy the Apache Commons database class to edit-webapp directory

```
cp /path/to/shibboleth-identity-provider-3.4.3/webapp/WEB-INF/lib/commons-
dbcp2-2.1.1.jar /opt/shibboleth-idp/edit-webapp/WEB-INF/lib/
```

And remember to rebuild using the build.sh command below.

Shibboleth IDP

Software installation

Unpack identity provider tar package.

```
tar -zxf wget shibboleth-identity-provider-3.4.3.tar.gz
```

Use Switch installation script for base install

```
wget https://www.switch.ch/aai/guides/idp/installation/idp-install.sh
sh idp-install.sh shibboleth-identity-provider-3.4.3
```

Database settings

Add SQL login information to /opt/shibboleth-idp/conf/idp.properties:

```
psql.username= shibboleth
```

Add SQL password information to /opt/shibboleth-idp/conf/credentials.properties:

```
psql.password= some_complex_password
```

At it's current state there is a problem with PostgreSQL tables declared as large objects and the way IDP/Postgres handle them and it needs to be dealt with. Switch has created an XML override that will prevent this from happening.

Issue following commands :

```
mkdir /opt/shibboleth-idp/edit-webapp/WEB-INF/classes/META-INF
cd /opt/shibboleth-idp/edit-webapp/WEB-INF/classes/META-INF/
vim orm.xml
```

Then paste following contents to the file orm.xml :

```
<?xml version="1.0" encoding="UTF-8"?>
<entity-mappings xmlns="http://java.sun.com/xml/ns/persistence/orm"
                 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence/orm
http://java.sun.com/xml/ns/persistence/orm_1_0.xsd" version="1.0">
  <package>org.opensaml.storage.impl</package>

  <entity class="JPAStorageRecord" access="PROPERTY">
    <attributes>
      <basic name="value">
        <column name="value" nullable="false"/>
      </basic>
    </attributes>
  </entity>
</entity-mappings>
```

Rebuild IDP :

```
JAVACMD=/usr/bin/java /opt/shibboleth-idp/bin/build.sh -
Didp.target.dir=/opt/shibboleth-idp
```

Restart PostgeSQL (**use postgresql instead of postgresql-9.4 with distribution Postgres**) and tomcat.

```
systemctl restart postgresql-9.4
systemctl restart tomcat
```

Reading metadata

Open /opt/shibboleth-idp/conf/metadata-providers.xml and start with metadata chain to enable more than single metadata source (without idp.properties multiple metadata definitions).

```
<MetadataProvider id="ShibbolethMetadata"
xsi:type="ChainingMetadataProvider"
        xmlns="urn:mace:shibboleth:2.0:metadata"
        xmlns:resource="urn:mace:shibboleth:2.0:resource"
        xmlns:security="urn:mace:shibboleth:2.0:security"
        xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="urn:mace:shibboleth:2.0:metadata
http://shibboleth.net/schema/idp/shibboleth-metadata.xsd
                urn:mace:shibboleth:2.0:resource
http://shibboleth.net/schema/idp/shibboleth-resource.xsd
                urn:mace:shibboleth:2.0:security
http://shibboleth.net/schema/idp/shibboleth-security.xsd
                urn:oasis:names:tc:SAML:2.0:metadata
http://docs.oasis-open.org/security/saml/v2.0/saml-schema-metadata-2.0.xsd">
```

Local metadata such as test sp's and other non-federated stuff is simplest to set with file provider as follows :

```
<MetadataProvider id="LocalMetadata"  xsi:type="FilesystemMetadataProvider"
metadataFile="%{idp.home}/metadata/local-metadata.xml"/>
```

Federation test metadata needs to be set without validity interval requirements. See HAKA test information.

Get test metadata certificate from HAKA and save to /opt/shibboleth-idp/credentials.

```
wget
'https://wiki.eduuni.fi/download/attachments/27297785/haka_testi_2015_sha2.c
rt?version=1&modificationDate=1430212953940&api=v2' -O /opt/shibboleth-
idp/credentials/haka_testi_2015_sha2.crt
```

Set test metadata configuration to /opt/shibboleth-idp/conf/metadata-providers.xml:

```
        <MetadataProvider id="FederationTestMetadata"
                xsi:type="FileBackedHTTPMetadataProvider"
                xmlns="urn:mace:shibboleth:2.0:metadata"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:schemaLocation="urn:mace:shibboleth:2.0:metadata
http://shibboleth.net/schema/idp/shibboleth-metadata.xsd"
metadataURL="https://haka.funet.fi/metadata/haka_test_metadata_signed.xml"
                maxRefreshDelay="PT1H"
                backingFile="%{idp.home}/metadata/federation-test-
metadata.xml">
                <MetadataFilter xsi:type="ChainingFilter">
                        <MetadataFilter xsi:type="SignatureValidation"
requireSignedRoot="true"
certificateFile="%{idp.home}/credentials/haka_testi_2015_sha2.crt"/>
                </MetadataFilter>
        </MetadataProvider>
```

Federation production metadata definition is a copy of the test definition with added validity interval requirement and different certificate. Look HAKA production metadata information and get certificate from HAKA.

```
wget
'https://wiki.eduuni.fi/download/attachments/27297775/haka-sign-v3.pem?versi
on=1&modificationDate=1428925492807&api=v2' -O /opt/shibboleth-
idp/credentials/haka-sign-v3.pem
```

Add production metadata configuration to /opt/shibboleth-idp/conf/metadata-providers.xml :

```
        <MetadataProvider id="FederationTestMetadata"
                xsi:type="FileBackedHTTPMetadataProvider"
                xmlns="urn:mace:shibboleth:2.0:metadata"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:schemaLocation="urn:mace:shibboleth:2.0:metadata
http://shibboleth.net/schema/idp/shibboleth-metadata.xsd"
metadataURL="https://haka.funet.fi/metadata/haka-metadata.xml"
                maxRefreshDelay="PT1H"
                backingFile="%{idp.home}/metadata/federation-test-
metadata.xml">
                <MetadataFilter xsi:type="ChainingFilter">
                        <MetadataFilter xsi:type="RequiredValidUntil"
maxValidityInterval="P7D"/>
                        <MetadataFilter xsi:type="SignatureValidation"
requireSignedRoot="true"
certificateFile="%{idp.home}/credentials/haka-sign-v3.pem"/>
                </MetadataFilter>
```

```
        </MetadataProvider>
```

Publishing metadata

Usually there is no need to publish IDP metadata but it is possible to make metadata visible in IDP address https://n.n.n.n/idp/shibboleth by setting up the the following value to /opt/shibboleth-idp/conf/idp.properties.

```
idp.entityID.metadataFile= %{idp.home}/metadata/idp-metadata.xml
```

The file %{idp.home}/metadata/idp-metadata.xml is generated in the installation and should be updated according to the current (federation) configuration.

Authentication

For normal authentication use Password authentication bean. Check that Password authentication bean is enabled in /opt/shibboleth-idp/conf/idp.properties:

```
idp.authn.flows= Password
```

Check that Password authentication bean is defined and enabled in /opt/shibboleth-idp/conf/authn/general-authn.xml:

```
        <bean id="authn/Password" parent="shibboleth.AuthenticationFlow"
                p:passiveAuthenticationSupported="true"
                p:forcedAuthenticationSupported="true" />
```

Other authentication beans can be enabled in /opt/shibboleth-idp/conf/authn/general-authn.xml as well. You can enable or disable them with idp.properties idp.authn.flows value.

Save LDAP cert chain to a file and edit out other certs but the issuing CA with commands:

```
openssl s_client -connect localhost:636 -showcerts > /opt/shibboleth-
idp/credentials/ldapca.crt
vim /opt/shibboleth-idp/credentials/ldapca.crt
```

Configure LDAP authentication to file /opt/shibboleth-idp/conf/ldap.properties.

For authentication with binded subtree user search edit the following:

```
idp.authn.LDAP.authenticator                    = bindSearchAuthenticator
idp.authn.LDAP.ldapURL                          = ldaps://your-host
idp.authn.LDAP.useStartTLS                      = false
idp.authn.LDAP.useSSL                           = true
```

```
idp.authn.LDAP.subtreeSearch                    = true
idp.authn.LDAP.baseDN                           = o=org
idp.authn.LDAP.userFilter                       = (cn={user})
idp.authn.LDAP.bindDN                           =
cn=myidpuser,ou=someorgunit,o=someorg
```

In order to use the CA file from above add:

```
idp.authn.LDAP.sslConfig                        = certificateTrust
idp.authn.LDAP.trustCertificates                =
%{idp.home}/credentials/ldap-ca.crt
```

Add LDAP password to /opt/shibboleth-idp/conf/credentials.properties -file

```
idp.authn.LDAP.bindDNCredential                 = your_password
```

Attribute resolving

As the documents say open attribute-resolver.xml and start with organization custom attribute definitions followed by data connectors. Utilize configuration parameters from properties files. Switch prefers to split the resolvers into multiple files and define them in idp.properties but we do it the original way to keep it more in line with original configuration.

Namespaces for attribute-resolver.xml :

```
<resolver:AttributeResolver
                xmlns:resolver="urn:mace:shibboleth:2.0:resolver"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                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"
                xsi:schemaLocation="urn:mace:shibboleth:2.0:resolver
classpath:/schema/shibboleth-2.0-attribute-resolver.xsd
                        urn:mace:shibboleth:2.0:resolver:pc
classpath:/schema/shibboleth-2.0-attribute-resolver-pc.xsd
                        urn:mace:shibboleth:2.0:resolver:ad
classpath:/schema/shibboleth-2.0-attribute-resolver-ad.xsd
                        urn:mace:shibboleth:2.0:resolver:dc
classpath:/schema/shibboleth-2.0-attribute-resolver-dc.xsd
                        urn:mace:shibboleth:2.0:attribute:encoder
classpath:/schema/shibboleth-2.0-attribute-encoder.xsd
                        urn:mace:shibboleth:2.0:security
classpath:/schema/shibboleth-2.0-security.xsd">
```

Example attribute definition with dependency on myLDAP ldap connector defined below :

```
    <resolver:AttributeDefinition id="id-urn:mace:dir:attribute-def:uid"
xsi:type="Simple" xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        sourceAttributeID="uid">
        <resolver:Dependency ref="myLDAP" />
        <resolver:DisplayName
xml:lang="fi">Käyttäjätunnus</resolver:DisplayName>
        <resolver:DisplayName xml:lang="en">Username</resolver:DisplayName>
        <resolver:DisplayName xml:lang="se"></resolver:DisplayName>
        <resolver:DisplayDescription xml:lang="fi">Käyttäjätunnus
kotiorganisaatiossa</resolver:DisplayDescription>
        <resolver:DisplayDescription xml:lang="en">Username in the home
organisation</resolver:DisplayDescription>
        <resolver:DisplayDescription
xml:lang="se"></resolver:DisplayDescription>
        <resolver:AttributeEncoder xsi:type="SAML1String"
xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:mace:dir:attribute-def:uid" />
        <resolver:AttributeEncoder xsi:type="SAML2String"
xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:oid:0.9.2342.19200300.100.1.1" friendlyName="uid" />
    </resolver:AttributeDefinition>
```

Example static data connector:

```
    <resolver:DataConnector id="staticAttributes" xsi:type="Static"
xmlns="urn:mace:shibboleth:2.0:resolver:dc">
        <Attribute id="funetEduPersonHomeOrganization">
            <Value>domain.com</Value>
        </Attribute>
        <Attribute id="logout-url">
            <Value>https://idp.domain.com/cas/logout</Value>
        </Attribute>
        <Attribute id="schacHomeOrganizationType">
            <Value>urn:schac:homeOrganizationType:fi:polytechnic</Value>
        </Attribute>
    </resolver:DataConnector>
```

Example LDAP connector:

```
    <resolver:DataConnector id="myLDAP" xsi:type="LDAPDirectory"
xmlns="urn:mace:shibboleth:2.0:resolver:dc"
        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}"
        connectTimeout="%{idp.attribute.resolver.LDAP.connectTimeout}"
        responseTimeout="%{idp.attribute.resolver.LDAP.responseTimeout}">
        <FilterTemplate>
            <![CDATA[
                %{idp.attribute.resolver.LDAP.searchFilter}
            ]]>
        </FilterTemplate>
        <StartTLSTrustCredential id="LDAPtoIdPCredential"
xsi:type="sec:X509ResourceBacked">
<sec:Certificate>%{idp.attribute.resolver.LDAP.trustCertificates}</sec:Certi
ficate>
        </StartTLSTrustCredential>
    </resolver:DataConnector>
```

Followed by legacy nameid data connectors if needed, as explained in the topic below.

PersistentId

Use Switch global.xml to define PostgreSQL connector for persistentID.

```
wget 'https://www.switch.ch/aai/guides/idp/installation/global.xml' -O
/opt/shibboleth-idp/conf/global.xml
```

Set persistent id generator, connector, source attribute and salting algorithm to file /opt/shibboleth-idp/conf/saml-nameid.properties:

```
idp.persistentId.store = myPersistentIdStore
idp.persistentId.generator = shibboleth.StoredPersistentIdGenerator
idp.persistentId.dataSource = shibboleth.PostgreSQLDataSource
idp.persistentId.sourceAttribute = cn
idp.persistentId.algorithm = SHA
```

Set salt secret to /opt/shibboleth-idp/conf/credentials.properties:

```
idp.persistentId.salt = my_secret_salt
```

Add bean for persistent ID storage to file /opt/shibboleth-idp/conf/saml-nameid.xml:

```
<bean id="myPersistentIdStore"
class="net.shibboleth.idp.saml.nameid.impl.JDBCPersistentIdStore">
  <property name="dataSource" ref="shibboleth.JPAStorageService.DataSource"
/>
</bean>
```

Enable (uncomment) persistent id generator in file /opt/shibboleth-idp/conf/saml-nameid.xml:

```
<ref bean="shibboleth.SAML2PersistentGenerator" />
```

Enable (uncomment) persistent id handling in /opt/shibboleth-idp/conf/c14n/subject-c14n.xml:

```
<ref bean="c14n/SAML2Persistent" />
```

Legacy nameid support

As long as eduPersonTargetedId is around and there are old SPs in the federation that need legacy StoredId and ComputedId (instead of new persistent and transient ids) do the following steps to enable legacy support.

Uncomment the following from /opt/shibboleth-idp/conf/saml-nameid.properties :

```
idp.nameid.saml2.legacyGenerator = shibboleth.LegacySAML2NameIDGenerator
idp.nameid.saml1.legacyGenerator =
shibboleth.LegacySAML1NameIdentifierGenerator
```

Set legacy datasources in attribute-resolver.xml :

```
<resolver:DataConnector xsi:type="ComputedId"
xmlns="urn:mace:shibboleth:2.0:resolver:dc"
        id="computedID"
        generatedAttributeID="computedID"
        sourceAttributeID="%{idp.persistentId.sourceAttribute}"
        salt="%{idp.persistentId.salt}">
        <resolver:Dependency ref="myLDAP" />
</resolver:DataConnector>

<resolver:DataConnector xsi:type="StoredId"
xmlns="urn:mace:shibboleth:2.0:resolver:dc"
        id="storedID"
        generatedAttributeID="persistentID"
        sourceAttributeID="%{idp.persistentId.sourceAttribute}"
        salt="%{idp.persistentId.salt}">
        <resolver:Dependency ref="myLDAP" />
        <BeanManagedConnection
xmlns="urn:mace:shibboleth:2.0:resolver:dc">shibboleth.PostgreSQLDataSource<
/dc:BeanManagedConnection>
</resolver:DataConnector>
```

There is no need to add PostgreSQL data source to saml-nameid.xml since it is already in global.xml with corresponding workarounds courtesy of Switch.

Concent

Set consent to ask again if information is changed, save unlimited amount of answers and store the data to our clustered database.

Set following to /opt/shibboleth-idp/conf/idp.properties :

```
idp.consent.compareValues= true
idp.consent.maxStoredRecords= -1
idp.consent.StorageService=shibboleth.JPAStorageService
```

Attribute filter

Use file backed http resource by adding following to /opt/shibboleth-idp/conf/services.xml :

```
bean id="FileBackedFederationAttributeFilter"
      class="net.shibboleth.ext.spring.resource.FileBackedHTTPResource"
      c:client-ref="shibboleth.FileCachingHttpClient"
      c:url="https://haka.funet.fi/metadata/haka-attribute-filter.xml"
      c:backingFile="%{idp.home}/conf/haka-attribute-filter.xml"/>
```

When using test federation we need to define test service providers' release policies in the local /opt/shibboleth-idp/conf/attribute-filter.xml .

Attribute filtering from metadata

If you do not receive a federated attribute filter you can do it via AttributeInMetadata configuration. This is the current practice in the finnish HAKA federation.

Open attribute-filter.xml and use following as an example.

Below we define all attributes as available if they are requested in the metadata. This applies to metadata EntitiesDescriptors with names of “urn:mace:funet.fi:haka” and “CSC testipalvelut”. Please note that there are no isRequired attributes defined in HAKA metadata at the time of writing this document. Also be aware that the namespaces may not be correct below.

```
       <AttributeFilterPolicy id="attributes-when-requested">
                <PolicyRequirementRule xsi:type="OR">
                        <Rule xsi:type="InEntityGroup"
groupID="urn:mace:funet.fi:haka" />
                        <Rule xsi:type="InEntityGroup" groupID="CSC
testipalvelut" />
                </PolicyRequirementRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
```

```
def:eduPersonPrincipalName">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:eduPersonTargetedID">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:uid">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:mail">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:cn">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:sn">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:displayName">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:l">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:street">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:title">
                        <PermitValueRule xsi:type="AttributeInMetadata"
```

```
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:givenName">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:Language">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:mobile">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:o">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:ou">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:postalAddress">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:postalCode">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:postOfficeBox">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:telephoneNumber">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
```

```
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:employeeNumber">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:preferredLanguage">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:eduPersonAffiliation">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:eduPersonPrimaryAffiliation">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:eduPersonScopedAffiliation">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:eduPersonEntitlement">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:eduPersonNickname">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:eduPersonOrgDN">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:eduPersonOrgUnitDN">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:eduPersonPrimaryOrgUnitDN">
```

```
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:schacMotherTongue">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:schacGender">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:schacDateOfBirth">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:schacPlaceOfBirth">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:schacCountryOfCitizenship">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:schac:attribute-
def:schacHomeOrganization">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:schac:attribute-
def:schacHomeOrganizationType">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:schacCountryOfResidence">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:schacUserPresenceID">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
```

```
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:schacPersonalUniqueCode">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:schacPersonalUniqueID">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:schacUserStatus">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonTargetDegree">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonProgram">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonSpecialisation">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonStudyStart">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonPrimaryStudyStart">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonStudyToEnd">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
```

```
def:funetEduPersonPrimaryStudyToEnd">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonCreditUnits">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonECTS">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonStudentCategory">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonStudentStatus">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonStudentUnion">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonHomeCity">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonEPPNTimeStamp">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:persistentId">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-urn:mace:dir:attribute-
def:funetEduPersonHomeOrganization">
                        <PermitValueRule xsi:type="AttributeInMetadata"
```

```
onlyIfRequired="false"/>
                </AttributeRule>
                <AttributeRule attributeID="id-
urn:mace:funet.fi:haka:logout-url">
                        <PermitValueRule xsi:type="AttributeInMetadata"
onlyIfRequired="false"/>
                </AttributeRule>
        </AttributeFilterPolicy>
```

Session storage

IDP uses DataSealer which uses an AES secret key to secure cookies and other data. This key must be kept secret AND must be shared between nodes of the cluster.

To protect resources a daily update and replication of the key (between nodes) is recommended.

Set session storage to database by setting following in /opt/shibboleth-idp/conf/idp.properties :

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

If running a cluster set key group ownership to sealer_ssh user and allow group writes :

```
chgrp sealer_ssh /opt/shibboleth-idp/credentials/sealer.*
chmod g+w /opt/shibboleth-idp/credentials/sealer.*
```

In the primary node create a periodically activated script /opt/shibboleth-idp/credentials/rotate-sealer.sh with contents :

```
#!/bin/sh

IDP_HOME=/opt/shibboleth-idp

java -cp "$IDP_HOME/webapp/WEB-INF/lib/*" \
net.shibboleth.utilities.java.support.security.BasicKeystoreKeyStrategyTool
\
  --storefile $IDP_HOME/credentials/sealer.jks \
  --versionfile $IDP_HOME/credentials/sealer.kver \
  --alias secret \
  --storepass "$(grep '^idp.sealer.password'
$IDP_HOME/conf/credentials.properties | cut -d = -f 2)"

scp $IDP_HOME/credentials/sealer.* sealer_ssh@node1:$IDP_HOME/credentials/
scp $IDP_HOME/credentials/sealer.* sealer_ssh@node2:$IDP_HOME/credentials/
```

Comment out the scp commands if running without clustering.

Add the script to crontab as /etc/cron.d/idp-rotate-sealer with example daily key rotation time 5:10 :

```
10 5 * * * tomcat /opt/shibboleth-idp/credentials/rotate-sealer.sh
```

Logout support

To enable support for logout we need to be able to track sessions which in turn be able to use HTML local storage. Enable these by uncommenting following from /opt/shibboleth-idp/conf/idp.properties :

```
idp.storage.htmlLocalStorage = true
idp.session.trackSPSessions = true
idp.logout.elaboration = true
idp.logout.authenticated = false
idp.session.secondaryServiceIndex = true
```

Enabling status page

Enable status page by downloading jstl and rebuilding IDP.

```
cd /opt/shibboleth-idp/edit-webapp/WEB-INF/lib
wget https://repo1.maven.org/maven2/jstl/jstl/1.2/jstl-1.2.jar
JAVACMD=/usr/bin/java /opt/shibboleth-idp/bin/build.sh -
Didp.target.dir=/opt/shibboleth-idp
```

Allow temporary access from your development IP by modifying /opt/shibboleth-idp/conf/access-control.xml :

```
<entry key="AccessByIPAddress">
       <bean id="AccessByIPAddress" parent="shibboleth.IPRangeAccessControl"
           p:allowedRanges="#{ {
                 '127.0.0.1/32',
                 '::1/128',
                 'your-ip-address/netmask'
        } }" />
</entry>
```

Logging

For testing increase LDAP verbosity to the /opt/shibboleth-idp/conf/logback.xml file:

```
<logger name="org.ldaptive" level="INFO"/>
```

Or alternatively just enable LDAP authentication logging:

```
<logger name="org.ldaptive.auth.Authenticator" level="INFO" />
```

In case you want to link logs to /var/log/ use /var/log/shibboleth-idp. Do not use /var/log/shibboleth since Shibboleth SP rpm does logging in /var/log/shibboleth/ and sets the directory permissions on install so that IDP Tomcat cannot write to the same directory.

```
ln -s /opt/shibboleth-idp/logs /var/log/shibboleth-idp
```

You may also want to consider logrotate.

Metadata certificates

If clustering we need to use same certificates in the metadata in every node. Copy /opt/shibboleth-idp/credentials/idp.* from primary to secondary nodes.

Other

```
mkdir /opt/shibboleth-idp/tmp
chown -R tomcat.tomcat /opt/shibboleth-idp
```

IDP deployment

IDP is deployed with the /opt/tomcat/conf/Catalina/localhost/idp.xml with each tomcat restart.

In case you need to build it use command:

```
JAVACMD=/usr/bin/java /opt/shibboleth-idp/bin/build.sh -
Didp.target.dir=/opt/shibboleth-idp
```

And restart tomcat.

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

Then restart tomcat.

```
systemctl restart tomcat
```

Accessing status page

Access the status page at address https://n.n.n.n/idp/status from the IP that is allowed in /opt/shibboleth-idp/conf/access-control.xml as stated above.

Customizing looks

We are using css and image files that we store in /var/www/html/css /var/www/html/images. Copy the content, set the rights and SELinux contexts :

```
chown -R apache.apache css
chown -R apache.apache images
chown -R apache.apache fonts
chcon -R --reference /var/www/html /var/www/html/css
chcon -R --reference /var/www/html /var/www/html/images/
chcon -R --reference /var/www/html /var/www/html/fonts
```

IDP forms are created with Apache Velocity templates which reside in directory /opt/shibboleth-idp/views/. There are different templates which corresponding actions :

- login.vm : login page
- login-error.vm : error message included in login page
- intercept/attribute-release.vm : attribute release page
- logout.vm : logout page
- logout-propagate.vm : concluding logout page with propagation
- logout-complete.vm : concluding logout page without propagation
- user-prefs.vm : user preferences
- error.vm : error page

In addition there are duo.vm and spnego-unavailable.vm templates for functionalities not included in this document.

Customizations to login page can be run runtime by editing .vm files in the /opt/shibboleth-idp/views directory.

Default settings for appearance are in file /opt/shibboleth-idp/system/messages/messages.properties which is good to check and edit if needed. At least these values are good to keep in check :

```
idp.title = Org login service
idp.title.suffix = Error
idp.logo = /images/orglogo.jpg
idp.logo.alt-text = Org - Our Best Organization
idp.message = An unidentified error occurred.
idp.footer = The Best Footer Text
```

```
idp.url.password.reset = https://pwdreset.domain.com
idp.url.helpdesk = https://helpdesk.domain.com
```

The LDAP engine seems to give incorrect timeout error information on login password failure. We don't want that and in general it is a good to not give any information from the underlying identity storage other than successful or failed authentication. Therefore edit /opt/shibboleth-idp/views/login-error.vm and set error message section to a static one :

```
   <section>
        ## <p class="form-element form-
error">$encoder.encodeForHTML($message)</p>
        <p class="form-element form-error"> Authentication failed</p>
    </section>
```

Login.vm example :

```
#set ($rpContext =
$profileRequestContext.getSubcontext('net.shibboleth.idp.profile.context.Rel
yingPartyContext'))
#set ($username =
$authenticationContext.getSubcontext('net.shibboleth.idp.authn.context.Usern
amePasswordContext', true).getUsername())
#set ($passwordEnabled = false)
#if (!$passwordPrincipals or $passwordPrincipals.isEmpty() or
$authenticationContext.isAcceptable($passwordPrincipals))
#set ($passwordEnabled = true)
#end
##
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width,initial-scale=1.0">
<title>#springMessageText("idp.title", "Web Login Service")</title>
<link rel="stylesheet" type="text/css"
href="$request.getContextPath()/css/main.css">
</head>
<body>
<header>
        <div id="top">
                <img src="/images/orglogo.jpg" alt="Org logo">
        </div>
</header>
<div id="loginbar">
        <div id="loginbartitle">
                ORG LOGIN
        </div>
```

```
</div>
#parse("login-error.vm")
Username and password.<br/>
<form action="$flowExecutionUrl" method="post">
    #set ($serviceName = $rpUIContext.serviceName)
    #if ($serviceName &&
!$rpContext.getRelyingPartyId().contains($serviceName))
        <legend>
        #springMessageText("idp.login.loginTo", "Login to")
$encoder.encodeForHTML($serviceName)
        </legend>
    #end

    #if ($passwordEnabled)
        #springMessageText("idp.login.username", "Username")
        <input class="form-element form-field" id="username"
name="j_username" type="text"
        value="#if($username)$encoder.encodeForHTML($username)#end">
        <br/>

        #springMessageText("idp.login.password", "Password")
        <input class="form-element form-field" id="password"
name="j_password" type="password" value="">
        <br/>

        <input type="checkbox" name="donotcache" value="1" id="donotcache">
        #springMessageText("idp.login.donotcache", "Don't Remember Login")
        <br/>
    #end
    <input id="_shib_idp_revokeConsent" type="checkbox"
name="_shib_idp_revokeConsent" value="true">
    <label for="_shib_idp_revokeConsent">#springMessageText("idp.attribute-
release.revoke", "Clear prior granting of permission for release of your
information to this service.")</label>
    <br/>

    #if ($passwordEnabled)
        <button class="button" type="submit" name="_eventId_proceed"
onClick="this.childNodes[0].nodeValue='#springMessageText("idp.login.pleasew
ait", "Logging in, please wait...")'"
            >#springMessageText("idp.login.login", "Login")</button>
    #end

    #foreach ($extFlow in $extendedAuthenticationFlows)
        #if ($authenticationContext.isAcceptable($extFlow) and
$extFlow.apply(profileRequestContext))
            <button class="button" type="submit"
```

```
name="_eventId_$extFlow.getId()">
#springMessageText("idp.login.$extFlow.getId().replace('authn/','')",
$extFlow.getId().replace('authn/',''))
            </button>
        #end
    #end
</form>
#set ($logo = $rpUIContext.getLogo())
#if ($logo)
    <img src= "$encoder.encodeForHTMLAttribute($logo)"
    alt="$encoder.encodeForHTMLAttribute($serviceName)">
#end
#set ($desc = $rpUIContext.getServiceDescription())
#if ($desc)
    $encoder.encodeForHTML($desc)
#end

<ul class="list list-help">
        #if ($passwordEnabled)
            <li class="list-help-item"><a
href="#springMessageText("idp.url.password.reset", "#")"><span class="item-
marker">&rsaquo;</span> #springMessageText("idp.login.forgotPassword",
"Forgot your password?")</a></li>
        #end
        <li class="list-help-item"><a
href="#springMessageText("idp.url.helpdesk", "#")"><span class="item-
marker">&rsaquo;</span> #springMessageText("idp.login.needHelp", "Need
Help?")</a></li>
</ul>

<footer>
        ## <p class="footer-text">#springMessageText("idp.footer", "Insert
your footer text here.")</p>
</footer>
</body>
</html>
```

Adding IDP to test federation

Set up your organization user management description document (käyttäjähallintokuvaus) to a world accessible web page to be included in the federation configuration.

Go to HAKA resource registry and start identity provider add process.

- As an organization use HAKA test organization unless your organization is listed and you are

able use the appropriate selection.
- Use generated idp-metadata.xml as a reminder to fill out the necessary SSO addresses to the HAKA forms.
- It is allowed use self signed certificate unless you specifically wish to use known CA.

Test IDP with attribute test service and upon successful testing add IDP to federation with similar procedures and help from the excellent team professionals at HAKA federation.

Backups

Make directories for backups

```
mkdir /home/backups
mkdir /home/backups/postgresql

chown postgres:postgres /home/backups/postgresql
chmod 700 /home/backups/postgresql
```

PostgreSQL backup

```
vim /etc/cron.d/postgres-backup
```

```
10 5 * * * postgres pg_dumpall | gzip > /home/backups/postgresql/dumpall-
`date +\%a`.sql.gz
20 * * * * postgres pg_dumpall | gzip > /home/backups/postgresql/dumpall-
latest.sql.gz
```

Comments and suggestions

If you find bugs above please comment below. Also feel free to rate.