# Integration of Liferay with Shibboleth

## Shibboleth IdP:

### Preconfiguration

*Prepare your VM*

Check to see what IP address the VM has been assigned by using `ifconfig` and looking for `eth0`.
In your host environment, assign a unique FQDN hostname of the form host.domain.tld to your VM by editing the hosts file. Edit the hosts file generally at `/etc/hosts` using a text editor of your choice. Add a line following the existing mapping definitions with your chosen FQDN and the VM's IP address to create the mapping. The following is an example:

```
127.0.0.1 localhost
255.255.255.255 broadcasthost

x.y.z.w my.special.name
```

*(perhaps you need to configure iptables)*

Shibboleth requires that messages passed between the IdP and the SP are in close synchronization time-wise. Please check the `ntpd` service is running with:

```
service ntpd status
```

If it is not, please start the ntpd service:

```
service ntpd start
```

If it is not installed, please install with:

```
yum install ntp
```

### Prepare to install the Shibboleth IdP

Create a directory for the IdP installation

```
mkdir /opt/shibboleth-idp
```

Download the latest version of the IdP to `/opt/shibboleth-idp` and untar/zip the file

```
cd /opt/shibboleth-idp
curl -O http://shibboleth.net/downloads/identity-
provider/latest/shibboleth-identityprovider-2.3.8-bin.zip
unzip shibboleth-identityprovider-2.3.8-bin.zip
```

**Prepare Java/Tomcat**

Install java and tomcat6

```
yum install java-1.6.0-openjdk java-1.6.0-openjdk-devel
yum install tomcat6
service tomcat6 start
```

Copy all libraries from download/unzip directory's "endorsed" sub-directory (`/opt/shibboleth-idp/endorsed`) to Tomcat's `$CATALINA_HOME/endorsed` directory (you may need to create this folder)

```
mkdir $CATALINA_HOME/endorsed
cp /opt/shibboleth-idp/shibboleth-identityprovider-
2.3.8/endorsed/*.jar $CATALINA_HOME/endorsed
```

Download the file "`tomcat6-dta-ssl-1.0.0.jar`" and copy it to `$CATALINA_HOME/lib`.

```
curl -o $CATALINA_HOME/lib/tomcat6-dta-ssl-1.0.0.jar
https://build.shibboleth.net/nexus/content/repositories/releases
/edu/internet2/middleware/security/tomcat6/tomcat6-dta-
ssl/1.0.0/tomcat6-dta-ssl-1.0.0.jar
```

Configure Tomcat for endpoints on both ports 443 for SSL and 8443 in Tomcat's `server.xml`. Port 8443 will require client certificates on incoming connection (this is called the SOAP endpoint and the file you configure the endpoints in is called the `server.xml` which is found in `$CATALINA_HOME/conf`).
Add these endpoints next to the existing port 8080 Connector element.
*Remember the password you used; you will need it when you run the installation script.*

Here is a basic port 443 Configuration, the 8443 connector is documented on the wiki (see link below) and has been included below for clarity:

```
<Connector
   port="443"
   protocol="HTTP/1.1"
   SSLEnabled="true"
   maxThreads="150"
   scheme="https"
   secure="true"
   clientAuth="false"
   sslProtocol="TLS"
   keystoreFile="/opt/shibboleth-idp/credentials/idp.jks"
   keystorePass="YourSecretPassword" />


<Connector port="8443"
   protocol="org.apache.coyote.http11.Http11Protocol"
   SSLImplementation="edu.internet2.middleware.security.
   tomcat6.DelegateToApplicationJSSEImplementation"
   scheme="https"
   SSLEnabled="true"
   clientAuth="true"
   keystoreFile="/opt/shibboleth-idp/credentials/idp.jks"
   keystorePass="YourSecretPassword" />
```

The easiest way to allow Tomcat to listen on port 443 is to run it as root. Please modify `/etc/sysconfig/tomcat6` by uncommenting TOMCAT_USER and changing the TOMCAT_USER to root. We'll also tell it where the endorsed directory is located.

```
# What user should run tomcat
TOMCAT_USER="root"
# Where the endorsed directory is located
JAVA_ENDORSED_DIRS="$CATALINA_HOME/endorsed"
```

Production deployments should use a Tomcat "context deployment fragment" to automatically load the war file that is built by the Shibboleth installer. Upgrades and changes to the Shibboleth IdP's source that require a re-build (like the login page content) can then be done without messing with Tomcat's folder structure and caching. Create the file

`$CATALINA_HOME/conf/Catalina/localhost/idp.xml` and place the following content in it:

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

Finally, please set `JAVA_HOME`. We'll set `JAVA_HOME` for all users by adding it to `/etc/profile`, and then export it into the current shell so you don't need to relog:

```
echo "export JAVA_HOME=/path/to/jre" >> /etc/profile
export JAVA_HOME=/path/to/jre
```

### Install Shibboleth

Run the `install.sh` script found in the location you unzipped the shibboleth download:

```
cd /opt/shibboleth-idp
./install.sh
```

Accept default values other than hostname; enter the fully-qualified hostname for your IdP host (the external name your users' browsers will use and see). (use <MyIdPSpecialName> from above).

Enter the password for the java keystore created by the installer. This password should be the same password you configured in Tomcat's `server.xml` in the previous step.

Now, you will need to start tomcat.

### Configure User Authentication

Configure your IdP to authenticate your users using the UsernamePassword authentication handler

Configure the LDAP section of login.config in your shibboleth install:

Switch to the directory where the config file is kept [i.e. /opt/shibboleth-idp/conf]
Edit login.config
A completely configured login.config file will look like this:

```
ShibUserPassAuth {

   edu.vt.middleware.ldap.jaas.LdapLoginModule required
      host="idp.training.incommon.org"
      port="389"
      base="ou=people,dc=example,dc=org"
      bindDn="uid=IdPServiceAcct,ou=people,dc=example,dc=org"
      bindCredential="password"
      userField="uid={0}";
};
```

Edit `handler.xml`:

Uncomment the UsernamePassword login handler in `handler.xml`
Comment the RemoteUser login handler

Restart Tomcat.

### Configure User Attributes/Attribute Resolver

Uncomment/reconfigure/add the attribute definitions in the IdP's `conf/attribute-resolver.xml` file.

```
<resolver:AttributeDefinition xsi:type="ad:Simple" id="uid"
sourceAttributeID="uid">
        <resolver:Dependency ref="myLDAP" />
        <resolver:AttributeEncoder xsi:type="enc:SAML1String"
name="urn:mace:dir:attribute-def:uid" />
        <resolver:AttributeEncoder xsi:type="enc:SAML2String"
name="urn:oid:0.9.2342.19200300.100.1.1" friendlyName="uid" />
    </resolver:AttributeDefinition>
...
```

Uncomment and configure the `resolver:DataConnector` called `myLDAP` (towards the bottom of the file) to connect to your LDAP directory and retrieve information for the authenticated user:

```
<resolver:DataConnector id="myLDAP" xsi:type="dc:LDAPDirectory"
        ldapURL="ldap://idp.training.incommon.org:389"
        baseDN="ou=people,dc=example,dc=org"
        principal="uid=IdPServiceAcct,ou=people,dc=example,dc=org"
        principalCredential="password">
        <dc:FilterTemplate>
            <![CDATA[
                (uid=$requestContext.principalName)
            ]]>
        </dc:FilterTemplate>
    </resolver:DataConnector>
```

### Configure Attribute Release Policies

Create a new rule to release attributes to the SP located by adding this to your `attribute-filter.xml` inside the main `<AttributeFilterPolicyGroup>` element:

```
<afp:AttributeFilterPolicy id="releaseToAnyone">
<afp:PolicyRequirementRule xsi:type="basic:ANY" />

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

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


</afp:AttributeFilterPolicy>
```

### Verify that your new IdP is up and running

Access the following URL from the IdP server itself by running the following command:

```
curl -k https://localhost/idp/status
```

You should see lots of information about your IdP that starts with the following:

```
### Operating Environment Information
operating_system: Linux
```

You can also try from your web browser the old deprecated status handler at this URL (it simply responds with "ok"):

```
https://<MyIdPSpecialName>/idp/profile/Status
```

# Shibboleth SP:

### Preconfiguration

Prepare your VM for the Session
Check to see what IP address the VM has been assigned by using `ifconfig` and looking for `eth0`.
In your host environment, assign a unique FQDN hostname of the form host.domain.tld to your VM by editing the hosts file. Edit the hosts file generally at `/etc/hosts` using a text editor of your choice. Add a line following the existing mapping definitions with your chosen FQDN and the VM's IP address to create the mapping. The following is an example:

```
127.0.0.1 localhost
255.255.255.255 broadcasthost

x.y.z.w my.special.name
```

*(perhaps you need to configure iptables)*

Shibboleth requires that messages passed between the IdP and the SP are in close synchronization time-wise. Please check the `ntpd` service is running with:

```
service ntpd status
```

If it is not, please start the ntpd service:

```
service ntpd start
```

If it is not installed, please install with:

```
yum install ntp
```

### Installation

Download Shibboleth repo to /etc/yum.repos.d/:

```
curl -o /etc/yum.repos.d/security:shibboleth.repo
http://download.opensuse.org/repositories/security://shibboleth/
CentOS_CentOS-6/security:shibboleth.repo
```

Install the Shibboleth SP software on your web server:

```
yum install shibboleth
```

Basic Apache configuration:

Turn on canonical name support. This is necessary because the SP uses the server name that is provided by the browser and web environment when constructing a variety of links. If the server name is variable, some string matches can fail and trust failures may occur.
Disable http:// access to sensitive resources that require Shibboleth protection.

```
RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}
```

Add mod_ssl

```
yum install mod_ssl
```

Import the SP's certificate in the Java keystore

```
keytool -import -keystore $JAVA_HOME/jre/lib/security/cacerts -file sp-cert.pem -alias sp
```

Turn on canonical name use by Apache in /etc/httpd/conf/httpd.conf:

```
UseCanonicalName On
ServerName <MySPSpecialName>:80
```

**Verify that your new Shibboleth SP is alive**

First, start/restart Apache on your server to load mod_shib using:

```
service httpd restart
```

Next, start shibd using:

```
service shibd start
```

The status handler is protected by an IP address ACL, which means you need to either perform this task from the web server itself or edit the ACL for the `/Statushandler` in `/etc/shibboleth/shibboleth2.xml`. Keep in mind that the list is space delimited for example:

```
<Handler type="Status" Location="/Status" acl="127.0.0.1
10.0.0.2 10.0.0.3"/>
```

Access the URL https://127.0.0.1/Shibboleth.sso/Status from inside your VM:

```
curl -k https://127.0.0.1/Shibboleth.sso/Status
```

### Configure Shibboleth SP

Edit `/etc/shibboleth/shibboleth2.xml` file for configure the SP to support CASShib. Each service needs to have its own protected *Shibboleth* address for CAS validation. For mapping URLs with services we add these rows in the shibboleth configuration file.

Under the root tag SPConfig you need to add:

```
<RequestMapper type="Native">
      <RequestMap applicationId="default">
            <Host name="MySPSpecialName" port="443" scheme="https">
                  <PathRegex regex="casshib/shib/app2" applicationId="app2"
                  authType="shibboleth" requireSession="true"/>
            </Host>
      </RequestMap>
</RequestMapper>
```

After this, you need to configure the section with fake service. This means that if the request doesn't match the regular expressions return an error page. For configuring the registrated service we add these lines under the tag ApplicationDefaults:

```
<ApplicationOverride id="app2"
                     entityID="https://MySPSpecialName/casshib/app2"
                     homeURL="https://MySPSpecialName/app2/"
                     REMOTE_USER="shibattr-mail">
      <Sessions lifetime="28800" timeout="3600" checkAddress="false"
            handlerURL="/casshib/shib/app2/Shibboleth.sso" handlerSSL="true"
            exportLocation="/casshib/shib/app2/Shibboleth.sso/GetAssertion"
            exportACL="MySPSpecialAddress"
            idpHistory="false" idpHistoryDays="7"
            cookieProps="; path=/casshib/shib/app2">
      </Sessions>
</ApplicationOverride>
```

Edit `/etc/shibboleth/shibboleth2.xml`

Edit `/etc/shibboleth/attribute-map.xml` file to define the attributes used by Shibboleth SP:

```
<Attribute name="urn:mace:dir:attribute-def:uid" id="shibattr-uid"/>
<Attribute name="urn:oid:0.9.2342.19200300.100.1.1" id="shibattr-uid"/>
<Attribute name="urn:mace:dir:attribute-def:mail" id="shibattr-mail"/>
```

```
<Attribute name="urn:oid:0.9.2342.19200300.100.1.3" id="shibattr-mail"/>
```

## Prepare Java/Tomcat

Install java and tomcat6

```
yum install java-1.6.0-openjdk java-1.6.0-openjdk-devel
yum install tomcat6
```

Configure APR connector in Tomcat by editing the `server.xml` file:

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

Start Tomcat server:

```
service tomcat6 start
```

## Connect Tomcat with Apache Web Server

```
yum install httpd-devel

cd /tmp
download tomcat-connector source from mirror (tomcat website)
untar it

cd /tmp/tomcat-connectors-current.version-src /native
./buildconfig.sh
yum install gcc*
./configure --with-apxs=/usr/sbin/apxs
make
cd ./apache-2.0
cp mod_jk.so /usr/lib64/httpd/modules/

touch /etc/httpd/workers.properties
vi /etc/httpd/workers.properties
```

create a workers.properties file in your /etc/httpd/ dir, with at least:

```
workers.tomcat_home=/srv/tomcat7
workers.java_home=/usr/lib/jvm/java-1.6.0-openjdk-
1.6.0.0.x86_64/jre
```

```
worker.list=worker1
worker.worker1.type=ajp13
worker.worker1.host=localhost
worker.worker1.port=8009
```

create a `httpd-jk-init.conf` file:

```
LoadModule jk_module modules/mod_jk.so
<IfModule jk_module>
   JkWorkersFile conf/workers.properties
   JkShmFile logs/mod_jk.shm
   JkLogFile logs/mod_jk.log
   JkLogLevel info
   JkLogStampFormat "[%a %b %d %H:%M:%S %Y]"
</IfModule>
```

create a `httpd-jk-mount.conf` file:

```
<IfModule jk_module>
   JkMount /casshib/* worker1
   JkUnMount /casshib/shib/*/Shibboleth.sso worker1
   JkUnMount /casshib/shib/*/Shibboleth.sso/* worker1
</IfModule>
```

**Protect the login and Shibboleth.sso URLs.**

In your /etc/httpd/conf/httpd.conf the <LocationMatch> element activates mod_shib to protect a resource:

```
<LocationMatch "/casshib/shib/[^/]*/(login|Shibboleth.sso(/[^/]*)?)">
   AuthType shibboleth
   ShibRequireSession On
   ShibUseHeaders On
   require valid-user
</LocationMatch>
```

Restart `shibd`

## CASShib

### Installation (on the Shibboleth SP machine)

```
cd /tmp
wget http://casshib.googlecode.com/files/casshib-server-3.4.2a-release.tar.gz
tar xzf casshib-server-3.4.2a-release.tar.gz
cd  casshib-server-3.4.2a/modules/
cp casshib-server-webapp-3.4.2a.war $CATALINA_HOME/webapps/casshib.war
```

### Configuration

### You need to modify the CASShib Service Registrations file in this way:

```
nano $CATALINA_HOME/webapps/casshib/WEB-INF/classes /casshib-service-
registrations.xml

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<casShibServiceRegistrations>
  <service id="https://<MygCubePortalSpecialName>/c/portal/login"
           appname="app2"
           passcode="12345" />
</casShibServiceRegistrations>
```

## gCube Portal

### Installation

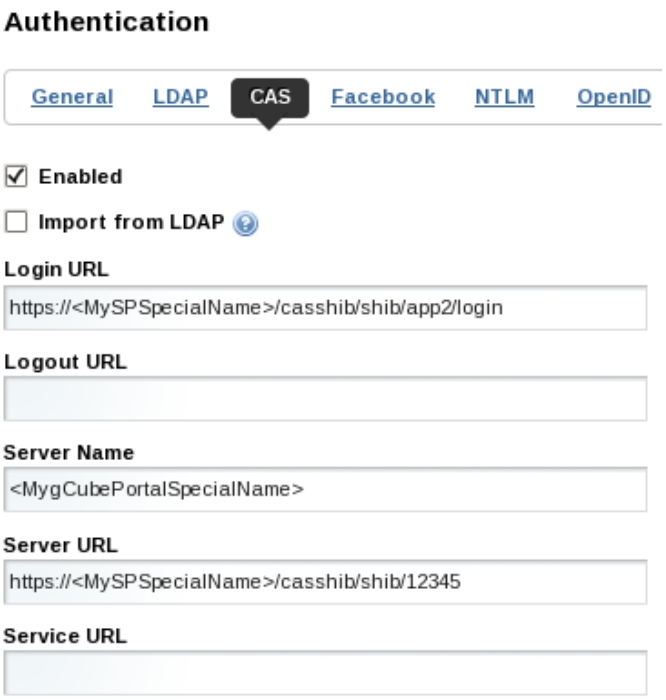
https://gcube.wiki.gcube-system.org/gcube/index.php/GCube_Portal_Installation

### Configure CAS authentication

From the portal's web page you navigate to

```
Manage/Control Panel/Settings/Authentication/CAS
```

and you configure Liferay to use CASShib in this way:




### Apache + Java keystore

Install apache and mod_ssl

```
yum install httpd mod_ssl
```

Import your portal's certificate into Java keystore

```
keytool -import -keystore $JAVA_HOME/jre/lib/security/cacerts
-file /path/to/portalcert/portal-cert.pem -alias portal
```

## Configure Shibboleth SP and IdP

## Metadata

### Download and store SP Metadata

```
curl -o /opt/shibboleth-idp/metadata/sp-metadata.xml
https://<MySPSpecialName>/Shibboleth.sso/Metadata (on the IdP)
```

### Download and store App2 Metadata

```
curl -o /opt/shibboleth-idp/metadata/app2-metadata.xml
https://<MySPSpecialName>/casshib/shib/app2/Shibboleth.sso/Metad
ata (on the IdP)
```

### Open the `relying-party.xml` of the Shibboleth IdP and change the Metadata Provider entry to:

```
<metadata:MetadataProvider id="ShibbolethMetadata"
      xsi:type="metadata:ChainingMetadataProvider">

        <metadata:MetadataProvider id="FileIDPMD"
xsi:type="metadata:FilesystemMetadataProvider"
                          metadataFile="/opt/shibboleth-idp/metadata/idp-
metadata.xml" maxRefreshDelay="P1D" />

        <metadata:MetadataProvider id="FileSPMD"
xsi:type="metadata:FilesystemMetadataProvider"
                          metadataFile="/opt/shibboleth-idp/metadata/sp-
metadata.xml" maxRefreshDelay="P1D" />

 <metadata:MetadataProvider id="FileApp2MD"
xsi:type="metadata:FilesystemMetadataProvider"
                          metadataFile="/opt/shibboleth-idp/metadata/app2-
metadata.xml" maxRefreshDelay="P1D" />
</metadata:MetadataProvider>
```

### Download and store IdP Metadata

```
curl -o /opt/shibboleth/idp-metadata.xml
https://<MyIdPSpecialName>/idp/profile/Metadata/SAML(on the SP)
```

### Open the `shibboleth2.xml` of the Shibboleth SP and change the MetadataProvider entry to:

```
<MetadataProvider type="XML" file="/opt/shibboleth/idp-
metadata.xml"/>
```

**References**

*Shibboleth Identity Provider*

*Shibboleth Service Provider*

*CASShib Official Wiki*

*CASShib Installation*

*Liferay Official Documentation*