

# DRAFT



## **Shibboleth Configuration for Colleges**

**Published 7-18-16**

# DRAFT

## Contents

<b>Purpose of This Document.....</b>	<b>3</b>
<b>Single Sign On High-Level Overview (Shibboleth Example).....</b>	<b>4</b>
<b>1. Shibboleth IdP Setup Steps.....</b>	<b>6</b>
Prerequisites.....	6
A. Install Java V8.....	7
B. Install Java Cryptography Extensions (JCE).....	7
<b>2. Upgrade Shibboleth IdP from V2.0 to V3.0 - Overview.....</b>	<b>8</b>
Sample Shibboleth IdP V3.0 Install/Upgrade.....	8
<b>3. Post-Install IdP Configuration.....</b>	<b>10</b>
A. Copy the Original IdP V3.0 Configuration Files.....	10
B. Configure Authentication Handling and Any UID Considerations.....	11
1. Copy the IdP V2.0 Files to IdP V3.0.....	11
2. Copy Over Signing Keys.....	15
3. Update Jetty Configuration.....	15
4. Add the Open CCC Metadata.....	16
5. Other Certificate Configuration.....	16
6. Configure Log Files.....	17
C. Apply CCCApply Customizations.....	17
D. Download the Active Directory (AD) Security Certificate.....	17
E. Rebuild the Shibboleth V3.0 IdP WAR File.....	17
<b>4. Test Your Shibboleth IdP V3.0 Server.....</b>	<b>19</b>
<b>5. Configure a Script and CRON Job for Your AES Key.....</b>	<b>20</b>
<b>Glossary of Terms.....</b>	<b>22</b>

# DRAFT

## Purpose of This Document

---

In order for students to access the Student Services Portal, Course Exchange, and Canvas, colleges need to configure their student population (student SIS) for Single Sign On (SSO). This process may include some complexities.

This document provides an overview to assist colleges in deciding whether to:

- Hire a third-party to do the necessary SSO installation and configuration
- OR
- Use your college resources to do the necessary SSO installation any configuration

To assist you in your decision-making, this guide provides:

- an overview of SSO using Shibboleth as an example
- an example workflow of upgrading your current Shibboleth SSO to IdP V3.0 and for the Student Services Portal, Course Exchange, and Canvas



**Note:** While the California Community Colleges already have Shibboleth IdP V2.0 in place for SSO for college staff who use the Administrator and CCC Report Center, instead of following the example path in this guide you may consider other third-party SSO products such as PortalGuard or Ping Identity in place of Shibboleth. Nonetheless, the focus of this guide will be in the context of Shibboleth IdP V3.0.

# DRAFT

## Single Sign On High-Level Overview (Shibboleth Example)

Single Sign On (SSO) is a session and user authentication process that permits a user to enter one name and password in order to access multiple applications. For example, when CCC students are configured for SSO, they can log in to either the Student Service Portal, Course Exchange, or Canvas and when they navigate to one of the other three web applications, they will not have to login again.

Right now:

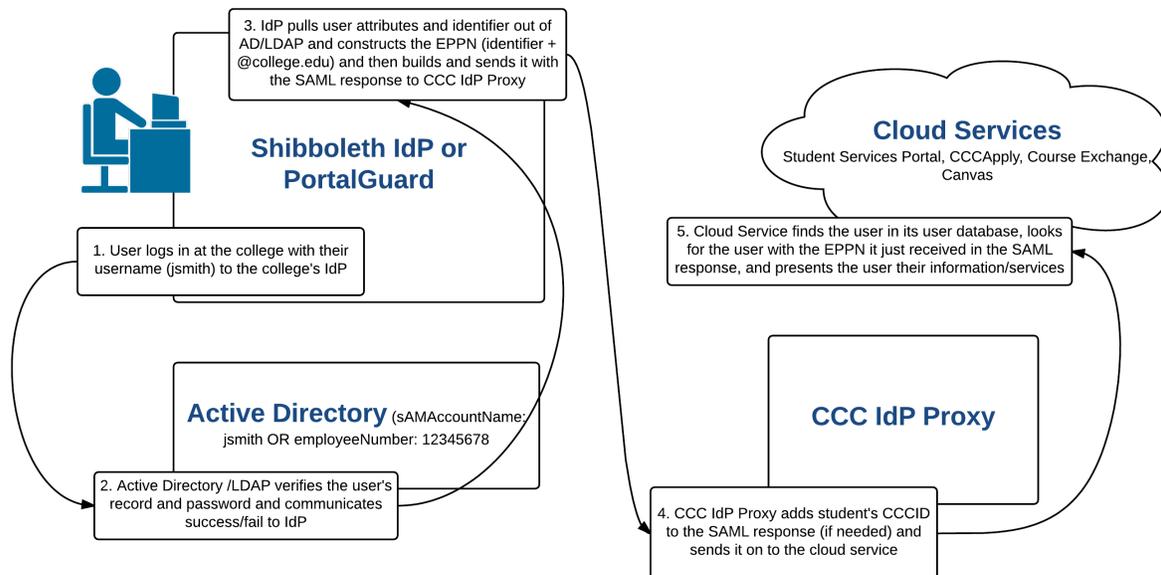
- All California Community Colleges already have Shibboleth Identity Provider (IdP) in place to authenticate college staff.
- College staff use Shibboleth's SSO to access the Administrator and the Report Center.

Next:

- Colleges must upgrade their SSO to include students so that they can access new resources. For example:

To Access:	Which...
Student Services Portal	houses the CCCApply application
Course Exchange	is available in the Student Service Portal and will need SSO authorization
Canvas (the state-wide adopted LMS)	is a Learning Management System that facilitates students' online courses

The SSO process involves authentication and authorization. Authentication is a confirmation that the person logging in is the person they claim to be. Authorization is a confirmation that the logged-in person is authorized to access a particular "resource" (i.e. Student Services Portal, etc.). The Tech Center has implemented an additional CCC IdP Proxy server to ease future upgrades.



In preparation for implementing SSO for students, CCCs will need to:

- Either:

DRAFT

- Determine if college IT staff will manage the Shibboleth IdP upgrade and configuration necessary for student SSO



**Note:** An understanding of security and risk in SSO configuration is necessary.

OR

- Determine if a third-party with expertise in Shibboleth IdP configuration should be hired to manage the Shibboleth upgrade and configuration necessary for student SSO

OR

- Select another third-party SSO software (i.e. PortalGuard or Ping Identity) and work with their support staff
- Consult with the Service Provider (SP) vendors for details and assistance:
  - CCC Tech Center for: Student Services Portal, Course Exchange, CCCApply
  - Canvas tech support for Canvas
- Get student accounts out of the college Student Information System (SIS) and into Active Directory (AD) in order to authorize students. Required student information includes CCCID, EPPN (eduPersonPrincipleName), and the college's MIS code.



**Note:** This step includes making a decision to store student accounts in either the same directory as the college staff or in a separate directory from the college staff so that there are two directories. For the purposes of this document, two ADs are assumed.

- Upgrade Shibboleth V2.0 Identity Provider (IdP) to Shibboleth 3.0 IdP
- Configure the Shibboleth Identity Provider (IdP):
  - with Metadata from the Service Providers (SPs):
    - Student Services Portal
    - Course Exchange
    - Canvas
  - to pass the required attributes to the SPs
  - to authenticate student accounts

CCC Tech Center will:

- Configure the CCC IdP Proxy, Student Services Portal, Course Exchange, and CCCApply SPs to accept authentication from the college's IdP (this requires metadata from each college's IdP)

# DRAFT

## 1. Shibboleth IdP Setup Steps

---

There are some prerequisites your server needs to meet prior to installing/upgrading Shibboleth IdP V3.0.

**Table 1: Overview of Shibboleth Install/Upgrade Prerequisites**

Prerequisite	
Planning to use the Student Services Portal or Course Exchange?	Consult with your CCC Tech Center representative prior to upgrading
Planning to integrate with Canvas?	Consult with Canvas tech support prior to upgrading
Install or upgrade to the latest Java 8 (32- and 64-bit) and Java Cryptography Extensions (JCE)	See <a href="#">A. Install Java V8</a> on page 7 and <a href="#">B. Install Java Cryptography Extensions (JCE)</a> on page 7
Additional recommendations/requirements: <ul style="list-style-type: none"> <li>• No firewall between IdP and Active Directory</li> <li>• Windows 2008 R2 preferred (if Windows)</li> <li>• 4 GB RAM, modern processor, ~24GB storage should be enough for: logging, OS, Apps</li> <li>• No failover needs</li> <li>• Commercial SSL cert availability for IdP (and DNS resolution for the selected name)</li> <li>• Remote access (RDP for Windows, SSH for Linux)</li> </ul>	

The overall workflow for upgrading Shibboleth V2.0 to V3.0 follows this path:

1. Ensure Prerequisites are in place (Install Java 8 (32- and 64-bit) and Java Cryptography Extensions (JCE))
2. Upgrade your Shibboleth IdP installation from V2.0 to V3.0 using the Windows MSI installer
3. Implement Post-Install Configuration
4. Test your IdP with TestShib
5. Generate an SSL Certificate
6. Make any "look and feel" updates
7. Configure a CRON job for AES key

## Prerequisites

---

The instruction steps that follow assume your college has the following:

- Knowledgeable IT staff aware of security scenarios, risks, and resolutions
- A working version of Shibboleth IdP V2.0 for CCCApply
- A Windows server
- An Active Directory for your student SIS population (separate from the staff AD) that includes:
  - the students' CCCID
  - your college MIS code
  - students' EPPN
- You have confirmed your test environment meets these CCCTC guidelines: [1. Shibboleth IdP Setup Steps](#) on page 6.

## A. Install Java V8

---

1. You may have Java installed on your server already. To check for it, enter the following command at a command prompt:

```
java -version
```

If your Java version is at 1.8.0\_91 or above, skip the steps below and go to [B. Install Java Cryptography Extensions \(JCE\)](#) on page 7.

If your Java version is earlier than 1.8.0\_91 or non-existent, then go to the next step, below.

2. Navigate to Oracle's site and download and install Java Runtime Environment (JRE) or Java Development Kit (JDK) V8 for your specific operating system.
  - <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>
3. Add JAVA\_HOME to the system variables to the 64 bit version: C:\Program Files (x86)\Java\jre1.8.0\_91\

## B. Install Java Cryptography Extensions (JCE)

---

1. Downloaded JCE Unlimited Strength (needed for encryption used by Shibboleth):
  - <http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html>
2. Replace the local\_policy.jar and US\_export\_policy.jar files in %JAVA\_HOME%\lib\security with those corresponding files in the JCE.

# DRAFT

## 2. Upgrade Shibboleth IdP from V2.0 to V3.0 - Overview

---

Most California Community Colleges have Shibboleth IdP V2.0 and must upgrade to V3.0. Shibboleth V2.0 is end of life and all security maintenance will end as of July 31, 2016.

- Review Shibboleth's IdP upgrade instructions on your development/test environment here: <https://wiki.shibboleth.net/confluence/display/IDP30/UpgradingFromV2>.

For the purposes of this guide, a Windows server environment is assumed.

When you reach the step (in the above link) to install Shibboleth V3.0 using the Windows installer, see:

[Sample Shibboleth IdP V3.0 Install/Upgrade](#) on page 8 in this guide. These instructions provide an example path for a Windows Server environment.

### Sample Shibboleth IdP V3.0 Install/Upgrade

---

The instructions in this section provide sample details for the Windows OS installer for Shibboleth V3.0.



**Note:** These instructions do not cover every detail of installation and configuration necessary but serve as an example only.

The instructions on this page assume that you have:

- Followed Shibboleth's upgrade instructions up to the Installing Shibboleth V3.0 step: <https://wiki.shibboleth.net/confluence/display/IDP30/UpgradingFromV2>
- Confirmed system requirements
- Installed the Java Cryptography Extensions (JCE)
  - Downloaded JCE Unlimited Strength (needed for encryption used by Shibboleth):
    - <http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html>
    - Replaced the JAR files in %JAVA\_HOME%\lib\security with JCE JAR files.
- Backed up your test environment before upgrading it

#### Shibboleth Installer for Windows

The instructions below provide configuration choices when running the Shibboleth V3.0 Windows installer. For reference, see: <https://wiki.shibboleth.net/confluence/display/IDP30/WindowsInstallation>

1. Download the shibboleth-identity-provider-3.0.0-x64.msi (for Windows) install file here:

- <http://shibboleth.net/downloads/identity-provider/latest/>



**Note:** If you're using a non-Windows computer select the appropriate download file for your OS.

2. Double-click the downloaded file and click **Run** when prompted to display the Shibboleth installer intro screen.
3. Click **Next** to display the license agreement and select the check box to accept it.
4. Click **Next** to display the Configure Shibboleth screen and:

- Select the *Install Jetty* and *Configure Active Directory* check boxes



**Note:** This installer selection handles all the configuration needed for the IdP to run under Jetty (except for pointing to the right certificate store it is using for the server) and will also install a service definition into the Windows Services Manager to start and stop Jetty. Additional post-install configuration will be needed for Active Directory as well.

- In the "...DNS Name for this IdP.." field, enter: idp.yourcollege.edu
- In the "...scope that the IdP will assert" field, enter: yourcollege.edu

# DRAFT

...where "yourcollege" is replaced by your actual college name.



**Note:** You are going to want Jetty/the Shibboleth IdP to "control/be available" on the standard SSL/https port, 443. So you do not want IIS running on the IdP server (if it is, ensure that it is running on any port other than 80).

5. Click **Next** to display the Configure for Active Directory screen and:
  - In the "*Specify the Active Directory Domain*" field enter: dom-d1.yourcollege.edu
  - Select the *Use Global Catalog* check box
  - In the "*Username (no domain)*" field, enter: ???
  - In the "*Password*" field enter: ???
  - CN=readonly web,CN=Users,DC=yourcollege,DC=edu
6. Click **Next** through the next to screens to begin installation.
7. Click **Finish** when the Completed the Shibboleth IdP V3 Setup Wizard screen displays to close the installer.

## Test the Basics

To test that your install is functioning, do one of the following:

### Test the Core

1. Open a command prompt (Powershell) window.
2. cd "C:\Program Files (x86)\Shibboleth\IdP"
3. bin\status

OR

### TestShib

1. Downloaded testshib-providers.xml and put it in ../metadata
2. At the command prompt, navigate to C:\Program Files (x86)\Shibboleth\IdP and enter: bin\status.bat.

You should see output summarizing the environment and information about the IdP's state.

# DRAFT

## 3. Post-Install IdP Configuration

---

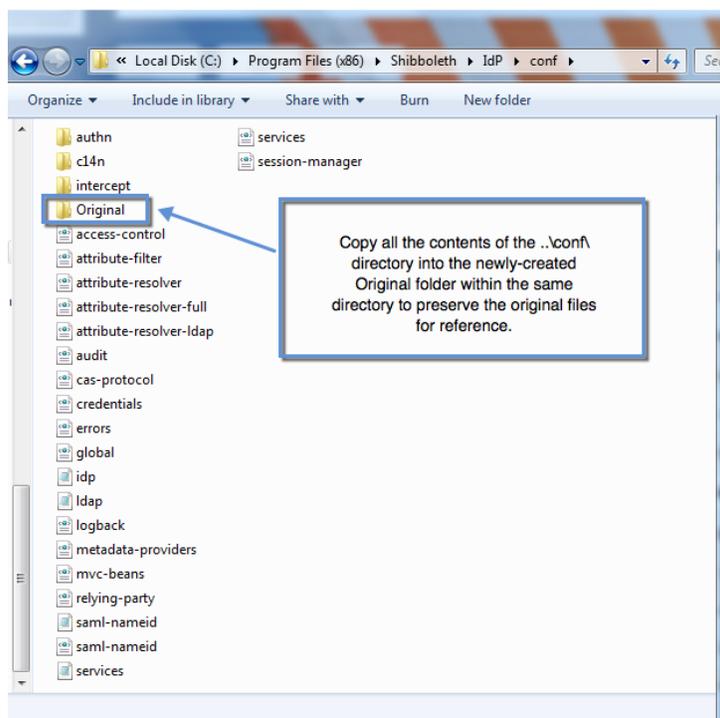
The majority of the Shibboleth IdP install/upgrade effort takes place in configuration files (i.e. `..\conf\`). The information in this section is an example path to configuring the necessary files when you upgrade from IdP V2.0 to IdP V3.0.

As a general reference to configuration file usage, see: <https://wiki.shibboleth.net/confluence/display/IDP30/Configuration>

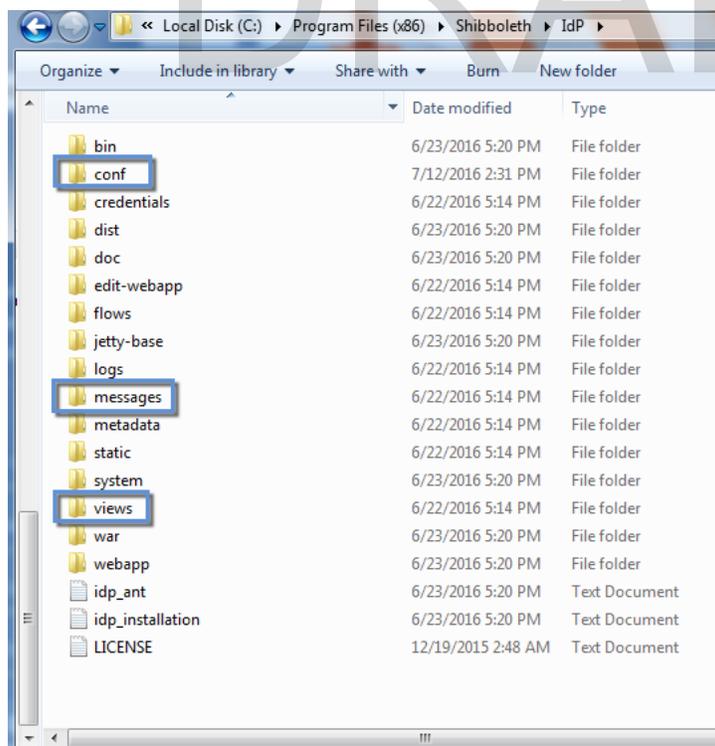
### A. Copy the Original IdP V3.0 Configuration Files

---

1. Create a folder named "Original" in the `C:\Program Files (x86)\Shibboleth\IdP\conf` directory and copy all the files from `..\conf\` into the original folder so you will preserve the original files. Keeping an original copy of the newly-installed configuration files is useful for reference.



2. Copy the `..\views` and `..\messages` directories to preserve their original files.



## B. Configure Authentication Handling and Any UID Considerations

After upgrading to IdP V3.0, you will need to configure authentication and make any adjustments required for advanced use. Authentication and UI considerations are where the bulk of configuration is needed to get the SSO system working.

Since most all CCCs already have Shibboleth V2.0 prior to the V3.0 upgrade, a lot of configuration needs can be handled by copying and pasting V2.0 configuration files to V3.0.

### 1. Copy the IdP V2.0 Files to IdP V3.0

To configure the IdP, copy over your current IdP V2.0 config files to the IdP V3.0 folders using the steps below.

1. Copy the following credentials from the V2.0 `..\credentials` directory into the V3.0 `..\credentials` directory:
  - IdP signing certificate and the key that goes with it
  - server credentials
2. Copy the V2.0 `..IdP\metadata` files into the V3.0 `..IdP\metadata` directory.
3. Copy the V2.0 logo file (image for Login page etc.) into the V3.0 `..\webapp\images` directory.

Many text strings, error message text, etc. shown on various user pages (error, logout, etc.) are configured in the `..\messages\` directory.

To make more substantial changes beyond these text strings and error message text, you will modify the Velocity templates themselves (located in the `..\views\` directory) and/or the CSS that is located in `..\edit-webapp\css\` directory.



**Note:** Any changes to the `..\edit-webapp\` directory require you rebuild the WAR file and restart of Jetty/IdP to take effect. Velocity template changes are "live" (i.e. they take place immediately).

Make the following changes to affect look and feel and messaging:

1. Use the table below to make changes in the `..\messages\` and `..\views\` directories:

Open this file:	And do this:
messages\error-messages.properties	update error.message.properties with the correct title, logo reference, etc.
messages\authn-messages.properties	update other messages and/or labels that impact the Login page
views\login.vm	comment out: <ul style="list-style-type: none"> <li>options for remember me</li> <li>consent</li> <li>the Don't Remember and Revoke Consent check boxes</li> </ul>
other views\*.vm files	for other user pages

- Copy the files in the IdP V2.0 `..\metadata\` directory into the IdP V3.0. This directory includes backing files for URL-consumed metadata and the InCommon metadata file. The signing cert for checking the validity of those feeds was added to the `..\credentials\` directory in step one above.

Some additional "look and feel" changes you may wish to make:

- Add the college logo to `edit-webapp\images`.

- Open a command window and run `bin\build.bat`. You will need admin privileges.

 **Note:** You have to press enter after first prompt.

- Restart Shibboleth service.

- Modify `views\login.vm` to use the new image and comment out the footer. This is not in the WAR file so it can be edited without having to rebuild.

 **Note:** However, to add the image you may need to rebuild above.

- Modify `views\intercept\attribute-release.vm` to use the new image.

- Log in to Shibboleth V2.0 and use the following information to help guide changes to:

- `..\views\login.vm`
- `..\messages\*` files

- Copy the V2.0 `attribute-filter.xml` file into your V3.0 `..\conf` directory.

- Open the `attribute-filter.xml` file you just copied into the V3.0 directory and make the following edits to simplify it. (V3.0 allows this simpler form that is easier to read.)

- Edit the `<AttributeFilterPolicyGroup>` element id attribute to the new v3 value.
- Remove namespace prefixes:
  - `afp: ->`
  - `basic: ->`
  - `saml: ->`
  - `AttributeRequesterString -> Requestor`
  - `AttributeValueString -> Value`
  - `saml:AttributeRequesterEntityAttributeExactMatch -> EntityAttributeExactMatch`
  - `saml:AttributeRequesterInEntityGroup -> InEntityGroup`

Make other organizational changes as seem appropriate.

- Copy the V2.0 `attribute-resolver.xml` file into your V3.0 `..\conf` directory and then make the following edits:

- Replace the `<resolver:AttributeResolver>` element with a copy of the one provided in the IdP V3.0 sample in the Original folder you created.
- Make needed changes to scripts to account for differences that Java 8 introduced in the Javascript/ECMA "engine".

- Remove (most) NameID definitions. Those now go into the saml-nameid.xml config file. (Some NameID-related options are specified in saml-nameid.properties, but you don't need to configure those unless you've been using a ComputedID.)

 **Note:** You no longer need the Principal elements at the end of the resolver.

- Add configuration for determining whether the user is student or staff based on a substring of their EPPN (the domain). For example:

```
<resolver:AttributeDefinition xsi:type="ad:Mapped"
id="eduPersonAffiliation" sourceAttributeID="userPrincipalName">
  <resolver:Dependency ref="myLDAP" />
  <resolver:AttributeEncoder xsi:type="enc:SAML1String"
name="urn:mace:dir:attribute-def:eduPersonAffiliation"
encodeType="false" />
  <resolver:AttributeEncoder xsi:type="enc:SAML2String"
name="urn:oid:1.3.6.1.4.1.5923.1.1.1.1"
friendlyName="eduPersonAffiliation" encodeType="false" />
  <ad:DefaultValue>affiliate</ad:DefaultValue>
  <ad:ValueMap>
    <ad:ReturnValue>student</ad:ReturnValue>
    <ad:SourceValue ignoreCase="true">.+@student\.yourSchool\.edu</
ad:SourceValue>
  </ad:ValueMap>
  <ad:ValueMap>
    <ad:ReturnValue>staff</ad:ReturnValue>
    <ad:SourceValue ignoreCase="true">.+@yourSchool\.edu</ad:SourceValue>
  </ad:ValueMap>
</resolver:AttributeDefinition>
```

- Open the following IdP V3.0 files and apply values in your previously-edited V2.0 file to the new, out-of-the-box V3.0 files:

- ..\conf\relying-party.xml
- ..\conf\metadata-providers.xml

 **Note:** The V2.0 relying-party.xml file has been split into two distinct config files in V3.0. The relying party information remains in the V3.0 relying-party.xml file, but the metadata configuration is now in the V3.0 metadata-providers.xml file.

Also, in V3.0's NameID support, there are more times where you'll need to add a relying-party.xml override to ensure that the right NameID type is "chosen" for that SP. You must do that if you want any form of "unspecified" format NameID to be chosen.

**Table 2: Summary of ..\config Files You Edit to Match Previously-Installed IdP V2.0/Add New Functionality**

File:	Action:
idp.properties	turn on: idp.encrypted.optional = true   <b>Note:</b> IdP SSO Sessions: There are various session-length settings that have default values (for now) in ..\conf\idp.properties and ..\conf\authn\general-authn.xml. You can read about the properties.settings that influence the length of the IdP's SSO session on the following page: <a href="https://wiki.shibboleth.net/confluence/display/IDP30/SessionConfiguration">https://wiki.shibboleth.net/confluence/display/IDP30/SessionConfiguration</a>

File:	Action:
	Under the default configuration, user authentication occurs hourly except in cases where the IdP session is idle for more than 30 minutes.
ldap.properties	<i>Configure the LDAP Properties</i> on page 14
attribute-filter.xml	
attribute-resolver.xml	
metadata-providers.xml	
relying-party.xml	
saml-nameid.xml	to define special NameIDs needed

9. Save and close the V3.0 files to save your changes.

### Configure the LDAP Properties

With Shibboleth IdP V3, LDAP configuration can be done in one place, in the `..\conf\ldap.properties` file. Those properties are then referenced by both the authentication component and the resolver component, configured in:

- `..\conf\authn\ldap-authn-config.xml` (which rarely needs editing)
- `..\conf\attribute-resolver.xml`

This replaces needing to configure it both in the JAAS config file (`login.config`) and the resolver in Shibboleth IdP V2.

1. Open the `..\conf\ldap.properties` file and edit/add the lines in the file that correspond to those highlighted in the image below. Note that the "schoolName" is where your school name should go.

```
## Authenticator strategy
idp.authn.LDAP.authenticator= bindSearchAuthenticator

## Connection properties ##
idp.authn.LDAP.authenticator= bindSearchAuthenticator
idp.authn.LDAP.idapURL= ldaps://adServerName.schoolName.edu:3269
idp.authn.LDAP.useStartTLS           = false
idp.authn.LDAP.useSSL                = true

# Search DN resolution, used by anonSearchAuthenticator, bindSearchAuthenticator
# for AD: CN=Users,DC=example,DC=org
idp.authn.LDAP.baseDN= DC=schoolName,DC=edu
idp.authn.LDAP.subtreeSearch         = true
idp.authn.LDAP.userFilter= (sAMAccountName={user})
# bind search configuration
# for AD: idp.authn.LDAP.bindDN=adminuser@domain.com
idp.authn.LDAP.bindDN= CN=readonly web,CN=Users,DC=schoolName DC=edu
idp.authn.LDAP.bindDNCredential= <password>
```

The highlighted items in the image above are configuration for searching the active directory for the username.

- The active directory search starts with:

```
idp.authn.LDAP.baseDN= DC=schoolName,DC=edu
```

And searches all substrings for `sAMAccountName` = the username (the student or staff authenticated with).

- The port 3269 corresponds to Global Catalog (instead of the normal port, which would correspond to LDAP).
2. Optional: To ensure the LDAP configuration is correct, add an LDAP Browser to the server and use it to configure connections to the LDAP server and test that you have the right settings:

DRAFT

- LDAP URL/servername
- Base DN
- bind account/credentials
- etc/

You may want to try the free/shareware LDAP Browser: <http://www.ldapbrowserwindows.com>.

## 2. Copy Over Signing Keys

### Copy Over Signing Keys

Installing the V3.0 IdP generates a group of certificates and keys, including:

- separate certificates for signing responses (the key cert that SPs care about), encryptions, and the backchannel port (e.g. 8443, if your IdP needed to accept an attribute query, usually just from SPs that only support SAMLv1.1)
- a user-facing cert as a placeholder (that must be replaced with a real CA-issued cert)

To avoid updating metadata with a new IdP signing key with federated partners, do the following:

1. Copy the IdP V2.0's signing key (public and private) and paste it into the `..\director\xx.xml` file to replace the one installed in the IdP V3.0 install.
2. Configure the right server cert keystore (PKCS12 format, .pfx format) to be what Jetty uses for port 443.

## 3. Update Jetty Configuration

### Update Jetty Configuration

Complete the following Jetty changes:

1. Ensure that Jetty has permissions to update files as needed, and in particular, that it can write log files to: `C:\Program Files (x86)\Shibboleth\IdP\logs, ...\metadata, ...\conf, etc.`
2. Change the Jetty setting for the certificate it uses for "user browser-facing/port 443" interactions. The "user-facing cert" is the one that is presented to the user's browser when they access port 443 on that server. The cert used is configured into Jetty, and has to be a keystore, where a .pfx keystore is perfect. That HAS to be a CA-issued cert, just like you'd use for any HTTPS web server. Wildcard certs are okay.

The keystore, and the password to unlock it, are configured in the file: `C:\Program Files (x86)\Shibboleth\IdP\jetty-base\start.d\idp.ini`

Key settings are:

- `jetty.browser.keystore.path=`
- `jetty.browser.keystore.password=`
- `jetty.browser.keystore.type= PKCS12`

By convention, all the certs/keystores are in: `C:/Program Files (x86)/Shibboleth/IdP/credentials/`

3. Make any JAVA environment changes for Jetty via the: `C:\Program Files (x86)\Shibboleth\Procrun\shib_idpw.exe` program.

Typical changes include:

- changing the memory requirements
- adding system variables (required by plugins)

These changes may not survive the IdP V2.0 re-installation, so they might need to be re-applied after the IdP V3.0 upgrade. Compare any changes you made to JAVA for IdP V2.0 to those after the IdP V3.0 upgrade as there is a chance that your configuration may have survived.

4. Restart Jetty.

# DRAFT



**Note:** When you use the MSI installer a service definition is added to the Windows Server Manager to (auto) start, stop, and restart Jetty/IdP. It's named (by default) 'Shibboleth 3 IdP Daemon' (service name: 'shibd\_idp').

## 4. Add the Open CCC Metadata

1. Download the two metadata files:
  - Production: <https://admin.openccc.net/Shibboleth.sso/Metadata>
  - QA: <https://ci.control.openccc.net/Shibboleth.sso/Metadata>
2. Save these files as "OpenCCC-production-metadata.xml" and "OpenCCC-QA-metadata.xml", respectively, in ..IdP\metadata\.

## 5. Other Certificate Configuration

### 1. Add InCommon and CCC-Wide Certifications

Copy the following certificates from your IdP V.2.0 into the IdP V3.0 ..\credentials\.. directory:

- InCommon public signing cert (used to validate the InCommon metadata feed)

(If you haven't joined InCommon already, do so at the following link: <https://www.incommon.org/join.html>,



**Note:** Your IdP supports a wide range of endpoints but it's usually better just to register the ones that you'll know may need to be used and not define the other ones to InCommon. That doesn't mean the other endpoints "won't work"--just that you don't advertise them to anyone else for now.

1. Register the idP's "signing certificate" here: C:\Program Files (x86)\Shibboleth\IdP\credentials\idp-signing.crt.
2. Confirm the property value in the ..\conf\idp.properties file for: idp.signing.cert= to determine the correct signing cert in use by your IdP. For example: C:\Program Files (x86)\Shibboleth\IdP\credentials\idp.crt.
3. Register the cert and then wait till you get acknowledgement that is ready "to reference".



**Note:** Skip registration for all of the following items:

- any Artifact endpoints
- any SSO endpoints \*other than\* the two SAMLv2 ones (POST & Redirect)
- the Attribute Service items (for backchannel calls to your IdP that used to be needed for the SAMLv1 protocol support, but almost no one needs to support these days)
- CCC-wide metadata distribution public signing cert

### 2. Configure for Handling Service Providers That Don't Provide an Encrypted SAML Response Certificate

In IdP V2, you had to create a relying-party exception to allow non-encrypted responses to SPs if the metadata available to the IdP for a given SP did not have a certificate in it that could be used for encryption.

For IdP V.3.0, use the following steps to allow non-encryption:

1. Open the ..\conf\idp.properties file.
2. Confirm that the "idp.encryption.optional" value is set to 'true' and not commented out.
3. Save and closet the idp.properties file.



**Note:** When this property is set to 'true' encryption happens whenever a key to use it can be located and requests will succeed even if encryption fails.

### 3. Update the ..\metadata\idp-metadata.xml File for Any Signing Cert Config Changes

Your IdP's own metadata (generated and stored in the file: C:\Program Files (x86)\Shibboleth\IdP\metadata\idp-metadata.xml when you install the IdP) is NOT auto-updated when you make any configuration changes to the IdP that impact the signing cert being used, the endpoints you are intending to support, etc.

# DRAFT

That file can be accessed at the URL: <https://idp.yourCollegeInitials.edu/idp/shibboleth>



**Note:** Where yourCollegeInitials = your college initials followed by "cc" (for community college).

If you want the metadata for your IdP available from that endpoint to be accurate, you need to hand-edit the `..\IdP\metadata\idp-metadata.xml` file when you make any changes to the IdP that impact the metadata. You may also want to add Contact and MDUI information to that file.

OR

If you register your IdP with InCommon, and are a designated site admin, an easy way to get a copy of your IdP's metadata (that you would share with partners who are unable to consume it out of the InCommon feed) is to download a copy of the InCommon feed, search the file for your entityID, and copy out just your IdP's metadata. See <http://md.incommon.org/InCommon/InCommon-metadata.xml>.



**Note:** This presumes that you are one of the two "designated site admins" for your campus and that you use the "InCommon Register Your IdP Wizard web interface" to register information about your IdP with InCommon. Once the InCommon staff approve your registration, they will publish it the next time they publish a new InCommon metadata aggregate (approximately once every business day.)

## 6. Configure Log Files

There are two sets of log files, logs for Jetty (i.e. the web server) and logs for the Shibboleth IdP itself. The names, rotations, number of days worth kept, etc. are set by these files:

- `..\IdP\conf\logback.xml` (configuration for the IdP logs)
- `C:\Program Files (x86)\Shibboleth\IdP\jetty-base\resources\` (for Jetty logs)
- `C:\Program Files (x86)\Shibboleth\IdP\jetty-base\logs` (Jetty logs)
- `C:\Program Files (x86)\Shibboleth\IdP\logs` (Shibboleth IdP logs)

## C. Apply CCCApply Customizations

---

Follow the instructions at the link below to add the necessary customizations for the CCCApply Standard and International applications as well as the BOG Fee Waiver.

- <https://cccnext.jira.com/wiki/display/PD/Shibboleth+V3+Customizations+for+CCCApply+Applications>

## D. Download the Active Directory (AD) Security Certificate

---

1. Download the AD security certificate from your college's certificate server and place it in: `C:\Program Files (x86)\Shibboleth\IdP\credentials\`
2. Configure the `C:\Program Files (x86)\Shibboleth\conf\ldap.properties` to use the certificate file.

## E. Rebuild the Shibboleth V3.0 IdP WAR File

---

Any changes that are made in the `..\edit-webapp\` directory require the IdP's WAR file to be rebuilt. After completing the steps above, you should be done with these edits.

Rebuild the WAR file using the new IdP V3.0 script `..\bin\build.bat`:

1. At the command prompt:
 

```
cd C:\Program Files (x86)\Shibboleth\IdP
```

2. Then:
 

```
bin\build
```

A line will print in the command shell indicating the home directory of the IdP the build is going to happen in.

DRAFT

3. Hit the Return key to proceed and complete the build. You should get a BUILD SUCCESSFUL message.

# DRAFT

## 4. Test Your Shibboleth IdP V3.0 Server

---

After you've updated your Shibboleth IdP V3.0 using the previous steps in this guide use the following steps to test your configuration.

It is assumed that you migrated the following information from IdP V2.0 to IdP V3.0 so that you are using the same information (the way you migrate is to keep everything the same as before so that your metadata doesn't change):

- entityID
- URL/hostname for your IdP
- IdP signing cert/key

The steps below provide a way for you to test your IdP V3.0 install without needing to make any changes to SPs (to start). These steps take advantage of the classic SAML V2 flow where all communication between the IdP and an SP is through your browser. One starts by testing using IdP-initiated, and then one can go to testing by starting at the actual services.

1. Ensure that the new IdP V3.0 is configured to use the same signing cert as your current IdP V2.0. Otherwise no SP will be able to verify the response that you send it using the same entityID as your current IdP.
2. Create a local hosts entry that lists the service/hostname you use for your current IdP V2.0, but with the IP address of the new IdP V3.0 server. The local host entry will override the DNS, allowing you to control the actual server the browser will go to.
  - Windows: C:\Windows\System32\drivers\etc
  - Linux/Mac: /etc/hosts

This allows you to test the IdP V3.0 with a browser on your desktop; the local hosts entry now has the IP address at which your browser will "see" the new IdP server.

3. In your browser, access any test SPs you have configured, and each and every (or at least some reasonable sampling) of SPs that your current IdP is integrated with.



**Note:** The Firefox browser with the SAML Tracer plugin added to it is a good choice.

You will use URLs of the form...

<https://myidp.campus.edu/idp/profile/SAML2/Unsolicited/SSO?providerId=https://some.sp.com/shibboleth>

...where you substitute:

- your IdP's name for *myidp.campus.edu*
- the entityID of the given SP you want to test with for *https://some.sp.com/shibboleth*

4. Open the ..IdP\conf\logback.xml file and set the idp.loglevel.messages variable to Debug in order to see information on the response recorded in the log.:

```
<variable name="idp.loglevel.messages" value="DEBUG" />
```

If you have any trouble with basic operation, or with your connectivity to LDAP, then you can adjust the following variables in the ..IdP\conf\logback.xml file to "DEBUG":

```
<variable name="idp.loglevel.idp" value="INFO" />
```

```
<variable name="idp.loglevel.ldap" value="WARN" />
```



**Note:** Your number of log entries grow substantially when you set multiple variables to DEBUG.

# DRAFT

## 5. Configure a Script and CRON Job for Your AES Key

Shibboleth IdP V3.0 has a default session-storage mechanism that uses an encrypted, client-side cookie to store data. For the basic features of supporting sessions, SAMLv2 SSO flow, and simple logout, this is sufficient and it easy to cluster the IdP. In order to encrypt the cookie stored in the user's browser, and then decrypt it when "handed back", the IdP needs a symmetric key. The key it uses is an Advanced Encryption Standard (AES) key.

It is recommended that you generate a new AES key once a day with a CRON job.

There are two files associated with that AES key: sealer.kver and sealer.jks.

File	Usage
..\IdP\credentials\sealer.kver	A simple text file that just indicates the timestamp the most recent key was generated, and an increasing number of how many have ever been generated.
..\IdP\credentials\sealer.jks	Holds the last N keys, where N defaults to 30. Everything new is always encrypted with the latest key, but the IdP can continue to decrypt anything encrypted with any one of the "still tracked" keys. Basically, it is session information that's stored in that cookie, but if you use the attribute consent, a limited amount of data might be stored in that cookie that you'd want to be able to "read back" a number of days later. (This does not impact you if you aren't using the attribute consent items.)

You can determine if the scheduled job you have set up is succeeding by checking the information in the sealer.kver and sealer.jks files.



**Note:** If you don't run multiple nodes, you can switch session storage to local server memory, and not worry about these keys.

1. Create a Windows Powershell script to generate a new key. Or, if you already have one in place for Shibboleth V2.0, copy it and edit the copy. For the purposes of the next few steps, assume the location of your script is: C:\Program Files (x86)\Shibboleth\KeyGenerator.PS1.
2. At a command prompt:
  - cd C:\Program Files (x86)\Shibboleth\IdP
  - .\bin\seckeygen.bat --storefile 'C:\Program Files (x86)\Shibboleth\IdP\credentials\sealer.jks' --storepass 'password' --versionfile 'C:\Program Files (x86)\Shibboleth\IdP\credentials\sealer.kver' --alias secret
3. Create a daily Task Scheduler job to execute it once a night. Example (assuming a :
  - a. Open Task Scheduler, Action -> Create Task to open the Create Task dialog box.
  - b. Enter values:
    - General Tab -> Name field: Generate New Shib IdP Secret Key
    - General Tab -> Description field: Generate new cookie encryption key for Shib IdPv3.
    - Triggers Tab -> New... to open the New Trigger dialog box. Select the Daily radio button and set a time (perhaps in the early morning hours).
    - Actions Tab -> New... to open the New Action dialog box.
      - Action drop-down list: Start a program
      - Program/script field: powershell
      - Add arguments (optional) field: -file KeyGenerator.PS1
      - Start in (optional) field: C:\program files (x86)\shibboleth\

DRAFT

See: <https://wiki.shibboleth.net/confluence/display/IDP30/SecretKeyManagement> for reference.



**Note:** If you have more than one IdP node, you need to run your script on one node, and then copy it to the other nodes from here. You only want to generate a new key on a single "master" node, and then have that same scheduler job copy over the new versions of the two sealer files to the other nodes. Add whatever is appropriate to the Powershell script that generates the new key in order to ensure the new versions of those two files get copied to each and every other IdP node.

# DRAFT

## Glossary of Terms

---

**Active Directory:** a directory service that Microsoft developed for Windows domain networks. It is included in most Windows Server operating systems as a set of processes and services. Initially, Active Directory was only in charge of centralized domain management.

 **Note:** AD has the capability to authenticate via LDAP.

A server running Active Directory Domain Services (AD DS) is called a domain controller. It authenticates and authorizes all users and computers in a Windows domain type network—assigning and enforcing security policies for all computers and installing or updating software.

A forest is a collection of one or more domains which may have one or more trees. What makes a forest unique is that it shares the same schema. The schema defines what and how Active Directory objects are stored.

Source: [https://en.wikipedia.org/wiki/Active\\_Directory](https://en.wikipedia.org/wiki/Active_Directory)

**Authentication:** the act of confirming the truth of an attribute of a single piece of data claimed true by an entity (in contrast with identification, which refers to the act of stating or otherwise indicating a claim purportedly attesting to a person or thing's identity, authentication is the process of actually confirming that identity).

**Authorization:** the function of specifying access rights to resources related to information security and computer security in general and to access control in particular.

**Directory Service:** A directory service or name service, maps the names of network resources to their respective network addresses. Active Directory (AD) and LDAP are two popular directory service providers. With the name service type of directory, a user does not have to remember the physical address of a network resource; providing a name locates the resource. Each resource on the network is considered an object on the directory server. Information about a particular resource is stored as attributes of that object. Information within objects can be made secure so that only users with the available permissions are able to access it. More sophisticated directories are designed with namespaces as Subscribers, Services, Devices, Entitlements, Preferences, Content and so on.

Source: [https://en.wikipedia.org/wiki/Directory\\_service](https://en.wikipedia.org/wiki/Directory_service)

**Global Catalog:** (Microsoft) The global catalog is a distributed data repository that contains a searchable, partial representation of every object in every domain in a multidomain Active Directory Domain Services (AD DS) forest. The global catalog is stored on domain controllers that have been designated as global catalog servers and is distributed through multimaster replication. Searches that are directed to the global catalog are faster because they do not involve referrals to different domain controllers.

Source: [https://technet.microsoft.com/en-us/library/cc728188\(v=ws.10\).aspx](https://technet.microsoft.com/en-us/library/cc728188(v=ws.10).aspx)

**IdP:** Identity Provider; also known as Identity Assertion Provider, is responsible for (a) providing identifiers for users looking to interact with a system, and (b) asserting to such a system that such an identifier presented by a user is known to the provider, and (c) possibly providing other information about the user that is known to the provider. This may be achieved via an authentication module which verifies a security token that can be accepted as an alternative to repeatedly explicitly authenticating a user within a security realm.

Source: [https://en.wikipedia.org/wiki/Identity\\_provider](https://en.wikipedia.org/wiki/Identity_provider)

**InCommon:** A "trust framework" that provides a scalable way for lots of metadata to be exchanged in a secure way. No actual "traffic" passes through InCommon. Instead, you "leverage" InCommon by downloading the metadata feed it provides. For example, Instructure is happy to put the metadata each college needs to interact with Canvas into that InCommon feed, so that you don't otherwise need to do anything special to get and use the Canvas metadata. See <https://www.incommon.org/join.html>.

**IdP Proxy:** A centralized Proxy to help colleges assert consistent SAML attributes to the various Service Providers within the CCC Federation. The main proxy use case is when a college is not able to send the CCCID SAML attribute for students. If the proxy discovers that the CCCID SAML attribute is not present, it will attempt to find the CCCID associated with the IDPs unique identifier (EPPN) for the student. If a CCCID is not found, the student will be

# DRAFT

redirected to OpenCCC to either recover or create a new OpenCCC account. Once the account is recovered or created, the CCCID will be cross referenced to the student's EPPN so that the next time a student attempts to enter the CCC Federation from their college IDP, the proxy will be find the students CCCID and add it to the SAML attributes presented to various CCC Federation service providers.

**LDAP:** Lightweight Directory Access Protocol; an open, vendor-neutral, industry standard application protocol for accessing and maintaining distributed directory information services over an Internet Protocol (IP) network.

Source: [https://en.wikipedia.org/wiki/Lightweight\\_Directory\\_Access\\_Protocol](https://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol)

**SAMAccountName:** The username the student or staff authenticated with.

**SAML:** Security Assertion Markup Language; an XML-based, open-standard data format for exchanging authentication and authorization data between parties, in particular, between an identity provider and a service provider. A set of specifications that encompasses the XML-format for security tokens containing assertions to pass information about a user and protocols and profiles to implement authentication and authorization scenarios.

Source: [https://en.wikipedia.org/wiki/Security\\_Assertion\\_Markup\\_Language](https://en.wikipedia.org/wiki/Security_Assertion_Markup_Language)

**Security Token:** Security tokens are used to prove one's identity electronically (as in the case of a customer trying to access their bank account). The token is used in addition to or in place of a password to prove that the customer is who they claim to be. The token acts like an electronic key to access something.

Source: [https://en.wikipedia.org/wiki/Security\\_token](https://en.wikipedia.org/wiki/Security_token)

**Service Provider:** A company that provides organizations with consulting, legal, real estate, education, communications, storage, processing, and many other services. Although the term service provider can refer to organizational sub-units, it is more generally used to refer to third party or outsourced suppliers, including telecommunications service providers (TSPs), application service providers (ASPs), storage service providers (SSPs), and Internet service providers (ISPs).

Source: [https://en.wikipedia.org/wiki/Service\\_provider](https://en.wikipedia.org/wiki/Service_provider)

**Shibboleth:** Shibboleth is a standards based, open source software package for web single sign-on across or within organizational boundaries. It allows sites to make informed authorization decisions for individual access of protected online resources in a privacy-preserving manner.

Source: <https://shibboleth.net/about/>

**Single Sign On:** A session/user authentication process that permits an user to enter one name and password in order to access multiple applications. A property of access control of multiple related, but independent software systems. With this property a user logs in with a single ID and password to gain access to a connected system or systems without using different usernames or passwords, or in some configurations seamlessly sign on at each system. This is typically accomplished using the Lightweight Directory Access Protocol (LDAP) and stored LDAP databases on servers also called directory servers.[1] A simple version of single sign-on can be achieved over IP networks using cookies but only if the sites share a common DNS parent domain.[2] For clarity it is best to refer to systems requiring authentication for each application but using the same credentials from a directory server as Directory Server Authentication and systems where a single authentication provides access to multiple applications by passing the authentication token seamlessly to configured applications as SSO. Single Sign-On is a service that is usually provided by a class of technology called Access management. And there's a good reason for that. Efficient SSO solution asks for user credentials just once. Then it needs to remember the fact that user is already authenticated. It needs session for this. That's what access management does: session management. It also needs to pass the information that the user was authenticated to the applications. And also pass some data about the identity of the authenticated user.

Source: <http://searchsecurity.techtarget.com/definition/single-sign-on>

**Software Token:** A type of two-factor authentication security device that may be used to authorize the use of computer services. Software tokens are stored on a general-purpose electronic device such as a desktop computer, laptop, PDA, or mobile phone and can be duplicated. This is in contrast to hardware tokens, where the credentials are stored on a dedicated hardware device and therefore cannot be duplicated (absent physical invasion of the device).

Source: [https://en.wikipedia.org/wiki/Software\\_token](https://en.wikipedia.org/wiki/Software_token)

DRAFT

**Testshib:** A testing service intended for new installations of Shibboleth. All SAML 2.0 implementations are welcome and may be tested against Shibboleth here. <https://www.testshib.org/>