Download/un-tar/zip Shibboleth 2.1.2

- **http://shibboleth.internet2.edu/downloads/shibboleth/idp/latest/shibboleth-identityprovider-2.1.2-bin.tar.gz**

Prepare to install shibb **(https://spaces.internet2.edu/display/SHIB2/IdPInstall)**

Prepare java for shibb **(https://spaces.internet2.edu/display/SHIB2/IdPApacheTomcatPrepare)**

- copy (unzip)/lib/shib-jce.jar -> (jre)/lib/ext
- edit (jre)/lib/security/java.security as directed (point to above file)
- set JAVA_HOME to point to base java directory (the one with a bin folder)

Prepare Tomcat for shibb

- Port 443 in (tomcat)/conf/server.xml (for recommended tomcat-only install, otherwise, this is an apache task in conjunction with mod_proxy_ajp/mod_jk). You will want to use a certificate that all of your browsers trust in this keystore since it will host your login page (see **here** for generating a tomcat cert/keystore).

```
<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="mypassword" />
```

  (Your jks filename may be different; This site will be where users hit the login page, so it will need an "public" SSL cert. If you need to import an existing key/cert to a jks, see **here** or **here**.)

- Port 8443 as per the directions (note the special config needed if running tomcat 6 on windows)
- Set tomcat to run automatically
  - Unix shell script template here - edit for your install, then place in appropriate location (/etc/init.d/tomcat):
  - **https://eco.tx-learn.net/downloads/tomcat-init-d.txt**

- Finish remaining config, including endorsed jars (from shibb distribution), JAVA_OPTS (on Windows, use tomcat GUI), and the context deployment fragment (**https://spaces.internet2.edu/display/SHIB2/IdPApacheTomcatPrepare** and **http://tomcat.apache.org/tomcat-6.0-doc/config/context.html**)

Install Shibboleth

- Customize Shibb error pages and login page located in (shibb-dist)/src/main/webapp (logo, wording, etc) *[do this first so that the resulting .war file will have your webpages as you want them]*
- Build tomcat .war file by running either "./install.sh" (unix) or "install.bat" (Windows) - you'll need hostname, use default file location

Define metadata for use with your Shibboleth IdP

- **https://spaces.internet2.edu/display/SHIB2/IdPMetadataProvider**
- Use a file-backed HTTP metadata provider. For filters, require a signature, validate the schema, and require a validUntil attribute. Optionally, you can filter out unneeded roles (other IdPs).

- Metadata URL:
  - InCommon: **http://wayf.incommonfederation.org/InCommon/InCommon-metadata.xml**

- Metadata signature validation cert:
  - InCommon: **https://wayf.incommonfederation.org/bridge/certs/incommon.pem**

Verify that shibb is running at a basic level

- Restart tomcat, then try the URL ***https://(your hostname)/idp/profile/Status*** - it should respond with 'ok'.

Register your IdP's metadata located in IDP_HOME/metadata (hostname-metadata.xml)

- Via the InCommon participant **admin interface**

Authentication: decide on UsernamePassword (JAAS) or REMOTE_USER (like the old shibb)

- for UsernamePassword, preferred (**https://spaces.internet2.edu/display/SHIB2/IdPAuthUserPass**):
  - uncomment UsernamePassword section in handler.xml
  - configure login.config for Kerberos (you'll need a keytab file) or ldap (you'll need service credentials)

- for RemoteUser (**https://spaces.internet2.edu/display/SHIB2/IdPAuthRemoteUser**):

  - protect the URL "/idp/Authn/RemoteUser" with your choice of authentication handler (CAS, etc)

Add your authentication method's handler to the DefaultRelyingParty in relying-party.xml

- **https://spaces.internet2.edu/display/SHIB2/IdPUserAuthn**
- For RemoteUser, add "urn:oasis:names:tc:SAML:2.0:ac:classes:unspecified"

- For UsernamePassword, add
  "urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport"

Review tasks so far...

- Java
- Tomcat
- Shibb
- Metadata
- Authentication

Discuss the config files

- relying-party.xml
- attribute-resolver.xml
- attribute-filter.xml
- login.config
- logging.xml
- handler.xml
- service.xml
- internal.xml

More config...

attribute-resolver.xml: uncomment LDAP attributes (except eduPersonTargetedID and the static element in eduPersonAffiliation)

- **https://spaces.internet2.edu/display/SHIB2/IdPAddAttribute**
- **Discussion on persistent IDs...**

attribute-resolver.xml: Configure your ldap connection for attributes

- **https://spaces.internet2.edu/display/SHIB2/ResolverLDAPDataConnector**
- requires a network path and a service account on your ldap server (an acct that respects FERPA restrictions)
- If you are using kerberos, you will need to split the kerberos realm out of the principal name for the ldap queries (**https://spaces.internet2.edu/display/SHIB2/ResolverRegexSplitAttributeDefinition**) – also see this related thread in the shibb-users mailing list archives:
  - **https://mail.internet2.edu/wws/arc/shibboleth-users/2008-07/msg00281.html**
- Using kerberos will also require a kerberos keytab file (usually generated on the kdc) and a krb5.conf/ini (on Windows, must be in the %SystemRoot% directory) - it's not hard, just requires some specific settings, documentation is available if you're interested.

Configure an attribute release policy in attribute-filter.xml

- **https://spaces.internet2.edu/display/SHIB2/IdPAddAttributeFilter**
- Test SP Requestor entity IDs to release attributes to are:
  - *https://narwhal.utsystem.edu/shibboleth*
- Discussion on attribute release...
  - Defining new attributes
  - done in attribute-resolver.xml
  - SAML1 attributes use readable names as their IDs
  - SAML2 attributes use OIDs with readable names as a separate XML attribute
  - See attribute-resolver.xml or wiki page for examples.
- Releasing new attributes
  - to a specific attribute requester (use it's entityID)
  - to an entire federation (see below)
  - See attribute-filter.xml or wiki page for examples.
  - A common approach for a "ReleaseCommonInfo" filter, like this:

```
<AttributeFilterPolicy id="releaseCommonInfo">
    <PolicyRequirementRule xsi:type="basic:OR">
          <basic:Rule xsi:type="basic:AttributeRequesterString"
                value="https://wwwdev.utsystem.edu/shibboleth" />
          <basic:Rule xsi:type="basic:AttributeRequesterString"
                value="https://narwhal.utsystem.edu/shibboleth" />
    </PolicyRequirementRule>

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

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

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

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

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

   <AttributeRule attributeID="eduPersonAssurance">
     <PermitValueRule xsi:type="basic:ANY" />
   </AttributeRule>
</AttributeFilterPolicy>
```

Test pages (these just dump all the headers, which shows you what you asserted):

- **https://narwhal.utsystem.edu/shibb2/dumpvars.asp** (Shibb 2.0 SP) <-- need to set this up 1st

Moving to production

Use LDAPS (SSL) for both authentication and attribute resolver

- For in-house cert on LDAP server, CA cert has to be in (jre)/lib/security/cacerts file (use java 'keytool' to add certs)

Security/cleanup/hardening

- Remove unneeded tomcat webapps from (tomcat)/webapps folder (manager, host-manager, root, examples, etc)
- comment out unneeded ports in (tomcat)/conf/server.xml (look for "Connector"): ports 8009, 8080
- consider running tomcat under a less privileged account
- possible issue with support for weak ciphers (see: **http://www.nessus.org/plugins/index.php?view=single&id=26928**)

- turn off any other unneeded ports in the operating system

Monitoring

- Have your monitoring system check the status URL (https://HOSTNAME/idp/profile/Status) for the word 'ok'

Reporting

- Achieved by writing scripts against shibb's log files (shib-error.log)
- Reporting possibilities: assertions issued per SP, successful logins, failed logins
- You can use my .NET version, if you can access your log file from a box that can run a .NET app.
- You can also do some neat things with **Orca**, like **this**.

Logging

- **https://spaces.internet2.edu/display/SHIB2/IdPLogging**
- Configured in the logging.xml config file
- Generally, leave logging level at INFO, but DEBUG can really help troubleshooting (it generates A LOT of output)
- Can change it on the fly - logging.xml is read every 5 minutes.
- Using the underlying *Logback* framework, it is possible to aggregate shibb logs to a syslog server or even to a database via JDBC.
- *Logback* also supports an SMTP appender that can email any ERROR level log messages to an administrator.
    - **https://spaces.internet2.edu/display/SHIB2/IdPProdLogging**
        - Other helpful items for logging: (**https://spaces.internet2.edu/display/SHIB2/IdPLogging**)
    - Logging authentication events (useful for reporting)
        - In *Logging.xml*:
          ```
          <logger name="edu.internet2.middleware.shibboleth.idp.authn">
             <level value="DEBUG" />
          </logger>
          ```
    - Logging events from the LDAP JAAS authentication module
        - In *Logging.xml*:
          ```
          <logger name=" edu.vt.middleware.ldap">
             <level value="DEBUG" />
          </logger>
          ```

Java (JVM) Tuning

- **https://spaces.internet2.edu/display/SHIB2/JVMTuning**
- Can improve scalability, especially important when using shibb for internal SSO across the campus.

Automatically reloading the config files

- **https://spaces.internet2.edu/display/SHIB2/IdPConfigConfig**
  ➔ add *configurationResourcePollingFrequency* to the service configuration of the attribute-filter in the service.xml config file – set it for 60 seconds (= 60000 msec)

  ```
  <Service id="shibboleth.AttributeFilterEngine"
  ```

```
        xsi:type="attribute-afp:ShibbolethAttributeFilteringEngine"
        configurationResourcePollingFrequency="60000">
                <ConfigurationResource file="/opt/shibboleth-idp/conf/attribute-
                filter.xml" xsi:type="resource:FilesystemResource" />
</Service>
```

Handling upgrades

Java

- install new java
- copy shib-jce-1.0.jar
- edit java.security
- change java home
- point tomcat at new java (Windows only)
- Restart tomcat
- test

Tomcat

- move/remove old tomcat (make copy of config)
- install new tomcat
- Set:
- JAVA_OPTS (windows)
- port 443
- port 8443
- endorsed jar files from shibb
- context deployment fragment
- Check/set service to run automatically on boot

Shibboleth

- unpack distribution
- copy/customize web/error pages
- run install.sh/bat (choose to preserve config)
- if location changed, update tomcat's port 443/8443 config and the context deployment fragment file
- check shibb wiki for any necessary changes to config files as a result of the upgrade (like this: https://spaces.internet2.edu/display/SHIB2/IdP2021Upgrade)
- make sure that the shibb-related files in tomcat's *endorsed* directory are still valid/current

Support Resources

Shibboleth Wiki site

- https://spaces.internet2.edu/display/SHIB2

Shibboleth-Users mailing list (one of the best supported lists ever, though it can be a bit busy at times)

- **http://shibboleth.internet2.edu/lists.html**

Advanced Topics

Multi-federation/local “federation”

- Add the various federations' metadata to your chaining metadata provider in relying-party.xml
- Avoid having your IdP's metadata look different for different federations/metadata groups

Single release policy for entire federation

- Use a PolicyRequirementRule, inside an AttributeFilterPolicy in your attribute-filter.xml that looks like this:

```
<AttributeFilterPolicy>
  <PolicyRequirementRule xsi:type="basic:OR">
      <basic:Rule xsi:type="saml:AttributeRequesterInEntityGroup"
              groupID="urn:mace:incommon" />
  </PolicyRequirementRule>
  <AttributeRule attributeID="givenName">
     <PermitValueRule xsi:type="basic:ANY" />
  </AttributeRule>
</AttributeFilterPolicy>
```

Metadata filtering

- See "Entity Role WhiteList Filter" here: **https://spaces.internet2.edu/display/SHIB2/IdPMetadataProvider**
- This might be good to do in a large federation with a large metadata file (since the metadata file sits in memory and could impact performance).

NameID

- Represents the "subject" of a transaction.
- Can be an issue when inter-operating with commercial SAML products that expect a non-transient NameID (something shibb originally avoided to preserve privacy)
- **https://spaces.internet2.edu/display/SHIB2/IdPNameIdentifier**

Advanced attribute handling

- Script (**https://spaces.internet2.edu/display/SHIB2/ResolverScriptAttributeDefinition**)
- RegEx split (**https://spaces.internet2.edu/display/SHIB2/ResolverRegexSplitAttributeDefinition**)
- Mapped (**https://spaces.internet2.edu/display/SHIB2/ResolverMappedAttributeDefinition**)
- Template (**https://spaces.internet2.edu/display/SHIB2/ResolverTemplateAttributeDefinition**)

Asserting binary data

- Useful for using shibb to assert binary attributes (byte arrays) like userCertificate or jpegPhoto
- Use the Base64 attribute encoder in the attribute definition in attribute-resolver.xml (you'll probably need to use both the SAML1 and SAML2 decoders).
- SAML1: **https://spaces.internet2.edu/display/SHIB2/SAML1Base64AttributeEncoder**
- SAML2: **https://spaces.internet2.edu/display/SHIB2/SAML2Base64AttributeEncoder**

Load balancing

- **https://spaces.internet2.edu/display/SHIB2/IdPClusterIntro**
- Uses Terracotta
- If you only need redundancy, an active/passive setup is much easier to build using heartbeat and rsync

eduPersontargetedID implementation

- Conceived to provide a different permanent, unique ID for each user to each SP they interact with.
- Preserves privacy, yet is still traceable for audit/security purposes, though, for some applications, the privacy feature is not necessarily good and may require "affiliations" of SPs (the multiple attribute authority problem).
- Requires a database to hold values (not LDAP).
- Easily supported by shibb, but can be a challenge to provision.
- The “StoredID Data Connector” is the best approach:
  - **https://spaces.internet2.edu/display/SHIB2/ResolverStoredIDDataConnector**
- The above approach requires a database, however. The older simpler approach is avail, but has drawbacks (and is technically deprecated):
  - **https://spaces.internet2.edu/display/SHIB2/ResolverComputedIDDataConnector**

Things to watch in the future...

Microsoft CardSpace

- **http://en.wikipedia.org/wiki/Windows_CardSpace**

Inter-federation, or federation peering

- **http://middleware.internet2.edu/fedsoup/docs/soup-final.pdf**

Dynamic metadata, or metadata discovery

- **http://www.computer.org/portal/pages/security/2008/n2/bsi.xml**

Attribute aggregation, or how to deal with multiple attribute authorities

- **http://sec.cs.kent.ac.uk/shintau/**

SP Installation/Configuration

- How it works/components
  - Web server plugin
  - Daemon/service
  - Session Mgmt
  - The role of PKI

- Download package (RPM)

- Install RPMs (https://spaces.internet2.edu/display/SHIB2/NativeSPLinuxRPMInstall)

- Where it puts everything
  - /etc/shibboleth
  - /usr/sbin/shibd
  - /var/log/shibboleth/shibd.log
  - /var/log/shibboleth/transaction.log
  - /var/log/httpd/native.log (mod_shib)
  - /usr/lib/shibboleth
  - /etc/httpd/conf.d/shib.conf

- Check status: https://localhost/Shibboleth.sso/Status (must be on localhost or edit ACL)

- **Shibb Config** (https://spaces.internet2.edu/display/SHIB2/NativeSPShibbolethXML):
  - RequestMap (https://spaces.internet2.edu/display/SHIB2/NativeSPRequestMap)
    - Host(s)
    - Path(s)
      - Complex paths not allowed (Path name="/this/that")
  - EntityID, homeURL (ApplicationDefaults/Sessions)
    - https://spaces.internet2.edu/display/SHIB2/NativeSPApplication
      - use URL for entity ID
      - handlerSSL="true"
      - cookieProps= "; path=/; secure"

  - SessionInitiators (https://spaces.internet2.edu/display/SHIB2/NativeSPSessionInitiator)
    - InCommon WAYF (default)
      - ***https://wayf.incommonfederation.org/InCommon/WAYF***
    - Local IdP

  - Metadata (https://spaces.internet2.edu/display/SHIB2/NativeSPMetadataProvider)
    - Filter for: signature, RequireValidUntil
      - https://spaces.internet2.edu/display/SHIB2/NativeSPMetadataFilter

  - Attribute-map.xml (uncomment LDAP attributes)
    - https://spaces.internet2.edu/display/SHIB2/NativeSPAddAttribute

  - Administrator email/Error pages

    - https://spaces.internet2.edu/display/SHIB2/NativeSPErrors

- **Apache Config** (https://spaces.internet2.edu/display/SHIB2/NativeSPApacheConfig):
  - Need <Location> element for secure URLs to activate mod_shib
  - Headers vs. Env Vars (ShibUseHeaders)
  - UseCanonicalName On
  - ServerName idp.foo.edu

- **Advanced topics:**
  - Authorization ACLs (https://spaces.internet2.edu/display/SHIB2/NativeSPProtectContent)
    - To use an external file, add this inside an appropriate Path element:
      *<AccessControlProvider path="/etc/shibboleth/shibacl.xml"
      type="XML"/>*

  - Application Override (https://spaces.internet2.edu/display/SHIB2/NativeSPApplication)
    - Used to allow for unique settings on an app-by-app basis
    - Use only if you must, minimally:
      <Host name="other.university.org"
      applicationId="other-app"
      authType="shibboleth"
      requireSession="true"/>

      <ApplicationOverride id="other-app"
        - *entityID=https://other.university.org/shibboleth/*

  - WAYF bypass / different WAYF
    - See *SessionInitiator* example for this in shibboleth2.xml
    - You can use multiple IdP-direct SessionInitiators to build your own WAYF without deploying the WAYF software

  - Attribute filtering (https://spaces.internet2.edu/display/SHIB2/NativeSPAttributeFilter)
    - Controlled vocabulary (eduPersonAffiliation)
    - Limit entitlement values to specific IdPs

  - Virtual hosts (https://spaces.internet2.edu/display/SHIB2/NativeSPRequestMap)

  - Metadata BlackListing (https://spaces.internet2.edu/display/SHIB2/NativeSPMetadataFilter)
    - If you want to keep protect network out of a particular SP

  - Logging
    - See *.logger files in /etc/shibboleth
    - Uses log4cpp/log4shib
    - defaults are typically just fine
    - turn up to DEBUG for troubleshooting
    - restart shibd for changes to be seen

- Clustering (https://spaces.internet2.edu/display/SHIB2/NativeSPClustering)

- LoadBalancer/SSL (https://spaces.internet2.edu/display/SHIB2/NativeSPNoSSL)

- Reporting (mainly from transaction.log)
  - Which apps are being used?
  - Which IdPs are asserting?