Delegating Authentication on a Shibboleth IdP

Jon Agland, Jisc

Last updated: 10th June 2020

Table of Contents

Acknowledgements ***4***

Changelog ***4***

Background ***4***

Advantages 4

Disadvantages 4

Support ***5***

How it works ***5***

Considerations ***6***

Attribute Resolution and LDAP 6

Metadata consumption 6

entityID and Metadata from other SAML IdP products 6

Microsoft ADFS 6

NetIQ Access Manager 7

Microsoft Azure 7

Preparing the Shibboleth IdP host ***7***

Installing Apache and Tomcat 7

Configuring Tomcat 7

Preparing certificate file(s) 8

Re-Configure Tomcat 8

Configure Apache 9

Configure the Shibboleth SP 11

Re-create trust certificates 11

Modify shibboleth2.xml 11

Consume the Other IdPs metadata 11

Modify attribute-map.xml 12

Testing 12

Configure Other IdP ***13***

Configure AD FS 13

Consume the SP metadata 13

Configure the RelyingParty Trust and Attribute Issuance Policy 13

Configure Azure IdP 14

Configure NetIQ Access manager 14

Set up the LDAP to SAML attribute mapping 14

Setting up the Shib SP to NAM IdP Trust 14

Import the encryption and signing certificates from the Shibboleth SP 15

Complete the configuration on the Shibboleth IdP host 16

Test that Other IdP works 16

Enable protection of the RemoteUser Location in Apache 16

Reconfigure the Shibboleth IdP 16

Testing 17

Conclusion 18

Acknowledgements

- Simon Palmer of Coleg Sir Gar for assisting in testing and documenting with NetIQ Access Manager
- Matthew Slowe of Jisc for testing and documenting with Microsoft Azure

Changelog

Version	Modification	Author	Date
0.1	Initial draft version	Jon Agland	26th October 2018
0.2	First published version	Jon Agland	9th June 2020

Background

Universities and Colleges operating within the UK federation often have multiple Identity Providers within their organisations, and we are frequently asked about the Interoperability of SAML Identity Provider (IdP) products such as Microsoft Azure, Microsoft AD FS (Active Directory Federation Services) and Net IQ Access Manager (NAM). Unfortunately, many of these IdPs cannot inter-operate within a multi-lateral mesh federation such as the UK federation (and by extension eduGAIN). However, the scope of this documentation is not to talk about those issues, but about a common workaround to the problem, which is to delegate authentication from an IdP that can inter-operate within the UK federation to another other IdP that cannot.

There are advantages and disadvantages to delegated authentication.

Advantages

1. A true single sign-on experience for your end-users
2. Ability to leverage functionality available in the other IdP (e.g. within Azure, the Azure Multi-Factor Authentication solution and some aspects of conditional access), without requiring additional support within the Shibboleth IdP

Disadvantages

1. A more complex infrastructure that utilises multiple IdPs and a SAML SP, potentially has more single points of failure and is more difficult to ex
2. Users loss of experience of the authentication flow at the IdP that is participating in the UK federation
    a. Loss of Metadata User Information (mdui) from the SP metadata, this can be mitigated by using the Consent module in the Shibboleth IdP.
    b. Loss of control of elements of the process such as Multi-Factor Authentication, where this would either be on or off for all federated services, based on the Other IdPs configuration.

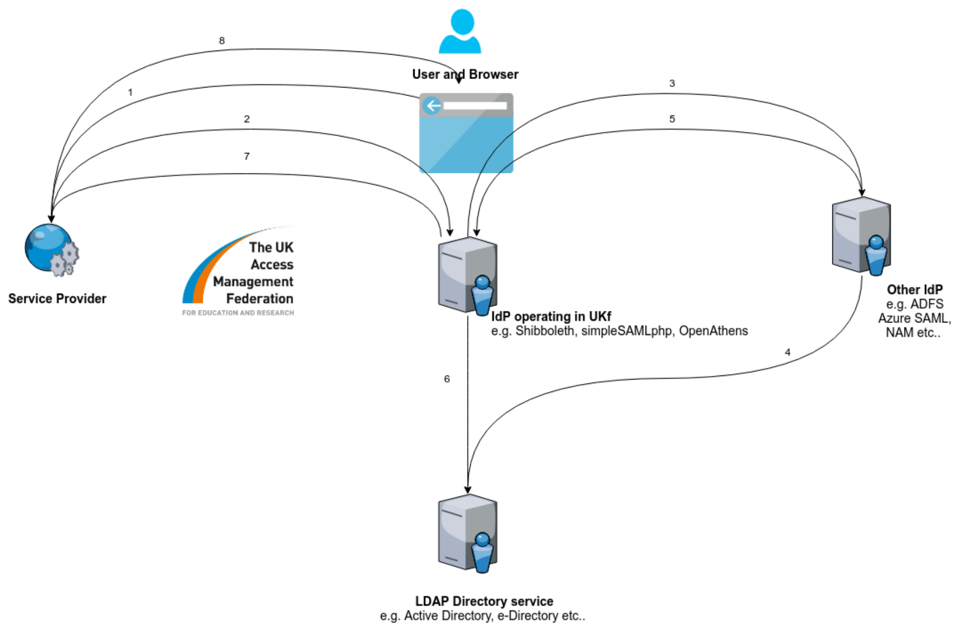
This documentation exclusively covers providing delegation authentication on a Shibboleth IdP, and assumes the deployer is providing this themselves, there are alternative solutions to this problem;

- OpenAthens (also part of Jisc), provide connectors within their OpenAthens hosted IdP service which can be used to delegate authentication.
- Overt Software Solutions, can provide a Shibboleth ADFS bridge, which provides similar delegated functionality.
- Other third parties may also offer similar services and solutions.

Support

This document is provided as-is with no warranty.   If you require support from Jisc on this matter, then this will only be provided via our Jisc Trust and Identity consulltancy service, there is no free support from the UK federation support desk for this guide or any configurations derived from it.

How it works


User and Browser
Service Provider
The UK Access Management Federation
FOR EDUCATION AND RESEARCH
IdP operating in UKf
e.g. Shibboleth, simpleSAMLphp, OpenAthens
Other IdP
e.g. ADFS
Azure SAML,
NAM etc..
LDAP Directory service
e.g. Active Directory, e-Directory etc..
1
2
3
4
5
6
7
8

1. User accesses a web resource which is part of the UK federation (they may have to selects their Organisation via a Discovery service, which is not displayed here)
2. Users browser is redirected to the Shibboleth IdP (by a SAML 2.0 AuthnRequest), this step is invisible to the user.
3. Users browser is redirected to the Other IdP (by a SAML 2.0 AuthnRequest), user is presented with the authentication page for the Other IdP.
4. User authenticates at the IdP, and this is looked up against LDAP Directory service, and the Other IdP performs any other authorisation it requires e.g. MFA
5. Assuming user successfully authenticates, a SAML assertion is issued by the other to the Shibboleth SAML IdP (our built-in SP), it contains an attribute identifying the user (e.g. `sAMAccountName`)
6. Shibbolelth IdP takes the attribute identity the user (e.g. `sAMAccountName`) and uses this to match it within the LDAP Directory service
7. A SAML assertion is issued by the Shibboleth IdP to the SP (which operates in the UK federation), it contains attributes based on the policy configured e.g. `eduPersonTargetedID`, `eduPersonScopedAffiliation`.
8. Assuming user is authorised by the SP, they are granted access to the resource.

Note: LDAP Directory service for each IdP could be separate or replicas, but the attribute identifying the user must contain the same value

Considerations

This documentation covers deploying this against Shibboleth IdP v3.4.6, we know this will work against earlier versions, but you are advised at this point (June 2020) to be running v3.4.6 or v4. We have not at this point tested against IdP v4, whilst this should work as-is against IdP v4, there will be better methods to integrate with v4 via (this) that remains untested by us.

The instructions assume you are using Tomcat as your Java web container, you maybe able to use Jetty, but you as it doesn't support AJP you would need to proxy using http(s) differently, and it may not carry the REMOTE_USER attribute or header in the same way.

You will need to use Apache webserver to serve the IdP front-channel via HTTPS (TCP port 443), so that the RemoteUser handler path can be protected by the Shibboleth SP

The IdP back-channel which uses SOAP (TCP port 8443) could either move to Apache, or continue to operate using the Java web container.

The instructions assume you are using CentOS or RedHat Enterprise Linux, it should be possible to complete this with other Linux distributions. However, for deployments on Windows we expect the steps to be significantly different.

In terms of the other IdP, this has been tested against Microsoft AD FS, Microsoft Azure and Net IQ Access Manager.

Attribute Resolution and LDAP

The instructions assume that Attribute Resolution will continue to use an LDAP connection, therefore this is unlikely to be a suitable solution if you want to use a service such as Azure Active Directory exclusively. If you have an on-site Active Directory alongside this, or Azure Active Directory Domain Services, then it should be possible to deploy this solution

Metadata consumption

This process involves the consumption of metadata by the other IdP of the Shibboleth SP and visa-versa. There are two methods to consuming the metadata this can either be;

- Dynamic, in which case the metadata URL is used, and each routinely fetches the metadata. However, it is possible that if a change takes place that it may not be picked-up on right away, and also there is potential scope for a man in the middle attack if signature checking is not used. Whilst using https for the metadata URL is a mitigation, it does not completely rule out this possibility.
- Static, in which case the metadata is downloaded from the metadata URL once by the operator and loaded statically from the file, this removes any scope for a man-in-the-middle attack, but means this process will need to be repeated when any changes are made that may have affected the metadata, such as a certificate rollover, which is an automatic process e.g. in AD FS.

Neither is ideal, and we have observed scenarios in both that can result in a breakage for either party.

Note: Microsoft Azure IdP offers a signed metadata file, but as the signature certificate with the same certificate as used for signing, then it doesn't resolve any issues related to man in the middle or dynamic consumption.

entityID and Metadata from other SAML IdP products

These are examples of the entityID and Metadata URLs for each of the other SAML IdP products. You will need to determine what yours are at a later stage.

Microsoft ADFS

entityID :

```
http://adfs.example.ac.uk/adfs/services/trust
```

Metadata URL

```
https://adfs.example.ac.uk/FederationMetadata/2007-06/FederationMetadata.xml
```

NetIQ Access Manager

entityID:

```
https://nam.example.ac.uk/nidp/saml2/metadata
```

Metadata URL:

```
https://nam.example.ac.uk:8443/nidp/saml2/metadata
```

Microsoft Azure

(You may not know these until you have created Non-Gallery Application in Azure)

entityID:

```
https://sts.windows.net/e410798b-70c5-4083-9d79-3d612ad85bb5/
```

Metadata URL:

```
https://login.microsoftonline.com/e410798b-70c5-4083-9d79-
3d612ad85bb5/federationmetadata/2007-06/federationmetadata.xml?appid=1474fdab-ded0-
4d3e-b95f-499a804aaf20
```

Preparing the Shibboleth IdP host

There is significant preparation required to the IdP host, this would be service affecting, so should not be attempted on a production IdP.   There should be zero disruption to the other IdP.

Installing Apache and Tomcat

Install from source or from the linux distributions software repository

```
yum install mod_ssl mod_security httpd
```

For those using CentOS or Redhat Enterprise

```
cd /etc/yum.repos.d
curl -O
http://download.opensuse.org/repositories/security://shibboleth/CentOS_7/security:shi
bboleth.repo
yum install shibboleth
```

If Tomcat is already in-use then you need to comment out the existing connector serving the IdP front-channel (TCP port 443) and the IdP back-channel SOAP (TCP port 8443)

Configuring Tomcat

If you are configuring Tomcat from scratch or migrating from Jetty then you may need to configure the following.

Add to `/etc/tomcat/tomcat.conf`

```
CATALINA_OPTS="-server -Xmx1G -Didp.home=/opt/shibboleth-idp -
Djdk.tls.ephemeralDHKeySize=2048"
```

Create `/usr/share/tomcat/conf/Catalina/localhost/idp.xml` with the following contents

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

Permissions will need to be set on the `/opt/shibboleth-idp` appropriately for tomcat to be able to operate

Preparing certificate file(s)

Jetty and Tomcat can both can use a PKCS12 file for the IdP credentials, but Apache will need PEM files with no password(s), if you don’t have these to hand you may need to extract them from the PKCS12.

Retrieve the PKCS12 password from the existing configuration

Example for Tomcat

```
grep -i keystorePass /etc/tomcat/server.xml
```

Example for Jetty

```
grep -i jetty.sslContext.keyStorePassword /opt/jetty-idp/start.d/idp.ini
```

Warning: Check carefully the filenames before continuing to ensure that you do not overwrite a file or certicate that is required

Identify the PKCS12 file concerned in this example we’ve used `idp-browser.p12`, with `idp-browser.tmp` as a temporary file, and `idp-browser.crt` and `idp-browser.key` as a destination file(s) for the public and private key respectively

```
cd /opt/shibboleth-idp/credentials
```

extract the public key

```
openssl pkcs12 -in idp-browser.p12 -out idp-browser.tmp -nodes -nokeys
```

remove the password from the public key

```
openssl x509 -in idp-browser.tmp -out idp-browser.crt
```

extract the private key

```
openssl pkcs12 -in idp-browser.p12 -out idp-browser.tmp
```

remove the password from the private key

```
openssl rsa -in idp-browser.tmp -out idp-browser.key
```

remove the temporary file

```
rm idp-browser.tmp
```

Re-Configure Tomcat

Your connector for ajp on 8009, should have tomcatAuthentication="false" set

```
<Connector port="8009" protocol="AJP/1.3" tomcatAuthentication="false"
redirectPort="8443" />
```

Following a change you need to restart tomcat

```
systemctl restart tomcat
```

Configure Apache

Add the following to `/etc/httpd/conf.d/ssl.conf`

```
Listen 443 https
SSLPassPhraseDialog exec:/usr/libexec/httpd-ssl-pass-dialog
SSLSessionCache         shmcb:/run/httpd/sslcache(512000)
SSLSessionCacheTimeout  300
SSLRandomSeed startup file:/dev/urandom  256
SSLRandomSeed connect builtin
SSLCryptoDevice builtin
SSLUseStapling          on
SSLStaplingResponderTimeout 5
SSLStaplingReturnResponderErrors off
SSLStaplingCache        shmcb:/var/run/ocsp(128000)

# HSTS (mod_headers is required) (15768000 seconds = 6 months)
Header always set Strict-Transport-Security "max-age=15768000"

<VirtualHost _default_:443>
ServerName idp.example.ac.uk
ErrorLog logs/ssl_error_log
TransferLog logs/ssl_access_log
LogLevel warn
SSLEngine on
SSLProtocol             all -SSLv3 -TLSv1 -TLSv1.1
SSLCipherSuite          ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
AES256-GCM-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-
AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-
SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-
AES128-SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-
RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-
SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA:!DSS:!3DES
SSLHonorCipherOrder     on
SSLCompression          off
SSLCertificateFile /opt/shibboleth-idp/credentials/idp-browser.crt
SSLCertificateKeyFile /opt/shibboleth-idp/credentials/idp-browser.key
SSLCertificateChainFile /opt/shibboleth-idp/credentials/idp-browser-chain.crt
ProxyPass /idp ajp://localhost:8009/idp
<Files ~ "\.(cgi|shtml|phtml|php3?)$">
    SSLOptions +StdEnvVars
</Files>

<Directory "/var/www/cgi-bin">
    SSLOptions +StdEnvVars
</Directory>
BrowserMatch "MSIE [2-5]" \
     nokeepalive ssl-unclean-shutdown \
     downgrade-1.0 force-response-1.0
CustomLog logs/ssl_request_log \
         "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
</VirtualHost>
```

Add the following to `/etc/httpd/conf.d/ssl-backchannel.conf`

```
Listen 8443
```

```
# :8443 configuration for Shibboleth IdP backchannel

<VirtualHost _default_:8443>
ServerName idp.example.ac.uk:8443
ServerAlias idp.example.ac.uk
SSLEngine on
SSLProtocol             all -SSLv3
SSLCipherSuite          ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
AES256-GCM-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-
AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-
SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-
AES128-SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-
RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-
SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA:!DSS
SSLHonorCipherOrder     on
SSLCompression          off
SSLOptions              -StdEnvVars +ExportCertData
# The next two directives ensure that the port 8443 certificate is verified by
Shibboleth.
SSLVerifyClient optional_no_ca
SSLVerifyDepth 10
SSLUseStapling          off
# Trust-fabric certificate config
#
SSLCertificateFile /usr/local/shibboleth-idp/credentials/idp-backchannel.crt
SSLCertificateKeyFile /usr/local/shibboleth-idp/credentials/idp-backchannel.key
ProxyPass /idp ajp://localhost:8009/idp
ErrorLog logs/ssl_8443_error_log
TransferLog logs/ssl_8443_access_log
LogLevel warn
CustomLog logs/ssl_request_log "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"

</VirtualHost>
```

Add to httpd.conf

Not required for operation of the IdP, but it means the tools in /bin such as aacli.sh and status.sh will continue to work.

```
ServerName idp.example.ac.uk

ProxyPass /idp ajp://localhost:8009/idp
```

Configure the Shibboleth SP

Re-create trust certificates

This section is an optional step; we re-create the certificates for a longer lifetime and to provide the correct FQDN/entityID (neither of these are technically required, but it may aid interoperability with the Other IdP)

For Shibboleth SP v3 there is a pair of certificate(s), one for signing and one for encryption, so we remove those and re-create

```
cd /etc/shibboleth
rm sp-encrypt-cert.pem && rm sp-encrypt-key.pem
rm sp-signing-cert.pem && rm sp-signing-key.pem
./keygen.sh -u shibd -g shibd -h idp.example.ac.uk -y 20 -e
https://idp.example.ac.uk/sp -n sp-encrypt
./keygen.sh -u shibd -g shibd -h idp.example.ac.uk -y 20 -e
https://idp.example.ac.uk/sp -n sp-signing
```

Modify shibboleth2.xml

We configure an entityID for the Shibboleth SP running on our Shibboleth IdP host.

`https://idp.example.ac.uk/sp` is the entityID we have picked for our SP running on our IdP

You may need the _AJP prefix that on attributes such as `REMOTE_USER` for it to be passed to the "idp" application running in Tomcat. If you are neither using Tomcat nor AJP this may differ

sAMAccountName is both the attribute that the Other IdP will release to this SP, and also the attribute used as part of the `idp.authn.LDAP.userFilter` and `idp.attribute.resolver.LDAP.searchFilter` in the Shibboleth IdP's `` `ldap.properties` `` file

```
<ApplicationDefaults entityID="https://idp.example.ac.uk/sp"
REMOTE_USER="sAMAccountName"/>
```

In some cases, you may need to add the attributePrefix for the attribute to be passed by Tomcat to the Java web application I.e. the Shibboleth IdP

```
<ApplicationDefaults entityID="https://idp.example.ac.uk/sp"
attributePrefix="_AJP" REMOTE_USER="sAMAccountName"/>
```

We set the entityID of the Other IdP here (check the section on entityID and Metadata URLs), chances are we only need SAML2 enabled to interoperate with it, so we exclude SAML1 from the SSO parameter (this is now the default in Shibboleth SP 3.x).

```
<SSO entityID="http://adfs.example.ac.uk/adfs/services/trust">SAML2</SSO>
```

Note: For Azure IdP, you may not know your entityID at this stage, and may need to come back to this later.

Consume the Other IdPs metadata

We must now configure the SP to consume the Other IdPs metadata. This can be done one of two ways, and is discussed in the section on Metadata consumption, you will need the metadata URL from the well-known location.

1. Configure the SP to consume the metadata dynamically from the IdPs well-known location by adding the following to shibboleth2.xml

```
<MetadataProvider type="XML" uri="https://adfs.example.ac.uk/FederationMetadata/2007-
06/FederationMetadata.xml"
```

```
            backingFilePath="/etc/shibboleth/adfs-metadata.xml"
reloadInterval="14400">

        </MetadataProvider>
```

2. Consume the metadata statically from the IdPs well-known locationa and configure the SP to consume the static file.

```
curl https://adfs.example.ac.uk/FederationMetadata/2007-06/FederationMetadata.xml  >
/etc/shibboleth2/other-idp-metadata.xml
```

Add the following to shibboleth2.xml

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

Modify attribute-map.xml

You may match based on other attributes, but this relates directly to your IdP Filter configuration. Typically we will use `sAMAccountName` in Windows environments because they have their search and user filters set similar to the following (extract from the Shibboleth IdPs `ldap.properties`)

```
idp.authn.LDAP.userFilter                       = (sAMAccountName={user})
idp.attribute.resolver.LDAP.searchFilter        =
(sAMAccountName=$resolutionContext.principal)
```

For ADFS you will need to add the attribute definition for `sAMAccountName`

```
<Attribute
name="http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountname"
id="sAMAccountName" />
```

For Azure you will need to add the attribute definition for `sAMAccountName`

```
<Attribute
name="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/sAMAccountName"
id="sAMAccountName"/>
```

For NAM we used `cn`, the attribute definition exists as an example, we simply need to uncomment it. However, the line is provided here for reference;

```
<Attribute name="urn:oid:2.5.4.3" id="cn"/>
```

Testing

Restart the shibd service;

```
service shibd start
```

Check the Shibboleth SP Metadata and that it reflects the configuration of the SP by downloading the file and comparing with the above changed e.g. entityID, Certificates

```
curl https://idp.example.ac.uk/Shibboleth.sso/Metadata
```

Configure Other IdP

Configure AD FS

Consume the SP metadata

1. Run PowerShell as administrator
2. Issue the following command to consume the Service Provider (RelyingParty) metadata

```
Add-AdfsRelyingPartyTrust -MetadataURL
https://idp.example.ac.uk/Shibboleth.sso/Metadata
```

3. It will then prompt you to supply the following info;

   ```
   Name: Shibboleth IdP via Shibboleth SP on idp.example.ac.uk
   ```

4. There might be some warnings related to endpoints because the bindings are not supported, these can be ignored, because the binding we need only `urn:oasis:names:tc:SAML:2.0:bindings:POST` which is supported by AD FS.

Configure the RelyingParty Trust and Attribute Issuance Policy

1. Within the AD FS snap-in find your SP listed under RelyingParty Trusts
2. “Edit Access Control Policy”
3. Select the appropriate one in AD FS (you might choose ‘Permit everyone’ to start, but at a later stage choose ‘Permit everyone and required MFA’, of course you may have custom access control policies in place)
4. Next we need to release an attribute/issue a claim to this SP.
   - Click Edit Claim Issuance Policy.
   - Choose Add Rule.
   - On the claims rules template, choose Send LDAP Attributes as Claims, a
   - Enter a claim rule name, in this case we are using “`sAMAcccountName`”,
   - Choose Attributes Store, “Active Directory”
   - Under LDAP Attribute choose SAM-Account-Name
   - Under Outgoing Claim Type choose “Windows account name”
   - Click OK

Configure Azure IdP

1. Log into portal and find "Enterprise applications"
    a. Select "New Application"
    b. Select "Non-gallery application"
    c. Enter a name for the application such as "`Shibboleth IdP via Shibboleth SP on idp.example.ac.uk`"
    d. When app is created, find it in the list and go to "Properties"
        i. Set "User assignment required" to "No"
        ii. Set "Visible" to "No"
        iii. Select "Single sign-on" on the left and then "SAML"
    e. Either enter the details from the Shibboleth SP metadata (download from `https://idp.example.ac.uk/Shibboleth.sso/Metadata`) manually in the "Basic Details" section or use the "Upload metadata" button.
2. Get the correct IdP metadata from Azure for this entity and put into the Shibboleth SP configuration per the section “Consume the Other IdPs metadata” above.
3. Add an extra claim to release a SAML attribute as follows;
    a. Name `sAMAccountName`
    b. Namespace `http://schemas.xmlsoap.org/ws/2005/05/identity/claims/sAMAccountName`
    c. Source Attribute `user.onpremesissamaccountname`

Configure NetIQ Access manager

Set up the LDAP to SAML attribute mapping

1. From within your NetIQ Access Manager create an Attribute Set.
    a. Identity Servers
    b. Shared settings
    c. Attribute Sets
    d. New
    e. Provide a Name and hit Next
2. Within the Attribute Set, create a new Attribute Mapping
    a. Under the Mapping Tab choose New
    b. Complete details as follows
        i. Local Attribute: `LDAP Attribute:cn`
        ii. Remote Attribute: `urn:oid:2.5.4.3`
        iii. Remote namespace: none
        iv. Remote format: uri
        v. Attribute value encoding: Special characters encoded

Setting up the Shib SP to NAM IdP Trust

1. From within your NetIQ Access Manager
    a. Devices
    b. IDCluster
    c. SAML 2.0
2. Under Trust Providers, choose New and Enter the following details
    a. Metadata import from uri: `https://idp.example.ac.uk/Shibboleth.sso/Metadata`
    b. Attributes: choose the Attribute Set created above

Import the encryption and signing certificates from the Shibboleth SP

If your Shibboleth SP is v2 or was upgraded from v2 you may only have a single certificate for both encryption and signing, we assume here that you are using v3 where you have separate certificates for encryption and signing.

1. From within your NetIQ Access Manager, choose
    a. Security
    b. Trusted Roots
2. Choose Import
    a. Give the certificate a name “Shibboleth SP on idp.example.ac.uk - Signing”
    b. Paste the contents of `/etc/shibboleth/sp-signing-cert.pem` (or extract from the SAML metadata file for the Shibboleth SP)
3. Choose Import
    a. Give the certificate a name “Shibboleth SP on idp.example.ac.uk - Encyption”
    b. Paste the contents of `/etc/shibboleth/sp-encrypt-cert.pem` (or extract from the SAML metadata file for the Shibboleth SP)
4. For each of the certificates just imported
    a. Click “Add Trusted Roots to Trust Stores”
    b. Add to: “OCSP Trust Store”
    c. Add to: “Trust Store”

Complete the configuration on the Shibboleth IdP host

Test that Other IdP works

Contruct a WAYFLess URL to test with, this loads the "/Shibboleth.sso/Session" page after authentication. In the following example, you should change the first part of the URL and add the IdP entityID (discussed in the section on

```
https://idp.example.ac.uk/Shibboleth.sso/Login?entityid=IdPentityID&target=%2FShibbol
eth.sso%2FSession
```

For example;

```
https://idp.example.ac.uk/Shibboleth.sso/Login?entityid=http://adfs.example.ac.uk/adf
s/services/trust&target=%2FShibboleth.sso%2FSession
```

This example output for a Session from an AD FS IdP on our Shibboleth SPs session handler.

```
Miscellaneous
Session Expiration (barring inactivity): 480 minute(s)
SSO Protocol: urn:oasis:names:tc:SAML:2.0:protocol
Identity Provider: http://adfs.example.ac.uk/adfs/services/trust
Authentication Time: 2018-06-20T14:57:53.373Z
Authentication Context Class:
urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
Authentication Context Decl: (none)
Client Address: 192.168.1.1

Attributes
sAMAccountName: 1 value(s)
```

Enable protection of the RemoteUser Location in Apache

Add the following to relevant VirtualHost of your Apache configuration, this protects the Shibboleth IdP's RemoteUser Handler/Location with the Shibboleth SP

```
<Location /idp/Authn/RemoteUser>
  AuthType shibboleth
  ShibRequestSetting requireSession 1
  require shib-session
</Location>
```

Reconfigure the Shibboleth IdP

In idp.properties and change the following;

```
idp.authn.flows= RemoteUser
```

In idp.properties and change the following;

```
idp.authn.flows.initial = RemoteUser
```

Testing

Check Apache config and restart

```
apachectl configtest
```

Stop Jetty service and Restart Tomcat and Apache / HTTPd (service affecting)

```
systemctl restart tomcat && systemctl start httpd
```

Test the IdP to check it is operational again e.g. via https://test.ukfederation.org.uk, once it is proceed to the next step

Enable Apache / HTTPd

```
systemctl enable httpd && service httpd restart
```

Conclusion

Your Shibboleth Identity Provider should now be delegating via the Shibboleth Service Provider software and the authentication page your other IdP. At this point you should probably check to confirm that everything works as expected e.g. perform a SAML trace using a browser plugin so that you can see the what normal behaviour looks like.

If you experience a failure the most common scenarios are.

- You have not consumed the metadata of either party.
- The IdP is not releasing the correct attribute to the SP (use the Session handler on the SP or shibd.log files)
- The REMOTE_USER attribute and header is not reaching the IdP via Tomcat and Apache (likely to result in a 403 error from Tomcat)
- Apache and the Shibboleth SP are not protecting the RemoteUser handler and user recieves the Shibboleth IdP login page.