Installing and configuring Shibboleth as the backend IDP server for use with EFT SSO

THE INFORMATION IN THIS ARTICLE APPLIES TO:

- EFT v7.3.3 and later

NOTE: This article provides guidelines for using a third-party tool with EFT. This article is not meant as formal support for that tool, but only as an example of setup options. Contact the third-party seller's support for detailed information about their product. Globalscape is not responsible for any configuration errors involving the third-party tool.

Overview

- This document outlines the steps needed to install and configure Shibboleth as the backend IDP server for use with testing EFT's SSO Feature.
- There are several ways to configure shibboleth for testing this covers one very basic installation that authenticates users against an LDAP server.
- Shibboleth backend that authenticates against an LDAP server
- Shibboleth and ADFS are the most ideal test IDPs for long term EFT Automation testing in that the other test servers (safenet, salesforce) are reside in the cloud outside of our control. Shibboleth however requires the most setup time.

Shibboleth Installation

- Download Shibboleth Identity provider for Windows.
- Follow the basic installation instructions and tests to verify successful install. If you get service unavailable or unauthorized add your IP address to the attribute-filter.xml file located at shibboleth\idp\conf
- https://wiki.shibboleth.net/confluence/display/IDP30/Installation

Starting and Stopping Shibboleth

- To restart the shibboleth service locate the shibd_idpw.exe application located at ..\shibboleth\idp\bin\ . The executable contains service restart buttons.
- After restarting the shibboleth service you will also need to restart the jetty web server application start.jar located at ..\shibboleth\jetty

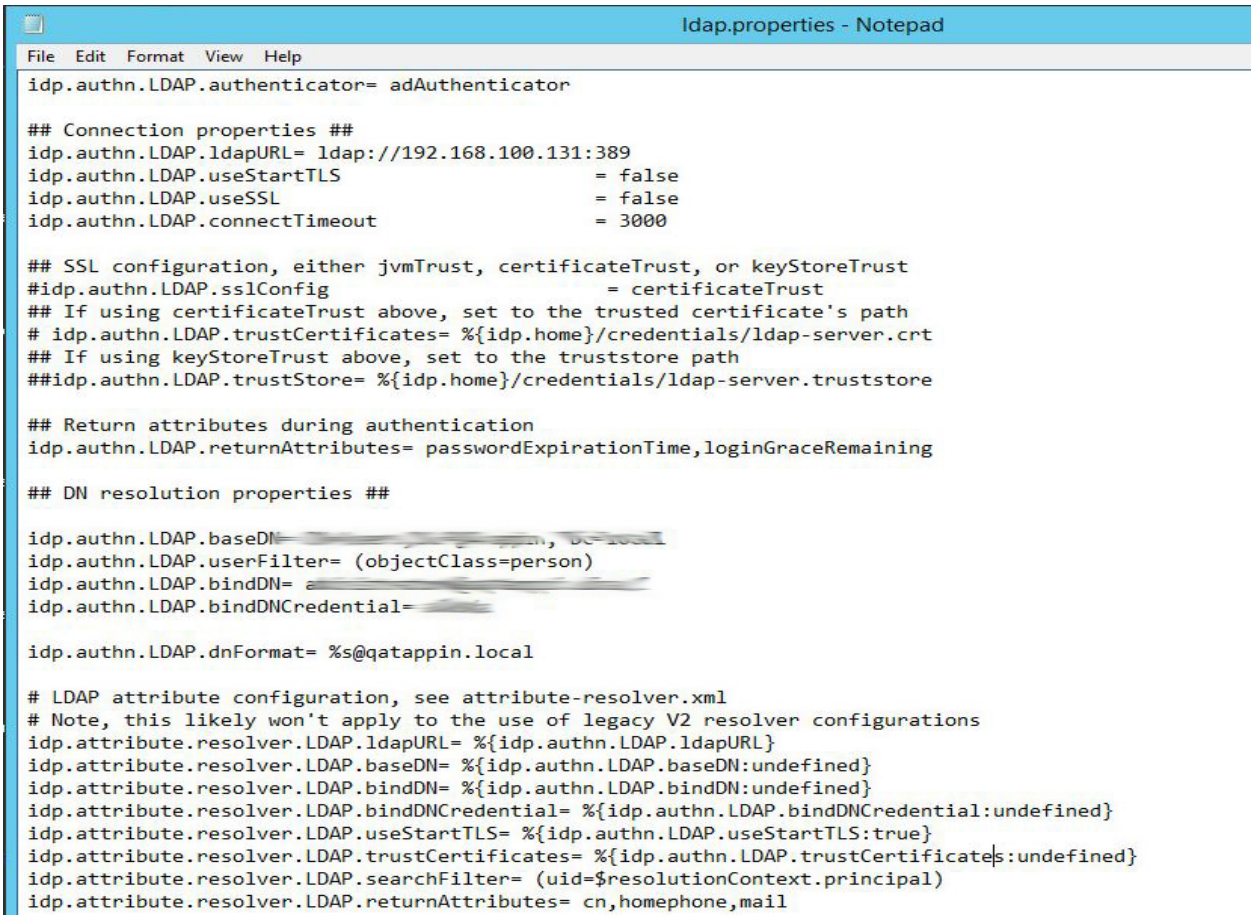
You may want to make shortcut buttons to the aforementioned services since you will likely need to restart them often.

Shibboleth Configuration

ldap.properties

Below is an excerpt of notable configuration items from the ldap.properties file located at ..\Shibboleth\idp\conf

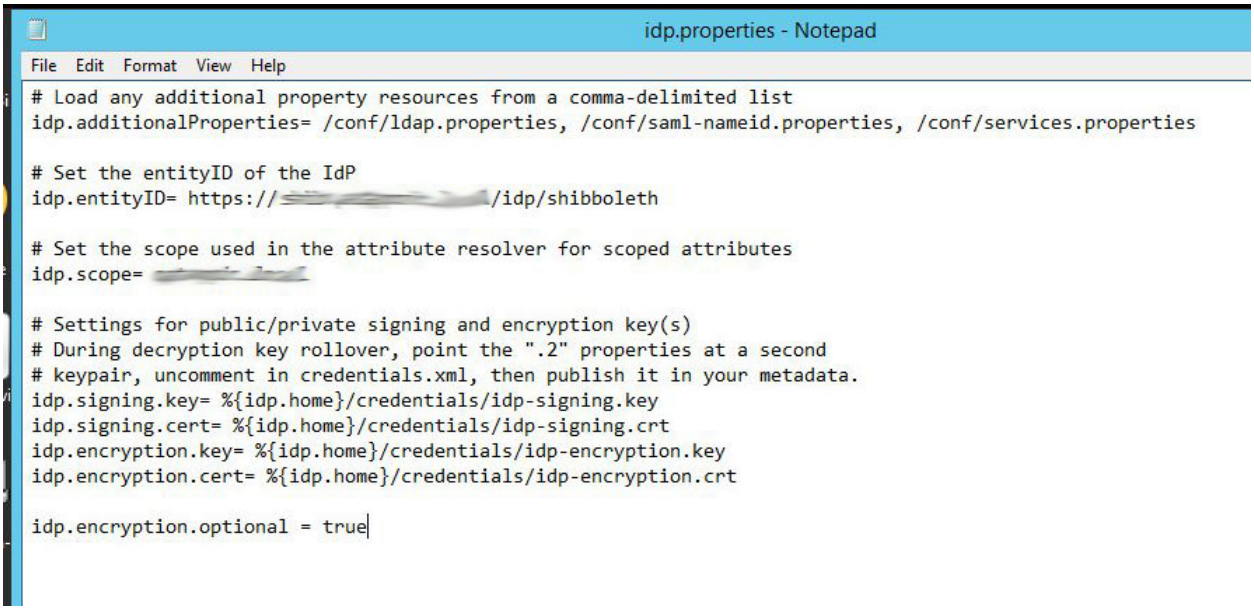
- You will need to edit the ldap.properties file with your LDAP configuration that Shibboleth will be authenticating against.
- Open up the ldap.properties file located at ..\Shibboleth\IDP\conf
- Configure the connection properties.
- In the example below, SSL settings are set to false, if you choose to use SSL and set useSSL to true then you will need to obtain your LDAP server's certificate information and and the SSL settings below.
- Configure the DN Resolution Properties,
- Configure the Attribute Resolver properties.



idp.properties

Below is an excerpt of notable configuration items from the idp.properties file located at ..\Shibboleth\idp\conf

- Take note of the idp.additionalProperites field, this field allows you to specify other properties files that will be imported into the shibboleth runtime.
- The idp.entityid can be found here, this value will be needed when configuring EFT's SSO Connection properties.
- Set the idp.scope equal to your domain.
- The location of the idp-signing.crt is also specified here. This certificate will need to be copied to your EFT System and specified in the SSO Configuration.



relying-party.xml

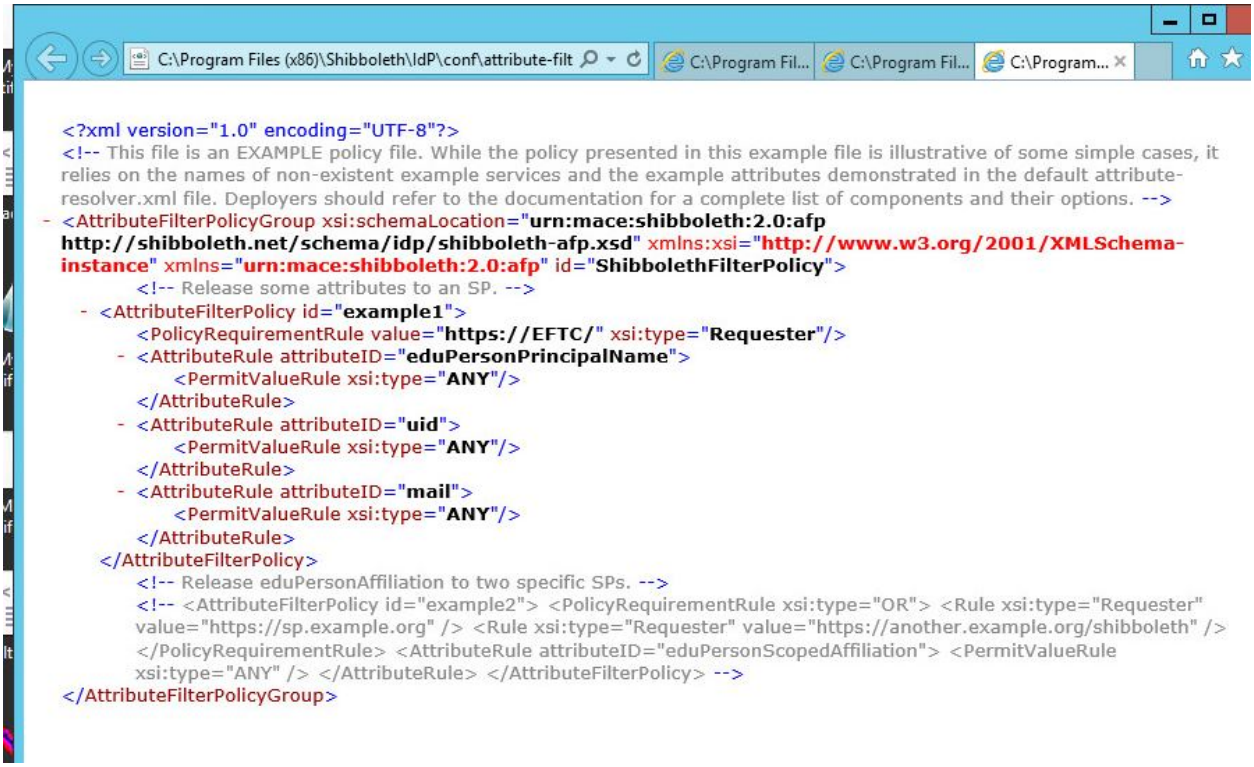
- For this example configuration, edit the rely-ing-party.xml file located in the config directory to signResponses, signEncryptions but to not encryptAssertions for communication with our EFT SP.

Mapping EFT Attributes to IDP > LDAP backend

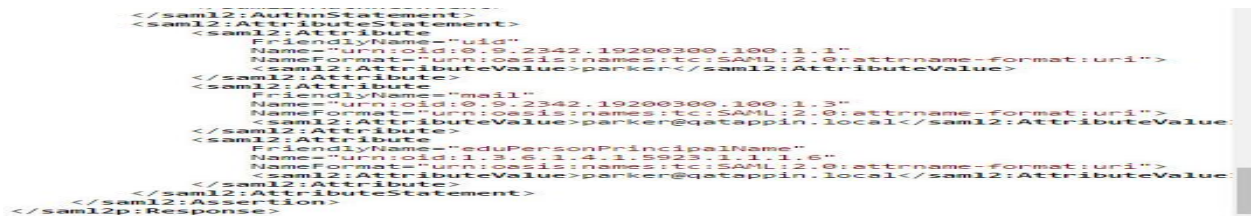
The attribute-filter, attribute-resolver and attribute-resolver-ldap files contain the attribute mapping/translations. In EFT we will supply a parameter attribute/nameid to the IDP. From there we need to define a mapping of the EFT attribute to the idp server to the ldap server.

Attibute-filter.xml

- This file determines which attributes we will return to the SP (EFT). Below if you sniff the SAML Response you will see the attributes defined in the attribute-filter.xml document.

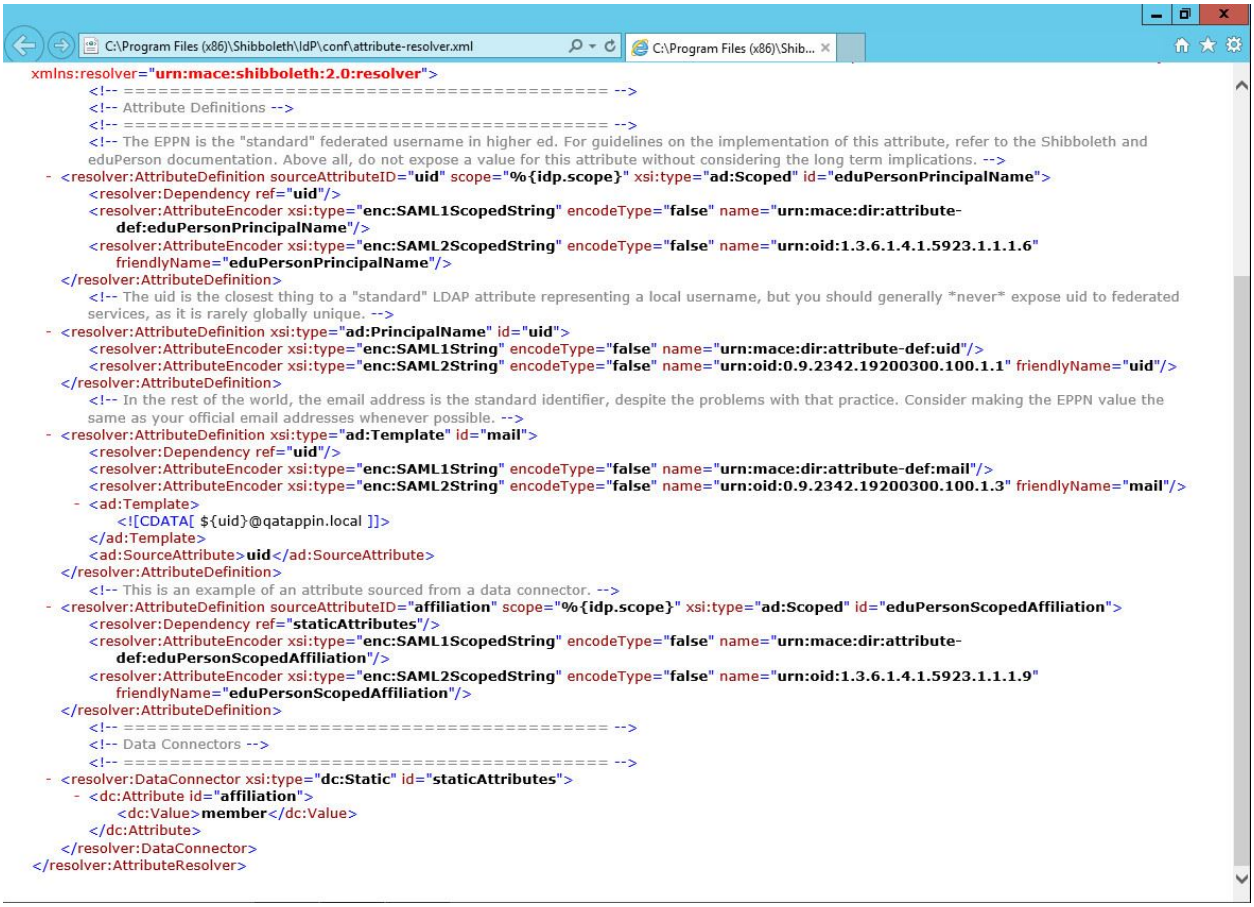


- Below Image capture of a SAML Response to an EFT SP request:



Installing and configuring Shibboleth as the backend IDP server for use with EFT SSO

Attribute-resolver.xml



```
C:\Program Files (x86)\Shibboleth\IdP\conf\attribute-resolver.xml
C:\Program Files (x86)\Shib...
xmlns:resolver="urn:mace:shibboleth:2.0:resolver">
        <!-- ========================================== -->
        <!-- Attribute Definitions -->
        <!-- ========================================== -->
        <!-- The EPPN is the "standard" federated username in higher ed. For guidelines on the implementation of this attribute, refer to the Shibboleth and
        eduPerson documentation. Above all, do not expose a value for this attribute without considering the long term implications. -->
  - <resolver:AttributeDefinition sourceAttributeID="uid" scope="%{idp.scope}" xsi:type="ad:Scoped" id="eduPersonPrincipalName">
        <resolver:Dependency ref="uid"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML1ScopedString" encodeType="false" name="urn:mace:dir:attribute-
            def:eduPersonPrincipalName"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML2ScopedString" encodeType="false" name="urn:oid:1.3.6.1.4.1.5923.1.1.1.6"
            friendlyName="eduPersonPrincipalName"/>
    </resolver:AttributeDefinition>
        <!-- The uid is the closest thing to a "standard" LDAP attribute representing a local username, but you should generally *never* expose uid to federated
        services, as it is rarely globally unique. -->
  - <resolver:AttributeDefinition xsi:type="ad:PrincipalName" id="uid">
        <resolver:AttributeEncoder xsi:type="enc:SAML1String" encodeType="false" name="urn:mace:dir:attribute-def:uid"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML2String" encodeType="false" name="urn:oid:0.9.2342.19200300.100.1.1" friendlyName="uid"/>
    </resolver:AttributeDefinition>
        <!-- In the rest of the world, the email address is the standard identifier, despite the problems with that practice. Consider making the EPPN value the
        same as your official email addresses whenever possible. -->
  - <resolver:AttributeDefinition xsi:type="ad:Template" id="mail">
        <resolver:Dependency ref="uid"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML1String" encodeType="false" name="urn:mace:dir:attribute-def:mail"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML2String" encodeType="false" name="urn:oid:0.9.2342.19200300.100.1.3" friendlyName="mail"/>
      - <ad:Template>
            <![CDATA[ ${uid}@qatappin.local ]]>
        </ad:Template>
        <ad:SourceAttribute>uid</ad:SourceAttribute>
    </resolver:AttributeDefinition>
        <!-- This is an example of an attribute sourced from a data connector. -->
  - <resolver:AttributeDefinition sourceAttributeID="affiliation" scope="%{idp.scope}" xsi:type="ad:Scoped" id="eduPersonScopedAffiliation">
        <resolver:Dependency ref="staticAttributes"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML1ScopedString" encodeType="false" name="urn:mace:dir:attribute-
            def:eduPersonScopedAffiliation"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML2ScopedString" encodeType="false" name="urn:oid:1.3.6.1.4.1.5923.1.1.1.9"
            friendlyName="eduPersonScopedAffiliation"/>
    </resolver:AttributeDefinition>
        <!-- ========================================== -->
        <!-- Data Connectors -->
        <!-- ========================================== -->
  - <resolver:DataConnector xsi:type="dc:Static" id="staticAttributes">
      - <dc:Attribute id="affiliation">
            <dc:Value>member</dc:Value>
        </dc:Attribute>
    </resolver:DataConnector>
</resolver:AttributeResolver>
```

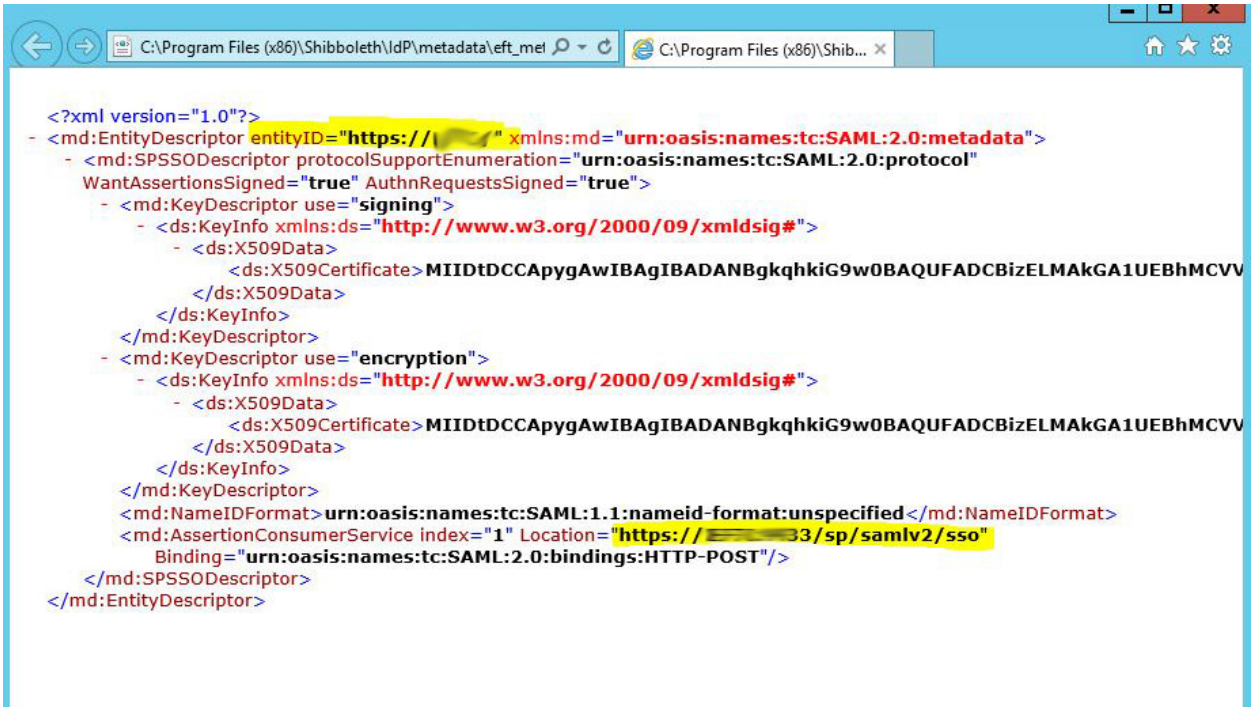
Attribute-resolver-ldap.xml

```
C:\Program Files (x86)\Shibboleth\IdP\conf\attribute-resolver-ldap.xml
C:\Program Files (x86)\Shib...
http://shibboleth.net/schema/idp/shibboleth-attribute-encoder.xsd urn:mace:shibboleth:2.0:security
http://shibboleth.net/schema/idp/shibboleth-security.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:sec="urn:mace:shibboleth:2.0:security" xmlns:enc="urn:mace:shibboleth:2.0:attribute:encoder"
xmlns:dc="urn:mace:shibboleth:2.0:resolver:dc" xmlns:ad="urn:mace:shibboleth:2.0:resolver:ad" xmlns:pc="urn:mace:shibboleth:2.0:resolver:pc"
xmlns:resolver="urn:mace:shibboleth:2.0:resolver">
        <!-- ========================================== -->
        <!-- Attribute Definitions -->
        <!-- ========================================== -->
        <!-- The EPPN is the "standard" federated username in higher ed. For guidelines on the implementation of this attribute, refer to the Shibboleth and
        eduPerson documentation. Above all, do not expose a value for this attribute without considering the long term implications. -->
  - <resolver:AttributeDefinition sourceAttributeID="eduPersonPrincipalName" xsi:type="ad:Prescoped" id="eduPersonPrincipalName">
        <resolver:Dependency ref="myLDAP"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML1ScopedString" encodeType="false" name="urn:mace:dir:attribute-
            def:eduPersonPrincipalName"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML2ScopedString" encodeType="false" name="urn:oid:1.3.6.1.4.1.5923.1.1.1.6"
            friendlyName="eduPersonPrincipalName"/>
    </resolver:AttributeDefinition>
        <!-- The uid is the closest thing to a "standard" LDAP attribute representing a local username, but you should generally *never* expose uid to federated
        services, as it is rarely globally unique. -->
  - <resolver:AttributeDefinition sourceAttributeID="uid" xsi:type="ad:Simple" id="uid">
        <resolver:Dependency ref="myLDAP"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML1String" encodeType="false" name="urn:mace:dir:attribute-def:uid"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML2String" encodeType="false" name="urn:oid:0.9.2342.19200300.100.1.1" friendlyName="uid"/>
    </resolver:AttributeDefinition>
        <!-- In the rest of the world, the email address is the standard identifier, despite the problems with that practice. Consider making the EPPN value the
        same as your official email addresses whenever possible. -->
  - <resolver:AttributeDefinition sourceAttributeID="mail" xsi:type="ad:Simple" id="mail">
        <resolver:Dependency ref="myLDAP"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML1String" encodeType="false" name="urn:mace:dir:attribute-def:mail"/>
        <resolver:AttributeEncoder xsi:type="enc:SAML2String" encodeType="false" name="urn:oid:0.9.2342.19200300.100.1.3" friendlyName="mail"/>
    </resolver:AttributeDefinition>
        <!-- ========================================== -->
        <!-- Data Connectors -->
        <!-- ========================================== -->
        <!-- Example LDAP Connector The connectivity details can be specified in ldap.properties to share them with your authentication settings if desired. -->
  - <resolver:DataConnector xsi:type="dc:LDAPDirectory" id="myLDAP" useStartTLS="%{idp.attribute.resolver.LDAP.useStartTLS:true}"
    principalCredential="%{idp.attribute.resolver.LDAP.bindDNCredential}" principal="%{idp.attribute.resolver.LDAP.bindDN}" baseDN="%
    {idp.attribute.resolver.LDAP.baseDN}" ldapURL="%{idp.attribute.resolver.LDAP.ldapURL}">
      - <dc:FilterTemplate>
            <![CDATA[ %{idp.attribute.resolver.LDAP.searchFilter} ]]>
        </dc:FilterTemplate>
        <dc:ReturnAttributes>%{idp.attribute.resolver.LDAP.returnAttributes}</dc:ReturnAttributes>
      - <dc:StartTLSTrustCredential xsi:type="sec:X509ResourceBacked" id="LDAPtoIdPCredential">
            <sec:Certificate>%{idp.attribute.resolver.LDAP.trustCertificates}</sec:Certificate>
        </dc:StartTLSTrustCredential>
    </resolver:DataConnector>
</resolver:AttributeResolver>
```

Specifying the EFT SP Metadata

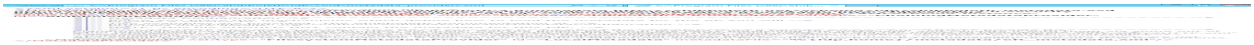
EFT Metadata.xml File

- Pay special attention to the entityID value.
- When specifying the entity location property be sure to include the port number if EFT is listening on any port other than the default 443.
- For this example IDP configuration set the properties WantAssertionsSigned and AuthRequestsSigned = true.
- Place this file in the shibboleth\idp\metadata directory.



metadata-providers.xml

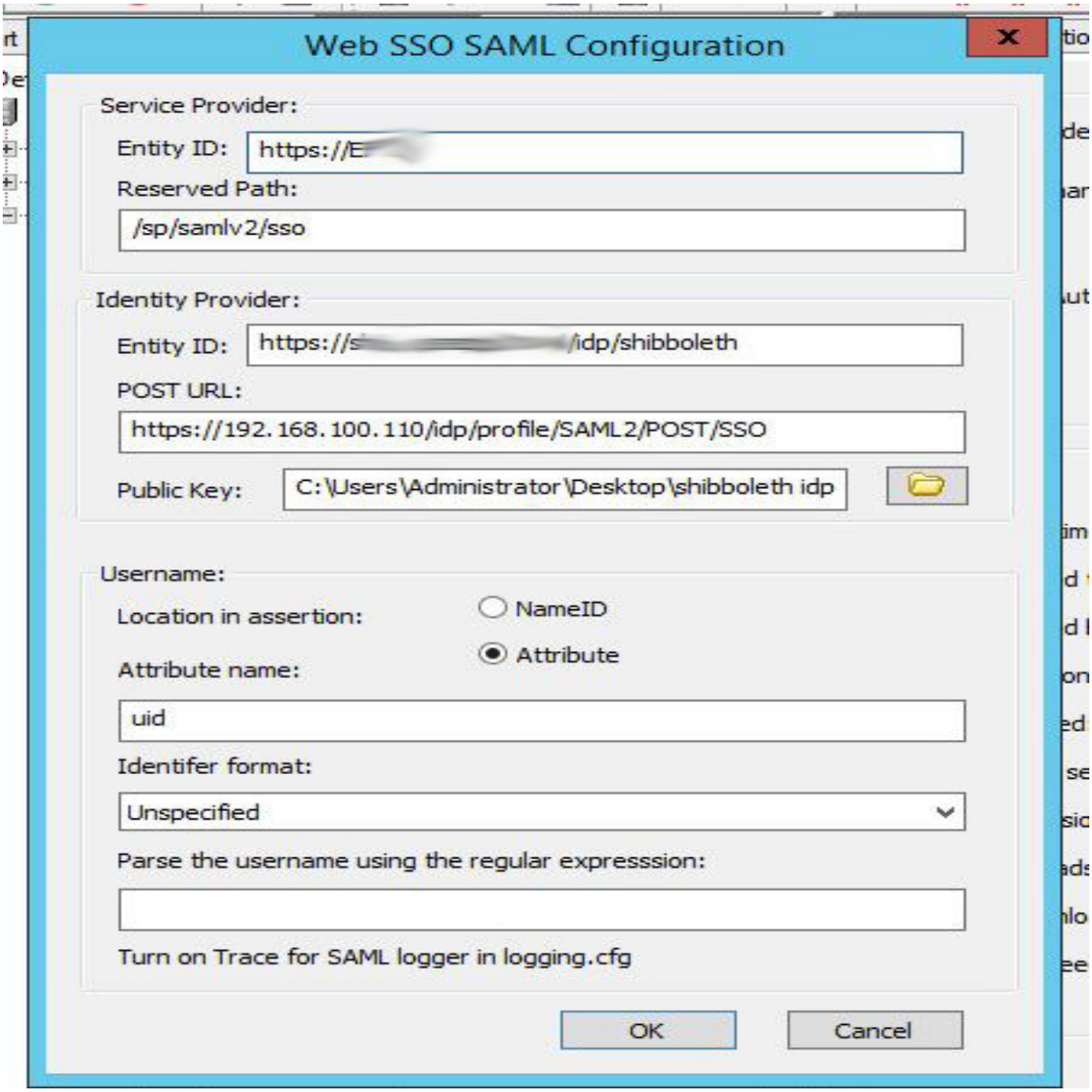
The metatdata-providers.xml file will need to be modified to show the location of the EFT Metadata file. The EFT_Metadata.xml file referenced in the example below is created and placed in the ...shibboleth\metatdata file location. The EFT_metadata file is how we communicate that EFT is an authorized Service Provider that will be using the Shibboleth IDP.



Installing and configuring Shibboleth as the backend IDP server for use with EFT SSO

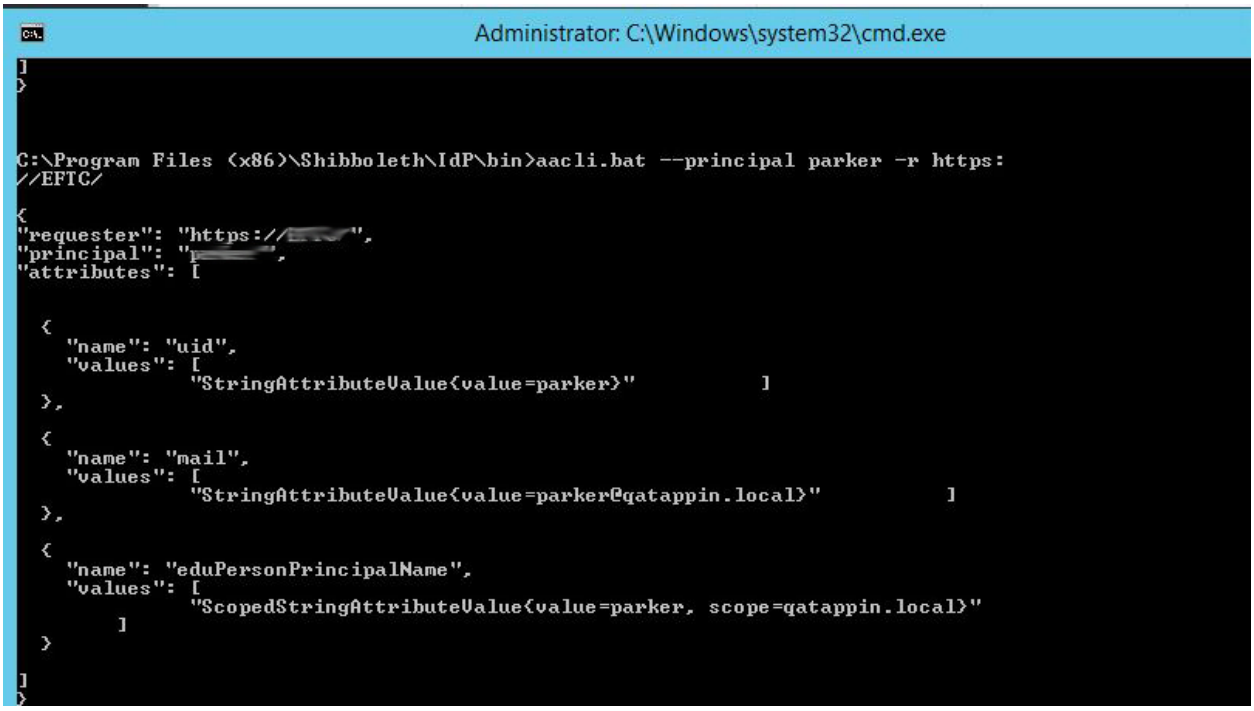
EFT Configuration

- The EFT site can just be a GS auth site with user names that exist on the LdAP backend.
- Configure EFT as follows, the idp properties can be found in the idp.properties as specified above.
- For the public key, copy the idp-signing.crt file from your shibboleth server to your EFT system and reference it in the SSO Settings. The idp-signing.crt file is automatically generated upon installation of the Shibboleth IDP server. It is located in the c:\program files(x86)\Shibboleth\idp\credentials folder.



Test configuration of release attributes

- Shibboleth has a tool called aacli.bat where you can test if you configured your release attributes correctly. If you do not get the expected results then you need to examine your attribute-filter and attribute-resolver files.



Shibboleth Logging

- To up the logging level can be changed in the logback.xml file located in the ..\shibboleth\idp\conf directory.
- Logs are located in the ..\shibboleth\idp\logs directory.

GlobalSCAPE Knowledge Base
https://kb.globalscape.com/Knowledgebase/11322/Installing-and-configuring-S...