Shibboleth SAML IdP in Azure AD B2C (Custom Policies)

# Scenario

This document will walk you through adding a SAML-based Identity provider to Azure AD B2C using Shibboleth.

This will enable scenarios such as:

* Contoso wants to build an app, *Contoso Rewards*, for their customers, who are consumers, to keep track of their rewards points obtained by purchasing Contoso products in retail stores. Contoso also wants to allow Contoso employees to sign in to the same app using their corporate credentials to create and manage activities that customers can participate in to get more rewards.
  + Use Azure AD B2C
  + Local accounts + social IdPs for consumers
  + SAML IdP for Contoso’s SAML-based IdP

This walkthrough will only focus on the Federated handshake using SAML as protocol and Shibboleth as the service.

Out of scope is how to integrate your application with B2C. This is described here: <https://docs.microsoft.com/en-us/azure/active-directory-b2c/active-directory-b2c-overview>

Prerequisite:

* Complete B2C Custom policy Get Started: setup: <https://docs.microsoft.com/en-us/azure/active-directory-b2c/active-directory-b2c-get-started-custom>
* Recommended: Complete Setup AAD as IDP to gain familiarity with approach to federation: <https://docs.microsoft.com/en-us/azure/active-directory-b2c/active-directory-b2c-setup-aad-custom>

Overview: Part 1: Setup Shiboleth as SAML IDP, Part 2: Setup B2C to work as service provider with Shiboleth.

# Walkthrough

## Setup

Make sure you first complete the Basic Setup tutorial. Setup for Shibboleth and underlying LDAP for shibboleth is completed. The current document describes configuration setup required with Shibboleth IDP 3.1.1

## Configure Shibboleth

1. Configure the SAML Relaying Party / App in Shibboleth

Configuration file path %Shibboleth\_Install\_Dir%/Idp/conf/relaying-part.xml

Use the shibboleth.DefaultRelayingParty bean in the relaying-party.xml file

Check for the presence of a SAML2.SSO profile for the relaying party

<!--

Default configuration, with default settings applied for all profiles, and enables

the attribute-release consent flow.

-->

<bean id="shibboleth.DefaultRelyingParty" parent="RelyingParty">

<property name="profileConfigurations">

<list>

<bean parent="Shibboleth.SSO" p:postAuthenticationFlows="attribute-release" />

<ref bean="SAML1.AttributeQuery" />

<ref bean="SAML1.ArtifactResolution" />

<bean parent="SAML2.SSO" p:postAuthenticationFlows="attribute-release" p:encryptAssertions="false" p:signAssertions="true" />

<ref bean="SAML2.ECP" />

<ref bean="SAML2.Logout" />

<ref bean="SAML2.AttributeQuery" />

<ref bean="SAML2.ArtifactResolution" />

<ref bean="Liberty.SSOS" />

</list>

</property>

</bean>

1. Configure ipd.properties:

Configuration file path: %Shibboleth\_Install\_Dir%/Idp/conf/idp.properties

-------------------

# Set the entityID of the IdP

idp.entityID= https://shibwin2.westus.cloudapp.azure.com/idp/shibboleth

# Set the scope used in the attribute resolver for scoped attributes

idp.scope= shibwin2.westus.cloudapp.azure.com

1. Configure Shibboleth IDP Metadata XML:

File path: %Shibboleth\_Install\_Dir%/Idp/metadata/idp-metadata.xml

This is the metadata for the IdP

It should contain the signing certificate used by the Shibboleth IdP

You should ensure that the Scope value is correct.

By default it includes 2 ArtificatResolutionService entries, and 4 SingleSignOnServiec binding entries.

Azure B2C will not accept these bindings. This should be edited to just contain bindings for HTTP -POST and HTTP-Redirect

By default, Shibboleth IDP metadata contains two signing certificates, ADB2C will throw exception while validating the SAML response. Need to remove one of the CERTs from the metadata.

The IdP Metadata should be viewable in a browser at https://<<domain\_name>>/idp/shibboleth

Eg: <https://shibwin2.westus.cloudapp.azure.com/idp/shibboleth>

1. Configure Relaying Party / App metadata in Shibboleth

Create a metadata file for the ADB2C policy and save as XML file in the below folder in Shibboleth installation:

%Shibboleth\_Install\_Dir%/Idp/metadata

EntityID in the metadata for Policy should contain the base policy URL eg: <https://login.microsoftonline.com/te/contoso.onmicrosoft.com/B2C_1A_base>

A sample Metadata file for Policy is as shown below:

<?xml version="1.0" encoding="UTF-8"?>

<EntityDescriptor entityID="https://login.microsoftonline.com/te/contoso.onmicrosoft.com/B2C\_1A\_base"

xmlns="urn:oasis:names:tc:SAML:2.0:metadata"

xmlns:ds="http://www.w3.org/2000/09/xmldsig#">

<SPSSODescriptor protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">

<NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</NameIDFormat>

<AssertionConsumerService index="1" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"

Location="https://login.microsoftonline.com/te/contoso.onmicrosoft.com/B2C\_1A\_base/samlp/sso/assertionconsumer" />

<KeyDescriptor>

<ds:KeyInfo>

<ds:X509Data>

<ds:X509Certificate><!--This tag should contain the public key for the certificate being uploaded in ADBC tenant for SamlMessageSigning or SamlAssertionsigning or SamlAssertionEncryption cryptographic keys --></ds:X509Certificate>

</ds:X509Data>

</ds:KeyInfo>

</KeyDescriptor>

</SPSSODescriptor>

</EntityDescriptor>

Modify metadata provider xml to configure the relaying party’s metadata

Configuration file path %Shibboleth\_Install\_Dir%/Idp/conf/metadata-provider.xml

Add a <MetadataProvider> tag with the path to the folder within shibboleth installation which contains the metadata xml for ADB2C SAML policy

<MetadataProvider id="contosoRP" xsi:type="FilesystemMetadataProvider" metadataFile="C:\Program Files (x86)\Shibboleth\IdP\metadata\contoso.xml"/>

</MetadataProvider>

Note: This metadata file can be replaced with the metadata URL for the ADB2C policy if any, Shibboleth supports that.

1. Configure Attribute resolver in Shibboleth: This is used to control the attributes that need to be sent as part of the SAML Assertions

Configuration File Path: %Shibboleth\_Install\_Dir%/Idp/conf/attribute-resolver.xml

Define Data connector in the resolver file to connect to LDAP for fetching attributes

<DataConnector id="myLDAP" xsi:type="LDAPDirectory"

ldapURL="ldap://localhost:389"

baseDN="ou=People,dc=example,dc=com"

principal="cn=Directory Manager"

principalCredential="Password@1234"

useStartTLS="false">

<FilterTemplate>

<![CDATA[

(uid=$resolutionContext.principal)

]]>

</FilterTemplate>

</DataConnector>

Add attribute definitions for all the attributes that need to be fetched and returned in claims

<AttributeDefinition id="userId" xsi:type="Simple" sourceAttributeID="uid">

<Dependency ref="myLDAP" />

<AttributeEncoder xsi:type="SAML2String" name="userId" friendlyName="userId" />

</AttributeDefinition>

1. Configure Attribute filter

Configuration file Path: %Shibboleth\_Install\_Dir%/Idp/conf/attribute-filter.xml

Create a new Attribute filter policy for ADB2C in Shibboleth and add all the attributes that need to be included in claims in the attribute filter

The value for Requester in Policy Requirement Rule must match the EntityId for the relaying party app.

<!-- Release some attributes to an SP. -->

<AttributeFilterPolicy id="example1">

<PolicyRequirementRule xsi:type="Requester" value=" https://login.microsoftonline.com/te/contoso.onmicrosoft.com/B2C\_1A\_base " />

<AttributeRule attributeID="givenName">

<PermitValueRule xsi:type="ANY" />

</AttributeRule>

<AttributeRule attributeID="sn">

<PermitValueRule xsi:type="ANY" />

</AttributeRule>

<AttributeRule attributeID="uid">

<PermitValueRule xsi:type="ANY" />

</AttributeRule>

<AttributeRule attributeID="mail">

<PermitValueRule xsi:type="ANY" />

</AttributeRule>

<AttributeRule attributeID="userId">

<PermitValueRule xsi:type="ANY" />

</AttributeRule>

</AttributeFilterPolicy>

## Create the SAML Claims Provider

Now that you’ve got working set of advanced policies (pre-requisite by completing: <https://docs.microsoft.com/en-us/azure/active-directory-b2c/active-directory-b2c-get-started-custom>)

, let’s go ahead and add the SAML IdP.

***Note****: The Policy Reference section at the end of this doc contains more details around each of the XML elements referenced in this section.*

1. Open the B2C\_1A\_base.xml policy from your working directory.
2. Find the section with the <ClaimsProviders> and add a new <ClaimsProvider> as follows:

<ClaimsProviders>

<ClaimsProvider>

<Domain>contososhibboleth</Domain>

<DisplayName>Contoso Shibboleth</DisplayName>

<TechnicalProfiles>

<TechnicalProfile Id="ContosoShibboleth">

<DisplayName>Contoso Shibboleth</DisplayName>

<Description>Login with your Contoso Shibboleth account</Description>

<Protocol Name="SAML2"/>

<Metadata>

<Item Key="RequestsSigned">false</Item>

<Item Key="WantsEncryptedAssertions">false</Item>

<Item Key="WantsSignedAssertions">false</Item>

<Item Key="PartnerEntity">

<![CDATA[

<EntityDescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:shibmd="urn:mace:shibboleth:metadata:1.0" xmlns:xml="http://www.w3.org/XML/1998/namespace" xmlns:mdui="urn:oasis:names:tc:SAML:metadata:ui" entityID="https://shibwin2.westus.cloudapp.azure.com/idp/shibboleth">

<IDPSSODescriptor protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol urn:oasis:names:tc:SAML:1.1:protocol urn:mace:shibboleth:1.0">

<Extensions>

<shibmd:Scope regexp="false">shibwin2.westus.cloudapp.azure.com</shibmd:Scope>

<mdui:UIInfo>

<mdui:DisplayName xml:lang="en">OpenJS directory on shibwin2</mdui:DisplayName>

<mdui:Description xml:lang="en">Description of your IdP at shibwin2</mdui:Description>

</mdui:UIInfo>

</Extensions>

<KeyDescriptor use="signing">

<ds:KeyInfo>

<ds:X509Data>

<ds:X509Certificate>

MIIDCzCCAfOgAwIBAgIUf8EwhMtatXd1mG7OFgY1DhqUYw4wDQYJKoZIhvcNAQEL

BQAwEzERMA8GA1UEAwwIc2hpYndpbjIwHhcNMTcwNDEwMjMzOTQ3WhcNMzcwNDEw

MjMzOTQ3WjATMREwDwYDVQQDDAhzaGlid2luMjCCASIwDQYJKoZIhvcNAQEBBQAD

ggEPADCCAQoCggEBAIRNMsplqWaTR8Q+2bDlFzPsVMDItijGbswrSGnZI/B+xmEq

uJn9E+vvtJdJdifKmEnkam99Phge373Rqkca2Jg7iZXVT5RIweMTQgdV3x/E9oML

8o31EzKf/sur4dHOrQdNKzLKn1eoT/jX2rAK8SwOpc0OyfIrzXa6ejqek+T0JvkU

JVqiDqo5OyDkO1BjbfmB8n+for1aCwLXCyv/ipWyAZtxWYMVv1oHjEKETraM3d7i

rUE0inQQh/1drLtvfMLPIWokwdNGAJ1HyV2pjbYM2BY981fJmXMj/0G4kVm2bS8b

8OQ1rnD4nBYsxNAle4cnM3MDx+Y5GQenghUMGf8CAwEAAaNXMFUwHQYDVR0OBBYE

FItdkqW9CzJ/kO1hcD9/o5Eez7DWMDQGA1UdEQQtMCuCCHNoaWJ3aW4yhh9odHRw

czovL3NoaWJ3aW4yL2lkcC9zaGliYm9sZXRoMA0GCSqGSIb3DQEBCwUAA4IBAQAa

CUfdgmQiHdL+pnAWKFRXwETAutwS9R4xZV6/shK6H/D3+MMzfMetGOCmgwtz4Rag

N7sYb1TmEgkvL2wLCEkzUfTZlw0pZDZrvnAecO/dPm2aH+vzo2nI6XXAaP6i0R0p

30rrNln6j0GnxOyUB42mgo8o1LKaav68pLEVbW9UUB5xQN5ZWv2MWsbPADPJ6WrB

bGr8shjVOH/yc+iDgBuVwKUH+e5mcZrBvxiJhnhpXlO3QcNddg4brrlqpMwet8YQ

pogPD8Od3pQN258S9U/Uu2o/zszLocfSl2KcMdSPWn36Fmsfhz/CImk+YW81GmdC

VKbmrDXVRrn8yK+LusEB

</ds:X509Certificate>

</ds:X509Data>

</ds:KeyInfo>

</KeyDescriptor>

<KeyDescriptor use="encryption">

<ds:KeyInfo>

<ds:X509Data>

<ds:X509Certificate>

MIIDCzCCAfOgAwIBAgIUOhNOdKSJNLRgMdS49/cmIVRJ9owwDQYJKoZIhvcNAQEL

BQAwEzERMA8GA1UEAwwIc2hpYndpbjIwHhcNMTcwNDEwMjMzOTQ4WhcNMzcwNDEw

MjMzOTQ4WjATMREwDwYDVQQDDAhzaGlid2luMjCCASIwDQYJKoZIhvcNAQEBBQAD

ggEPADCCAQoCggEBAJKc0pdi6ak7ud/YiOEmq4Ftz9pk552ndauLilsSfQzdajm8

6XtB8ZUW2LFb7ykqIjVFhgrKMgi8bRwHvNLb0DI6PrXyG+tk6kPM6pKlpctxrbQb

U8dc7/hCRHpO5VFbaefuWIHGAUuT8jo7d0z5U+zjB8vNUg2eRbVWruLdphjevotw

yaKdOsyJne4axowMVhzpt47vRHHMVl7sByy8/8RXX1yLpVUORf9R6PXHQhWOAbME

PwtkV89MS9F2vGjoTg2A7ysm5ZRpMIhaSaBjJ7EUWuOvRCAIkTQwwOw+37656ZJC

/0B0mDvTxuz3+AX8g0bJeMN4sEgYzdC0DvnJD28CAwEAAaNXMFUwHQYDVR0OBBYE

FDo/SFsj09qSVAnln+BTajcyxdVvMDQGA1UdEQQtMCuCCHNoaWJ3aW4yhh9odHRw

czovL3NoaWJ3aW4yL2lkcC9zaGliYm9sZXRoMA0GCSqGSIb3DQEBCwUAA4IBAQBv

NtKu0TeZNABNlRU0U1RwvQx6+/+GjjWCI1hZvtgz6PvKeHODfvdyi4v94buJbT69

htc7S/D1rPaImYhg5PWfU6LeON+z5utUGiCmN1lvy/2ZI6h0wr8NUvzCN7NeSBmC

Re3msOScqxiK1p/l9caTYSgF/p7I0YqKQ7/Oe22TLWaD0/DNlcl+xOOG0RR3dLOS

EV7+SDTOB8alpFodbzsN4IRlB4/+qsILH0tDTeTLQOGHDqR+TSrxe+KSpn/BDPKK

NJryktqB60WbuRs2HjlBINdTa+MMJe+h2/Miz56VMVBFye6uhY/hZI6Z22/HKTC1

3rmvRSzSexhyt9vfIVIi

</ds:X509Certificate>

</ds:X509Data>

</ds:KeyInfo>

</KeyDescriptor>

<SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://shibwin2.westus.cloudapp.azure.com/idp/profile/SAML2/POST/SSO"/>

<SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="https://shibwin2.westus.cloudapp.azure.com/idp/profile/SAML2/Redirect/SSO"/>

</IDPSSODescriptor>

</EntityDescriptor>]]>

</Item>

</Metadata>

<CryptographicKeys>

<Key Id="SamlAssertionSigning" StorageReferenceId="ShibbolethTrustFrameworkSigning"/>

<Key Id="SamlMessageSigning" StorageReferenceId="ShibbolethTrustFrameworkSigning"/>

<Key Id="SamlAssertionDecryption" StorageReferenceId="ShibbolethTrustFrameworkSigning"/>

</CryptographicKeys>

<OutputClaims>

<OutputClaim ClaimTypeReferenceId="userId" PartnerClaimType="uid"/>

<OutputClaim ClaimTypeReferenceId="identityProvider" DefaultValue="SAML Idp" />

<OutputClaim ClaimTypeReferenceId="email" PartnerClaimType="mail"/>

<OutputClaim ClaimTypeReferenceId="givenName" PartnerClaimType="givenName"/>

<OutputClaim ClaimTypeReferenceId="surname" PartnerClaimType="sn"/>

<OutputClaim ClaimTypeReferenceId="authenticationSource" DefaultValue="externalIdp"/>

</OutputClaims>

<OutputClaimsTransformations>

<OutputClaimsTransformation ReferenceId="CreateRandomUPNUserName"/>

<OutputClaimsTransformation ReferenceId="CreateUserPrincipalName"/>

<OutputClaimsTransformation ReferenceId="CreateAlternativeSecurityId"/>

<OutputClaimsTransformation ReferenceId="CreateSubjectClaimFromAlternativeSecurityId"/>

</OutputClaimsTransformations>

<UseTechnicalProfileForSessionManagement ReferenceId="SM-Noop"/>

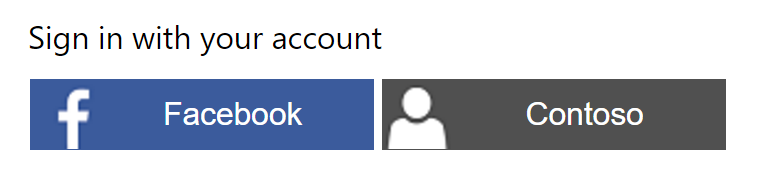
</TechnicalProfile>

</TechnicalProfiles>

</ClaimsProvider>

</ClaimsProviders>

1. Configure basic settings
   1. ClaimsProvider/Domain - Drives the value that can be passed in to [domain\_hint](http://www.cloudidentity.com/blog/2014/11/17/skipping-the-home-realm-discovery-page-in-azure-ad/) query string parameter, so make it url friendly, i.e. lowercase only, no spaces, only alphanumerical.
   2. ClaimsProvider/DisplayName – Currently not displayed anywhere
   3. TechnicalProfile@Id – Unique identifier for the technical profile, it is referenced elsewhere in the policy (see “Add the SAML Claims Provider to the User Journey”).
   4. TechnicalProfile/DisplayName - Displayed as the button's caption in the sign in screen.



* 1. TechnicalProfile/Description – Currently not displayed anywhere

1. Configure Metadata section
   1. PartnerEntity – Either the URL to the metadata endpoint or the metadata itself encapsulated by <![CDATA[ *…metadata…* ]]>
   2. Configure OutputClaims - Map each of the claims in this section to a claim in the SAML token response.
      1. ClaimTypeReferenceId is the name of the property in B2C, do not tweak these.
      2. For each of those, set either the DefaultValue or the PartnerClaimType.
2. Upload Certs - These are the certificates used to sign the SAML request and message. Even though we’ve configured the Claims Provider to not sign these, a certificate must still be provided.
   1. (If you don’t have a cert already) Create the cert using makecert (http://www.virtues.it/2015/08/howto-create-selfsigned-certificates-with-makecert/)
      1. makecert -r -pe -n "CN=idpsaml.yourtenant.onmicrosoft.com" -a sha256 -sky signature -len 2048 -e 12/21/2018 -sr CurrentUser -ss My ContosoIdpSamlCert.cer
      2. Go to cert store “Manage User Certificates” > Current User > Personal > Certificates > idpsaml.yourtenant.onmicrosoft.com
      3. Right click > All Tasks > Export
      4. Yes, export the private key
      5. Defaults (PFX and first checkbox)
   2. Open Powershell
   3. Go to ExploreAdmin
   4. Import-Module ExploreAdmin.dll (if it fails, it might be because the dll hasn’t been unblocked)
   5. Run New-CpimCertificate -TenantId yourtenant.onmicrosoft.com -CertificateId ContosoIdpSamlCert -CertificateFileName path -CertificatePassword password
      1. When you run the command, make sure you sign in with the onmicrosoft.com account local to the tenant.
      2. It’ll ask you for MFA
3. Save your changes and upload updated policy
   1. This time, make sure you check the *Overwrite the policy if it exists* checkbox.
   2. At this point, this will not have any effect, the intent of uploading is confirming that what you’ve added thus far doesn’t have any issues.

## Signing and Encryption Option

**Scenario 1: Configuration to receive Signed Assertions in SAML Message from IDP**

In the <ClaimsProviders> section under the new <ClaimsProvider> for SAML Profile add the WantsSignedAssertion item with value as “true” under metadata section

<Metadata>

<Item Key="WantsSignedAssertions">true</Item>

<Item Key="PartnerEntity">MetadataURL</Item>

</Metadata>

Add the respective cryptographic key that can be used to verify the assertion signing received from IDP

<CryptographicKeys>

<Key Id="SamlAssertionSigning" StorageReferenceId="SamlSigningCertShib2" />

<Key Id="SamlMessageSigning" StorageReferenceId="SamlSigningCertShib2" />

<Key Id="SamlAssertionDecryption" StorageReferenceId="SamlSigningCertShib2" />

</CryptographicKeys>

In Shibboleth installation set p:signAssertions to “true” in the relaying-party.xml file located at folder %Shibboleth\_Install\_Dir%/Idp/conf in the shibboleth.DefaultRelyingParty bean, as below

<bean id="shibboleth.DefaultRelyingParty" parent="RelyingParty">

<property name="profileConfigurations">

<list>

<bean parent="Shibboleth.SSO" p:postAuthenticationFlows="attribute-release" />

<ref bean="SAML1.AttributeQuery" />

<ref bean="SAML1.ArtifactResolution" />

<bean parent="SAML2.SSO" p:postAuthenticationFlows="attribute-release" p:encryptAssertions="true" p:signAssertions="true" />

<ref bean="SAML2.ECP" />

<ref bean="SAML2.Logout" />

<ref bean="SAML2.AttributeQuery" />

<ref bean="SAML2.ArtifactResolution" />

<ref bean="Liberty.SSOS" />

</list>

</property>

</bean>

**Scenario 2: Configuration to receive Signed and Encrypted Assertions in SAML Message from IDP**

In the <ClaimsProviders> section under the new <ClaimsProvider> for SAML Profile add the WantsSignedAssertion item with value as “true” under metadata section

<Metadata>

<Item Key="WantsSignedAssertions">true</Item>

<Item Key="WantsEncryptedAssertions">true</Item>

<Item Key="PartnerEntity">MetadataURL</Item>

</Metadata>

Add the respective cryptographic key that can be used to decrypt the assertions received from IDP

<CryptographicKeys>

<Key Id="SamlAssertionSigning" StorageReferenceId="SamlSigningCertShib2" />

<Key Id="SamlMessageSigning" StorageReferenceId="SamlSigningCertShib2" />

<Key Id="SamlAssertionDecryption" StorageReferenceId="SamlSigningCertShib2" />

</CryptographicKeys>

In Shibboleth installation set p:encryptAssertions to “true” in the relaying-party.xml file located at folder %Shibboleth\_Install\_Dir%/Idp/conf in the shibboleth.DefaultRelyingParty bean, as below

<bean id="shibboleth.DefaultRelyingParty" parent="RelyingParty">

<property name="profileConfigurations">

<list>

<bean parent="Shibboleth.SSO" p:postAuthenticationFlows="attribute-release" />

<ref bean="SAML1.AttributeQuery" />

<ref bean="SAML1.ArtifactResolution" />

<bean parent="SAML2.SSO" p:postAuthenticationFlows="attribute-release" p:encryptAssertions="true" p:signAssertions="true" />

<ref bean="SAML2.ECP" />

<ref bean="SAML2.Logout" />

<ref bean="SAML2.AttributeQuery" />

<ref bean="SAML2.ArtifactResolution" />

<ref bean="Liberty.SSOS" />

</list>

</property>

</bean>

## Add the SAML Claims Provider to User Journey(s)

At this point, the SAML IdP has been set up, but it’s not available in any of the sign-up / sign-in screens (aka User Journeys). In this section, we’ll make the IdP available in the SignUpOrSignIn User Journey.

1. Open the B2C\_1A\_base.xml policy from your working directory.
2. Find the section with the <UserJourneys> and duplicate the <UserJourney> with Id=”SignUpOrSignIn”
3. Rename the Id of that new <UserJourney> (i.e SignUpOrSignInSaml)
4. In the <OrchestrationStep> with Order=”2”, add a new <ClaimsExchange>
   1. Set the Id following the same convention as the others *[ClaimProviderName]Exchange*
   2. Set the TechnicalProfileReferenceId to the same Id value for the Technial Profile you set up in the previous section.
5. In the <OrchestrationStep> with Order=”1”, add a new <ClaimsProviderSelection>
   1. Set the TargetClaimsExchangeId to the same Id value you set up for the ClaimsExchange in the previous step.
6. Save your changes and upload the updated policy
7. Copy the SignUpOrSignIn.xml file
8. Rename it match the Id of the new journey you created (i.e. SignUpOrSignInSaml)
9. Modify its PolicyId to a new Guid.
10. Save your changes and upload this new policy.

## (Optional) Enable Debugging in User Journey(s)

You can enable debugging tools to help you follow through the each of the orchestration steps in the UserJourney and get details on issues that occur. This should only be enabled during development.

1. Open your new policy xml file (not the base.xml one)
2. Add the following attributes to the <TrustFrameworkPolicy> element:
   1. DeploymentMode=”Development”
   2. UserJourneyRecorderEndpoint="https://b2crecorder.azurewebsites.net/stream?id=<guid>"
      1. Replace <guid> with an actual GUID

This will allow you to troubleshoot by going to [https://b2crecorder.azurewebsites.net/trace\_102.html?id=<guid](https://b2crecorder.azurewebsites.net/trace_102.html?id=%3cguid)>