FusionAuth as SP, KeyCloak as IdP

SP-initiated Auth

In this integration, an application will take a user to FusionAuth to authenticate, and the user will click a `Login with KeyCloak` button to delegate that authentication to KeyCloak. The application could also pass an idp_hint parameter, which would cause FusionAuth to bypass the login page and take the user directly to KeyCloak.

In this case

- FusionAuth is the Service Provider, and has an application defined that the user wants to log in to
- KeyCloak is defined as an identity provider in FusionAuth
- KeyCloak has an client defined in it, which is what the user logs in to at KeyCloack

Setup

Run FusionAuth

- It's easiest to start up FusionAuth using a quickstart. This gives you a running FusionAuth instance and a working client application that comes already integrated.
- Log in with admin@example.com / password

Run KeyCloak

- There's a one-line Docker command for this
- Log in with admin / admin

Step 1: Create a KeyCloak Realm and User, Get KeyCloak's Signing Certificate

A realm in KeyCloak is like a tenant in FusionAuth. Users and clients are scoped to realms.

Create a Realm

In KeyCloak, open up the realm drop-down at the top-left of the app and select `Create Realm`.

Create a realm with the following values. You'll leave the Resource file section blank.

- Realm name: saml-idp
- Enabled: On

Click `Create` to create the realm.

Create a User

With your realm selected, click on the `Users` item in the left-hand nav, then click the `Add user` button.

Create a user with the following values.

- Username: testuser
- Email: testuser@fusionauth.io
- Email verified: Yes

Click `Create` to create the user.

Next, set the user's password. Click on the `Credentials` tab, then the `Set password` button.

Set the password using the following values.

- Password: password
- Password confirmation: password
- Temporary: Off

Click `Save`, and then `Save password` on the confirmation dialog to set testuser's password.

Get Your KeyCloak Realm's Signing Certificate

As the last part of this step, we'll get a signing certificate, which FusionAuth will use to verify the SAML response sent back from KeyCloak.

Click `Realm settings` in the left-hand nav. On the `General` tab, scroll down to the Endpoints section, and click on the SAML 2.0 Identity Provider Metadata link.

Find the `<ds:X509Certificate>` element, and copy it into a file, surrounded by certificate begin and end tags. You don't actually need this in a file, it's just a convenient place to hold a complete copy.

```
cat > certificate.crt <<EOF
-----BEGIN CERTIFICATE-----
MIICnzCCAYcCBgGJB9pxpzANBgkqhkiG9w0BAQsFADATMREwDwYDVQQDDAhzYW1sLWlkcDAeFw0yMzA2MjkxNTUwMD...
-----END CERTIFICATE-----
EOF
```

Keep the metadata page open. You'll need to grab another value from it later.

Step 2: Set Up FusionAuth

Relax CORS restrictions

We're going to allow requests from all origins, so that callbacks from KeyCloak are allowed. You could use a more specific origin setting here if you wanted to.

In FusionAuth, go to `Settings / System` in the left-hand nav. In the `CORS` tab, set:

- Allowed origins: *

Click the save button.

Import the KeyCloak Certificate

Go to `Settings / Key Master` in the left-hand nav.

From the top-right menu, select `Import certificate`.

Import the certificate you saved using the following values.

- Name: KeyCloak Signing Cert
- Certificate: Paste in the contents of the file that you created earlier

Click `Submit` to import the certificate.

Create an Identity Provider

In this section, you'll configure KeyCloak as an identity provider that can be used by applications to authenticate users.

Go to `Settings / Identity Providers` in the left-hand nav.

From the top-right menu, select `Add SAML v2`.

Create a SAML v2 identity provider using the following values.

- Name: KeyCloak IdP
- IdP endpoint: Take the value from the `Location` attribute of the `<md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" ...>` element.
  - If you're using a default setup, this should be: http://localhost:8080/realms/saml-idp/protocol/saml
- NameID format: Persistent
  - This is a hint that is passed to KeyCloak during authentication, suggesting that KeyCloak use a persistent (i.e. immutable) ID for this user, and not something editable like a name or email address
- Verification key: Select the cert you imported from KeyCloak
- Button text: `Login with KeyCloak`
- Linking strategy: Link on email. Create the user if they do not exist
- Debug enabled: On

In the Applications tab at the bottom, enable `Example App` (or whichever apps you'll be using)

In the Options tab at the bottom, set the following.

- Email claim: email
- Use POST method: On

Save the Identity Provider

View SAML Integration Information

In the IdP list, click View button for your identity provider

Scroll down to the `SAML v2 Integration details` section, and make note of two values, which you'll need to complete the KeyCloak setup.

- Callback URL (ACS)
- Issuer

Create a KeyCloak Client

A KeyCloak `Client` is analogous to a FusionAuth `Application`. In this step, you'll set up a KeyCloak client, which is the thing a user will be logging into at KeyCloak.

In the KeyCloak admin UI, select `Clients` from the left-hand nav, and then click the `Create client` button.

In the General Settings step, use the following values.

- Client type: SAML
- Client ID: Paste the `Issuer` value from the FusionAuth IdP's `SAML v2 Integration details`.
- Name: FusionAuth SP

Click the `Next` button.

In the `Login settings` step, use the following values.

- Valid redirect URIs: Paste the value of `Callback URL (ACS)` from the FusionAuth IdP's `SAML v2 Integration details`.

Click the `Save` button.

On the Settings tab for the client, scroll down to the `SAML capabilities` section, and set the following.

- Name ID format: persistent
- Force POST binding: On

Click the `Save` button.

On the Keys tab, turn off `Client signature required`.

- TODO: setup and document this

Configure KeyCloak to Pass an `email` Assertion

You'll need to jump through some hoops to get KeyCloak to add an `email` assertion to its SAML response.

In KeyCloak, select `Client scopes` from the left-hand nav and then click the `Create client scope` button.

On the `Create client scope` page, use the following values.

- Name: saml-email
- Protocol: SAML

Click `Save` to create the new client scope.

Next, you need to map this to a SAML attribute. Click on the `Mappers` tab for your new client scope.

Click the `Configure a new mapper` button. This will open a dialog with what looks like a table of information, but each row is actually a button. Click the `User Attribute` row to create a new user attribute mapper.

On the `Add mapper` page, use the following values.

- Name: email
- User Attribute: email
- SAML Attribute Name: email
- SAML Attribute NameFormat: Unspecified

Click `Save` to create the mapper.

Finally, configure the client to use this attribute.

Select `Clients` from the left-hand nav and then click your FusionAuth client link.

Select the `Client scopes` tab, and click the `Add client scope` button.

Check the box for `saml-email`, then click the `Add` button, selecting `Default` from the menu that opens up.

Test the Integration

In FusionAuth, go to the Applications area, find the application that you enabled with your identity provider, and click the `View` action button. Under the `OAuth2 & OpenID Connect Integration details` section of the dialog, copy the `OAuth IdP login URL` value and paste it into your browser. You'll need to do this in a new browser or in a private browser window/tab where you aren't already logged in to FusionAuth.

On the login page, click the `Login with KeyCloak` button. You should be taken to a KeyCloak login page, where you can log in with the credentials you set up for the KeyCloak user (testuser / password).

FusionAuth as SP, KeyCloak as IdP using IdP-initiated Auth

In this integration, the user logs into Keycloak (IdP), which then initiates login to FusionAuth (SP). As noted in the UI, this method is less secure and not recommended.

We'll assume you've followed the above steps for SP-initiated Auth and make a few changes needed for IdP initiated.

FusionAuth IdP-initiated setting

In FusionAuth, go to `Settings -> Identity Providers` and edit your KeyCloak identity provider. Turn on the `Enable IdP initiated login` toggle.

This will open an Issuer text box. For this value, go back to your KeyCloak realm settings XML and get the value of the `entityId` attribute of the `<md:EntityDescriptor>` element. If you followed this guide, it will be

```
http://localhost:8080/realms/saml-idp
```

Click `Save` to save your changes.

Keycloak IdP initiated setting

In KeyCloak, go to `Clients` in the left-hand nav, then click into your client.

In the `Settings` tab

- Go to the `Access settings` section
  - In the `IDP-Initiated SSO URL name` box, enter `fusionauth`
  - Enter your application's redirect URI in the `IDP Initiated SSO Relay State` box. This needs to match one of the `Authorized Redirect URLs` in your FusionAuth application's OAuth config.
- Click `Save`.

Next, go to the `Advanced` tab for your client.

- In the `Fine Grain SAML Endpoint Configuration` section, set the value of `Assertion Consumer Service POST Binding URL`. To get this value from FusionAuth
  - Go to `Settings -> Identity Providers` and click the `View` button for your KeyCloak identity provider
  - Go to the `SAML v2 Integration details` section
  - In the table next to `IdP Initiated Callback URLs (ACS)`, take the URL value for your application
- Put the value from FusionAuth into the `Assertion Consumer Service POST Binding URL` field
- Click the `Save` button at the bottom of the `Fine Grain SAML Endpoint Configuration` section.

Test the Integration

Now go to

```
http://localhost:8080/realms/saml-idp/protocol/saml/clients/fusionauth
```

and log in as `testuser`. You should be taken into your FusionAuth application. Note that `fusionauth` is the value you used in the `IDP-Initiated SSO URL name` setting above.