PingFederate Administration Training

STUDENT WORKBOOK

PF-300-BVP Rev A

Contents

Lab 1 - Introduction to Adapters

Introduction

This lab will guide you through configuring the adapters and required components for both your SP and IdP PingFederate servers.

This lab should take 45 minutes to complete.

Objectives

- Create a Simple Password Credential Validator
- Configure an HTML Form Adapter instance
- Configure a Reference ID adapter instance

Description of tasks to perform

Before you can configure a connection on either an SP or an IdP PingFederate requires an adapter.

For an IdP instance that connects to an SP the adapter is responsible for interfacing with the authentication system and providing a mechanism for authenticating the user prior to running the SSO transaction. For your IdP PingFederate (wal-ping.com) you will start with an HTML Form Adapter that uses a Simple PCV for authentication. The Simple PCV provides a hard coded list of users and passwords that PingFederate can use to validate credentials.

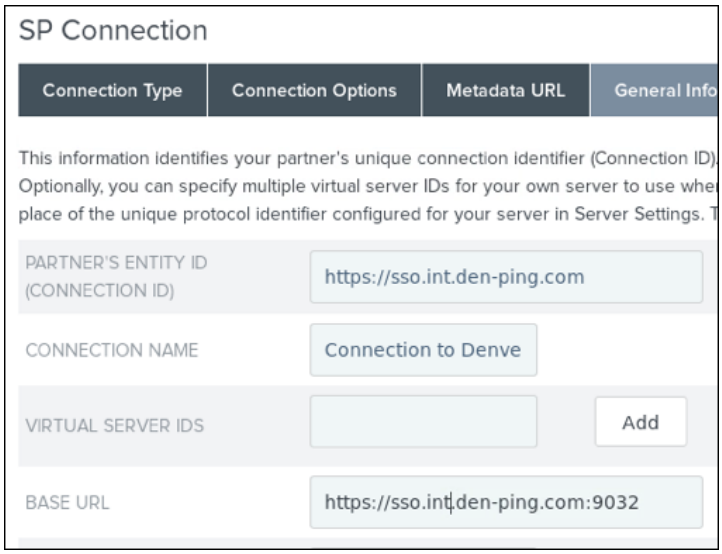
For the SP instance (den-ping.com) you will configure the Reference ID adapter. The Reference ID adapter is part of the Agentless Integration Kit and is also included with your PingFederate installation. The SP provides the last-mile integration with your application and is responsible for communicating incoming user attributes to the application.

Create a Simple Password Credential Validator

The first mile of integration to the application will perform authentication and provide the identity information needed by PingFederate; this is done using an IdP adapter. Your PingFederate instance will be prompting users for credentials, but it will need a way to check that those credentials are valid. This is the purpose of the Password Credential Validator (PCV), which you will configure first.

1. If you do not have a browser window open, double-click the **Firefox** icon on your desktop to open one.
2. On the toolbar click the **Admin Consoles** bookmark folder and select the **PingFed IdP Admin** bookmark to access the IdP admin console, the **Sign On** page will display.
3. Enter the following credentials to sign into the admin console:
    - Username: **Administrator**
    - Password: **2Federate**

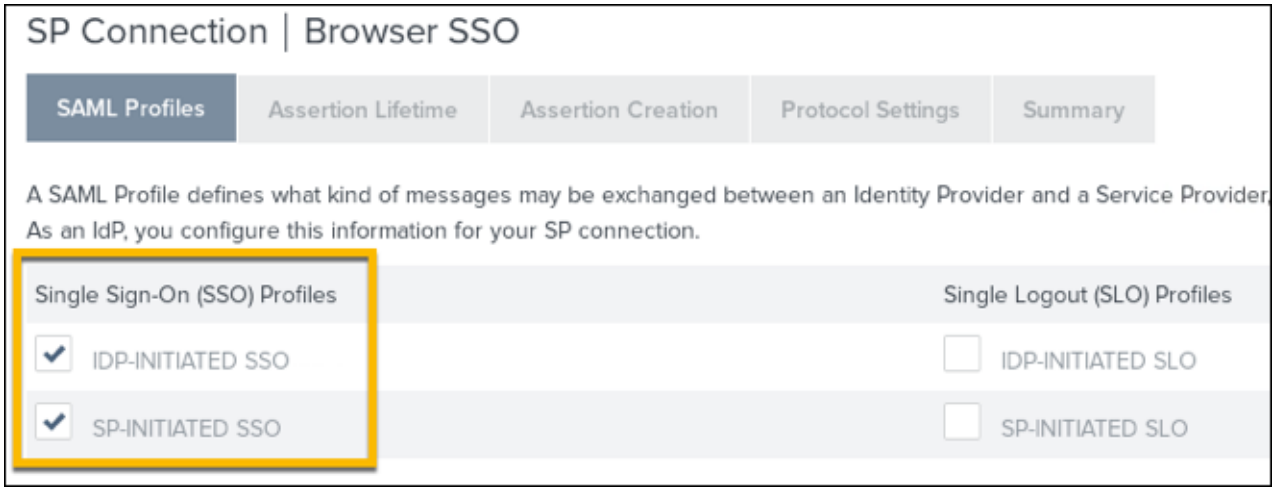
4. Navigate to the **System > Data & Credential Stores > Password Credentials Validators** page.



5. Click **Create New Instance**, the ``**Create Credential Validator Instance - Type** page will display.
6. Enter the following parameters:

   - INSTANCE NAME: **PCV: User Joe**
   - INSTANCE ID: **PCVUserJoe**
   - TYPE: **Simple Username Password Credential Validator**
   - PARENT INSTANCE: **None**

   The instance name is simply a human readable name you can assign to the PCV, the instance ID parameter is the name used internally by PingFederate to reference this particular PCV.

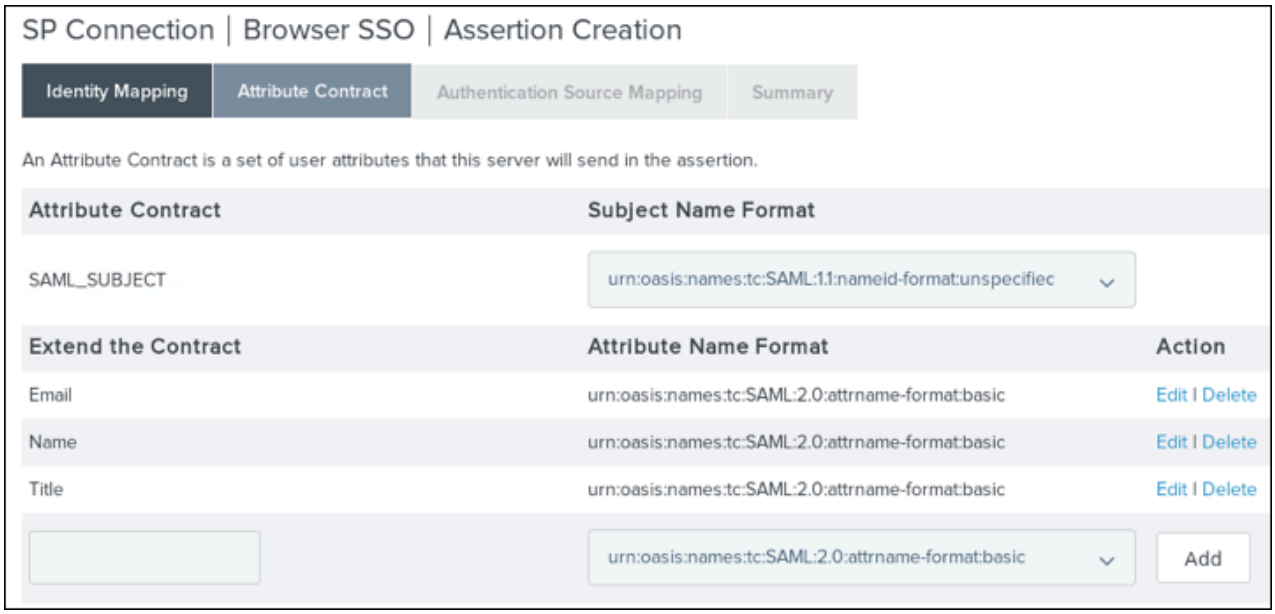


7. When you are finished click **Next**, the **Instance Configuration** page will display.
8. Click **Add a new row to 'Users'** to add a new user to the PCV.
9. Create a new user by entering the following (Note: User names are case sensitive):

   - USERNAME: **joe**
   - PASSWORD: **2Federate**

10. In the **Action** column click **Update**.

11. When you have finished adding the joe user click **Next**, the **Summary** page will display.



12. Review the configuration and when you are finished click **Save** to commit your changes.

Configure the HTML Form Adapter on the IdP

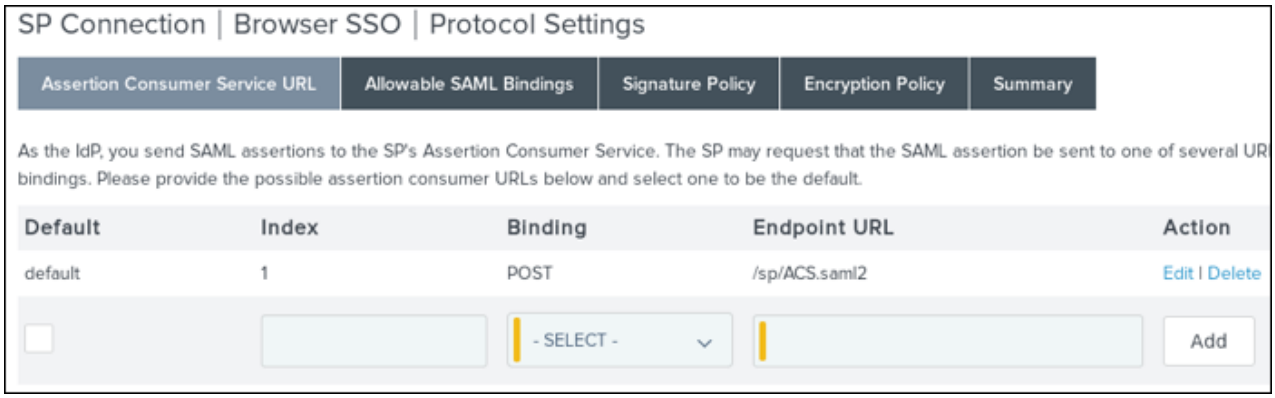
The PCV gives PingFederate a way to authenticate users, but in order for a user to authenticate they need a way to enter their credentials. PingFederate has a built-in, customizable HTML form which can be deployed for this purpose, this is the HTML Form Adapter.

1. Navigate to the **Authentication > Integration > IdP Adapters** page.

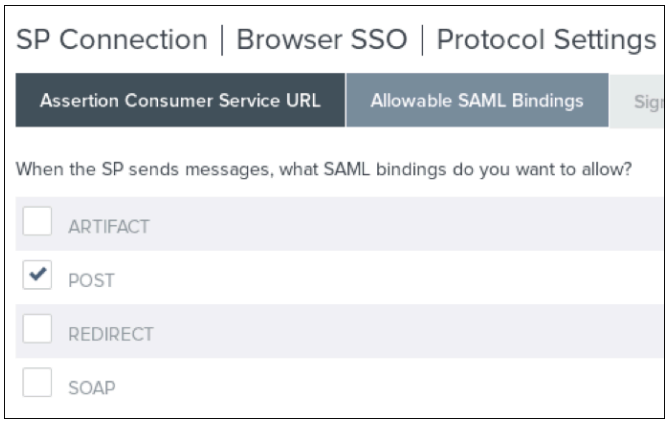
2. Click **Create New Instance**, the **Create Adapter Instance: Type** page will display.

3. Enter the following parameters:
   - INSTANCE NAME: **My Log In Form**
   - INSTANCE ID: **MyFormAdapter**
   - TYPE: **HTML Form IdP Adapter**
   - PARENT INSTANCE: **None**

   Just like the PCV configuration, the instance name is a human readable name you can give to the adapter and the instance ID is how PingFederate refers to the adapter internally.

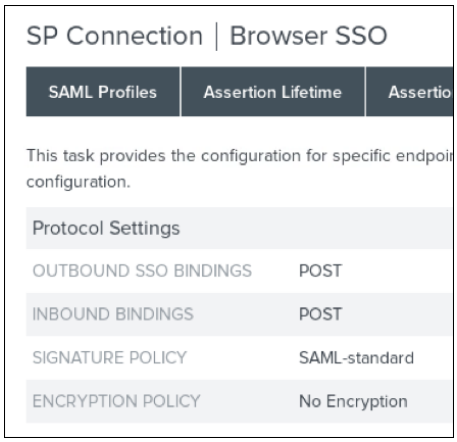


4. When you are finished click **Next**, the **IdP Adapter** page will display.
5. Click **Add a new row to 'Credential Validators'**.
6. In the PASSWORD CREDENTIAL VALIDATOR INSTANCE drop-down select **PCV: User Joe**.
7. In the Action column click **Update**.



   Assigning the PCV: User Joe password credential validator to the HTML Form Adapter provides PingFederate a method to validate user credentials. There are other methods of checking user credentials which will be covered later in this course.
8. Leave the remaining fields at their defaults and click **Next**, the **Extended Contract** page will display.
9. The Extended Contract page allows you to specify additional attributes that are not provided by the PCV. We will not be using this at this time. Click **Next** to continue, the **Adapter Attributes** page will display.
10. In the **Unique User Key Attribute** drop-down, select **username**. This is the value that will be used to identify a particular user across multiple adapters for session management purposes.

11. In the **username** row check the box under **Pseudonym**.



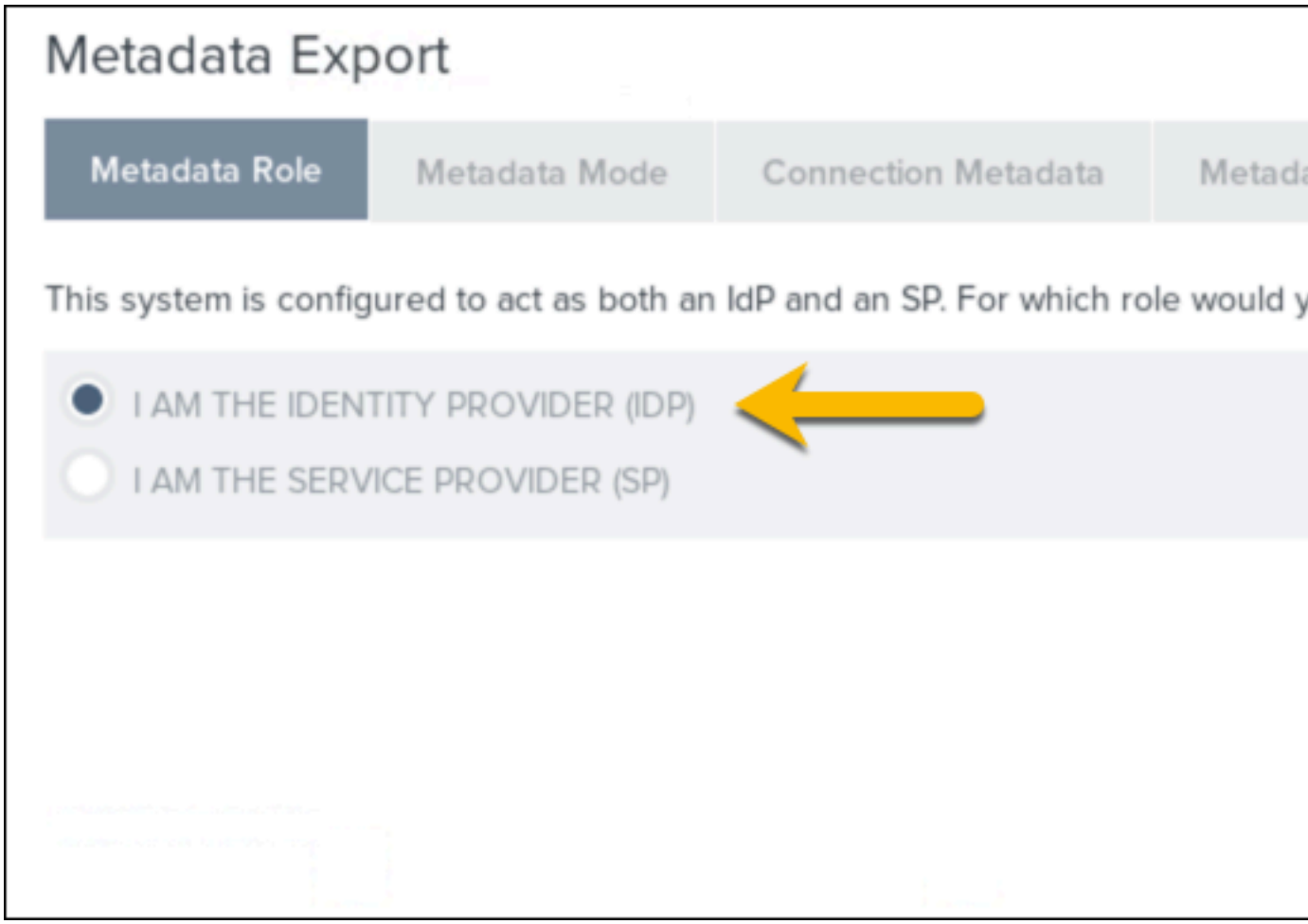
12. Click **Next**, the **Adapter Contract Mapping** page will display.
13. We will not be mapping any additional attributes into the adapter contract at this time. Click **Next** to continue, the **Summary** page will display.
14. Review the adapter configuration and when you are finished click **Save** to commit your changes.

Configure the Reference ID Adapter on the SP

The next step in our lab setup is going to be configuring the Reference ID adapter on the SP. The SP adapter acts as the last-mile integration and represents the link between our SP PingFederate and our application.

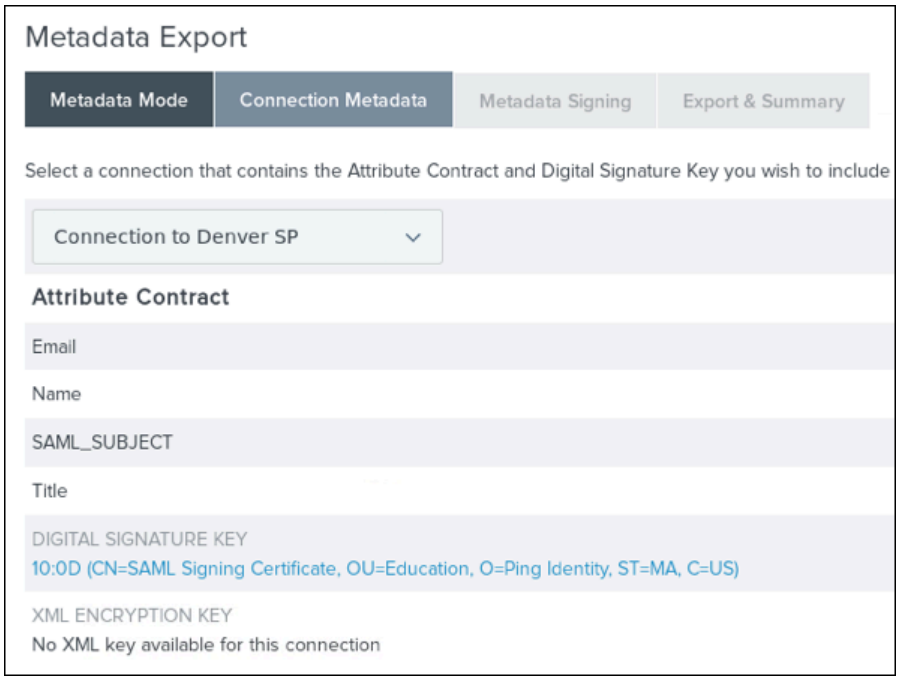
1. On the Firefox bookmark toolbar click **Admin Consoles** and then click the **PingFed SP Admin** link.
2. Enter the following credentials to sign into the admin console:
   - Username: **Administrator**
   - Password: **2Federate**
3. Navigate to the **Applications > Integration > SP Adapters** page.
4. Click the **Create New Instance** button, the **Create Adapter Instance – Type** page will display.

5. Enter the following parameters:
   - INSTANCE NAME: **Reference ID Adapter**
   - INSTANCE ID: **RefIDAdapter**
   - TYPE: **Reference ID SP Adapter**
   - PARENT INSTANCE: **None**



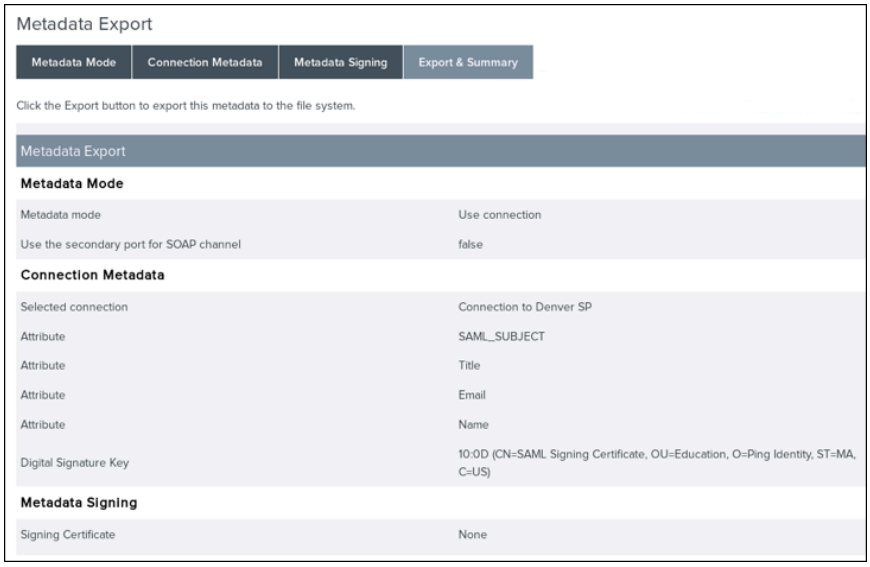
6. Click **Next**, the **Instance Configuration** page will be displayed.
7. Enter the following parameters, anything not mentioned here can be left at the defaults.
   - USER NAME: **pingUser**
   - PASS PHRASE: **2Federate**
8. Click **Show Advanced Fields**.
9. Verify that the OUTGOING ATTRIBUTE FORMAT drop-down is set to **JSON**.
10. Click **Next**, the **Actions** page will display.
11. Click **Next**, the **Extended Contract** page will display.
12. In the Extend the Contract column, enter **Email** into the text box.
13. Click **Add**.

14. Repeat steps 13-15 to add the following two attributes:
    - Name
    - Title



Note: These attributes are what PingFederate will be sending to the application. In the next lab, when you create the connections, you will populate these using attributes that are send by the IdP in the SAML assertion.

15. Click **Next** to continue, the **Target App Info** page will display.
16. In the APPLICATION NAME field enter **Agentless SP Sample App**.



17. Click **Next**, the **Summary** page will display.
18. Review the adapter summary, when you are finished click **Save** to commit your changes to the console.

Summary

Congratulations, you have successfully configured a PCV for authenticating users and an HTML Form Adapter for entering user credentials. You have also configured the SP adapter to connect to the sample application in your lab environment. This is the first step in configuring a connection for an IdP or an SP.

On the IdP PingFederate needs a way for users to enter their credentials for validation and a list of users to validate those credentials against. On the SP PingFederate needs a way to communicate the incoming user attributes to the target application. Both of these functions are managed by using the PingFederate adapters.

Lab 2 - Introduction to Connections

Introduction

This lab will guide you through the steps required to create the connections between your IdP and your SP PingFederate instances. Remember, that each side needs to have a connection configured and there are a number of parameters that have to match between the two parties.

This lab should take 75 minutes to complete.

Objectives

- Create a service provider connection (IdP to SP) on the IdP
- Export the connection metadata
- Use the IdP metadata to create an identity provider connection (SP to IdP) on the SP
- Test the connection

Description of tasks to perform

Before your users can begin using browser SSO in your environment you will need to connect to one or more service providers. In this lab, you will act as the identity provider and configure a connection to a service provider. As part of the configuration you will need to specify which partner you are connecting to, what attributes your connection will send, where the attributes will come from, and whether you will be signing your assertions. Finally, you will export a metadata file and use this file to facilitate creating the connection in the SP instance.

Create a Connection to a Service Provider

As the identity provider, you have to create an assertion with attributes and send those attributes to a service provider. What remains on the identity provider setup is to define your connection to your service provider partner. Here is an outline of the steps you will take in the next exercise:

- Define and describe the connection. How do you want to refer to this connection?
- Provide service provider connection details. Who are you connecting to?
- Define an attribute contract. What attributes are you sending?
- Select an adapter. How are you authenticating the user?
- Define assertion mapping details. Which values from the adapter go with which assertion attributes?
- Define the federation protocol. How are you sending the assertion?
- Configure signing credentials. How are you assuring the validity of the assertion?

- Select digital signing options. Do you want to provide extra validity?
- Select encryption options. Do you want to provide additional security?

Create a New Service Provider Connection

1. From the IdP admin console, navigate to **Applications > Integration > SP Connections**.
2. Click the **Create Connection** button.
3. We will not be using a template for this connection. Click **Next**, the **SP Connection - Connection Type** screen will display.
4. Check the **BROWSER SSO PROFILES** checkbox.
5. Ensure **SAML 2.0** is selected in the **PROTOCOL** drop-down
6. Click **Next**, the **Connection Options** screen will display.
7. Ensure the **BROWSER SSO** box is checked.
8. Click **Next**, the **Import Metadata** screen will display.
9. You will not be importing metadata for this connection so no changes are necessary. Ensure the **NONE** radio button is selected.
10. Click **Next**, the **General Info** page will display.
11. On the General Info page enter the following parameters:

    - Partner's Entity ID (CONNECTION ID): **https://sso.int.den-ping.com**

    This is the SAML entity ID of your Service Provider, which you would have received in advance.

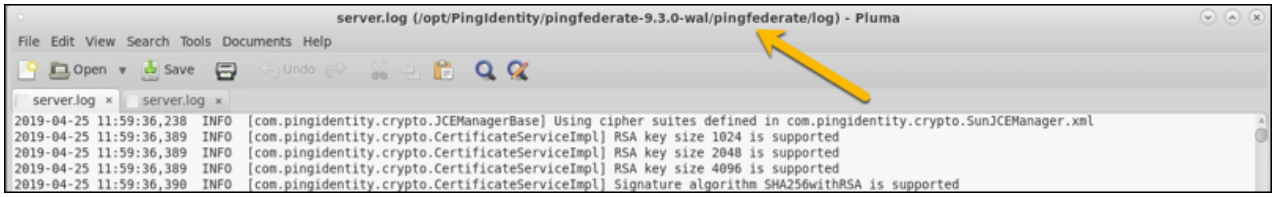
    - Connection Name: **Connection to Denver SP**

    The connection name is a descriptive field which will appear in the connection list on the admin console.

    - Base URL: **https://sso.int.den-ping.com:9032**

    The base URL is the fully qualified hostname and port on which your partner's federation server runs.

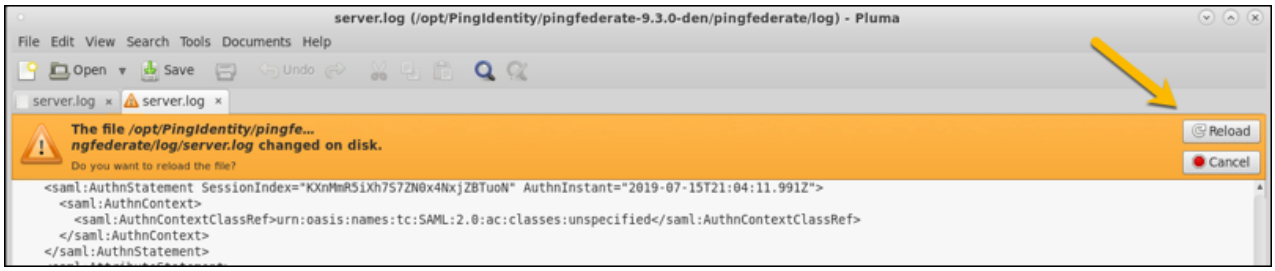
    Note: Any fields not mentioned above should be left at their defaults.



12. Click **Next**, the **Browser SSO** screen displays.

Configure Browser SSO

1. Click **Configure Browser SSO**, the **Browser SSO - SAML Profiles** screen displays.
2. In the Single Sign-On (SSO) Profiles column check the boxes for both **IDP-INITIATED SSO** and **SP-INITIATED SSO**.



3. Click **Next**, the **Assertion Lifetime** page displays.
4. Leave both fields at the default of 5 minutes and click **Next**, the **Assertion Creation** page displays.

Configure Assertion Creation

1. Click the **Configure Assertion Creation** button, the **Assertion Creation - Identity Mapping** page displays.
2. Make sure the **STANDARD** radio button is selected.
3. Click **Next**, the **Attribute Contract** page will display.
4. In the Extend the Contract column, enter **Email** into the text box.
5. Leave the Attribute Name Format drop-down at the default.
6. Click **Add**.
7. Repeat steps 18-20 to add the following two attributes:
   - Name
   - Title

   Note: An attribute contract is the set of user attributes that you and your partner have agreed will be sent in the SAML assertions for this connection. You identify these attributes on this screen. Attribute names are case sensitive and must correspond to the names expected by your partner. By default, the SAML_SUBJECT attribute is required. We are also choosing to send three additional attributes in our assertions to the service provider.
8. Click **Next**, the **Authentication Source Mapping** page will display.

Map the Adapter Instance

1. Click the **Map New Adapter Instance** button, the **IdP Adapter Mapping - Adapter Instance** page will display.
2. In the ADAPTER INSTANCE drop-down select **My Log In Form**.
3. Click **Next**, the **Mapping Method** page will display.
4. Make sure the **USE ONLY THE ADAPTER CONTRACT VALUES IN THE SAML ASSERTION** radio button is selected.
5. Click **Next**, the **Attribute Contract Fulfillment** page will display.
6. For each item in the Attribute Contract column enter the following for Source and Value.
   - Email - **Text**, **Email Placeholder**
   - Name - **Text**, **Name Placeholder**
   - SAML_SUBJECT - **Adapter**, **username**
   - Title - **Text**, **Title Placeholder**

   Note: This is telling PingFederate to send the attributes on the left to the service provider, filling in with the values on the right. We are hardcoding all of this, telling the identity provider exactly what to send, the same for every user. This will not normally be the case and we will update these values in a later lab.
7. Click **Next**, the **Issuance Criteria** page will display.
8. You will not be configuring any issuance criteria, leave everything at the defaults. Click **Next**, the **Summary** page will display.
9. Review the summary and click **Done**, the **Assertion Creation – Authentication Source Mapping** page will display.

   Note: Now that you have configured an adapter instance for this connection you can see it on the screen.
10. Click **Next**, the **Summary** page will display.
11. Review the Summary then click **Done**, the **Browser SSO – Assertion Creation** page will display.

    Note: Now that the assertion has been configured you can see the configured attributes on this page.

Configure Protocol Settings

You have told PingFederate how to create the assertion, but you will still need PingFederate to actually send it to the service provider.

1. Click **Next**, the **Protocol Settings** page will display.
2. Click the **Configure Protocol Settings** button, the **Protocol Settings - Assertion Consumer Service URL** page will display.
3. In the Default column check the check-box.
4. In the Index column enter **1** (one).
5. In the Binding drop-down select **POST**.
6. In the Endpoint URL column enter **/sp/ACS.saml2**.

   Note: The endpoint URL is case sensitive. This is a relative path from the base URL entered in the General Information Screen. This information would be supplied by your service provider partner.
7. Click **Add**.
8. Click **Next**, the **Allowable SAML Bindings** page will display.
9. Select only the **POST** parameter and uncheck the rest.
10. Click **Next**, the **Signature Policy** page will display.

Configure Signature and Encryption

To coordinate digital signature policy, partners must first agree about whether they will sign SAML messages or tokens. In some cases, the protocol specifications require signatures – for example, all SAML STS tokens and all SSO assertions sent across the POST binding must be signed. The PingFederate administrative console and the runtime protocol engine enforce these requirements; you do not need to keep track.

Other uses of the digital signatures are optional between partners and enforced if specified for a partner connection. You have determined with your service provider partner to have minimal signatures and encryption.

1. Leave the signature policy options unchecked.
2. Click **Next**, the **Encryption Policy** page will display.
3. Leave the encryption policy page at the default with the **NONE** radio button selected.
4. Click **Next**, the **Summary** page will display.
5. Review the configuration and when you are finished click **Done**, the **Browser SSO – Protocol Settings** page will display.
6. You can review the protocol settings. When you are finished click **Next**, the **Summary** page will display.
7. Review the configuration summary.
8. When you are finished click **Done**, the **SP Connection – Browser SSO** page will display.

Configure Digital Signing Credentials

The next step is to configure how you will sign the assertion. You can create a self-signed certificate or use an existing certificate. For this lab you will use a certificate that has already been prepared.

1. Click **Next**, the **Credentials** page will display.
2. Click the **Configure Credentials** button, the **Credentials - Digital Signature Settings** page will display.
3. Click the **Manage Certificates** button, the **Manage Digital Signing Certificates** page will display.
4. Click the **Import** button, the **Import Certificate - Import Certificate** page will display
5. Click the **Choose File** button, the **File Upload** dialog will display.
6. Navigate to the `home/training/certs/wal-ping` directory.

7. Select the `saml.wal-ping.com.p12` certificate file and click **Open**.
8. In the PASSWORD field enter **2Federate**.
9. Click **Next**, the **Summary** page will display.
10. Ensure the **MAKE THIS THE ACTIVE CERTIFICATE** checkbox is selected.
11. Review the summary page, when you are finished click **Save**. The **Certificate Management** page will display. You will now see your imported certificate.
12. Click **Done**, the **Credentials - Digital Signature Settings** page will display.
13. The newly imported certificate is set to be used for signing all digital assertions.
14. Click **Next**, the **Summary** page will display.
15. Review the summary. When you are finished click **Done**, the **SP Connection - Credentials** page will display.
16. Click **Next**, the **Activation & Summary** page will display.
17. The new connection should default to enabled, check the Connection Status toggle to ensure it is set to **enabled** (green).
18. Review the connection summary. When you are finished click **Save** to commit your changes.

Export the Metadata File

Now that you have configured a connection to your service provider, you as the identity provider will create a metadata file to send to your service provider partner. The metadata file contains XML-formatted data that aids your partner in establishing a connection to your identity provider site.

You can create metadata from an existing connection or from scratch. Creating from an existing connection is easier because information about the assertion attributes and signing of the assertion is already defined in the connection. When you create the metadata from scratch you must provide this information.

1. Navigate to the **System > Protocol Metadata > Metadata Export** page.
2. Select the **I AM THE IDENTITY PROVIDER (IDP)** radio button.
3. Click **Next**, the **Metadata Mode** page will display.
4. Select the **USE A CONNECTION FOR METADATA GENERATION** radio button.
5. Click **Next**, the **Connection Metadata** page will display.
6. In drop-down list select **Connection to Denver SP**, the Attribute Contract section will be back filled with the connection data.
7. Click **Next**, the **Metadata Signing** page will display.
8. Leave this page at the defaults, with no signing certificate selected, and click **Next**, the **Export & Summary** page will display.
9. Review the summary. At the bottom of the page click the **Export** button.
10. The file will be saved to the user's Download directory.
11. Click **Done** to return to the admin console.

Create a Connection to an Identity Provider

Now you have the tools in place to consume an identity assertion, received from an identity provider, and use it to create a session token for your service provider application. Eventually in your own organization you will configure a variety of connections that facilitate this process with a variety of session and token types.

In PingFederate here are the things you will need to set up:

- Define and describe the connection. How do you want to refer to this connection?
- Import identity provider metadata, this will provide most of the connection information for you.
- Define an attribute contract. What values are you expecting to get?
- Define assertion mapping details. Which values for the application go with which values from the assertion?
- Select an adapter for sending attributes. How are you sending the values to the application?
- Define the federation protocol. How are you receiving the assertion?
- Configure signing credentials. How are you checking the validity of the assertion?
- Select digital signing and encryption options. Do you want to provide extra security?

Create a New Identity Provider Connection Using Metadata

The first step to finalizing your configuration is to set up an identity provider connection. Earlier in this lab, you created a service provider connection on your PingFederate identity provider instance and used that connection to create a metadata file. In this section, you will import that metadata file into the SP instance of your environment and create an IdP connection.

1. If you have not done so already, log into the SP admin console.
2. Navigate to the Authentication > Integration > IdP Connections page.
3. Click the Create Connection button, the IdP Connections - Connection Type page will display.
4. Ensure the BROWSER SSO PROFILES checkbox is checked.
5. Click Next, the Connection Options page will display.
6. Check the BROWSER SSO checkbox.
7. Click Next, the Import Metadata page will display.
8. Select the FILE radio button.
9. Click the Choose file button, the File Upload window will display.
10. Navigate to Home > Downloads and select your `metadata.xml` file.
11. Click Next, the Metadata Summary page will display.
12. Review the Entity ID of the identity provider, you will also notice that the metadata in the file is unsigned.
13. Click Next, the General Info page will display.

    Note: The connection information has been populated from the metadata file sent from the identity provider. Review the connection information.
14. You will want to give your connection a friendlier name. In the CONNECTION NAME field enter Connection to Waltham IdP.
15. Click Next, the Browser SSO page will display.

Configure Browser SSO

1. Click the Configure Browser SSO button, the Browser SSO – SAML Profiles page will display.
2. Check the boxes for both IDP-INIATED SSO and SP-INITIATED SSO.
3. Click Next, the User-Session Creation page will display.
4. Click the Configure User-Session Creation button, the User-Session Creation – Identity Mapping page will display.
5. Ensure the ACCOUNT MAPPING radio button is selected.
6. Click Next, the Attribute Contract page will display.

7. These attributes came in the metadata file, this is what the identity provider will send. Review the attributes.

8. Click **Next**, the **Target Session Mapping** page will display.

Mapping to Your Adapter

You must map at least one adapter to your connection. You configured the Reference ID adapter previously; now you have to tell PingFederate to use it. This is telling the service provider PingFederate how the attributes from the SAML assertion, like the user's name and email, will get passed from PingFederate to the target application.

1. Click the **Map New Adapter Instance** button, the **Adapter Mapping & User Lookup – Adapter Instance** page will display.
2. In the ADAPTER INSTANCE drop-down select **Reference ID Adapter**.

3. Click **Next**, the **Adapter Data Store** page will display.
4. All of the attributes required to fulfill this adapter's contract are contained in the SAML assertion. Ensure the **USE ONLY THE ATTRIBUTES AVAILABLE IN THE SSO ASSERTION** radio button is selected.
5. Click **Next**, the **Adapter Contract Fulfillment** page will display.
6. Since you will be fulfilling the adapter contract from values contained only in the assertion select **Assertion** for each drop-down in the Source column.
7. Map the assertion data to the adapter contract using the following Adapter Contract/Value pairs:
   - Email / Email
   - Name / Name
   - Title / Title
   - subject / SAML_SUBJECT

   Note: You are telling Pingfederate to take the values of the attributes from the assertion and pass them to the application.
8. Click **Next**, the **Issuance Criteria** page will display.
9. You will not use any issuance criteria for this connection. Click **Next**, the **Summary** page will display.
10. Review the configuration summary.

11. When you are finished click **Done**, the **User-Session Creation – Target Session Mapping** page will display.
12. Click **Next**, the **Summary** page will display.
13. Review the configuration summary.
14. When you are finished click **Done**, the **Browser SSO – User-Session Creation** page will display.
15. Review the configuration information.
16. When you are finished click **Next**, the **Protocol Settings** page will display.

Configure Protocol Settings

1. Click the **Configure Protocol Settings** button, the **Protocol Settings – SSO Service URLs** page will display.
2. The Binding and Endpoint URL values have been provided by the metadata file. Review the configuration.

3. Click **Next**, the **Allowable SAML Bindings** page will display.
4. Ensure only the **POST** checkbox is selected.

5. Click **Next**, the **Overrides** page will display.

6. In the DEFAULT TARGET URL text box enter **https://servapp.int.den-ping.com:8444/AgentlessSPSample/app**.

   Note: This is the default application the user will be sent to. The URL we are using here is for the SP Sample application. Hover over the SP Sample bookmark and you will see these URLs are identical.
7. Click **Next**, the **Signature Policy** page will display.

Configure Signatures and Encryption

1. Ensure the **USE SAML-STANDARD SIGNATURE REQUIREMENTS** radio button is selected.
2. Click **Next**, the **Encryption Policy** page will display.
3. Ensure the **NONE** radio button is selected.
4. Click **Next**, the **Summary** page will display.
5. Review the configuration summary.

6. When you are finished click **Done**, the **Browser SSO – Protocol Settings** page will display.
7. Click **Next**, the **Summary** page will display.
8. Review the browser SSO settings.
9. When you are finished click **Done**, the **IdP Connection – Browser SSO** page will display.
10. Click **Next**, the **Credentials** page will display.
11. The credentials used for this connection were provided by the metadata. This is the public saml.wal-ping.com certificate.
12. Click **Next**, the **Activation & Summary** page will display.
13. Ensure that the connection is **enabled** (selection toggle is green).
14. Review the configuration. When you are finished click **Save** to commit your changes and return to the admin console.
15. Your connection to the identity provider at wal-ping.com is complete

Test Connections With the Sample Applications

As the service provider, with the assistance of a metadata file, you have now configured a connection to an identity provider on the wal-ping.com domain.

You will now test the connection between the service provider and the identity provider. The Sample SP application has been preinstalled on your service provider. The sample application has been pre-configured for your lab environment.

Now you can test your configuration with an IdP initiated SSO. If your connection is set properly, the IdP will prompt you to authenticate using the HTML Form Adapter you created, and the PingFederate instance on the IdP will send an assertion to your service provider. When the service provider PingFederate receives the assertion, it will redirect you to the target application and pass the attributes using the Reference ID adapter you configured.

1. Open a new tab in Firefox
2. Select the **SSO to SP** bookmark folder and click the **SSO to SP** bookmark. This is equivalent to the user clicking on the bookmark to their service provider application.
3. Sign on with using the username/password of **joe**/**2Federate**.

   Note: Remember, this is the user that you configured in the Simple PCV on the IdP PingFederate.

4. If everything has been configured correctly you will be redirected to the service provider application. The user attributes will be displayed on the right side.

   Note: Remember that some of the user attributes are text placeholders being sent by the IdP. Even if a different user is used the Email, Title, and Name attributes will always be the same. You will update these attributes in a later lab to use values that are being pulled from an external data store.

Lab 3 - Review the Server Logs

Introduction

The server log records all PingFederate runtime and administrative events, including status and error messages that can be used for troubleshooting. In this lab exercise you will examine the server log in more detail.

This lab should take 45 minutes to complete.

Objectives

- Review the current log entries
- Perform an SSO using the connection you created in the previous labs
- Review the transaction in the identity provider logs
- Review the transaction in the service provider logs

Description of tasks to perform

In this exercise, you will perform various operations and observe the entries made in the logs on both the service provider and identity provider sides.

Follow an SSO Transaction in the PingFederate Logs

By default, the logging level in PingFederate is set to INFO level, not DEBUG level. This means that PingFederate will not, by default, log things like the assertion and mapping attributes. For your lab environment the log level in all PingFederate instances has already been set to DEBUG level.

Look at the Current Server Log

First, take a look at the current server log file for the identity provider (wal-ping.com).

1. In your file manager window navigate to the /opt/PingIdentity/pingfederate-12.0-wal/pingfederate/log directory.
2. Double-click the `server.log` file to open it, the server.log - Pluma window will display. This is the identity provider log and will track everything the identity provider PingFederate instance is doing until the point it sends the assertion to the service provider.
3. In the File Manager window navigate to the /opt/PingIdentity/pingfederate-12.0-den/pingfederate/log directory.

4. Locate the `server.log` file and double-click it to open it in Pluma. This is the service provider log and will track what the service provider PingFederate instance is doing from when it receives the assertion to when it passes the user attributes to the service provider application.

   Note: Both log files will be displayed as "server.log" on their respective tabs. You can hover over the tab to get the full path to the file. The full path will also be shown on the application title bar. Use this to determine if you are looking at the IdP (wal-ping) or the SP (den-ping) logs.

5. Go to the bottom of each log file.
6. Make a note of the time on your virtual machine, this will help you find the correct part of the log to look at after you run your transaction.
7. VM Time: _____
8. Open Firefox, on the bookmark toolbar click the SSO to SP > SSO to SP bookmark.
9. Log in using the joe/2Federate credentials.
10. When you are successfully redirected to the service provider application minimize your Firefox window and return to the Pluma window.
11. When prompted click the Reload button, you will need to do this as you select each file.

12. Begin with the identity provider log; double-check the timestamps to make sure you are at the beginning of the transaction.

Review the Transaction in the Identity Provider Server Log

The identity provider's log will show you everything that happened in the transaction up until the SAML assertion is sent to the service provider. With debug logging enabled PingFederate will show all steps, from the receipt of the authentication request, mapping the user attributes from the adapter, and then sending the assertion to the service provider as a SAML Response message.

1. Review the server log to see what type of information it contains. Some important items to watch for are listed below.

   Note: The log entries below will be truncated to remove the timestamp and some other log fields that are included on every log entry. They may also be formatted to fit in the lab guide and may look slightly different depending on if you have word wrap enabled or not. Any sections truncated for formatting will be annotated with elipses (...).

   - The initial request to PingFederate:

   ```
   GET: https://sso.int.wal-ping.com:9031/idp/startSSO.ping
   ```

   - Which attributes were taken from the identity provider adapter (source attributes) and which attributes were mapped into the SAML assertion (resulting attributes). HINT: look for the entries to find these:

   ```
   Source attributes: ... username=joe}, ... Resulting attributes:
   {username=joe}
   ```

```
Source attributes: ... Resulting attributes:{Email=Email Placeholder,
 Title=Title Placeholder, SAML_SUBJECT=joe, Name=Name Placeholder}
```

- At the beginning of the SAML message is the destination URL. This is the base URL plus the ACS (Assertion Consumer Service) URL configured in the connection. This line also includes the SAML version and timestamp (not shown here) for the SAML Response message:

```
OutMessageContext XML: <samlp:Response Version="2.0" ...
 Destination="https://sso.int.den-ping.com:9032/sp/ACS.saml2"
```

- Immediately below this is the issuer, this is the identity provider's entity ID:

```
... <saml:Issuer
 xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">https://sso.int.wal-
ping.com</saml:Issuer>
```

2. The actual SAML assertion is included as part of the SAML Response message and is contained within the *saml:Assertion* </saml:Assertion> tags. Some important items to look for in the assertion are listed below.

- The SAML issuer and issue instant. The issuer is the identity provider's entity ID, the issue instant is the time the assertion was created:

```
<saml:Assertion ... IssueInstant="2017-11-01T17:20:26.874Z"
 Version="2.0" ...
 <saml:Issuer>https://sso.int.wal-ping.com
 </saml:Issuer>
```

- The SAML subject:

```
<saml:Subject>
 <saml:NameID ... >joe</saml:NameID>
 <truncated>
</saml:Subject>
```

- The conditions of the assertion, including the audience restriction. The audience restriction is the entity ID of the service provider. The service provider will check the audience restriction to ensure the assertion is intended for it.

```
<saml:Conditions NotBefore="2017-11-01T17:15:26.874Z"
 NotOnOrAfter="2017-11-01T17:25:26.874Z">
 <saml:AudienceRestriction>
  <saml:Audience>
  https://sso.int.den-ping.com
  </saml:Audience>
 </saml:AudienceRestriction>
</saml:Conditions>
```

- The attributes being sent. Each attribute will include an attribute name and an attribute value as shown below:

```
<saml:AttributeStatement>
 <saml:Attribute Name="Email" ...
  <saml:AttributeValue ...
  Email Placeholder</saml:AttributeValue>
 </saml:Attribute>
 <saml:Attribute Name="Title" ...
  <saml:AttributeValue ...
  Title Placeholder</saml:AttributeValue>
 </saml:Attribute>
 ...
```

```
</saml:AttributeStatement>
```

- Below the assertion you will have the entity ID of the service provider, the binding being used, and the endpoint that the assertion is being sent to:

```
entityId: https://sso.int.den-ping.com (SP)
virtualServerId: https://sso.int.wal-ping.com
Binding: ...:HTTP-POST
Endpoint: https://sso.int.den-ping.com:9032/sp/ACS.saml2
```

Reviewing the Transaction in the Service Provider's Log

The service provider's logs will show you everything that happens in the transaction after the service provider receives the assertion from the identity provider. This will include a copy of the assertion, followed by the assertion attributes being mapped to the adapter and sent to the service provider's application.

Note: The service provider PingFederate instance has already been set to DEBUG logging level for you.

1. In your Pluma window select the service provider's server.log file, if prompted reload the file.
2. The service provider log will contain a copy of the assertion as it was received from the identity provider. Remember, SAML assertions are part of the SAML standard, and this assertion may have come from an identity provider that is not using PingFederate.
3. The service provider PingFederate will first check the issuer, which is the identity provider's entity ID, and will match that to what is configured in the connection.

```
<saml:Issuer>
 https://sso.int.wal-ping.com
</saml:Issuer>
```

4. At the bottom of the assertion you will find the entity ID, binding, and if the signature was valid:

```
entityId: https://sso.int.wal-ping.com (IDP)
virtualServerId: https://sso.int.den-ping.com
Binding: <truncated>:bindings:HTTP-POST
SignatureStatus: VALID
```

5. Below that you will find the attributes that PingFederate took from the assertion and how they were mapped to the adapter:

```
Incoming Assertion Attributes: ... Email=Email Placeholder, ...
 Title=Title Placeholder, SAML_SUBJECT=joe, ... Name=Name Placeholder

Source attributes:... TargetResource=https://servapp.int.den-
ping.com:8443/AgentlessSPSample/app, Email=Email Placeholder, ...
 Title=Title Placeholder, SAML_SUBJECT=joe, <truncated> Name=Name
 Placeholder}, ... Resulting attributes:{Title=Title Placeholder,
 Email=Email Placeholder, subject=joe, Name=Name Placeholder}

Mapped SP adapter Attributes: {Title=Title Placeholder, Email=Email
 Placeholder, subject=joe, Name=Name Placeholder}
```

Lab 4 - Attribute Mapping

Introduction

Until this point you have only used attributes provided by an IdP adapter or hard coded attributes. PingFederate can also pull attributes from one or more data sources such as a database or an LDAP directory.

This lab should take 60 minutes to complete.

Objectives

- Add a data store to your identity provider configuration
- Modify the existing identity provider adapter to use the data store
- Modify the connection to use the attributes from the data store in the SAML assertion
- Test the new configuration and make sure all attributes are sent as expected

Description of tasks to perform

You will configure PingFederate to look up users in an OpenLDAP directory. PingFederate will use this directory to pull user information to send in the assertion.

The directory is pre-configured with two users, smanager and jsaml. You will configure PingFederate to look up the users in OpenLDAP by their username. The username will be provided by the HTML Form Adapter you configured in the previous labs.

Currently the HTML Form adapter authenticates users against a simple Password Credential Validator. You will add a user to the PCV by the name of jsaml, so that jsaml can authenticate against the PCV and PingFederate can look them up in the OpenLDAP directory.

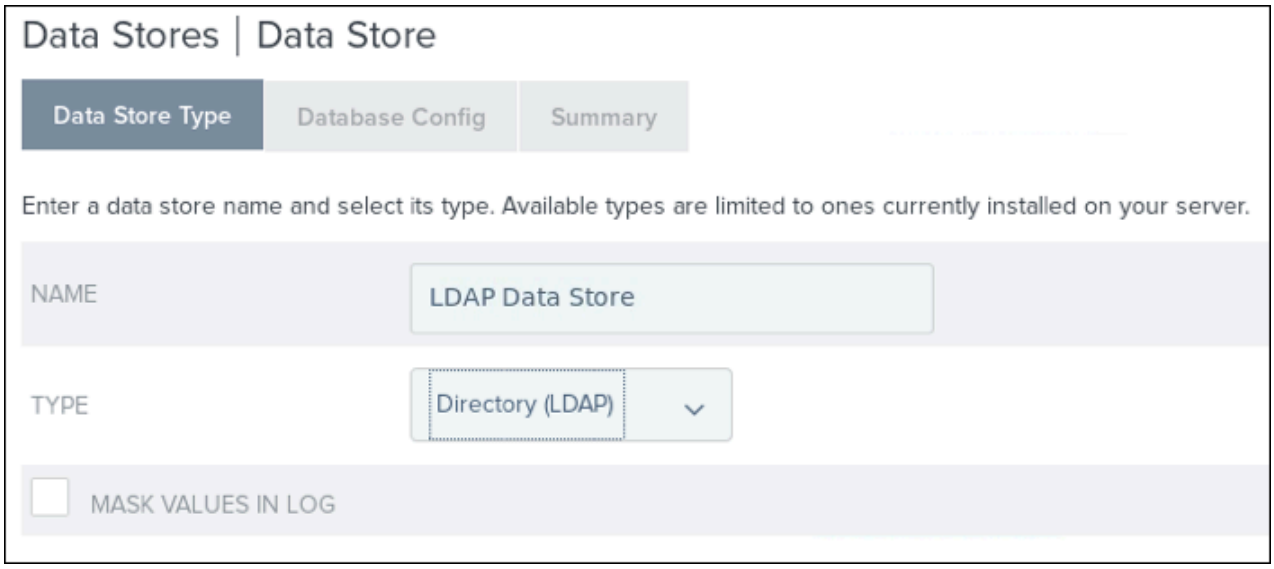
Add a Data Store

First, you will need to configure your identity provider PingFederate to pull user attributes from the OpenLDAP directory service.

The OpenLDAP directory is already configured on your virtual machine. You will need to connect PingFederate to the directory, then add the directory mapping to the connection you built in the previous lab. Then, you will check the identity provider server log to see what has changed.#

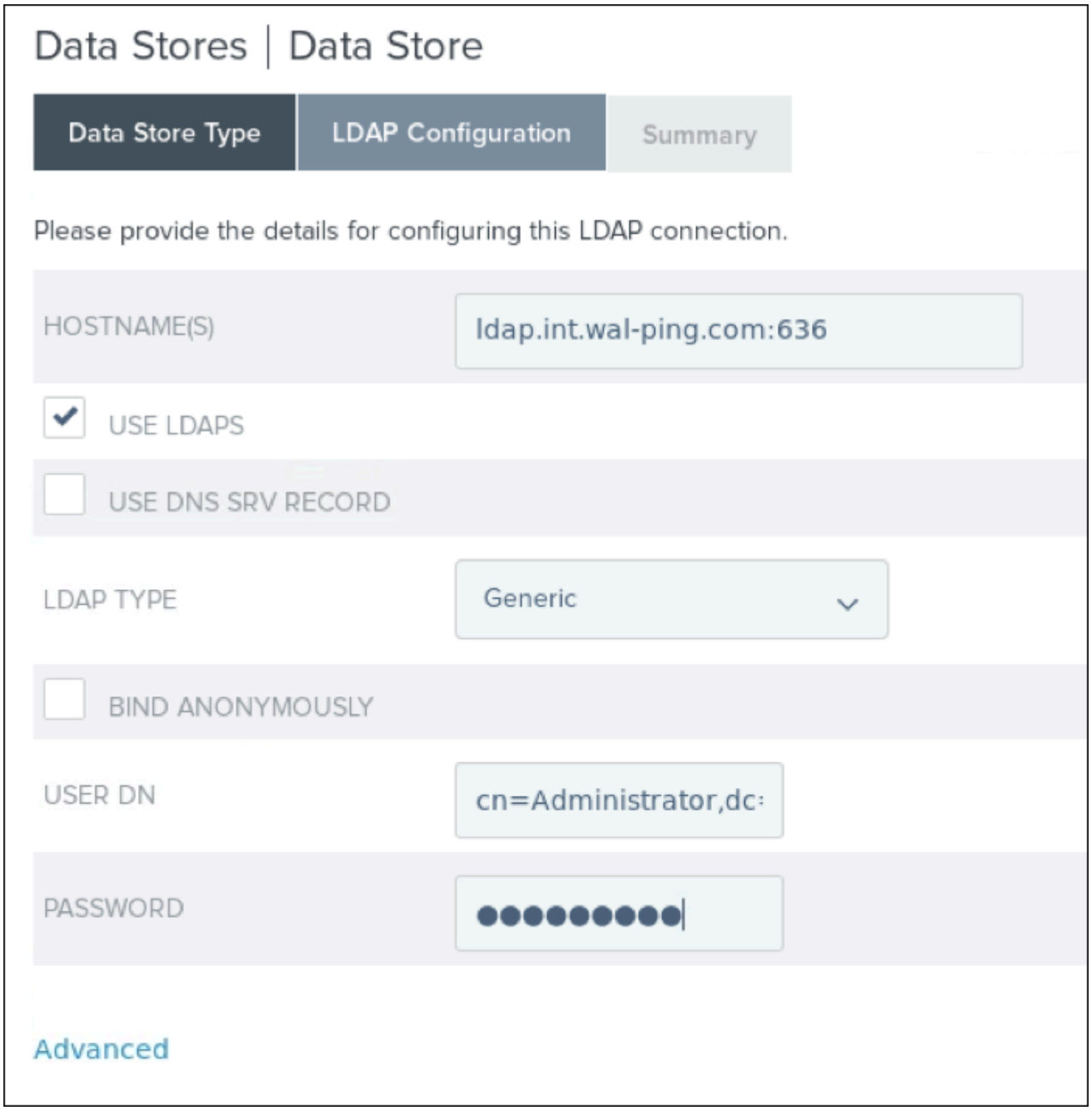
1. Open Firefox and navigate to the **IdP Admin Console**.
2. Navigate to the **System > Data & Credential Stores > Data Stores** page.
3. Click the **Add New Data Store** button, the **Data Stores – Data Store Type** page will display.
4. In the NAME field enter **LDAP Data Store**.

5. In the TYPE drop-down select **Directory (LDAP)**.



6. Click **Next**, the **LDAP Configuration** page will display.
7. Enter the following parameters, make sure to click the **Add** button next to the hostname:
   - HOSTNAME: **ldap.int.wal-ping.com:636**
   - LDAP TYPE: **Generic**
   - USER DN: **cn=Administrator,dc=wal-ping,dc=com**
   - PASSWORD: **2Federate**

8. Check the box for **USE LDAPS**.

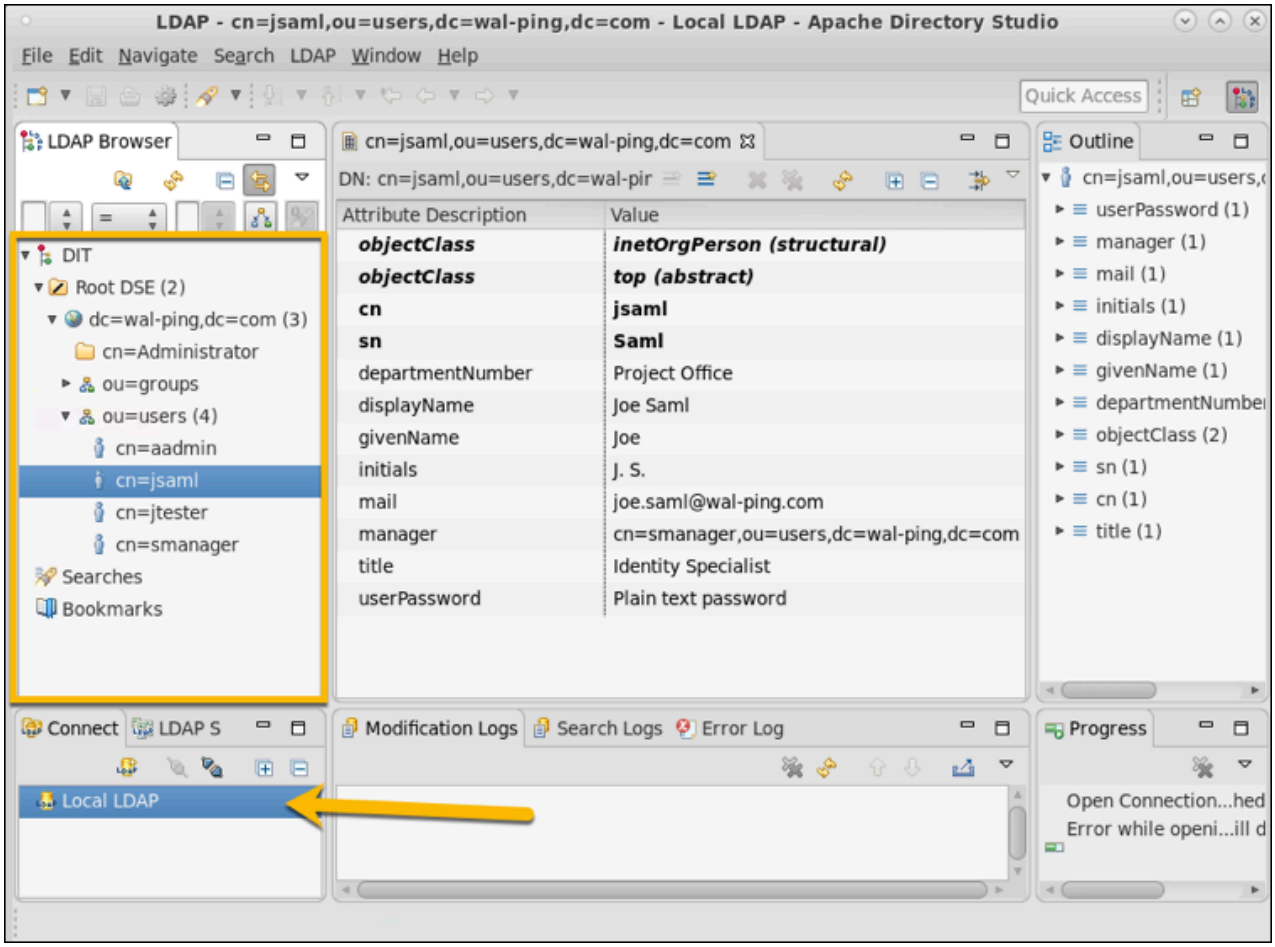


9. Click **Next**, the **Summary** page will display.
10. Review the summary.
11. When you are finished click **Save** to commit the configuration.

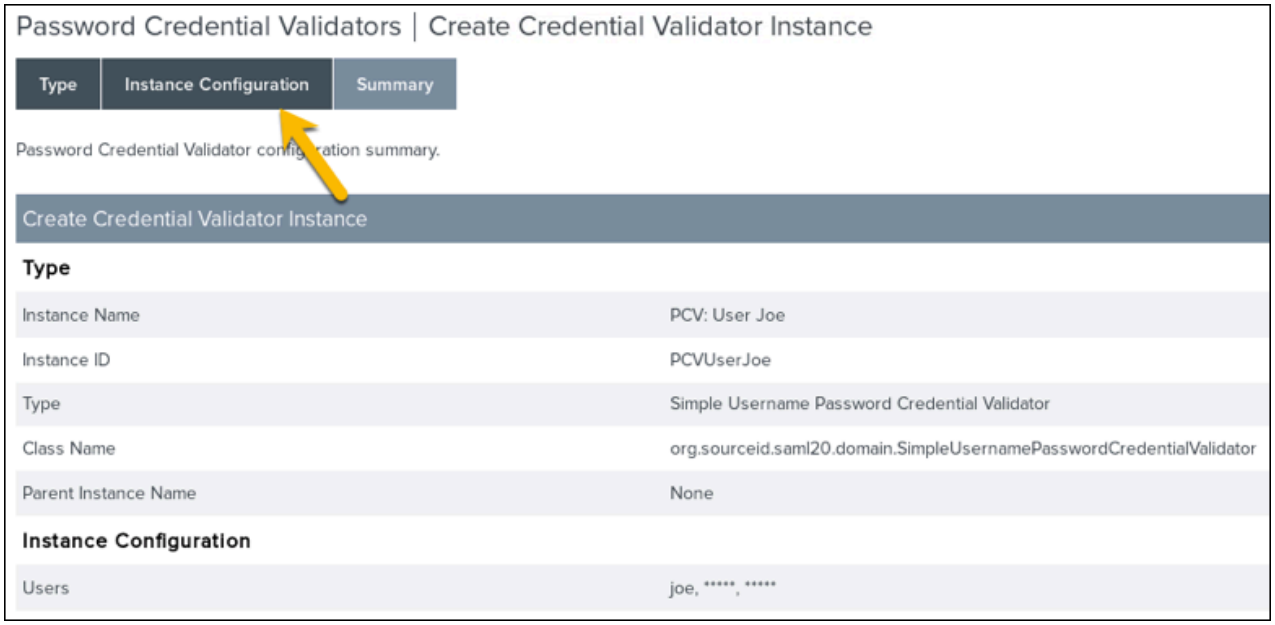
Add a New User to the PCV

Now PingFederate can connect to the OpenLDAP directory. The current user, joe, is not in the OpenLDAP directory. Although you can authenticate him with your password credential validator you won't be able to get any additional information about him. However, a user that is in the directory is Joe Saml, with a username of jsaml. You will add jsaml to the simple password credential validator. This way, Joe Saml can authenticate against the HTML Form adapter, which checks credentials against the simple PCV list.

If you want to view the users and attributes in the OpenLDAP directory you can use the Apache Directory Studio application. The application icon is located on the desktop of your lab environment.



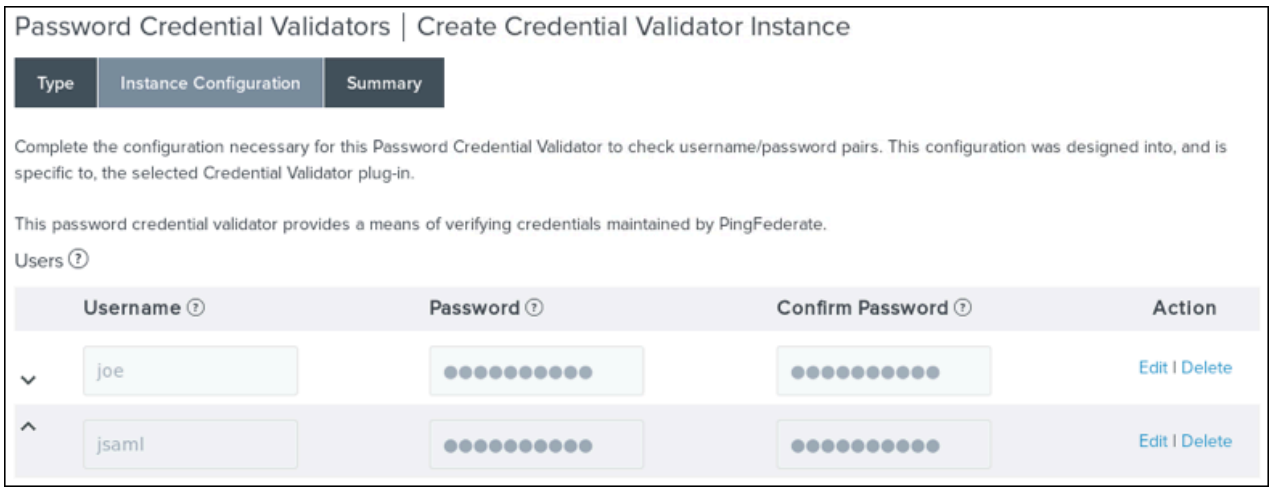
1. In the IdP Admin Console navigate to the System > Data & Credential Stores > Password Credential Validators page.
2. Click the PCV: User Joe instance, the Summary page will display.
3. Click the Instance Configuration tab, the Instance Configuration page will display.



4. Click Add a new row to 'Users'.

5. Add a new user with USERNAME **jsaml** and PASSWORD **2Federate**.

   Note: Later in this lab we will use the username attribute returned from the HTML Form Adapter to search the OpenLDAP directory. However, we are still using the Simple PCV to validate users, and there is no "joe" user in the OpenLDAP directory. But there is a "jsaml" user in the OpenLDAP directory, so we are adding that user name to the Simple PCV so the directory lookup will return user data.
6. Click **Update**.



7. Click **Save** to commit your changes.
8. Use the following steps to test the new account and make sure it authenticates correctly:
   - Close and reopen Firefox to clear your cache and cookies.
   - In the toolbar, click the **SSO to SP > SSO to SP** bookmark.
   - Sign in with the new account credentials: **jsaml**/**2Federate**.
   - A successful login will take place.

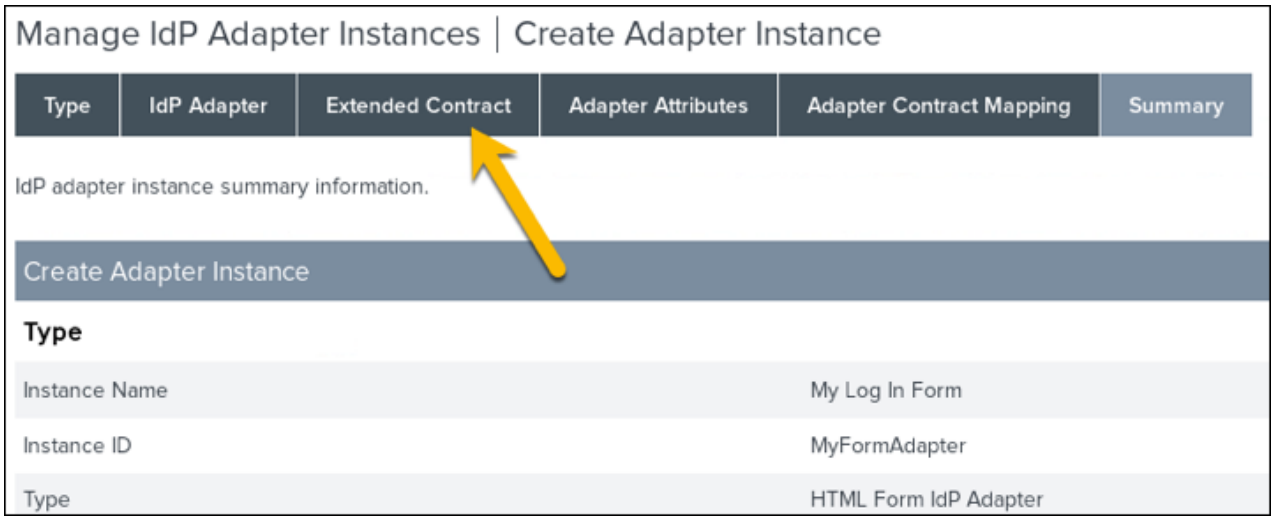
Modify the Adapter

Now PingFederate can retrieve user attributes for a user that is found in the OpenLDAP directory. The next step is to configure your HTML Form Adapter to lookup users in the OpenLDAP directory and retrieve some additional attributes. Then, PingFederate can send those attributes to the service provider in the assertion.

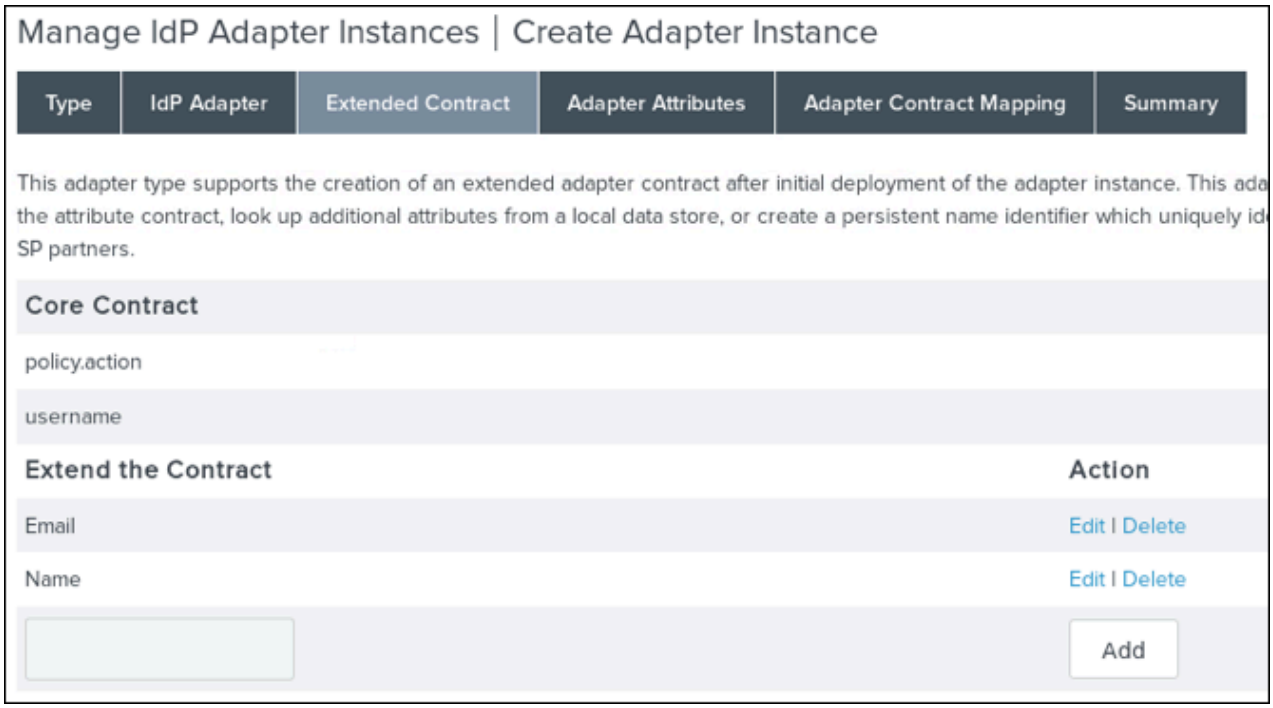
Add the Directory Lookup to the Adapter

1. Open Firefox and navigate to the **IdP Admin Console**.
2. Navigate to the **Authentication > Integration > IdP Adapters** page.
3. Click the **My Log In Form** adapter instance, the **Summary** page will display.

4. Click the **Extended Contract** tab, the **Extended Contract** page will display.

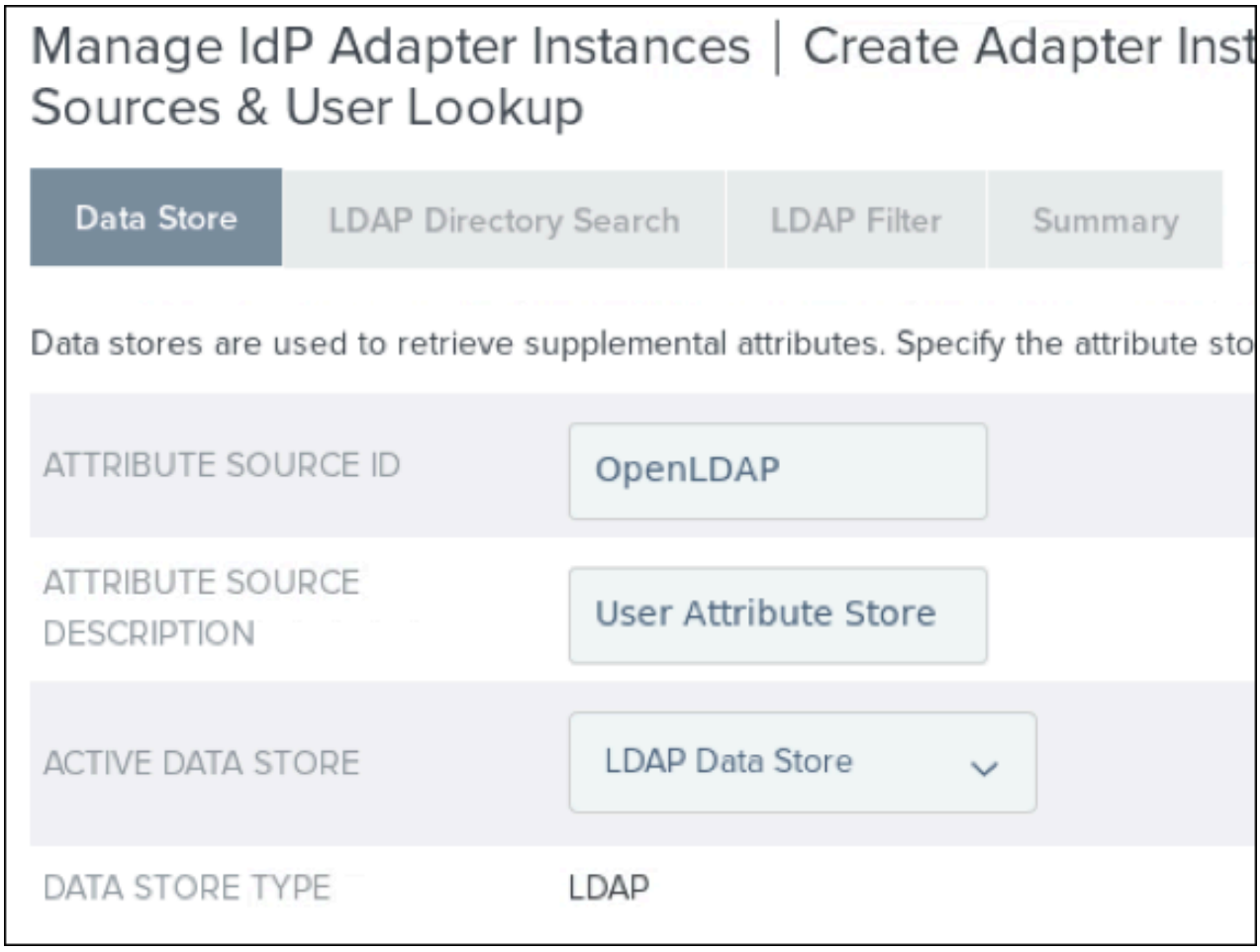


5. In the Extend the Contract text field enter **Name**.
6. Click **Add**.
7. Repeat steps 6-7 to add **Email** to the contract.



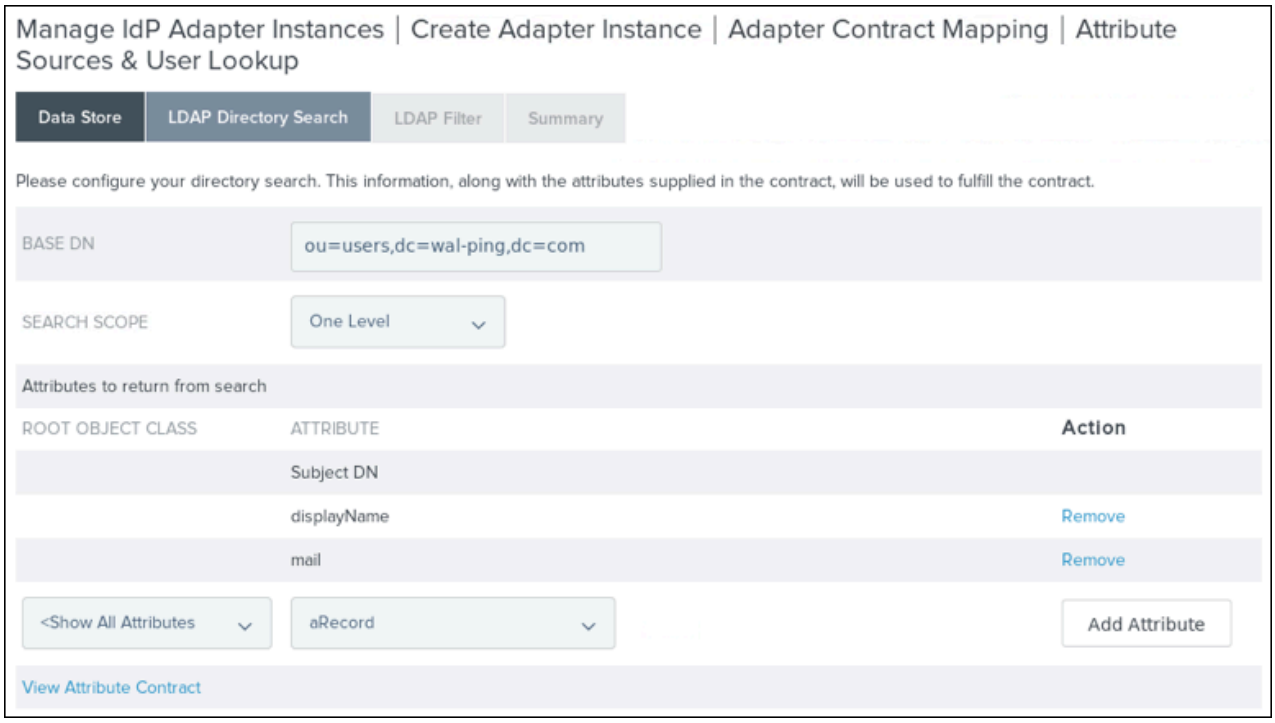
8. Click the **Adapter Contract Mapping** tab, the **Adapter Contracting Mapping** page will display.
9. Click the **Configure Adapter Contract** button, the **Adapter Contract Mapping – Summary** page will display.
10. Click the **Attribute Sources & User Lookup** tab, the **Attribute Sources & User Lookup** page will display.
11. Click the **Add Attribute Source** button, the **Attribute Sources & User Lookup – Data Store** page will display.

12. Enter the following parameters:
    - ATTRIBUTE SOURCE ID: OpenLDAP
    - ATTRIBUTE SOURCE DESCRIPTION: User Attribute Store
    - ACTIVE DATA STORE: LDAP Data Store



13. Click **Next**, the **LDAP Directory Search** page will display.
14. Enter the following parameters:
    - BASE DN: ou=users,dc=wal-ping,dc=com
    - SEARCH SCOPE: One Level
15. In the **ROOT OBJECT CLASS** drop-down select **Show All Attributes**.
16. In the **ATTRIBUTE** drop-down select **mail**.
17. Click the **Add Attribute** button.

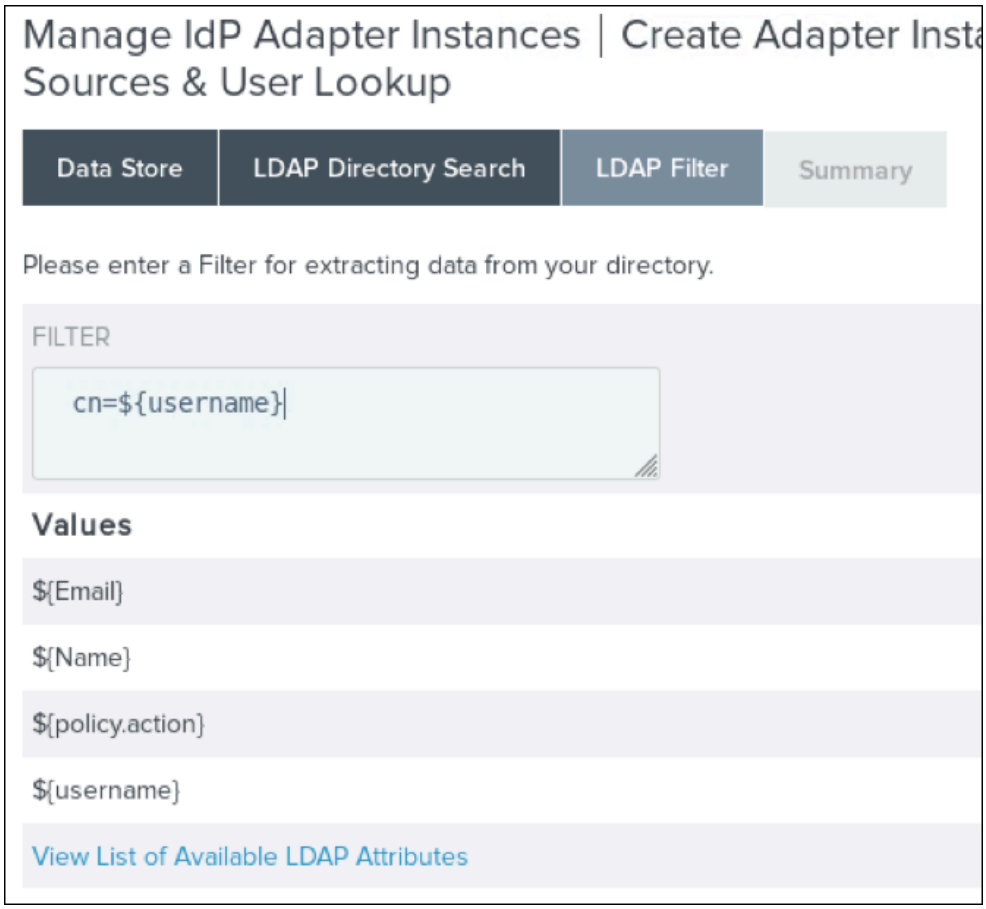
18. Repeat steps 17-18 to add the **displayName** attribute.



Note: Now when any user authenticates using the HTML Form adapter, PingFederate will pull these user attributes from the OpenLDAP directory.

19. Click **Next**, the **LDAP Filter** page will display.

20. In the FILTER field enter **cn=${username}**.

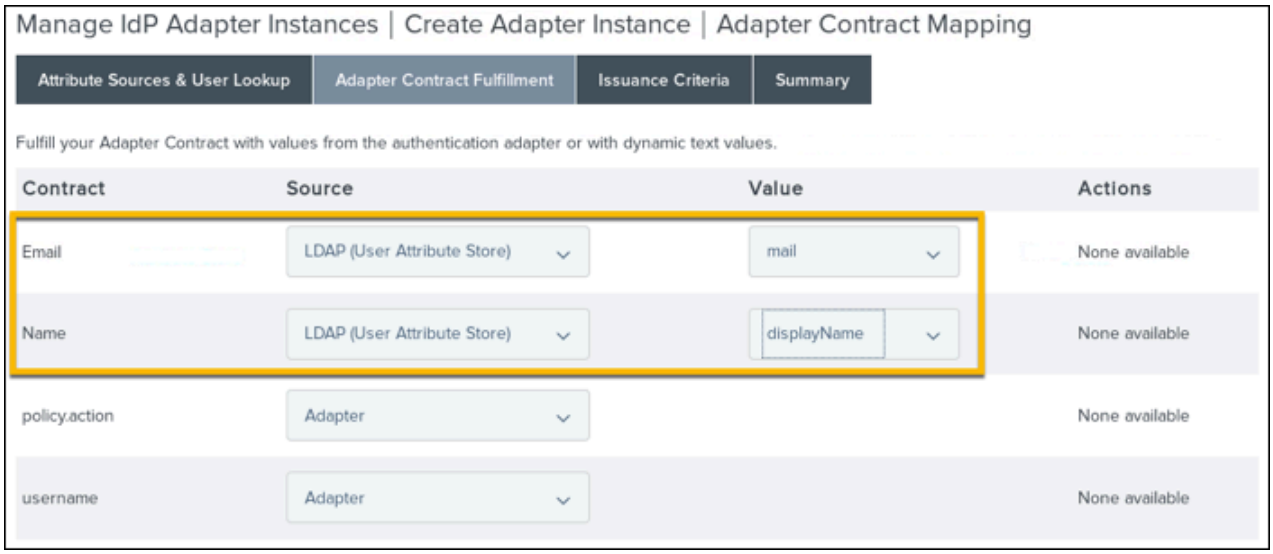


Note: This settings, will tell PingFederate which user to look for in the directory. PingFederate will search for a user whose username in the database is the same as the username entered on the HTML Form adapter. This is why we added "jsaml" to our Simple PCV earlier. There is no "joe" username in our OpenLDAP directory.

21. Click **Next**, the **Summary** page will display.
22. Review the configuration summary.
23. When you are finished click **Done**, the **Adapter Contract Mapping – Attribute Sources & User Lookup** page will display.
24. Click **Next**, the **Adapter Contract Fulfillment** page will display.

    Note: You are now mapping the adapter again. This tells PingFederate which attributes to return when this HTML Form adapter is used.
25. In the **Source** column for the **Email** attribute choose **LDAP (User Attribute Store)** from the drop-down.
26. In the **Value** column for the **Email** attribute select **mail** from the drop-down.

27. Repeat steps 26-27 to map the **displayName** value to the **Name** attribute.



28. Click **Done**, the **Create Adapter Instance – Adapter Contract Mapping** page displays.
29. Click **Save** to commit your configuration changes.

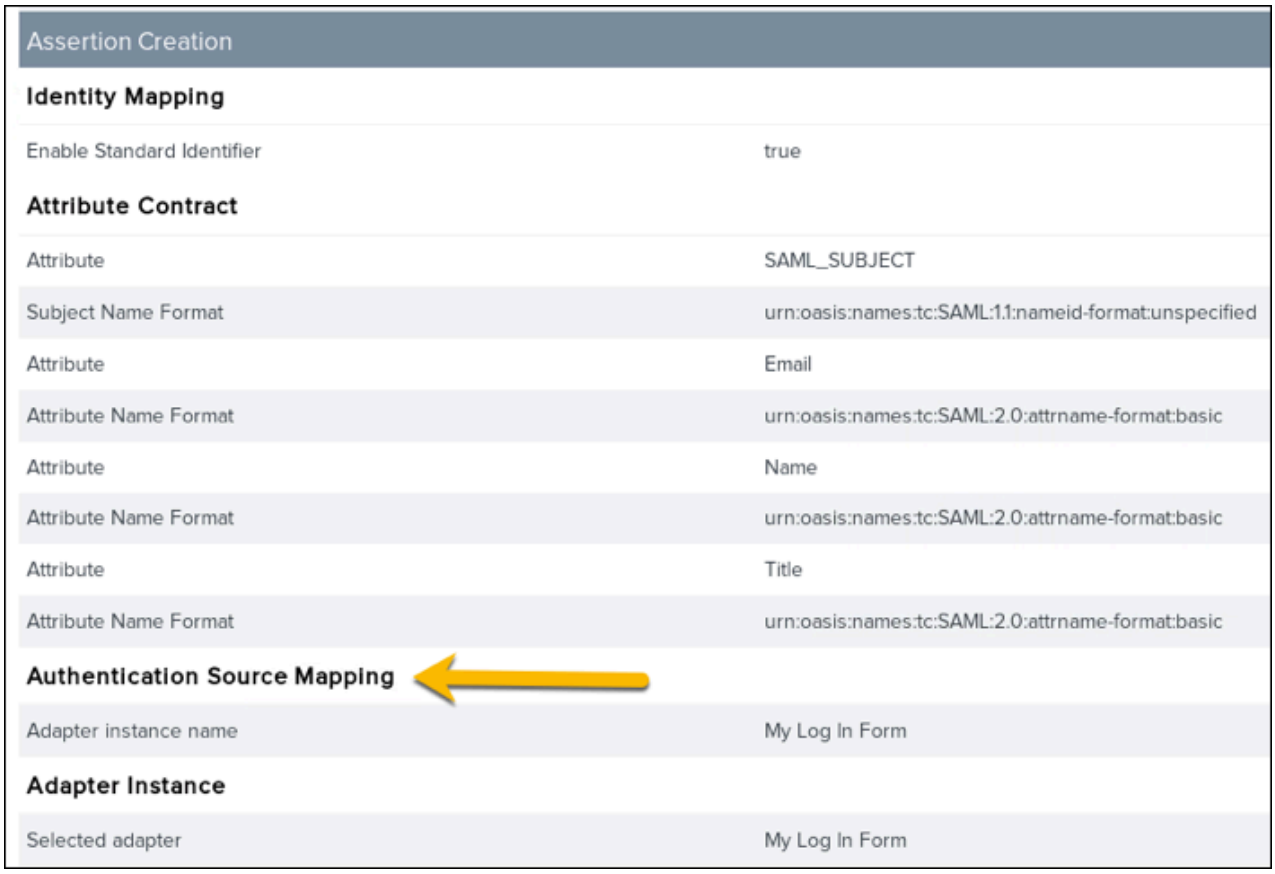
Modify the Connection

Now PingFederate is getting attributes from both the adapter (username) and OpenLDAP when it authenticates a user with the HTML Form adapter. Next you will need to configure PingFederate to use those new values in the connection.

Remap the Assertion Creation

1. Navigate to the **Applications > Integration > SP Connections** page.
2. Click **Connection to Denver SP**, the **Activation & Summary** page for the connection will display.

3. In the summary page locate the **Authentication Source Mapping** section and click the heading link, the **Assertion Creation – Authentication Source Mapping** page will display.



4. Click the **My Log In Form** adapter instance, the **IdP Adapter Mapping – Summary** page for the adapter will display.
5. Click the **Attribute Contract Fulfillment** tab, the **Attribute Contract Fulfillment** page will display.
6. In the **Source** column for the **Email** attribute select **Adapter** from the drop-down.
7. In the **Value** column for the **Email** attribute select **Email** from the drop-down.
8. Repeat steps 5-6 and change the **Name** attribute to map the **Name** attribute from the adapter.

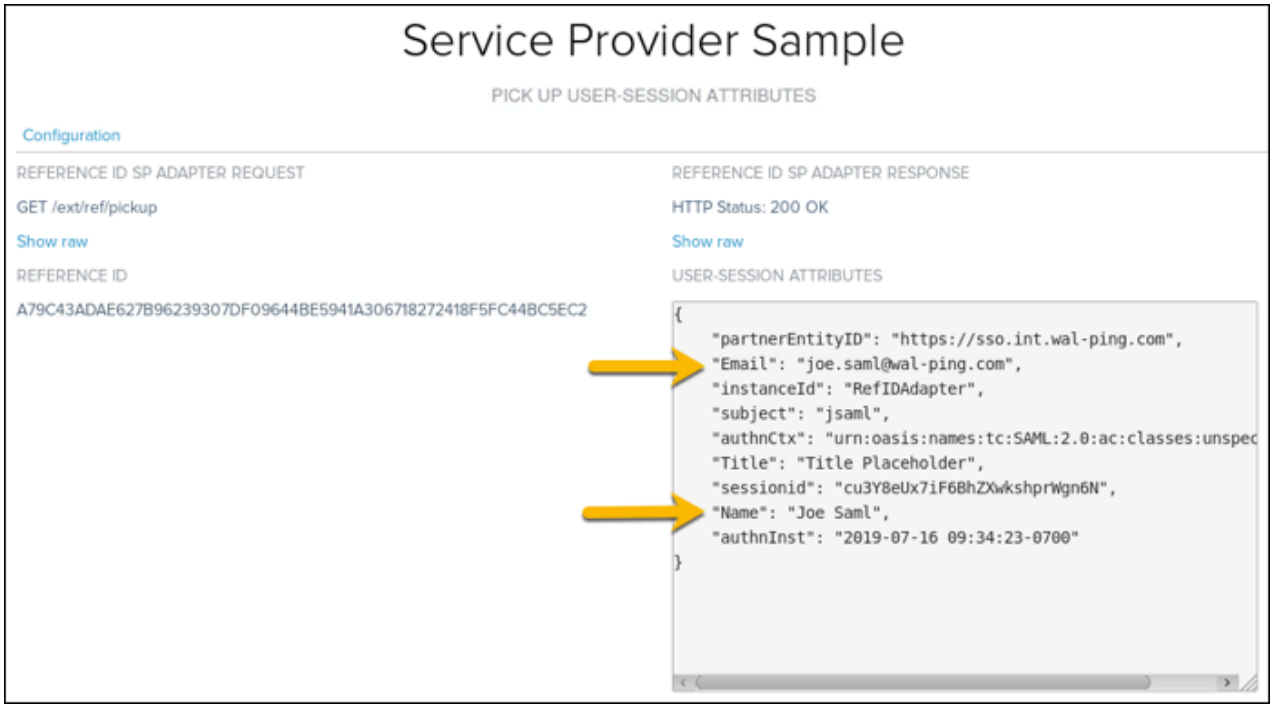
9. Click **Save** to commit your changes.

Test the Datastore Connection

Now you can test your SSO transaction as before, logging in using the HTML Form adapter as jsaml and ending in a session at the target application. Now, the attributes displayed in the target application will be what you mapped from the datastore. So instead of just the hard-coded attributes, the identity provider PingFederate from the OpenLDAP directory will pull these attributes.

Test the Datastore Connection

1. Restart Firefox or open a private browser window to clear your cookies.
2. Click the **SSO to SP > SSO to SP** bookmark.
3. Use the **jsaml/2Federate** credentials to login.
4. Check the user attributes and pay particular attention to the **Email** and **Name** attributes, you will see that the values displayed are no longer the placeholders you configured earlier, but instead match what was pulled by the adapter from the OpenLDAP directory.



Examine the Logs

Since the test worked, take a look at the log files for the identity provider PingFederate instance.

Note: The log entries here have been truncated and formatted to fit in the lab guide. Your logs may appear slightly different.

1. Open the identity provider (**wal-ping.com**) PingFederate log file in Plume. The log file is located at `/opt/PingIdentity/pingfederate-10.2-wal/pingfederate/log/server.log`.

2. Towards the bottom of the log, above the SAML assertion, look for the following entries:
   - LDAP Attr Src search: The search that PingFederate did in OpenLDAP to find the jsaml user. These are the two attributes that are being pulled from the directory.

```
LDAP Attr Src search: search base=ou=users,dc=wal-ping,dc=com |
 filter=cn=jsaml | searchScope=ONELEVEL_SCOPE | attrNames=[mail,
 displayName] | nestedGroup=false
```

   - LDAP Attr Src search result: The entry in the directory for the jsaml user.

```
LDAP Attr Src search result: {Subject DN=cn=jsaml,ou=users,dc=wal-
ping,dc=com, mail=joe.saml@wal-ping.com, displayName=Joe Saml}
```

   - Attributes from Datasource: The attributes returned from the directory as a result of the search.
   - Resulting attributes: These are the final attributes mapped to the assertion.

```
Source attributes:{..., username=jsaml}, context={...}} Attributes from
 Datasource:{OpenLDAP.mail=joe.saml@wal-ping.com, OpenLDAP.displayName=Joe
 Saml, OpenLDAP.Subject DN=cn=jsaml,ou=users,dc=wal-ping,dc=com} Resulting
 attributes:{Email=joe.saml@wal-ping.com, Name=Joe Saml, username=jsaml}

Source attributes:{..., Email=joe.saml@wal-ping.com, username=jsaml,
 Name=Joe Saml}, context={...}} Resulting attributes:{Email=joe.saml@wal-
ping.com, Title=Test Subject 15, SAML_SUBJECT=jsaml, Name=Joe Saml}
```

Summary

Congratulations, you have successfully configured PingFederate with a datastore. Using a datastore you can have user attributes pulled from a central directory location and have them mapped into a SAML assertion.

Lab 5 - Authenticating With External Data Stores

Introduction

Initial user authentication is normally handled outside of the PingFederate server using an application or ID management system logon module. PingFederate's adapter and application agents are typically used to integrate with these local authentication mechanisms.

PingFederate packages an HTML Form Adapter that delegates user authentication to a configured password credential validator. This authentication mechanism validates credentials based on either an LDAP directory or a simple username validator that authenticates credentials maintained by PingFederate. The simple PCV is what you have been using in the labs so far.

This lab should take 60 minutes to complete.

Objectives

- Create a new HTML Form Adapter that can use LDAP to lookup user attributes
- Customize the HTML Form Adapter
- Test the new HTML Form Adapter
- Review the sever log entries

Description of tasks to perform

You will first build a new HTML Form adapter, customize how it looks, and set it to check the user's credentials against the OpenLDAP instance. Next, you will chain the new and old HTML Form adapters together, so the user will have to successfully authenticate against both.

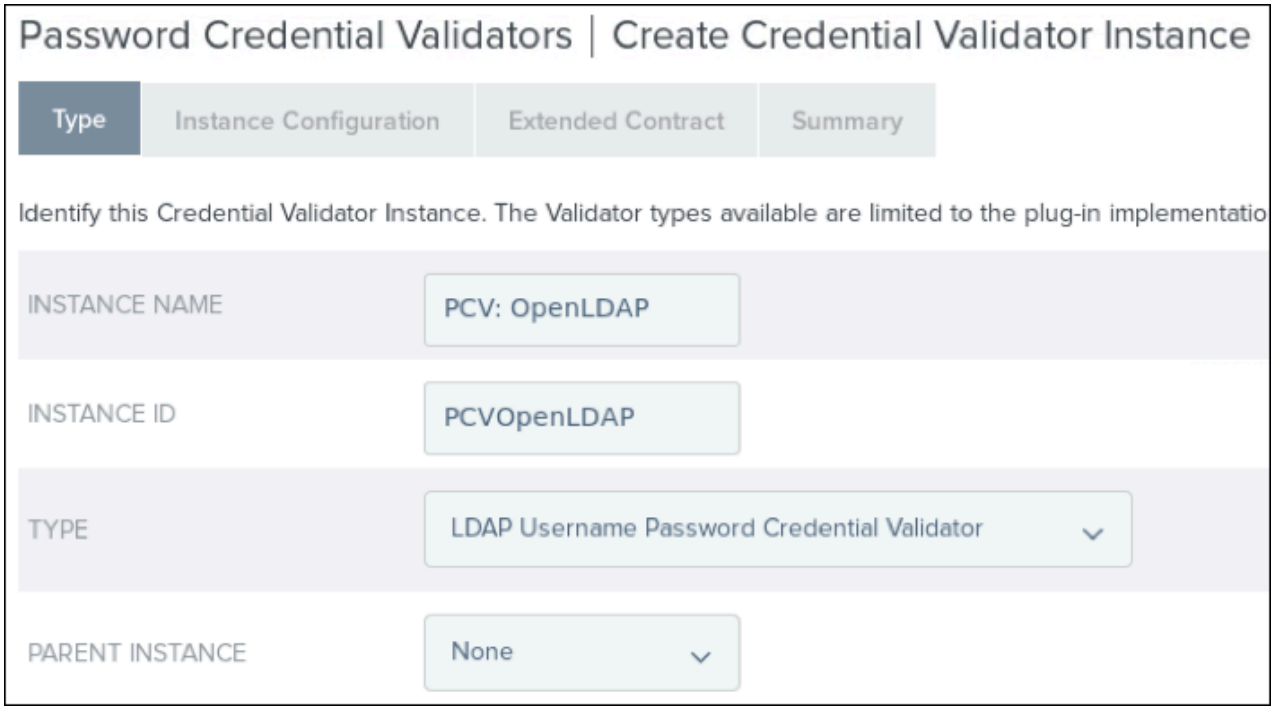
Then you will create an authentication selector, having PingFederate automatically decide which adapter should be used based on the request URL. You can look at your bookmarks to see the different URLs used.

Create a New HTML Form Adapter

You will first create a new HTML Form Adapter instance for this lab. Before you create the new HTML Form you will first create a new password credential validator.

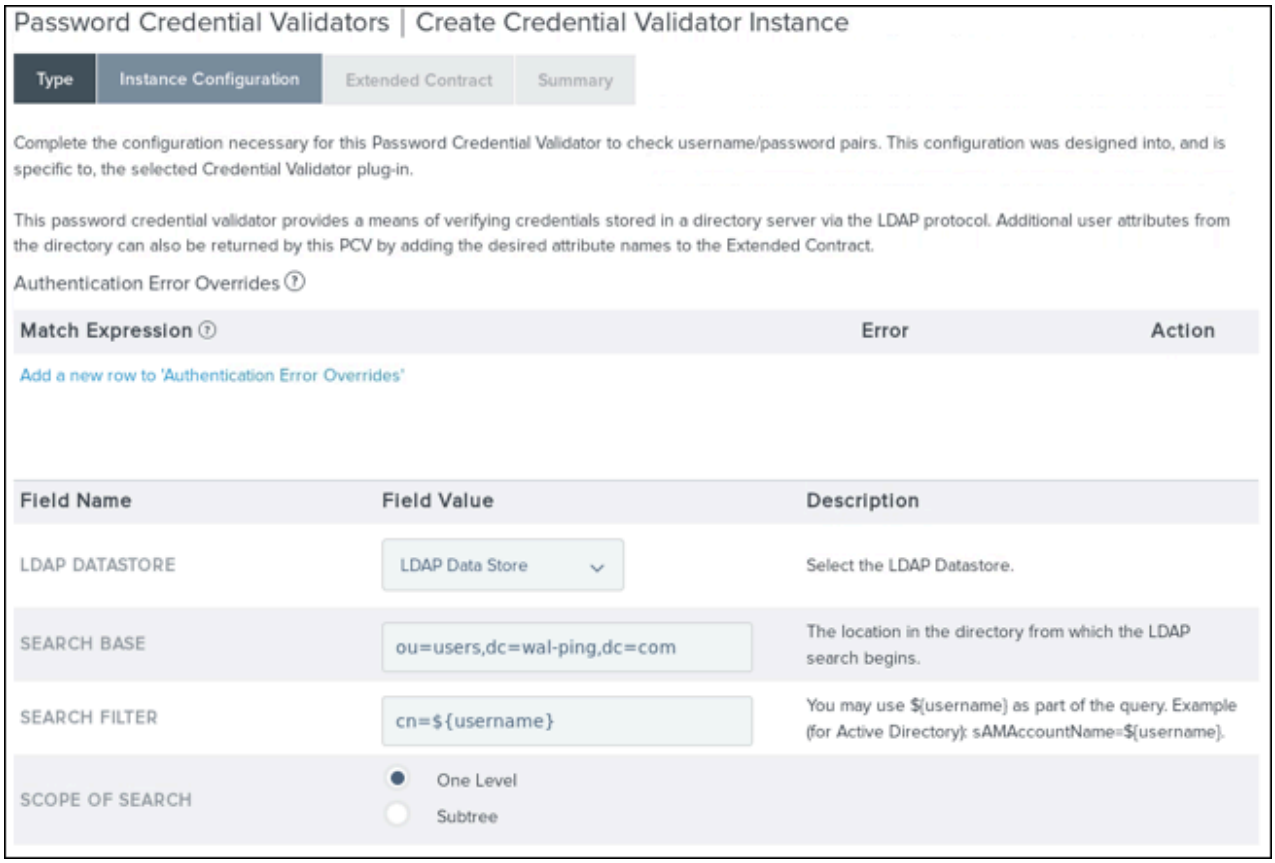
Configure a New Password Credential Validator

1. Log into the **IdP Admin Console**.
2. Navigate to the **System > Data & Credential Stores > Password Credential Validators** page.
3. Click the **Create New Instance** button, the **Create Validator Instance – Type** page will display.
4. Enter the following parameters:
   - INSTANCE NAME: **PCV: OpenLDAP**
   - INSTANCE ID: **PCVOpenLDAP**
   - TYPE: **LDAP Username Password Credential Validator**
   - PARENT INSTANCE: **none**



5. Click **Next**, the **Instance Configuration** page will display.
6. Enter the following parameters:
   - LDAP DATASTORE: **LDAP Data Store**
   - SEARCH BASE: **ou=users,dc=wal-ping,dc=com**
   - SEARCH FILTER: **cn=${username}**
   - SCOPE OF SEARCH: **One Level**

7. Leave all other settings at their defaults

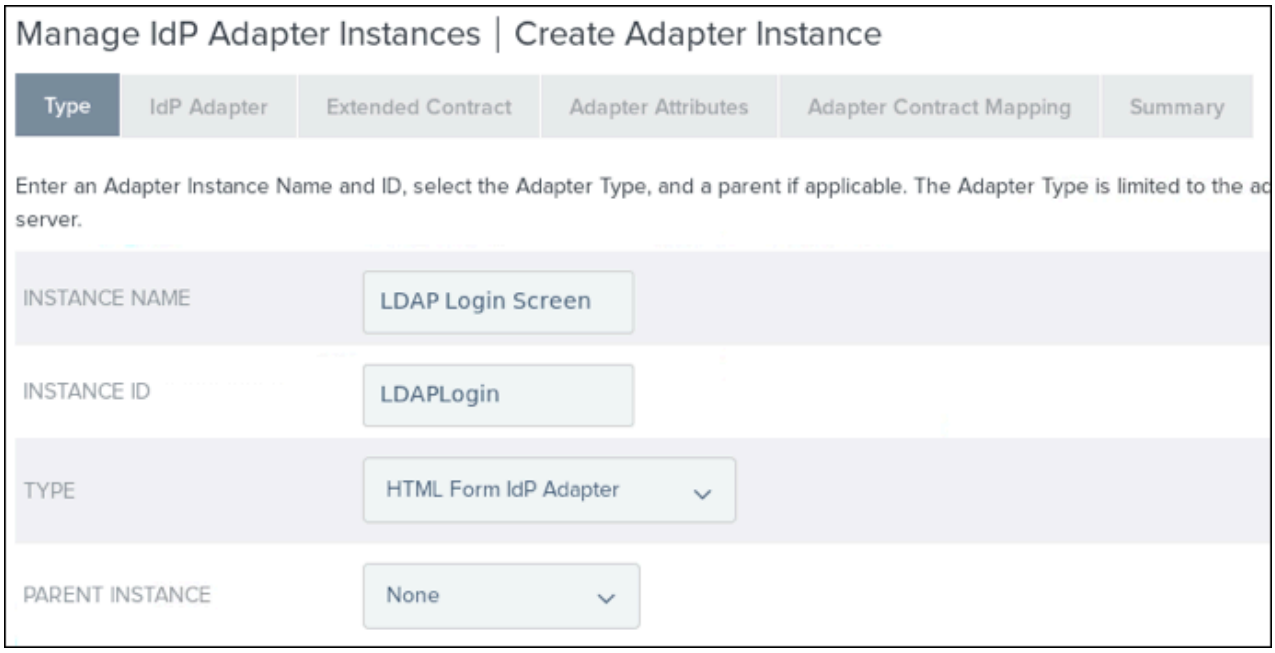


8. Click **Next**, the **Extended Contract** page will display. You will not be making any changes to the PCV attribute contract.
9. Click **Next**, the **Summary** page will display.
10. Review the summary.
11. Click **Save** to commit your changes.

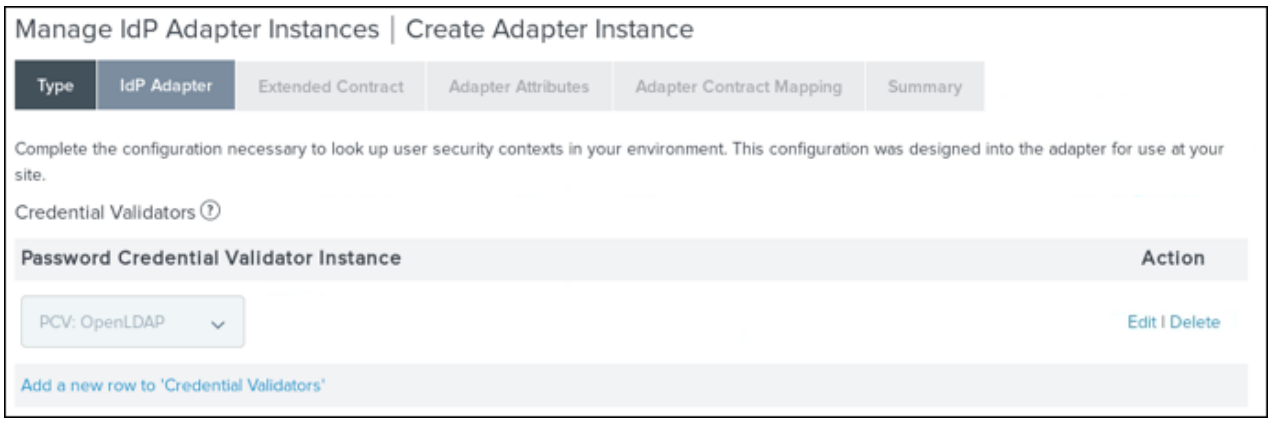
Configure the Adapter

1. Navigate to the **Authentication > Integration > IdP Adapters** page.
2. Click the **Create New Instance** button, the **Create Adapter Instance – Type** page will display.

3. Enter the following parameters:
   - INSTANCE NAME: **LDAP Login Screen**
   - INSTANCE ID: **LDAPLogin**
   - TYPE: **HTML Form IdP Adapter**
   - PARENT INSTANCE: **none**

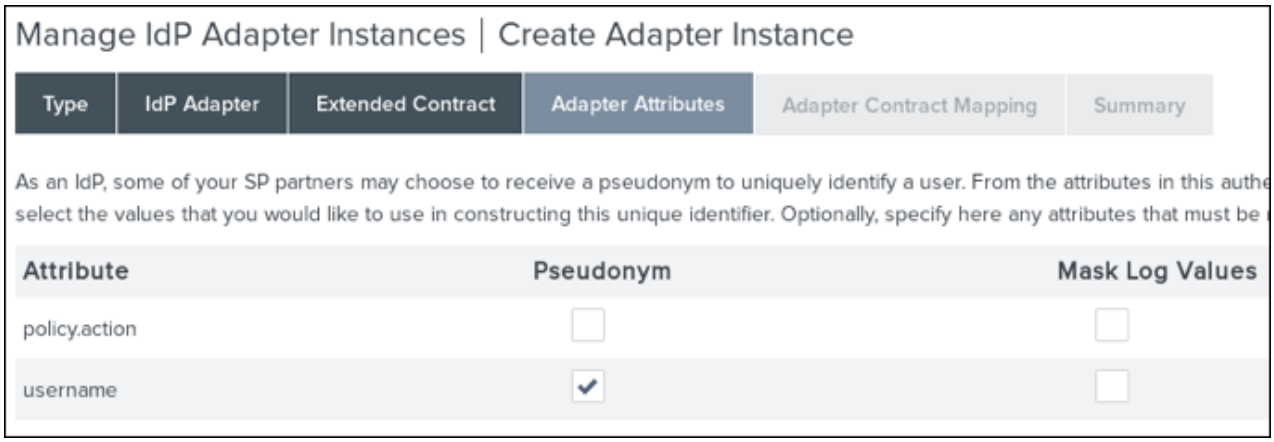


4. Click **Next**, the **IdP Adapter** page will display.
5. Click **Add a New Row to 'Credential Validators'**.
6. In the drop-down select **PCV: OpenLDAP**, this is the PCV you just created.
7. Click **Update**.



8. Leave the remaining fields at their defaults and click **Next**, the **Extended Contract** page will display.
9. You will not be making any changes to the contract. Click **Next**, the **Adapter Attributes** page will display.

10. For the username attribute, check the box for **pseudonym**.



11. Click **Next**, the **Adapter Contract Mapping** page will display.
12. You will not be configuring an adapter contract. Click **Next**, the **Summary** page will display.
13. Review the summary.
14. Click **Save** to commit your changes.

Customize the LDAP Form Adapter

So you can tell them apart, you will customize the newly created HTML Form adapter. You will create a new HTML form template and give it a yellow background.

Customizing the LDAP Form Adapter

1. In File Manger navigate to **/opt/PingIdentity/pingfederate-12.0-wal/pingfederate/server/default/conf/template**.
2. Right-click on **html.form.login.template.html** and select **Copy**.
3. Click the **Edit** menu and select **Paste**, a copy of the file will be placed in the directory.
4. Change the filename to **html.form.login.template.withLDAP.html**.
5. Right-click on the **html.form.login.template.withLDAP.html** file and select **Open With > Pluma**.
6. On line 51 change the `href=` section as shown below:

   Note: This line, and following lines, may be broken up for formatting in this lab guide. In the files they are a single line.

   ```
   <link rel="stylesheet" type=
   "text/css" href="assets/css/mainLDAP.css"/>
   ```

7. Click **Save** and close the file.
8. In your **File Manager** window navigate to the **assets/css** folder.
9. Right-click the **main.css** file and choose **Copy**.
10. Click the **Edit** menu and select **Paste**, a copy of the file will be placed in the directory.
11. Change the name to **mainLDAP.css**.
12. Right-click **mainLDAP.css** and select **Open With Pluma**.
13. On line 3979 change background-color to #fff000.

    Note: In Pluma use **Search > Find** and enter "html " (html followed by a space), then click **Find**. This will take you directly to the CSS that needs changing.

14. Delete lines 3982 through 3989.

```
background-color: #3d454d;
```

through

```
background-image: -webkit-radial-gradient(40% 40%, circle cover, rgba(61,
 69, 77, 0.9) 30%, #3d454d 80%);
```

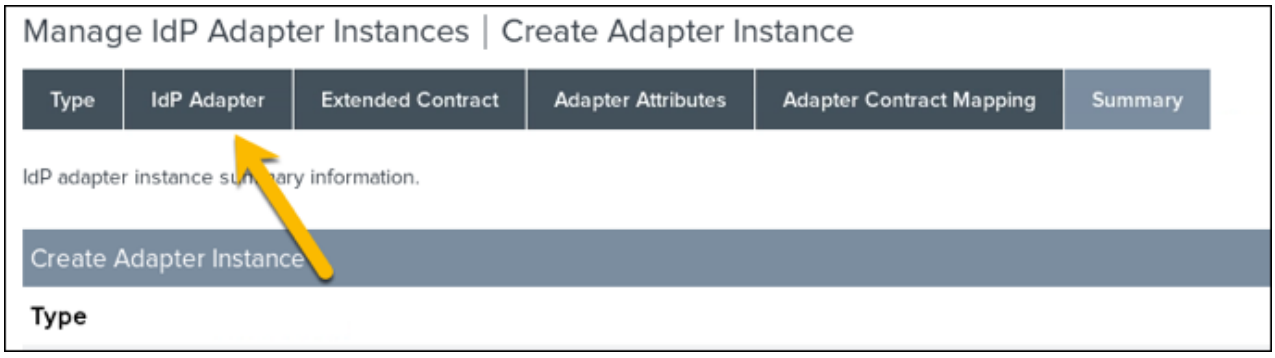
15. The html section of this file should now look like this:

```
html {
 height: 100%;
 background-color: #fff000;
 background-repeat: no-repeat;
 background-attachment: fixed;
 -webkit-background-size: cover;
 -moz-background-size: cover;
 -o-background-size: cover;
 background-size: cover;
}
```

16. Click **Save**.
17. Close Pluma and the File Manager windows.
18. Open a new **Terminal** window.
19. Use the command below to restart your IdP PingFederate instance. When prompted use **2Federate** as the password.

```
systemctl restart pingfed-wal.service
```

20. Wait a minute for the service to restart then return to the admin console for your **IdP PingFederate** instance.
21. Navigate to the **Authentication > Integration > IdP Adapters** page.
22. Click the **LDAP Login Screen** adapter, the **Create Adapter Instance – Summary** page will display.
23. In the top navigation bar click **IdP Adapter**, the **IdP Adapter** page will display.

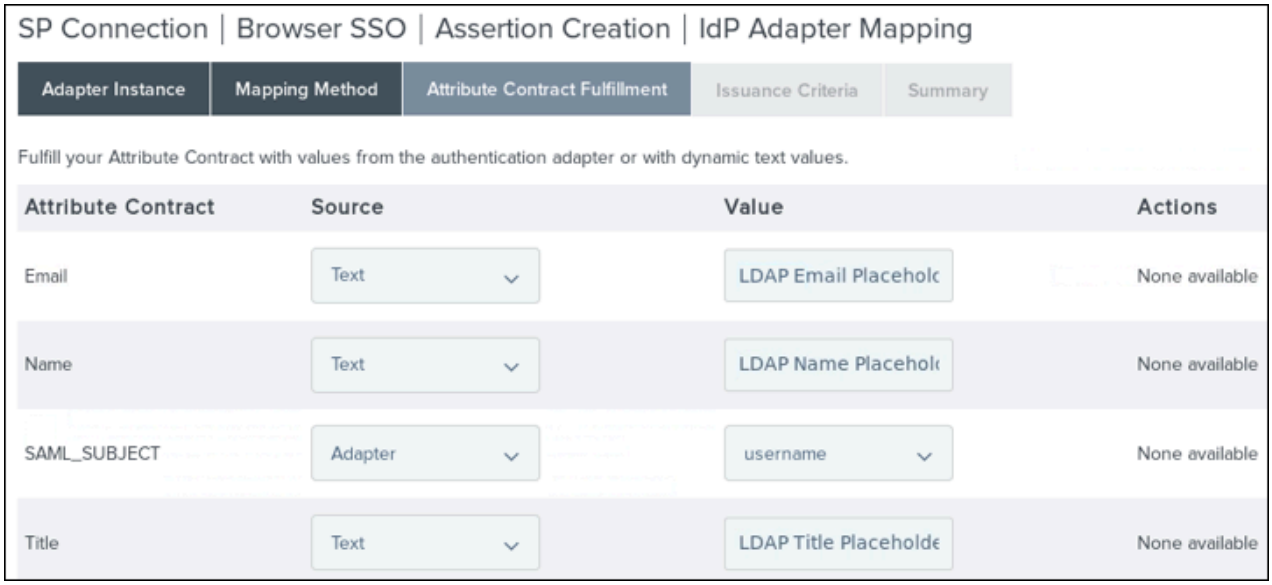


24. At the bottom click **Show Advanced Fields**.
25. In the LOGIN TEMPLATE field enter **html.form.login.template.withLDAP.html**.

26. Click **Save** to commit your changes.

Add the New Customized Adapter to the Connection

1. Navigate to the **Applications > Integration > SP Connections** page and click the existing **Connection to Denver SP** the **SP Connection – Activation & Summary** page will display.
2. Scroll down and click the **Authentication Source Mapping** header, the **Assertion Creation – Authentication Source Mapping** page will display.
3. Click the **Map New Adapter Instance** button, the **IdP Adapter Mapping – Adapter Instance** page will display.
4. In the ADAPTER INSTANCE drop-down select **LDAP Login Screen**.
5. Click **Next**, the **Mapping Method** screen will display.
6. Keep the default settings, **USE ONLY THE ADAPTER CONTRACT VALUES IN THE SAML ASSERTION**.
7. Click **Next**, the **Attribute Contract Fulfillment** page will display.
8. For each attribute set the following Source and Value pairs:
   - Email: **Text** / **LDAP Email Placeholder**
   - Name: **Text** / **LDAP Name Placeholder**
   - SAML_SUBJECT: **Adapter** / **username**
   - Title: **Text** / **LDAP Title Placeholder**



9. Click **Next**, the **Issuance Criteria** page will display.
10. You will not be setting any issuance criteria. Click **Next**, the **Summary** page will display.
11. Click **Save** to commit your changes.

Test the New HTML Form Adapter

Now that the new HTML Form Adapter is all setup it is time to test the connection and make sure everything is working correctly. You'll notice that you now have two adapters mapped into the same connection. How does PingFederate determine which adapter to use during an SSO transaction?

Testing the SSO to SP Link

In previous labs when you selected the SSO to SP bookmark you were taken directly to an HTML Form Adapter (My Log In Form), this is because that adapter instance was the only one configured for your connection. Before proceeding with the next steps take a look at the properties of the SSO to SP bookmark. You will see that the address can be broken into three pieces.

```
https://sso.int.wal-ping.com:9031
```

```
/idp/startSSO.ping?
PartnerSpId=https://sso.int.den-ping.com
```

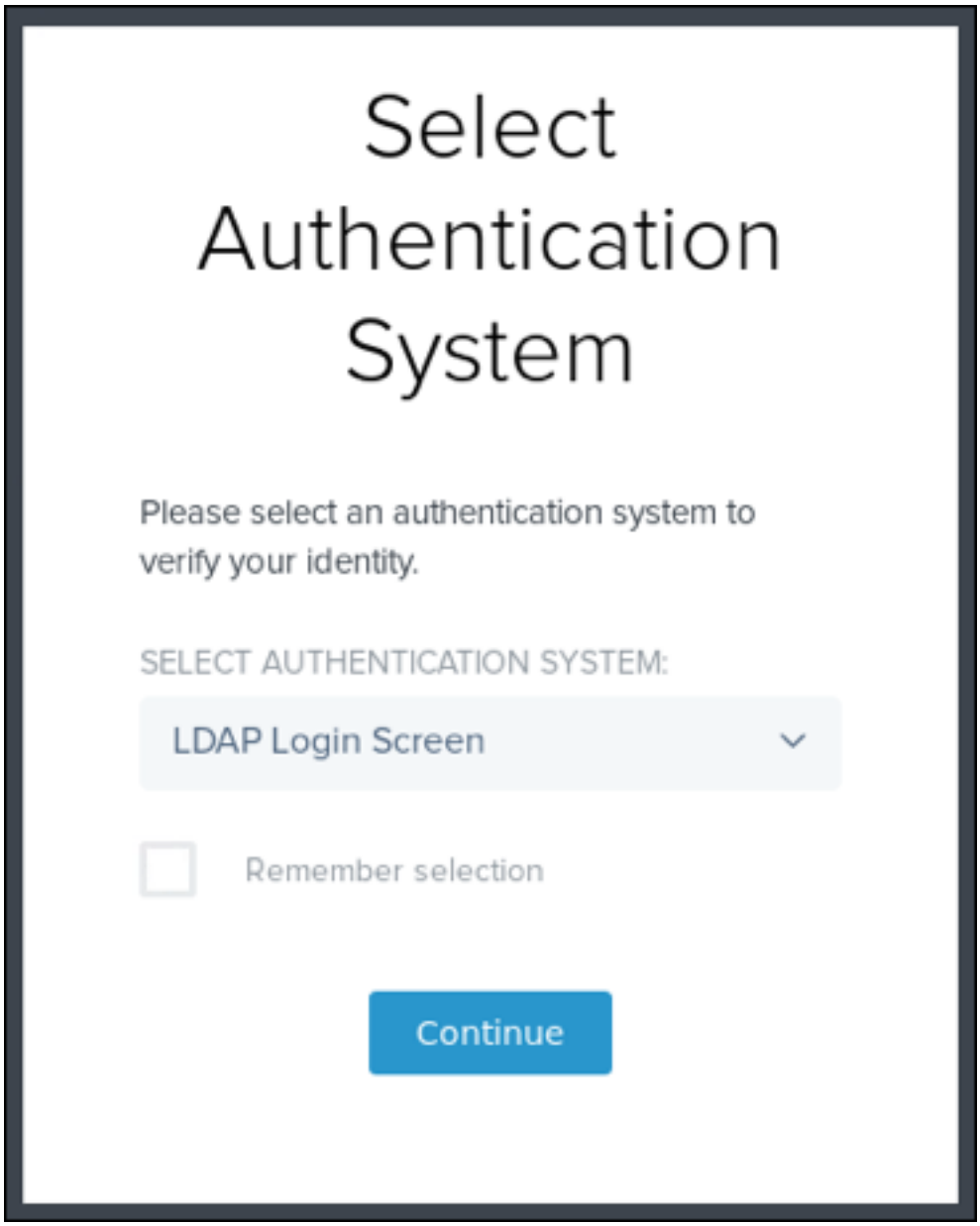
- The first part is the base URL and port of the IdP PingFederate server.
- The second part is the endpoint used to start an SSO transaction. PingFederate has a number of endpoints that serve different purposes, more information about these can be found in the manual.
- The last part is a query parameter. The /idp/startSSO.ping endpoint accepts a number of different parameters. In this case we are specifying the partner ID of the SP connection we want to use.

Keep these concepts in mind as you perform the remainder of this lab.

1. Open your Firefox browser and click the SSO to SP > SSO to SP bookmark. Since this connection has two adapters configured the user is prompted to select which adapter (authentication system) they wish to use.
2. In the SELECT AUTHENTICATION SYSTEM drop-down select LDAP Login Screen.

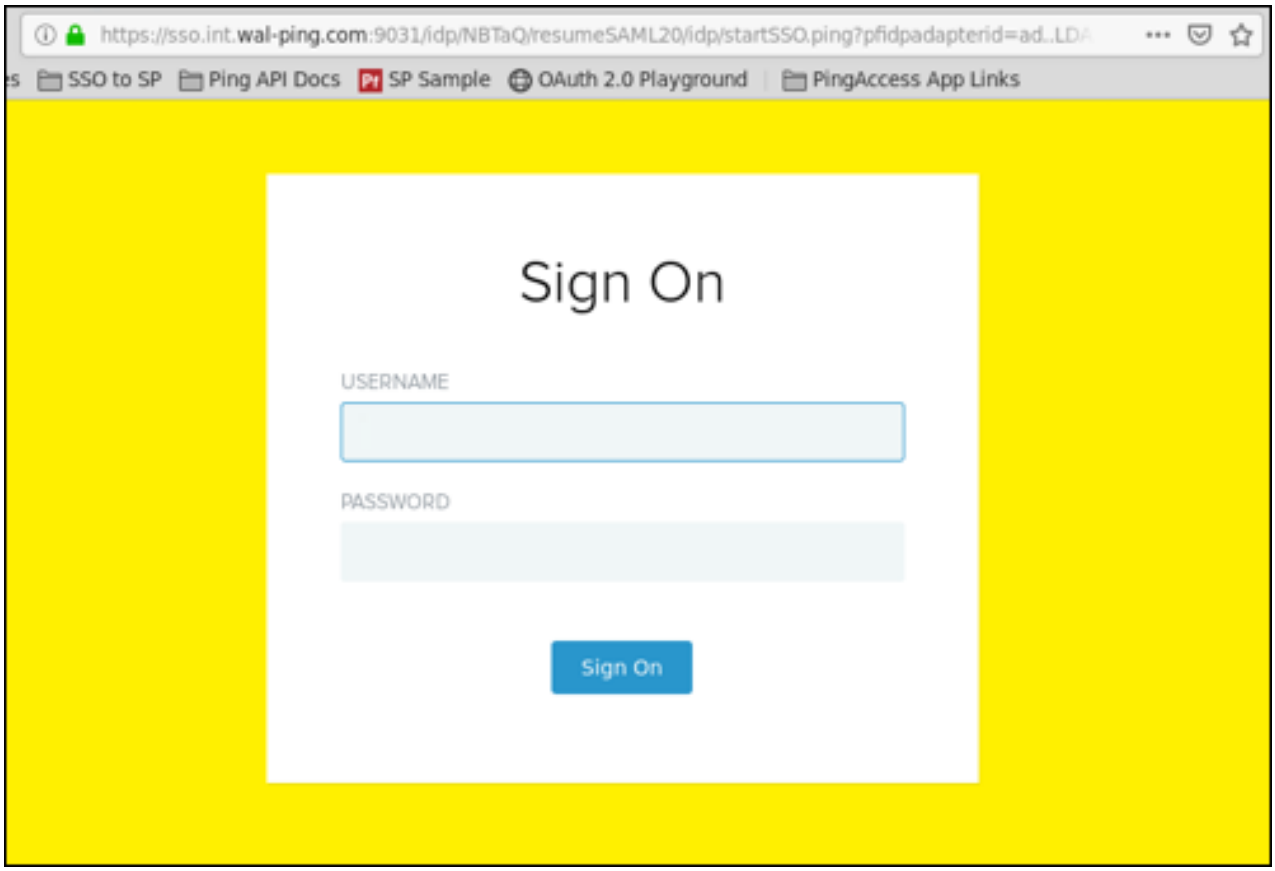
   Note: You should see that both adapters configured in the connection are available in the drop-down. In this instance, without any further input, PingFederate allows the user to select which adapter to use to authenticate.

3. Uncheck the **Remember selection** checkbox.



4. Click **Continue**.

5. If everything was configured correctly you will see the HTML Form Adapter Sign On page with the yellow background as shown below.



6. Login using the **jsaml/2Federate** credentials.

7. You will be redirected to the sample application, notice that the attributes passed are the placeholder attributes you configured for the LDAP Login adapter instance.

Using Parameters to Select a Specific Adapter

With PingFederate you can specify a specific adapter instance to use with the `IdpAdapterId` query parameter.

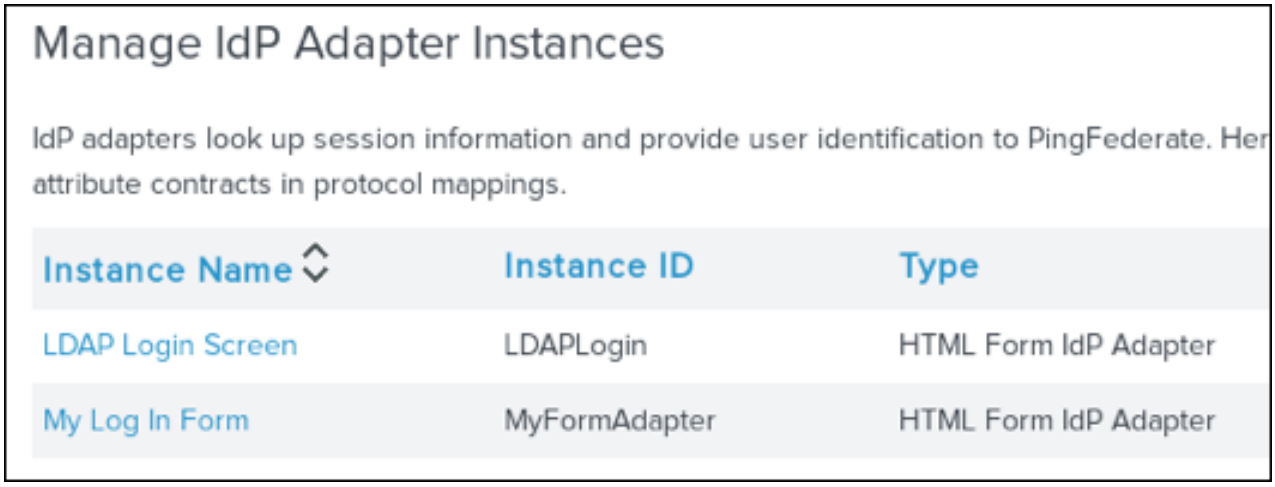
Examine the Log In Screen SSO bookmark:

```
https://sso.int.wal-ping.com:9031
 /idp/startSSO.ping?
 PartnerSpId=https://sso.int.den-ping.com&IdpAdapterId=MyFormAdapter
```

Notice the addition of the `IdpAdapterId` query parameter. Now look at the SSO to SP (LDAP Login) bookmark:

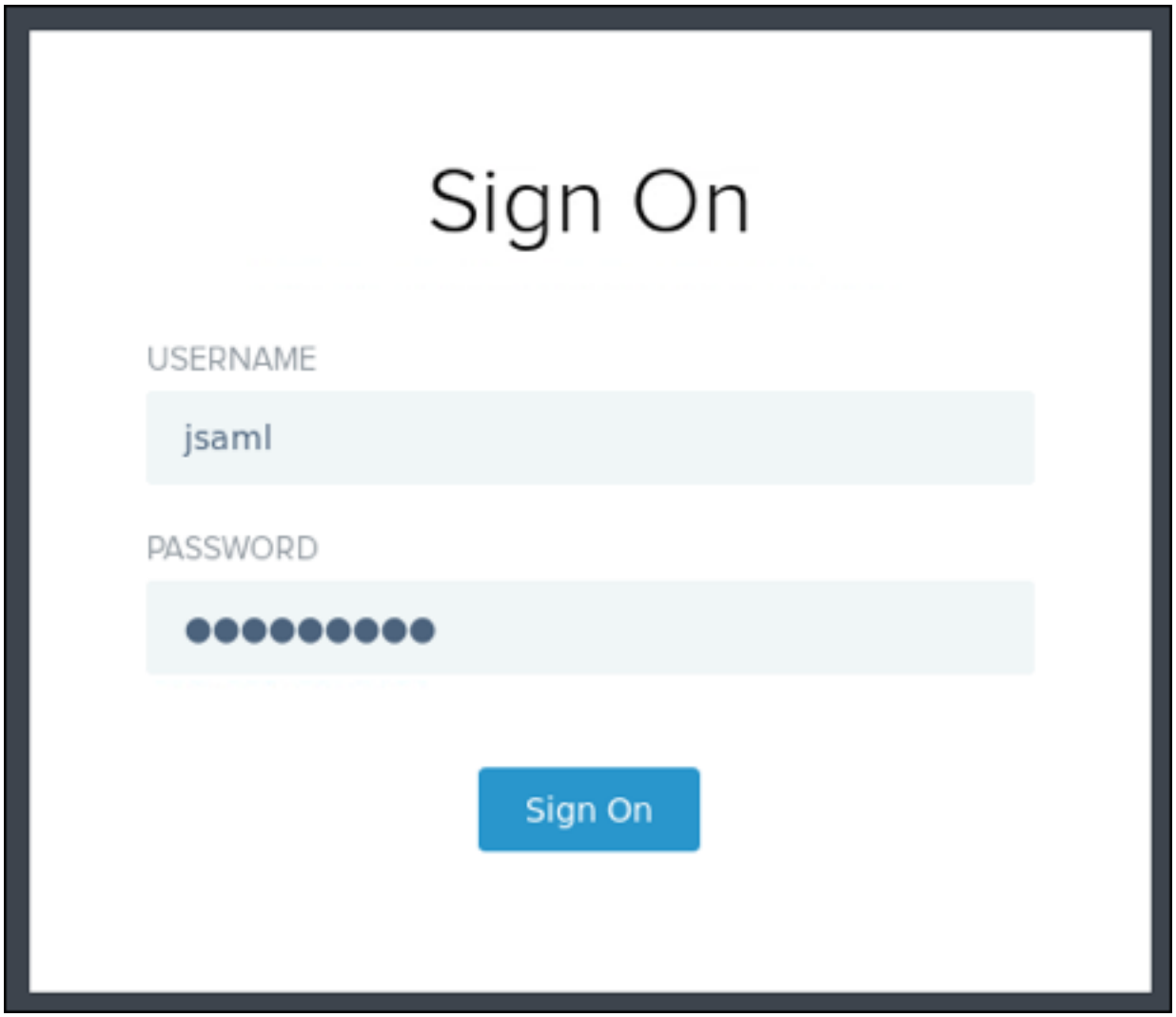
```
https://sso.int.wal-ping.com:9031
 /idp/startSSO.ping?
 PartnerSpId=https://sso.int.den-ping.com&IdpAdapterId=LDAPLogin
```

The `IdpAdapterId` parameter allows you to specify a specific adapter that the user should use when using this connection. The adapter ID is specified when the adapter is created:



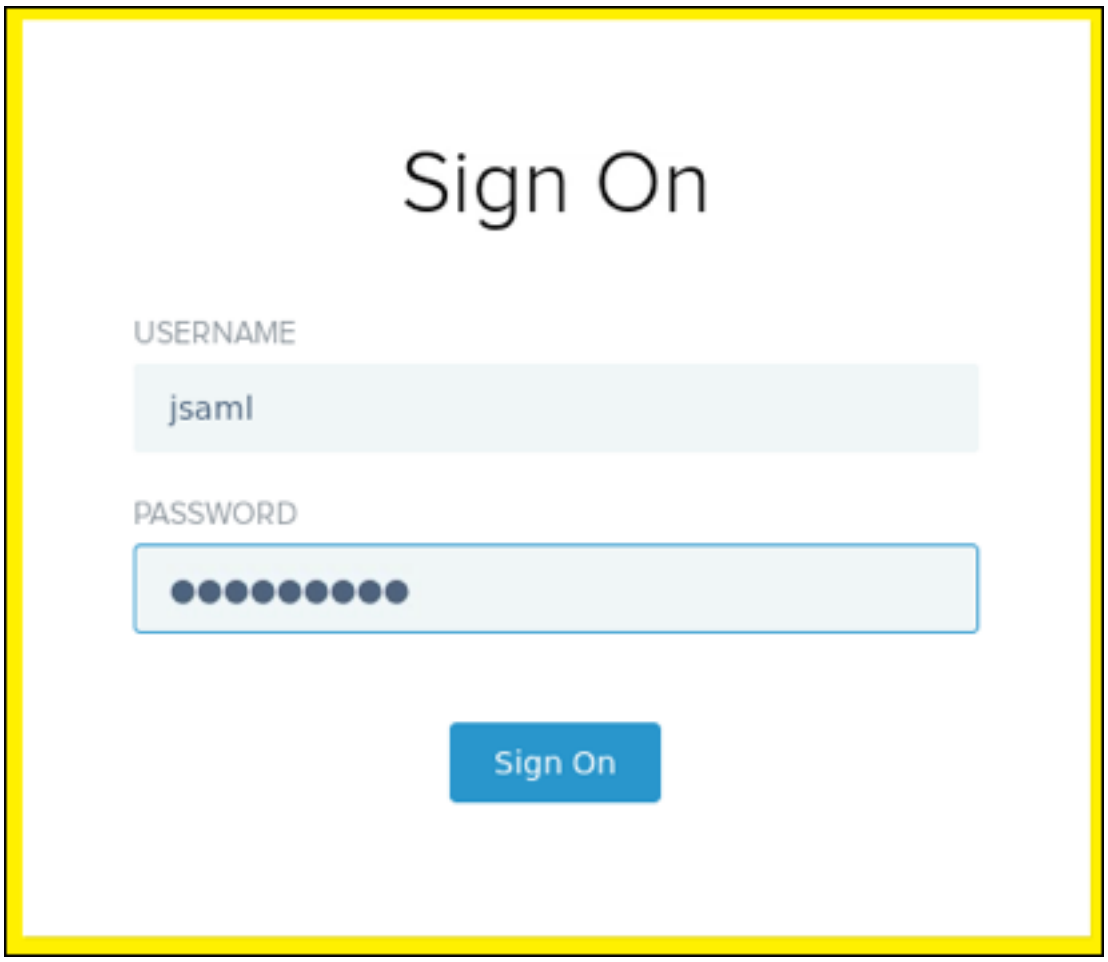
1. Use the **Log In Screen SSO** bookmark to reach the sample application. Authenticate using the **jsaml/2Federate** credentials.

2. Remember that the MyFormAdapter instance is configured to pull user attributes from the LDAP data store, you will see these attributes in the sample application.



3. Now use the SSO to SP (LDAP Login) bookmark to access the application. Since this adapter is configured to use the LDAP PCV for authentication (instead of the simple PCV) you can use any of the user accounts in the LDAP directory, including the jsaml account.

4. Notice that the adapter being used is now the LDAP Login adapter with the yellow background. You will also notice that the attributes in the application are the placeholder attributes you configured when you added this adapter to the connection.



Summary

In this lab you have seen how it is possible to create a PCV that authenticates users against an external data store, in this case an LDAP directory. You've also seen how PingFederate handles multiple adapters configured in a single connection. If no adapter is specified as part of the SSO URL PingFederate presents the user with a list of adapters to choose from. However you, as the PingFederate administrator, can use custom crafted URLs to control this behavior for your users.

Lab 6 - Authentication Policies

Introduction

This lab will walk you through the process of setting up an Authentication Policy that uses a Policy Contract to hold the user attributes. Policy Contracts are reusable sets of attributes that can be shared between multiple connections. Policy Contracts are populated using Authentication Policies.

This lab should take 45 minutes to complete.

Objectives

- Create an authentication policy contract with user attributes
- Create an authentication selector to use with the authentication policy
- Create an authentication policy that populates a contract

- Update the connection to use the new policy contract
- Use authentication policies to implement MFA

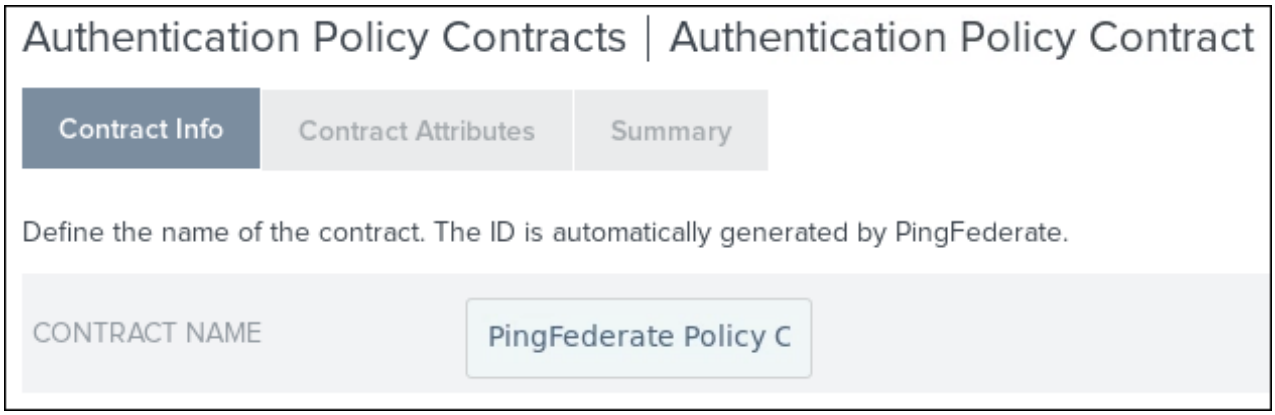
Description of tasks to perform

In this lab, you will set up an authentication policy contract and a basic corresponding authentication policy. You will start with an authentication selector to choose which adapter to use in the contract. From there you will create a new authentication policy that mimics an MFA setup.

Create a Policy Contract

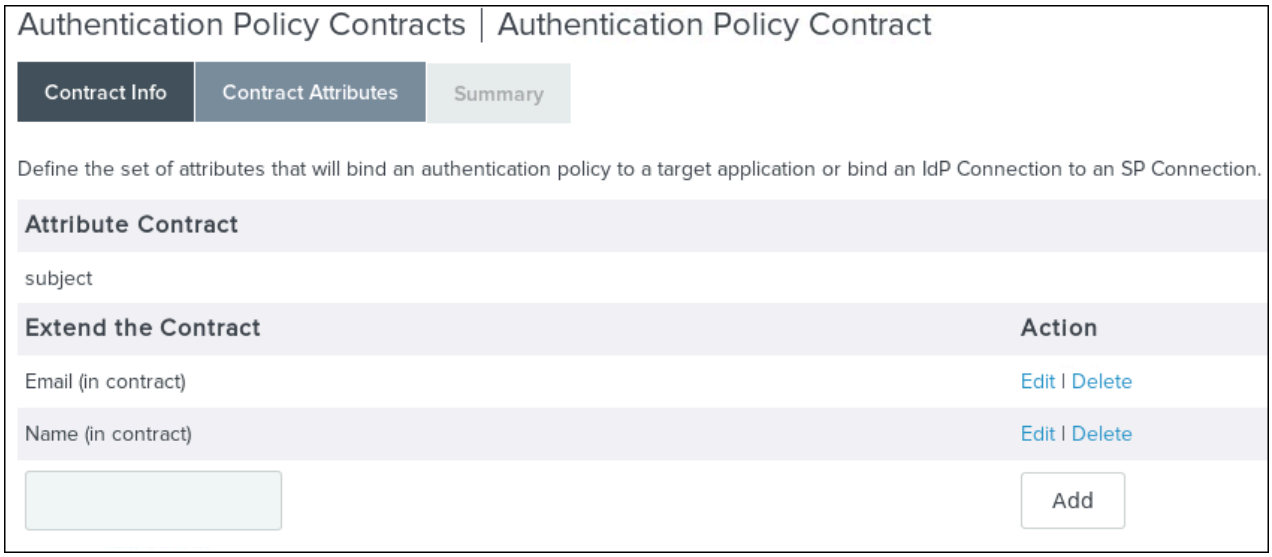
First, you have to make a list of the attributes the contract will return. Just like an adapter returns attributes that can be used in a connection (such as username), so does a policy contract. You will tell PingFederate that when it uses this policy contract, it can get the attributes, like username and email, to use. Since you can name the attributes anything, you will call phone number "Email (in Contract)" to remember where it came from.

1. Log into your **IdP PingFederate** admin console if you have not done so already.
2. Navigate to the **Authentication > Policies > Policy Contracts** page.
3. Click the **Create New Contract** button, the **Authentication Policy Contract – Contract Info** page will display.
4. In the CONTRACT NAME field enter **PingFederate Policy Contract**.



5. Click **Next**, the **Contract Attributes** page will display.
6. In the Extend the Contract field enter **Email (in contract)**.
7. Click the **Add** button to extend the contract.

8. Repeat steps 5 and 6 to add **Name (in contract)** as well.



9. Click **Next**, the **Summary** page will display.
10. Review the configuration summary.
11. Click **Save** to commit your changes.

Create an Authentication Selector

Authentication selectors can be used within the authentication policy tree to tell PingFederate which policy branch to follow.

Now you are going to create an authentication selector, specifically an HTTP Request Parameter selector. This will allow you to easily see the difference. The general URL to connect to the SP is:

```
https://sso.int.wal-ping.com:9031/idp/startSSO.ping?PartnerSpId=https://
sso.int.den-ping.com`
```

You will add a request parameter of `ThisSelector` and check for one of two values: `LDAP` or `Login`.

The two URLs you will be using are:

```
https://sso.int.wal-ping.com:9031/idp/startSSO.ping?PartnerSpId=https://
sso.int.den-ping.com&ThisSelector=LDAP
```

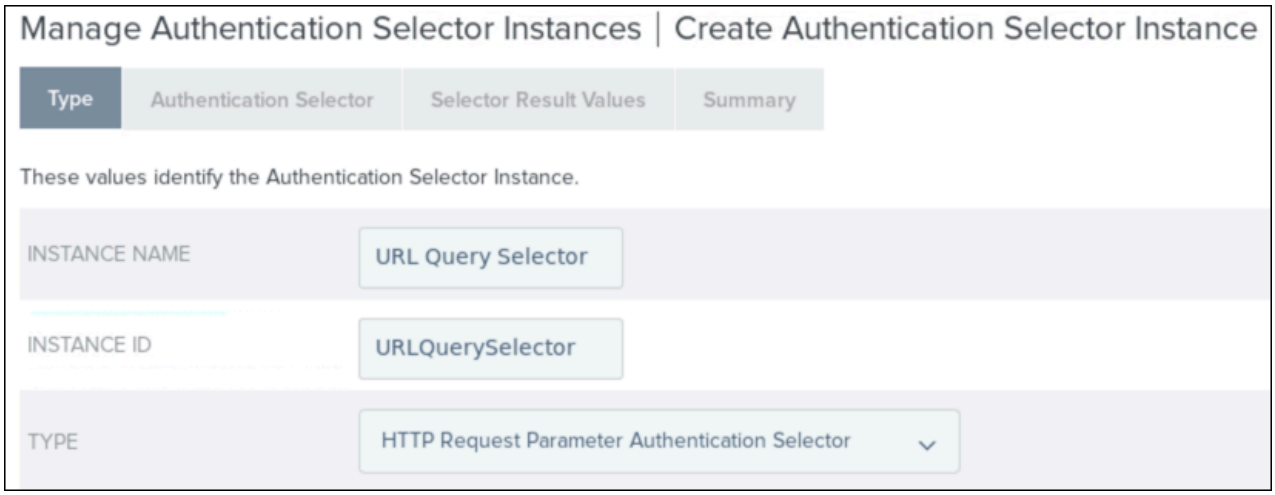
which will result in the LDAP form adapter. The Log in Screen adapter URL will be:

```
https://sso.int.wal-ping.com:9031/idp/startSSO.ping?PartnerSpId=https://
sso.int.den-ping.com&ThisSelector=Login
```

Note that all these values can be anything when configured in production.

1. On the **IdP Admin Console** navigate to the **Authentication > Policies > Selectors** page.
2. Click the **Create New Instance** button, the **Create Authentication Selector Instance – Type** page will display.

3. Enter the following parameters:
   - INSTANCE NAME: **URL Query Selector**
   - INSTANCE ID: **URLQuerySelector**
   - TYPE: **HTTP Request Parameter Authentication Selector**



4. Click **Next**, the **Authentication Selector** page will display.
5. In the HTTP REQUEST PARAMETER NAME field enter **ThisSelector**.
6. Leave all other settings at their defaults and click **Next**, the **Selector Result Values** page will display.
7. In the Result Values text box enter **LDAP**.
8. Click **Add**.
9. Repeat steps 8 and 9 to add the **Login** value.
10. Click **Next**, the **Summary** page will display.
11. Review the summary and click **Save** to commit your changes.

Create an Authentication Policy

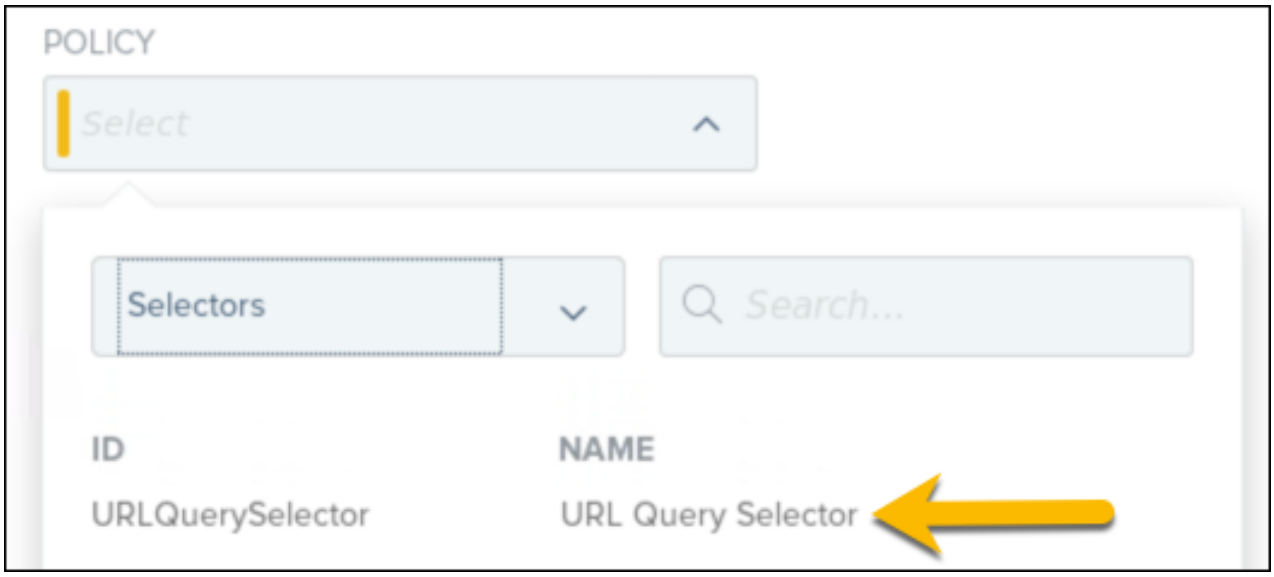
Authentication Policies are powerful, tree driven, decision paths that can be implemented in PingFederate. Up to this point you have configured authentication sources directly in the connection and handled your attribute mappings either through the adapter or in the connection configuration.

With Authentication Policies you can build complex authentication paths to support concepts like MFA, step-up authentication, or to determine an authentication method based on specific requirements.

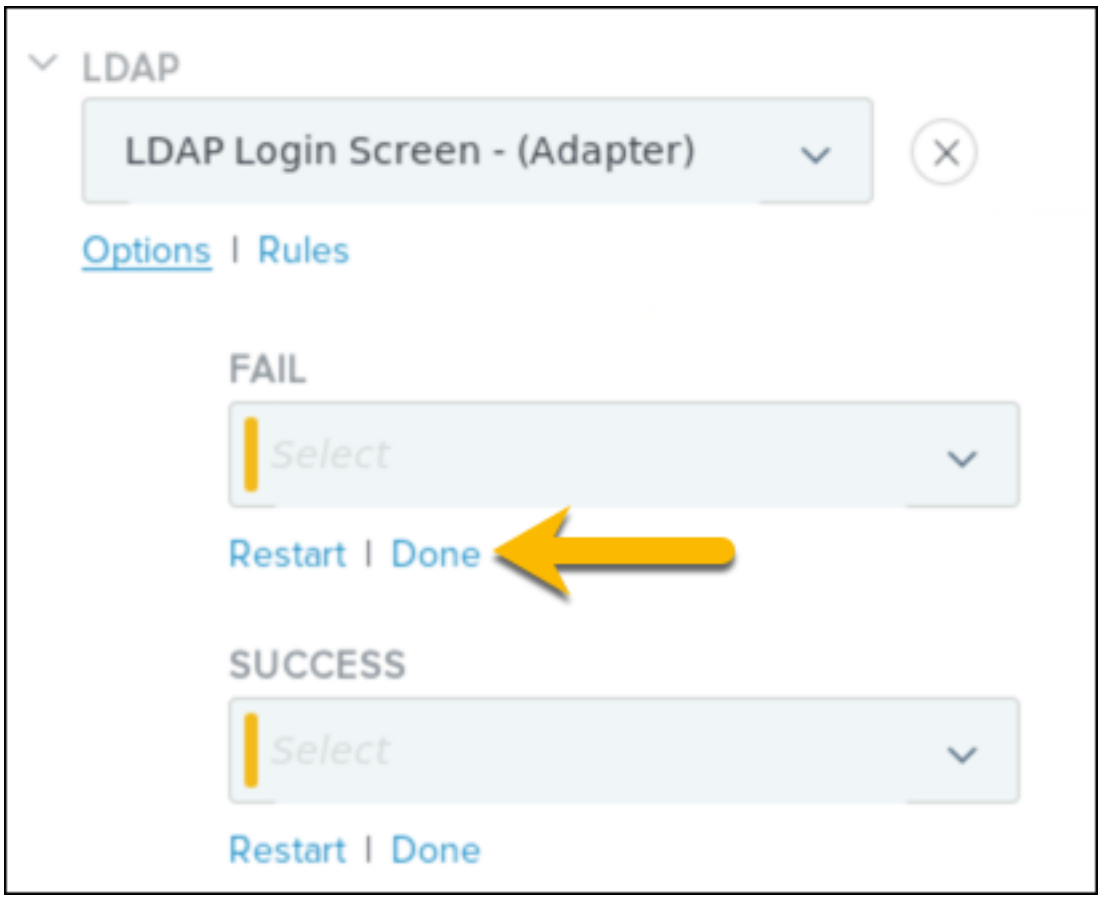
In this section you will build a simple Authentication Policy that uses one of two authentication methods based on a parameter supplied by the SSO URL. This policy will terminate in a Policy Contract which will then be used to provide the attributes for the connection.

1. Navigate to the **Authentication > Policies > Policies** page.
2. Check the **IDP AUTHENTICATION POLICIES** checkbox to enable policies.
3. Click the **Add Policy** button, the **Policies** screen will display.
4. In the NAME field enter **Authentication Policy with URL Selector**.
5. Click the **POLICY** drop-down to display the menu.
6. Change the menu drop-down from **IdP Adapters to Selectors**.

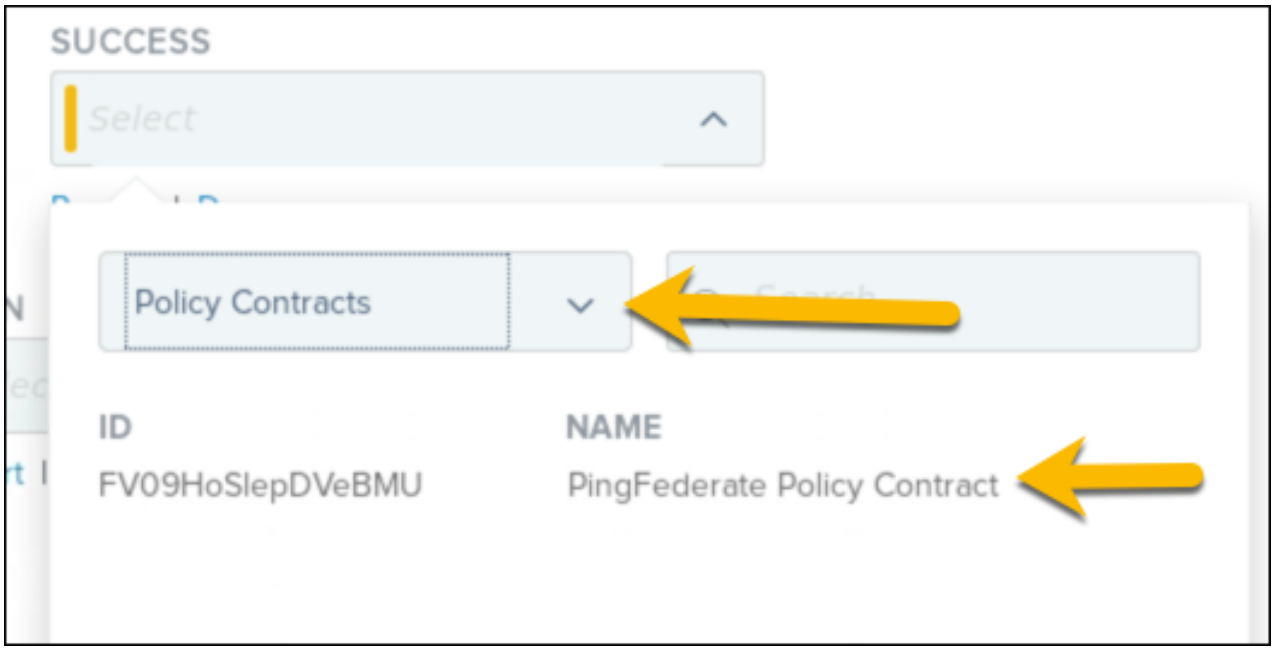
7. Select the **URL Query Selector** you created previously.



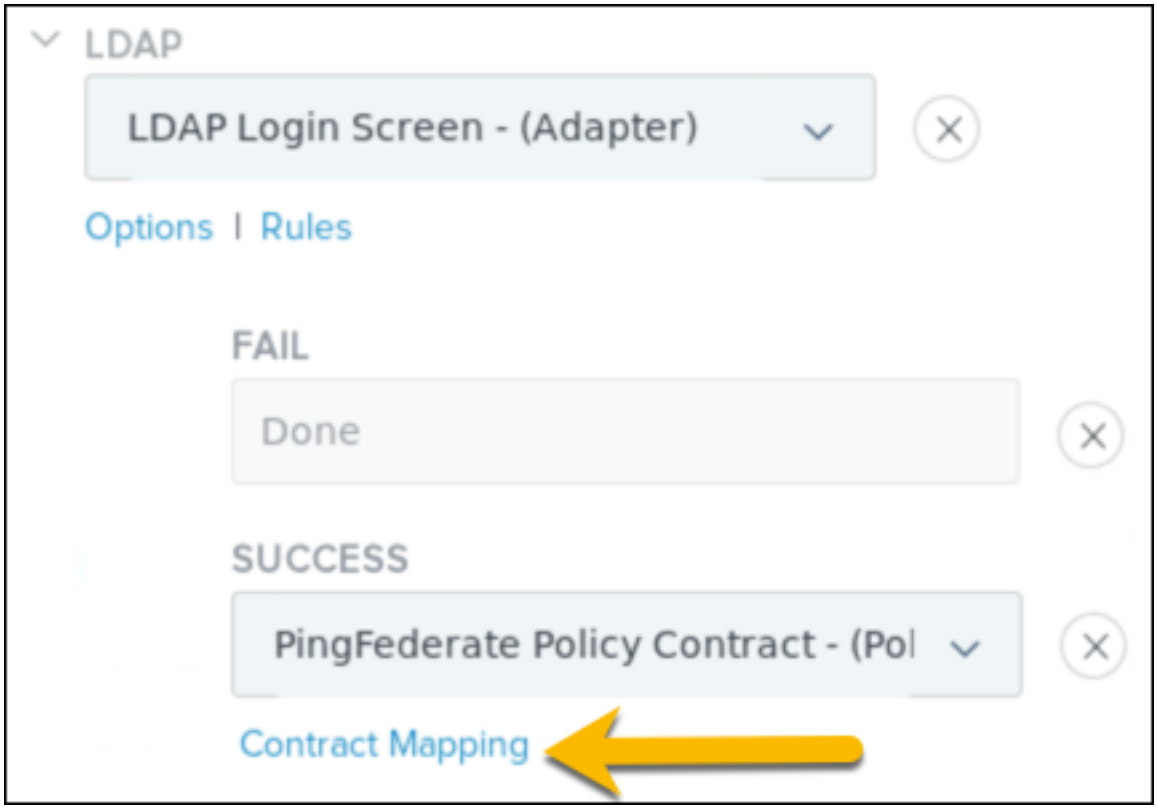
8. Click the **LDAP** drop-down and select the **LDAPLogin** adapter.
9. Under the FAIL drop-down click the **done** link. This tells PingFederate that if the user fails to authenticate then the transaction should be terminated.



10. In the SUCCESS drop-down select the **PingFederate Policy Contract** you created earlier.



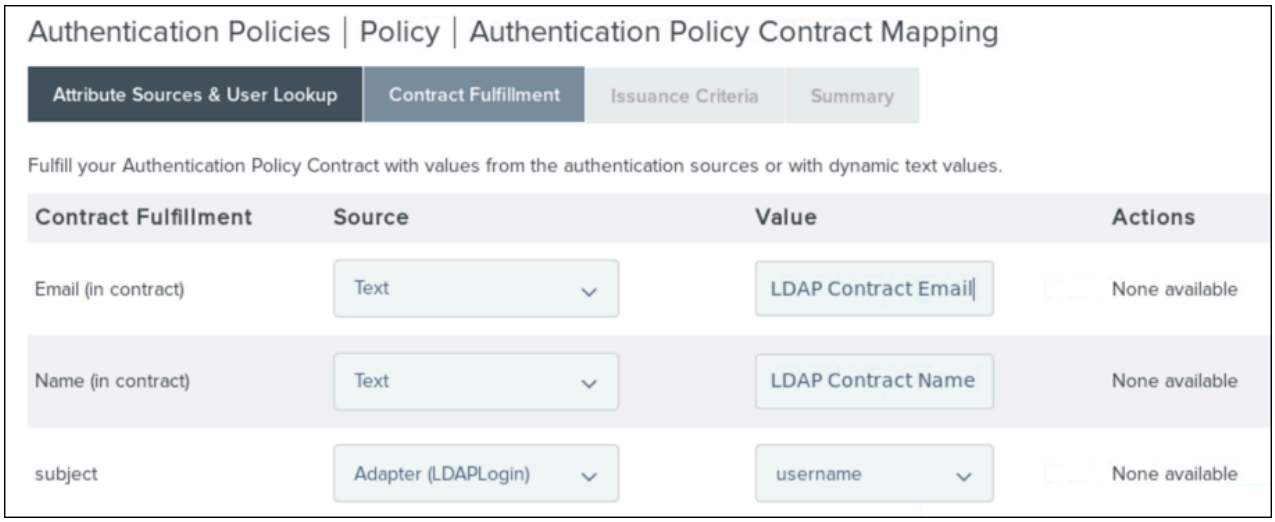
11. Click the **Contract Mapping** link to map your attributes into the contract, the **Attribute Sources & User Lookup** page will display.



Note: For testing purposes, we will only be using text attributes. When using policy contracts the process for mapping attributes from an external data store is identical to doing the same thing within an adapter or within a connection.

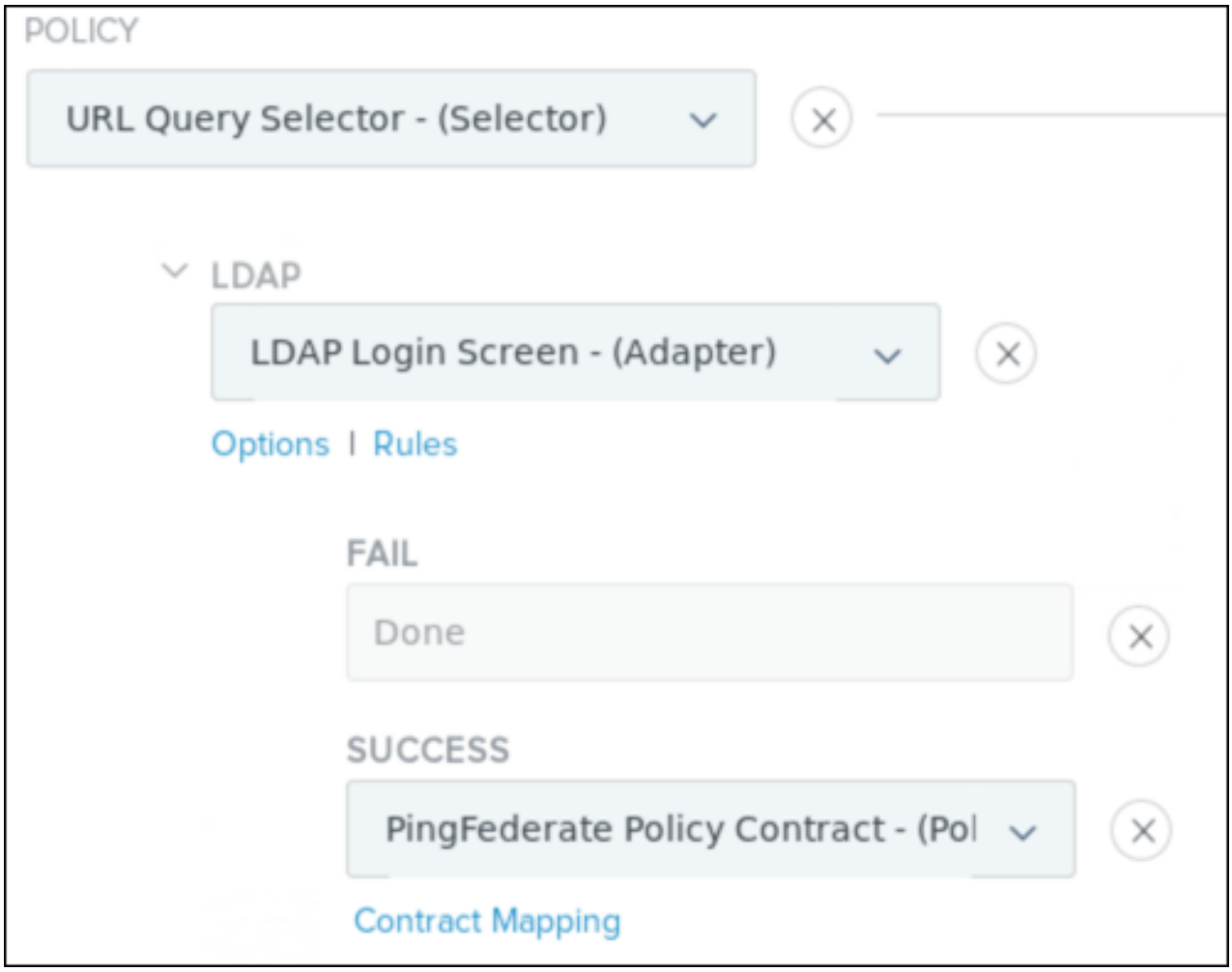
12. We will not be adding any external data stores at this time, click **Next**. The **Contract Fulfillment** page will display.

13. For the subject attribute select **Adapter (LDAPLogin)** in the source drop-down.
14. Select **username** in the Value drop-down.
15. For the other two attributes select **Text** in the source drop-down.
16. Enter the following values:
    - Email (In contract): **LDAP Contract Email**
    - Name (In contract): **LDAP Contract Name**



17. Click **Next**, the **Issuance Criteria** page will display.
18. We will not be configuring any issuance criteria, click **Next** to continue to the **Summary** screen.
19. Review the contract mapping configuration, when you are finished click **Done**.

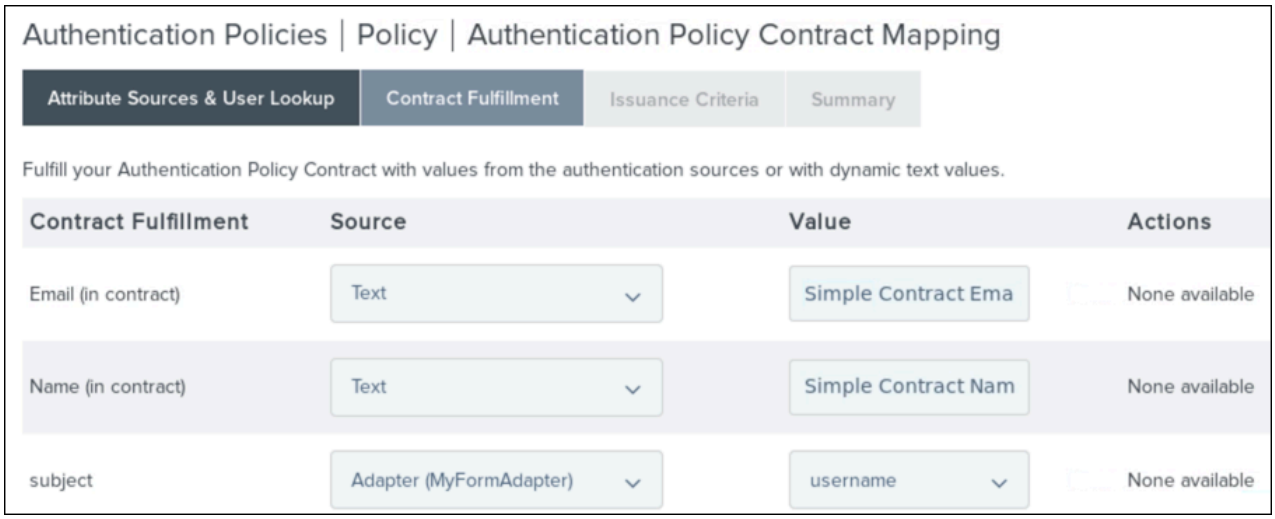
20. The first part of your policy should look similar to the image below.



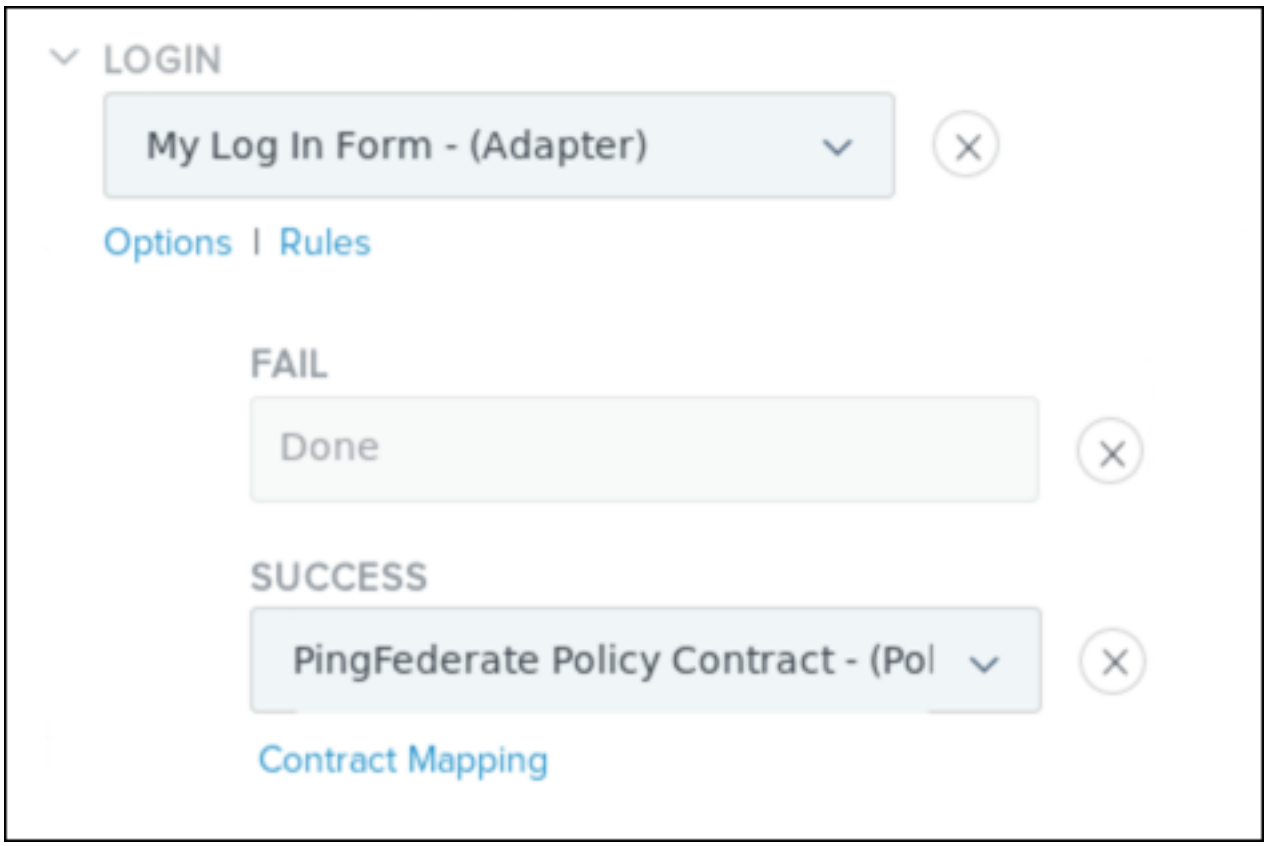
21. In the LOGIN drop-down select the **MyFormAdapter** adapter.
22. Under the FAIL drop-down click **Done**.
23. In the SUCCESS drop-down select the **PingFederate Policy Contract**.

    Note: You can use the same contract for multiple mappings and in multiple policies. The attributes that are mapped into the contract will ultimately be decided by which branches are taken in the policy tree.
24. Under the SUCCESS drop-down click the **Contract Mapping** link.
25. Click **Next**, the **Contract Fulfillment** page will display.
26. For the subject attribute select the **Adapter (MyFormAdapter)** as the source.
27. For the subject attribute select **username** as the Value.
28. For the other two attributes select **Text** in the source drop-down.

29. Enter the following values:
    - Email (In contract): **Simple Contract Email**
    - Name (In contract): **Simple Contract Name**



30. Click **Next**, the **Issuance Criteria** page will display.
31. You will not be configuring any issuance criteria, click **Next** to continue to the **Summary**.
32. Click **Done**.
33. The second part of your policy should look like the image below:

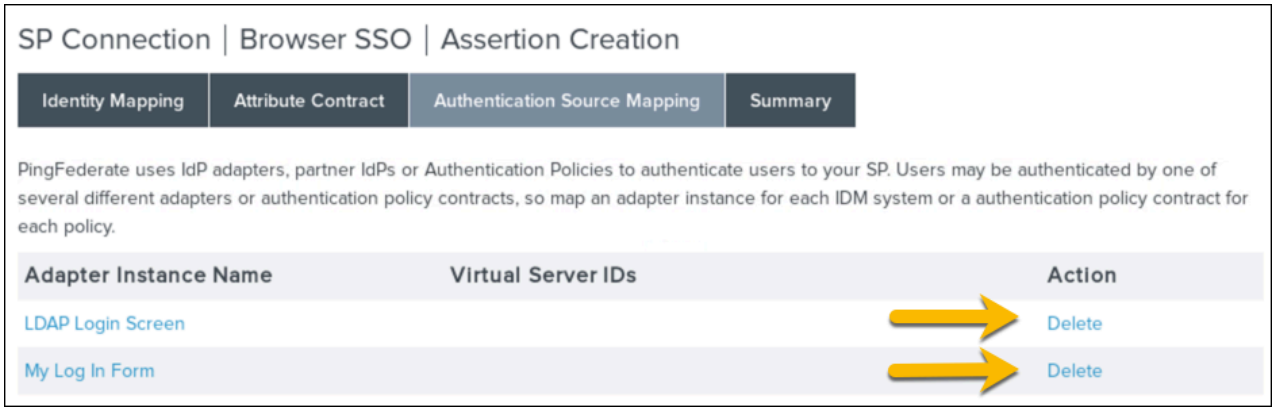


34. Click **Done**.
35. Click **Save** to commit your changes to the console.

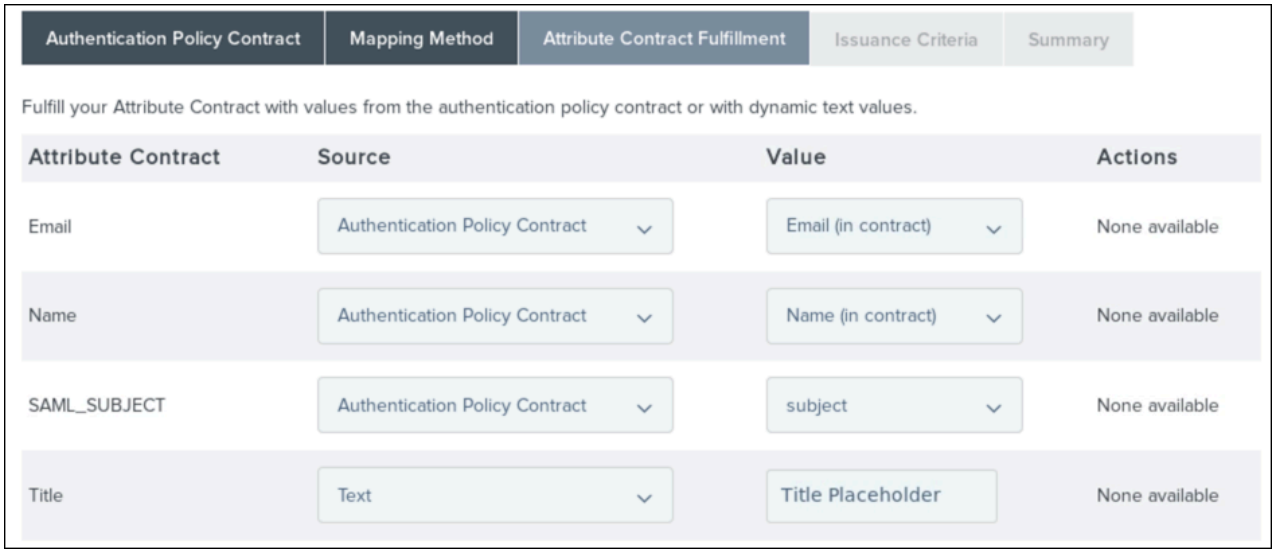
Update the Connection

Now that you have an Authentication Policy setup you can tell PingFederate to start using it with your connection. In this section you will edit the connection configuration to use the new policy contract.

1. Navigate to the **Applications > Integration > SP Connections** page.
2. Click the **Connection to Denver SP** to edit the connection.
3. In the Summary find and click the **Authentication Source Mapping** header.
4. Click the **Delete** link next to both adapters to remove them from the connection.



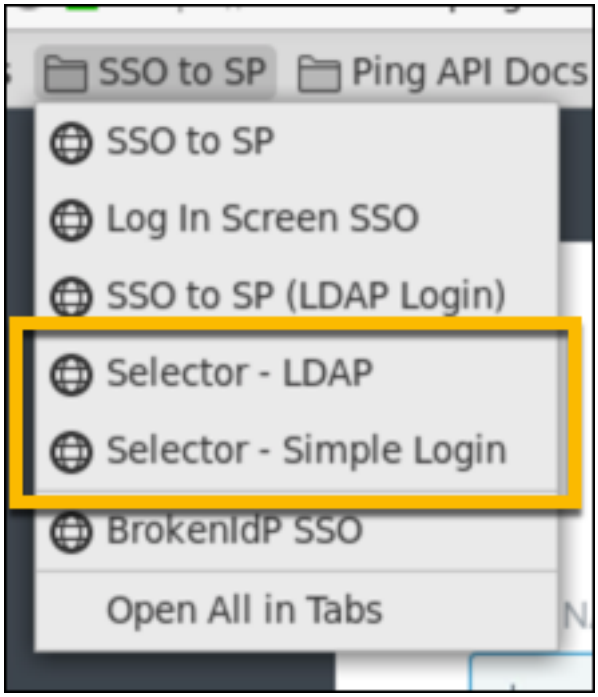
5. Click the **Map New Authentication Policy** button, the **Authentication Policy Mapping – Authentication Policy Contract** screen will display.
6. In the AUTHENTICATION POLICY CONTRACT drop-down select **PingFederate Policy Contract**.
7. Click **Next**, the **Mapping Method** screen will display.
8. You will not be mapping attributes from external sources, click **Next**.
9. Select the following parameters for the attribute mapping:
   - Email: **Authentication Policy Contract** / **Email (In contract)**
   - Name: **Authentication Policy Contract** / **Name (In contract)**
   - SAML_SUBJECT: **Authentication Policy Contract** / **subject**
   - Title: **Text** / **Title Placeholder**



10. Click **Next**.
11. Click **Save**.

Testing the Contract

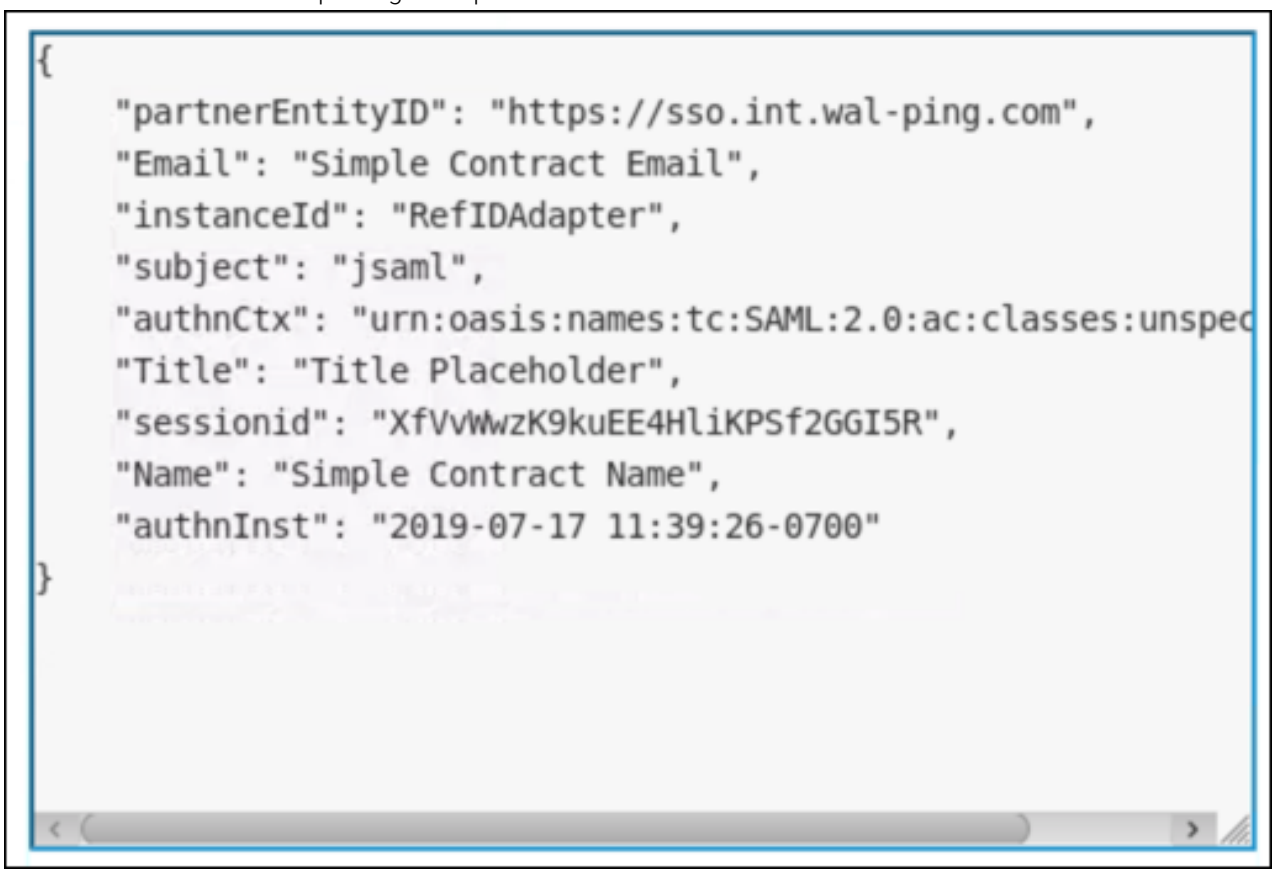
Now that your connection is configured to use the policy contract it is time to test it out. Your Firefox browser already has two bookmarks setup in the **SSO to SP** folder with the appropriate query parameters.



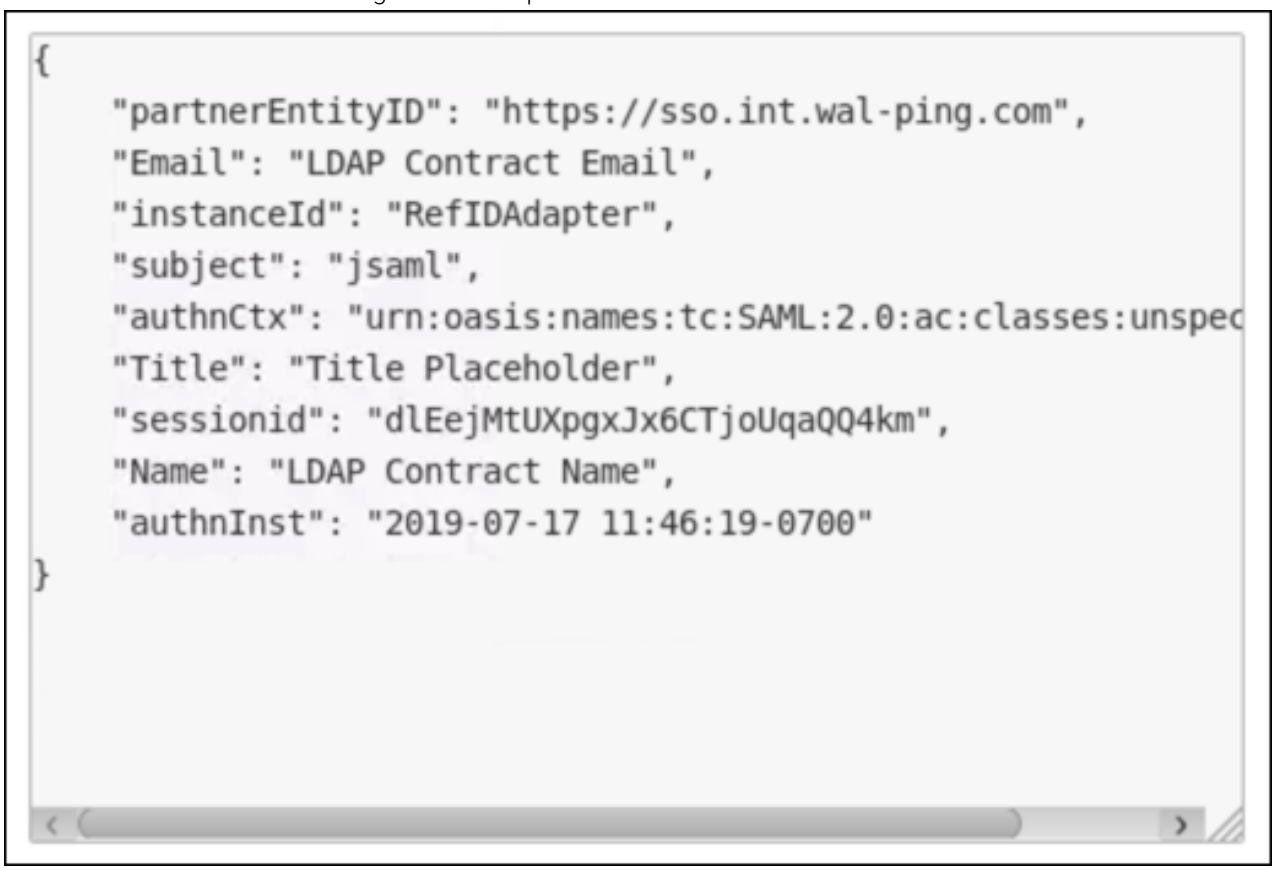
Using the **Selector – LDAP** bookmark should take you to the LDAPLogin adapter (with the yellow background), and using the **Selector – Simple Login** bookmark will take you to the MyFormAdapter adapter.

Try both bookmarks and login using the **jsaml** account. Pay attention to the attributes received by the SP Sample App, you should see that the attribute are different depending on which adapter was used to authenticate.

Authenticated with the Simple Login adapter:



Authenticated with the LDAP Login Form adapter:



Summary

As an administrator Authentication Policy Contracts can be a powerful tool for building complex authentication mechanisms into your SSO flows. They can be used to specify adapters used for MFA, step-up authentication, or to direct the user to a specific authentication mechanism depending on the user's IP address (internal or external to the company for example).

Lab 7 - Troubleshooting SSO Events

Introduction

In this exercise, you will practice troubleshooting various problems with Single Sign On (SSO). Your classroom systems contain instances of PingFederate that been preconfigured with common issues which, break the SSO process. You will attempt to discover the cause of these errors by analyzing PingFederate logging data and critical thinking.

This lab should take 45 minutes to complete.

Objectives

- Troubleshoot common problem scenarios in PingFederate
- Use the PingFederate server logs to determine the root cause of connection problems

Description of tasks to perform

You will be making use of pre-configured PingFederate instances that have been seeded with common SSO and configuration problems. Follow the lab instructions and work your way through each scenario.

Scenario A

Prepare Your Environment

1. Stop your PingFederate SP instance using the `systemctl stop pingfed-den.service` command.
2. Change to the `/home/ping/Desktop/Trouble Shooting/pingfederate-12.0-BrokenSP1/pingfederate/bin` directory.
3. Start the broken SP instance using the `./run.sh` command.

Problem Statement

You have received a report from a user that they cannot log on to a Service Provider's application. You've been asked to fix this. Hint: Once you fix this, be aware that you might have the same problem in a later lab as well.

Steps to Reproduce the Error

1. Open Firefox.
2. Click the SSO to SP > Selector - LDAP bookmark.
3. Log in using the smanager/2Federate credentials.

Scenario B

Prepare Your Environment

1. Stop the instance of the SP PF you just fixed by going to the terminal prompt where it is running and clicking ctrl-c.
2. Change to the directory of the next broken SP instance using the `cd ../../../pingfederate-12.0-BrokenSP2/pingfederate/bin` command.

   Note: The double dot tells Terminal to back up a directory. We are going back two directories, then into the BrokenSP2 directory.
3. Use the `./run.sh` command to start this PingFederate instance.

Problem Statement

You have received a report from a user that they are cannot log on to a Service Provider's application.

Steps to Reproduce the Error

1. Open Firefox.
2. Click the SSO to SP > Selector - LDAP bookmark.
3. Log in using the smanager/2Federate credentials.

Scenario C

Prepare Your Environment

This scenario builds off the last, finish Scenario B first.

Note: Values in an earlier lab may be of help here.

Problem Statement

The user is being routed to the application, but no attributes are there.

Steps to Reproduce the Error

1. Close and reopen Firefox.
2. Click the SSO to SP > Selector - LDAP bookmark.
3. Login using the smanager/2Federate credentials.
4. Notice under User Attributes there are no attributes listed! There should be a phone number listed in the Email field.

Note: Remember, in an earlier lab you hard-coded the Email attribute to contain the phone number for the smanager user.

Summary

Congratulations, you have completed the SSO Troubleshooting lab. In this lab you learned how to troubleshoot some of the most common configuration errors made when setting up SSO using PingFederate.

Lab 8 - Configure PingFederate as an OAuth Authorization Server, OAuth Scopes and Access Tokens

Introduction

In this exercise you will configure PingFederate as an Authorization Server (AS) to manage the issuance of access tokens. The configuration involves enabling OAuth on your Identity Provider instance (wal-ping), defining some scopes, creating a default Access Token, mapping from the IdP Adapter into the Persistent Grant Contract, and mapping from the Persistent Grant Contract to the Access Token.

This lab should take 30 minutes to complete.

Objectives

- Configure PingFederate as an Authorization Server for the OAuth 2.0 protocol.
- Create OAuth scopes.
- Create an Access Token.
- Map attributes from an IdP Adapter to the Persistent Grants Contract.
- Map attributes into an Access Token, from the Persistent Grant Contract and Data Stores.

Description of tasks to perform

Follow the lab instructions and work your way through each scenario.

Configure PingFederate as an Authorization Server

Define OAuth Scopes

In this section you will configure general settings and policy for your OAuth Authorization Server, including the scope descriptions, authorization code policy, and refresh-token and persistent-grant policy.

Do the following:

1. On the **IdP admin console**, navigate to the **System > OAuth Settings > Scope Management** page.

   Note: Common scopes are available to OAuth clients by default unless individual clients are configured to restrict common scopes.
2. Enter the following new scopes (not scope groups):
   - Scope Value: **admin** {case sensitive}
   - Scope Description: **Request Admin Privileges**
3. Click **Add**.
   - Then add another scope. Scope Value: **edit** {case sensitive}
   - Scope Description: **Request Edit Privileges**
4. Click **Add**.
5. Click **Next**, the **Exclusive Scopes** page will display.
6. Exclusive scopes aren't available to OAuth clients by default unless individual clients are configured to allow exclusive scopes. We are not using exclusive scopes
7. Click **Next**, the **Default Scope** page will display.
8. You need to provide a default scope description for the PingFederate Authorization server. If a client does not send a scope with a request, this will be the default.
   - In the DEFAULT SCOPE DESCRIPTION text field enter: **Read permissions (Default)**
9. Click **Save** to commit your changes.

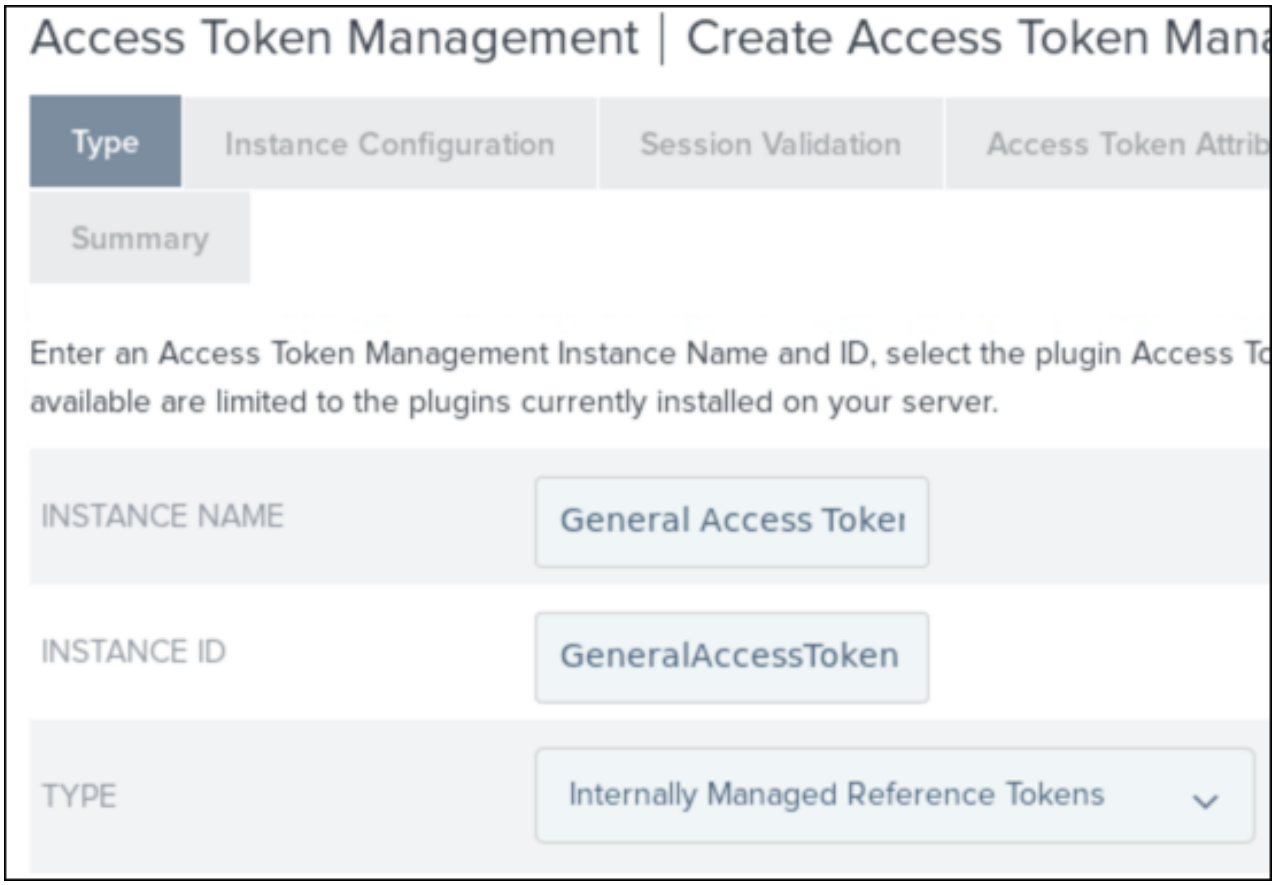
   Note: The scope values have been pre-defined in advance with the client applications; these are the scopes the client applications can request. They are case-sensitive. Remember, scopes are public knowledge. In a real life scenario you would communicate these scopes with the client application developers, typically by listing them on a developer help page on a website you control.

Create an Access Token

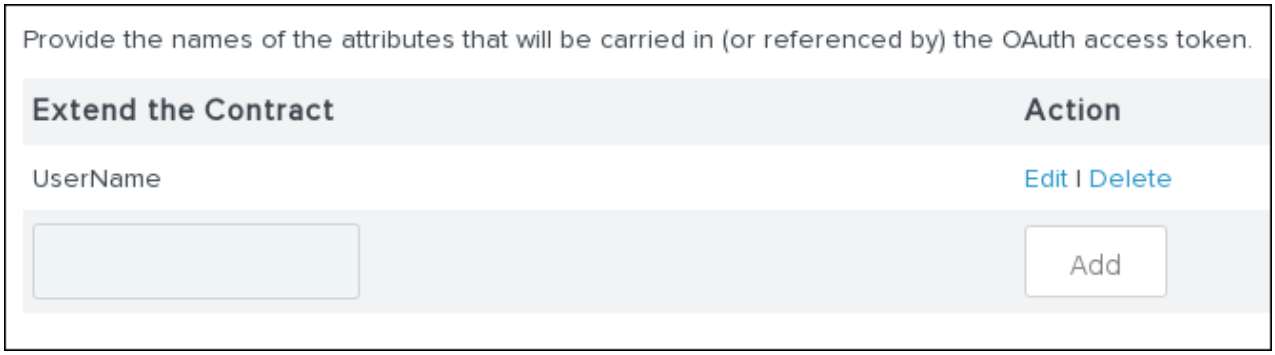
PingFederate, as an authorization server, will be handing out access tokens to clients. In this section you will specify how the PingFederate AS manages those OAuth access tokens. Do the following:

1. Navigate to the **Applications > OAuth > Access Token Management** page.
2. Click the **Create New Instance** button, the **Create Access Token Management Instance – Type** page will display.
3. Set the INSTANCE NAME to **General Access Token**.
4. Set the INSTANCE ID to **GeneralAccessToken**.

5. In the TYPE drop-down select **Internally Managed Reference Tokens**.



6. Click **Next**, the **Instance Configuration** page will display.
7. The Instance Configuration tab has default configuration necessary to issue and validate access tokens. This configuration is specific to the type of token selected on the previous page.
8. Click **Next** to accept the default instance configuration, the **Session Validation** page will display.
9. The Session Validation page allows you to define a policy to bind together the validity of access tokens with the user's session. You can require that the session has not been revoked through a logout or enforce that the user still has an authentication session that has not timed out.
10. Click **Next** to accept the default session validation settings, the **Access Token Attribute Contract** page will display.
11. In the Extend the Contract column enter **UserName**. (Note the capital N)
12. Click the **Add** button.
13. Enter **SecretAcctNumber**. (Note the capital N)
14. Click the **Add** button.



15. Click **Next**, the **Resource URIs** page will display.

16. On the Resource URIs tab you can specify a list of resource URIs that PingFederate OAuth AS can use to select this access token management instance. An OAuth client can optionally include one of these requested resource URIs in a query parameter (aud) when sending its request to the authorization endpoint on the PingFederate OAuth AS.
17. We are not creating resource URIs for this access token management instance. Click **Next**, the **Access Control** page will display.
18. On the Access Control tab you can optionally select a subset of OAuth Clients that are allowed to use this access token management instance.
19. We are not configuring any access control restrictions. Click **Next**, the **Summary** page will display.
20. On the **Summary** tab review the settings.
21. When you are finished click **Save** to commit your changes.

Mapping to the Persistent Grant Contract

Map an IdP Adapter

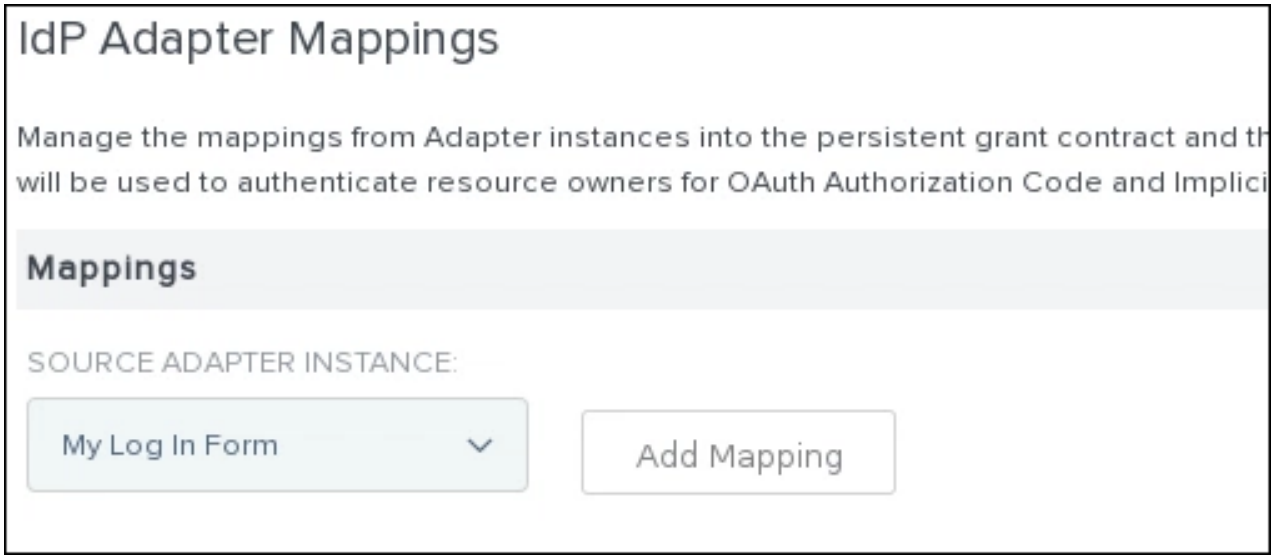
When PingFederate issues tokens, it stores a persistent grant in its own database. This is essentially a way of remembering that the user has granted authorization for the client, so when a client presents PingFederate with tokens, PingFederate can check if those tokens are valid.

Each persistent grant is identified by a USER_KEY. Also, PingFederate displays USER_NAME to the user when requesting authorization. We need to define where PingFederate will get the values for both of these.

In our lab, we are going to authenticate the user using an HTMLForm, and we are going to use the username the user enters as the value for both attributes.

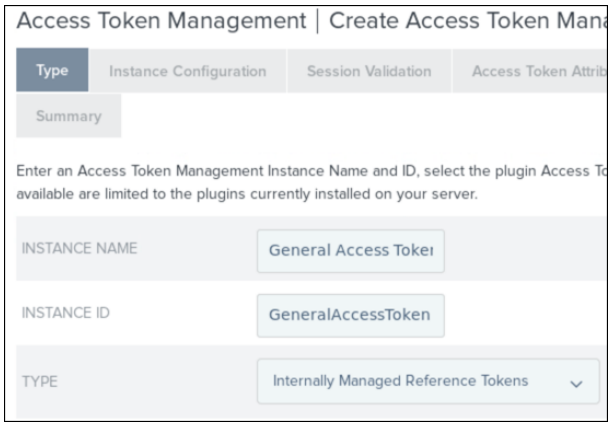
Do the following:

1. Navigate to the **Authentication > OAuth > IdP Adapter Grant Mappings** page.
2. Select **My Log in Form** from SOURCE ADAPTER INSTANCE dropdown.



3. Click the **Add Mapping** button, the **IdP Adapter Mapping – Attribute Sources & User Lookup** page will display.
4. We're not fulfilling the mapping from data store. Click **Next** the **Contract Fulfillment** page will display.
5. For USER_KEY, select **Adapter** and **username**.

6. For USER_NAME, select **Adapter** and **Name**.



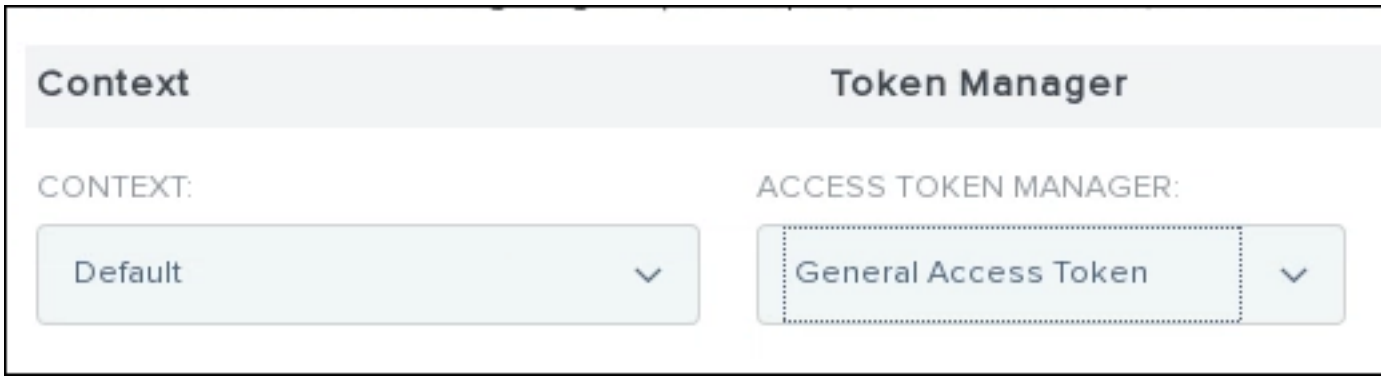
7. Click **Next**, the **Issuance Criteria** page will display.
8. You will not be configuring any issuance criteria, leave everything at the defaults.
9. Click **Next**, the **Summary** page will display.
10. Review the settings, then click **Save** to commit your changes and return to the OAuth Server page.

Map the Access Token

So now PingFederate is going to send the client an access token, but the access token is going to need values to go with it. In this required configuration, we will map attributes to be requested from the OAuth resource server with the access token — the token attribute contract.

Do the following:

1. Navigate to the **Applications > OAuth > Access Token Mappings** page.
2. For Context, select **Default** from the drop-down list
3. For Access Token Manager, select **General Access Token** from the dropdown list.
4. Click the **Add Mapping** button, the **Access Token Mapping – Attribute Sources & User Lookup** page will display.



5. You are not mapping from a data store. Click **Next**, the **Contract Fulfillment** page will display.
6. On the Contract Fulfillment tab you will set the attributes that only a Resource Server will receive when it successfully validates the Access Token:
   - Set the **UserName** to map from the **Persistent Grant** source and choose **USER_KEY** attribute.
   - Set the **SecretAcctNumber** to map from a **Text** source value of **999-7777-1234**.
7. Click **Next**, the **Issuance Criteria** tab will display.
8. You will not be configuring any issuance criteria, leave everything at the defaults.
9. Click **Next**, the **Summary** tab will display.
10. Review the settings, then click **Save** to commit your changes.

Configure Cross-Origin Resource Sharing (CORS)

Allow Cross-Origin Resource Sharing

As needed, administrators can add or remove allowed origins using the administrator console. Once configured, client-side web applications from the trusted origins are allowed to make requests to the PingFederate authorization server for the purpose of accessing protected resources, such as obtaining (or renewing) access tokens (with refresh tokens), presenting access tokens for revocation, querying additional claims (user attributes), and retrieving OpenID Provider configuration information and JSON Web Key Sets.

Before we can test our clients with the OAuth 2.0 Playground app, we need to add its URL to the Cross-Origin Resource Sharing list on our PingFederate server:

1. From the PingFed IdP Admin console, navigate to the System > OAuth Settings > Authorization Server Settings page.
2. Towards the bottom of the page locate Cross-Origin Resource Sharing Settings.
3. In Allowed Origin field enter the URL of the OAuth 2.0 Playground app: https://api.int.wal-ping.com:8443
4. Click the Add button.
5. Click Save at the bottom of the screen.

Summary

Congratulations, you have completed the initial configuration of your Identity Provider instance as an OAuth Authorizations server. In this lab you learned how to enable the OAuth protocol in Server Settings, setup a list of Scope names and a default scope description, create an Access Token, map IdP adapter attributes to USER_KEY and USER_NAME attributes in the Persistent Grant Contract, and map attributes into an Access Token including the unique ID of the user who authorized the grant (USER_KEY) from the Persistent Grant Contract.

Lab 9 - Create an OAuth Client for Client Credentials Grant Type

Introduction

In this exercise, you will practice how to define an OAuth Client in PingFederate that uses the Client Credentials Grant Type.

To help us understand the flow of the grant type, we'll be using a simulator, called the OAuth 2 Playground, to emulate how a typical client interacts with the PingFederate AS to obtain an access token.

In this first exercise testing the Client Credentials client with Playground will get a failure when we try to validate the access token. This is because we have not yet defined a Resource Server Client in PingFederate. You will do that in the next lab, and then re-test this client.

This lab should take 25 minutes to complete.

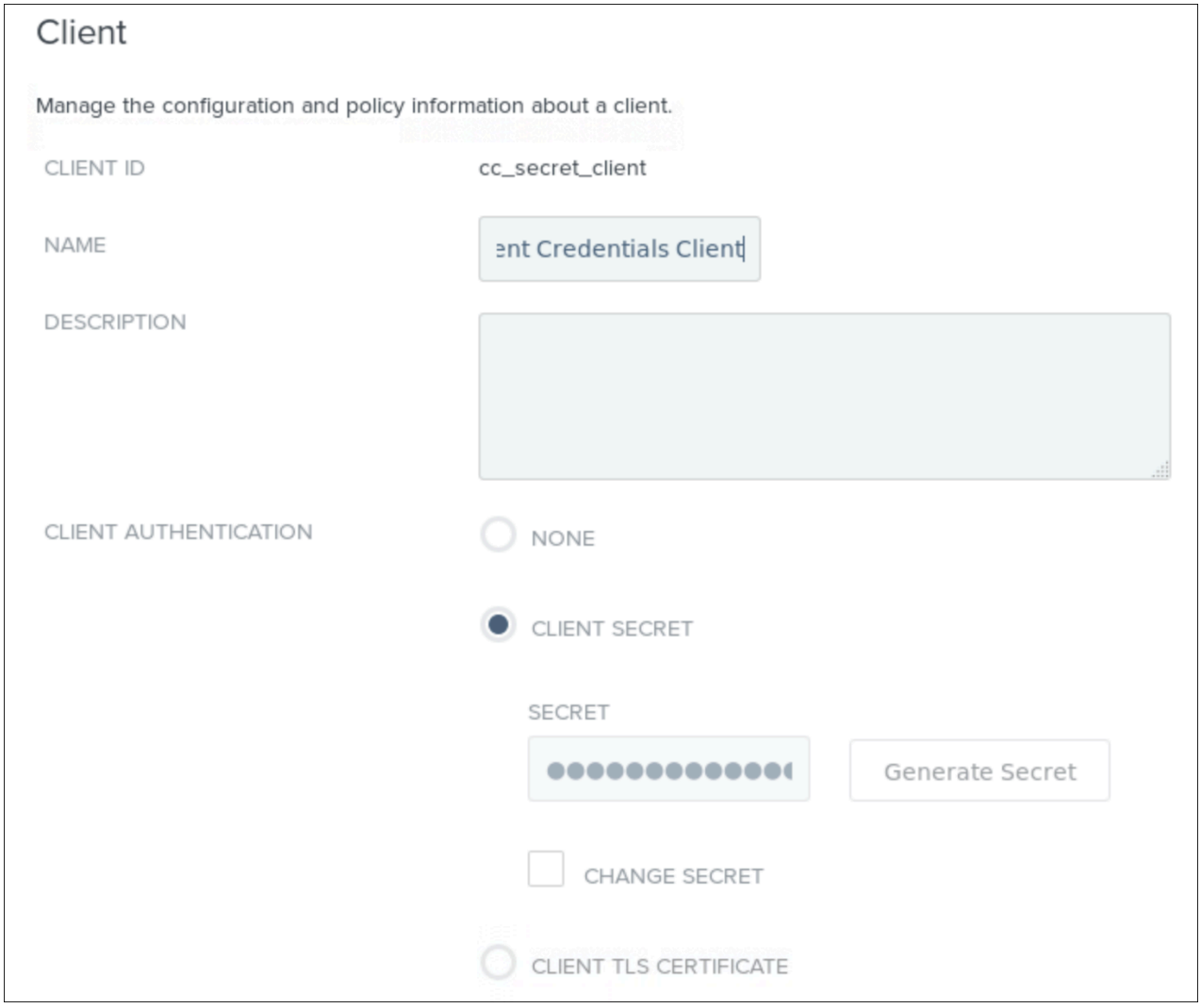
Objectives

- Configure OAuth Playground for testing PingFederate OAuth Client configurations
- Create OAuth Client using the Client Credentials Grant Type

Create the OAuth Client using Client Credentials Grant Type

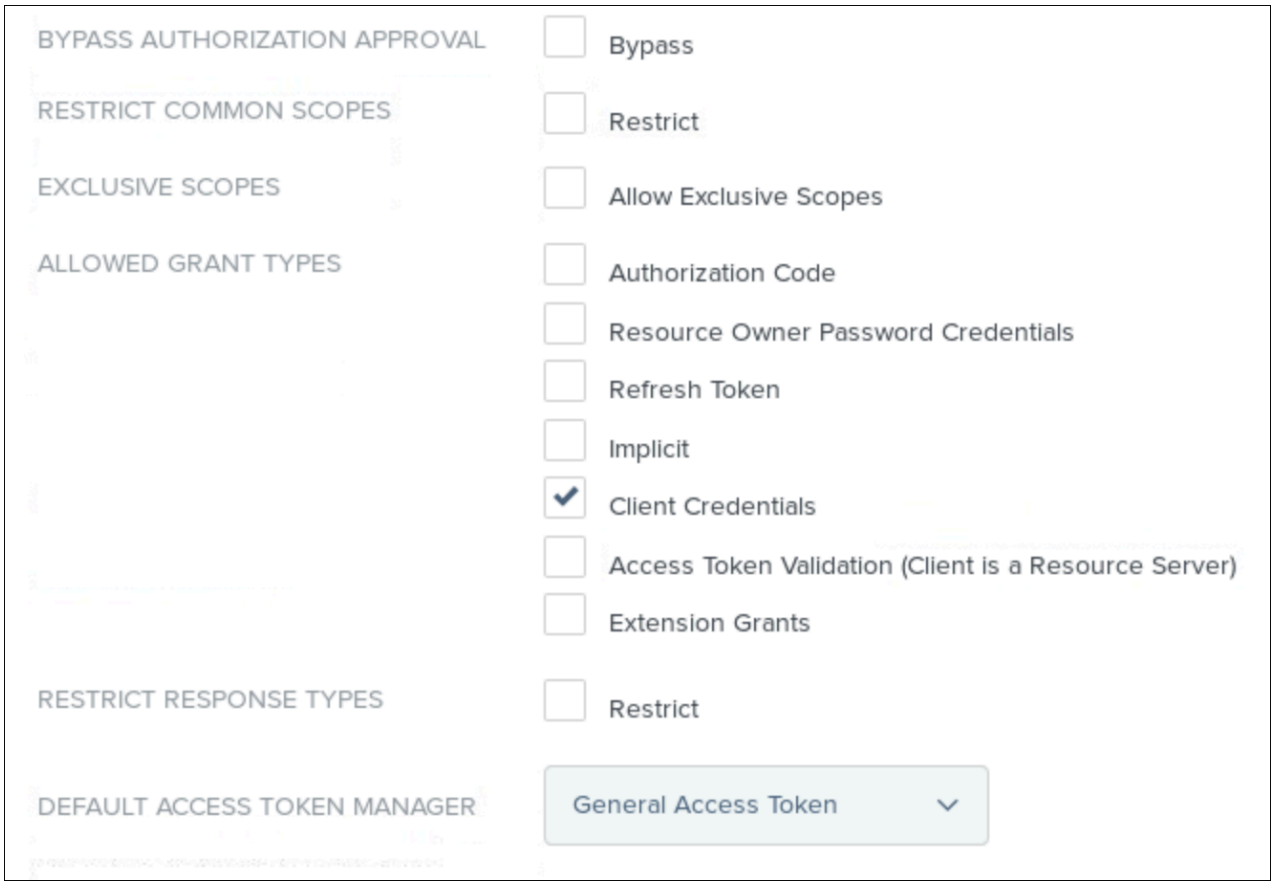
1. Navigate to the **Applications > OAuth > Clients** page.
2. Click the **Add Client** button, an empty **Client** page will display.
3. Configure the client:
   - CLIENT ID: **cc_secret_client** {case sensitive}
   - NAME: **Client Credentials Client**
   - CLIENT AUTHENTICATION: **Client Secret** (you will need to check the **CHANGE SECRET** box to enable the field)
   - SECRET: **2Federate**

   Note: Don't click **Generate Secret**. This generates a new random client secret, and is recommended for production use.



4. Scroll down to ALLOWED GRANT TYPES: **Select Client Credentials**

5. Scroll down to DEFAULT ACCESS TOKEN MANAGER. Select **General Access Token** from the drop-down list.



Note: We could leave it as Default, as the default will use the General Access Token we created previously. Here we are explicitly choosing General Access Token and that will not change if we were to change the Default Access Token later.

6. Click **Save**.

Configure the OAuth 2.0 Playground

Before we can test the client we need to configure OAuth Playground to work with your Authorization Server.

Configure Playground Settings

1. In Firefox, click the **OAuth 2.0 Playground** bookmark, the **OAuth 2.0 Playground Welcome** page will display.
2. In the left-hand navigation bar, click **Settings**. The **Settings** page will display.
3. Change the BASE URL field to the base URL and port number of your IdP instance: **https://sso.int.wal-ping.com:9031**.
4. Scroll to the bottom of the page and click **Save**.

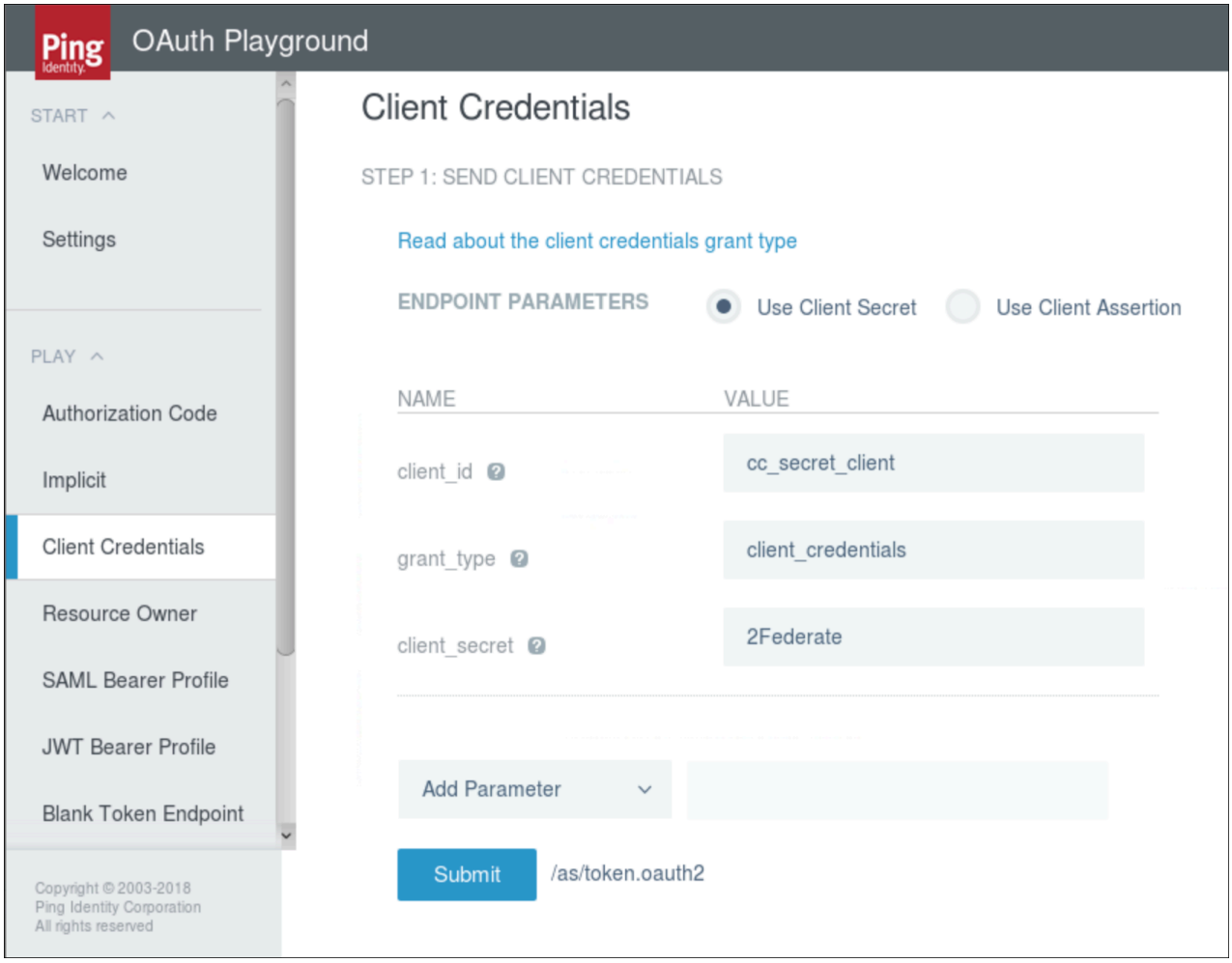
Test Your Configuration

PingFederate is now configured with an OAuth Client using the Client Credentials Grant Type. In this section you will complete the exercise by testing your new Client.

An example of this use case is when a 3rd party server or applications wants to access a protected resource, such as REST API.

Do the following:

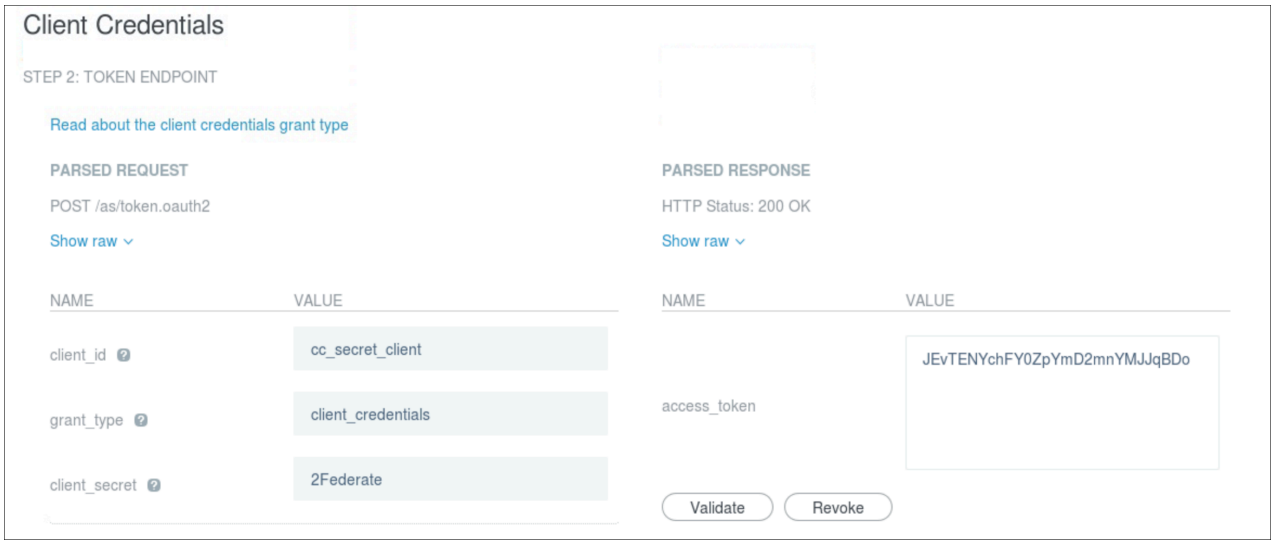
1. In a new tab, click the **OAuth 2 Playground** bookmark.
2. In the PLAY section of the left-hand navigation bar click **Client Credentials**.



Note: The OAuth2 Playground simulates a server client connecting to PingFederate AS using the cc_secret_client client ID, the client_credentials grant type, and the 2Federate client secret that you configured in PingFederate earlier.

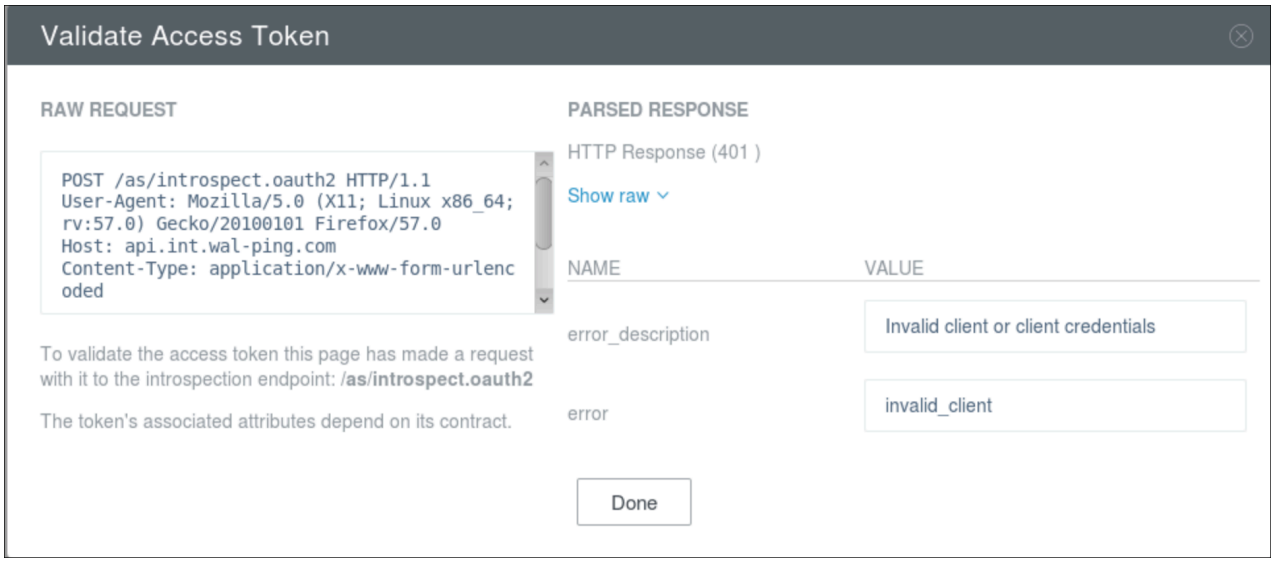
3. Click **Submit**.

   OAuth 2.0 Playground sends the request to PingFederate and will receive an Access Token in return with an HTTP Status: 200 OK, which is summarized on the STEP 2: TOKEN ENDPOINT page.



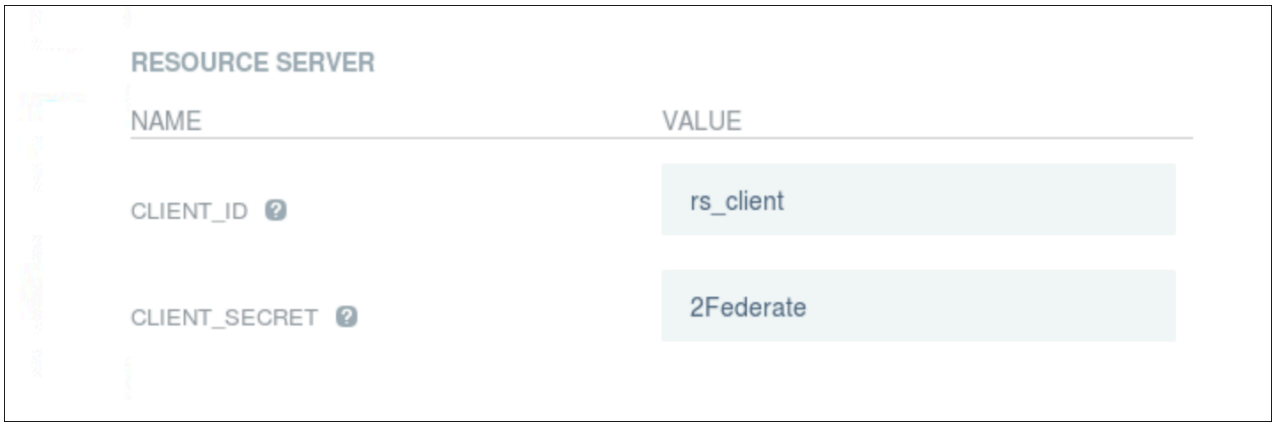
4. Now to validate the Access Token: Click on **Validate**.

   Notice we receive a 401 HTTP Response with an **Invalid client or client credentials** error message.



5. This is because when we click **Validate** OAuth Playground is now acting as a Resource Server and is making a request to the Authorization Server using a Client ID that has not yet been created
6. In the START section of the left-hand navigation bar click **Settings**.

7. Scroll to the bottom and note the settings that Playground is using to act as a Resource Server



So that we can successfully validate Access Tokens in Playground we need to create a Client for the Resource Server on the Authorization Server, and that is what we will do in the next lab, at which point we can re-test the Client Credentials workflow and successfully validate the Access Token returned.

Summary

Congratulations, you have enabled CORS on the PingFederate OAuth Server to allow cross origin resource sharing with the OAuth 2.0 Playground app, configured OAuth Playground for testing PingFederate OAuth Client configurations, and create an OAuth Client on the Authorization Server using the Client Credentials Grant Type.

Lab 10 - Create an OAuth Client for a Resource Server

Introduction

In this exercise, you will practice how to define an OAuth Client for a Resource Server using the Access Token Validation Grant Type.

In the last exercise we got an “Invalid client or client credentials” error message when we tried to validate the access token. This is because we have not yet defined a Resource Server Client in PingFederate. You will do that now.

When you initially re-test the Client Credentials client you will now get a successful Access Token validation step. However, there is an issue with the contents of that Access Token, and you will fix that by creating a second Access Token Management Instance specifically for clients using the Client Credentials Grant Type.

This lab should take 25 minutes to complete.

Objectives

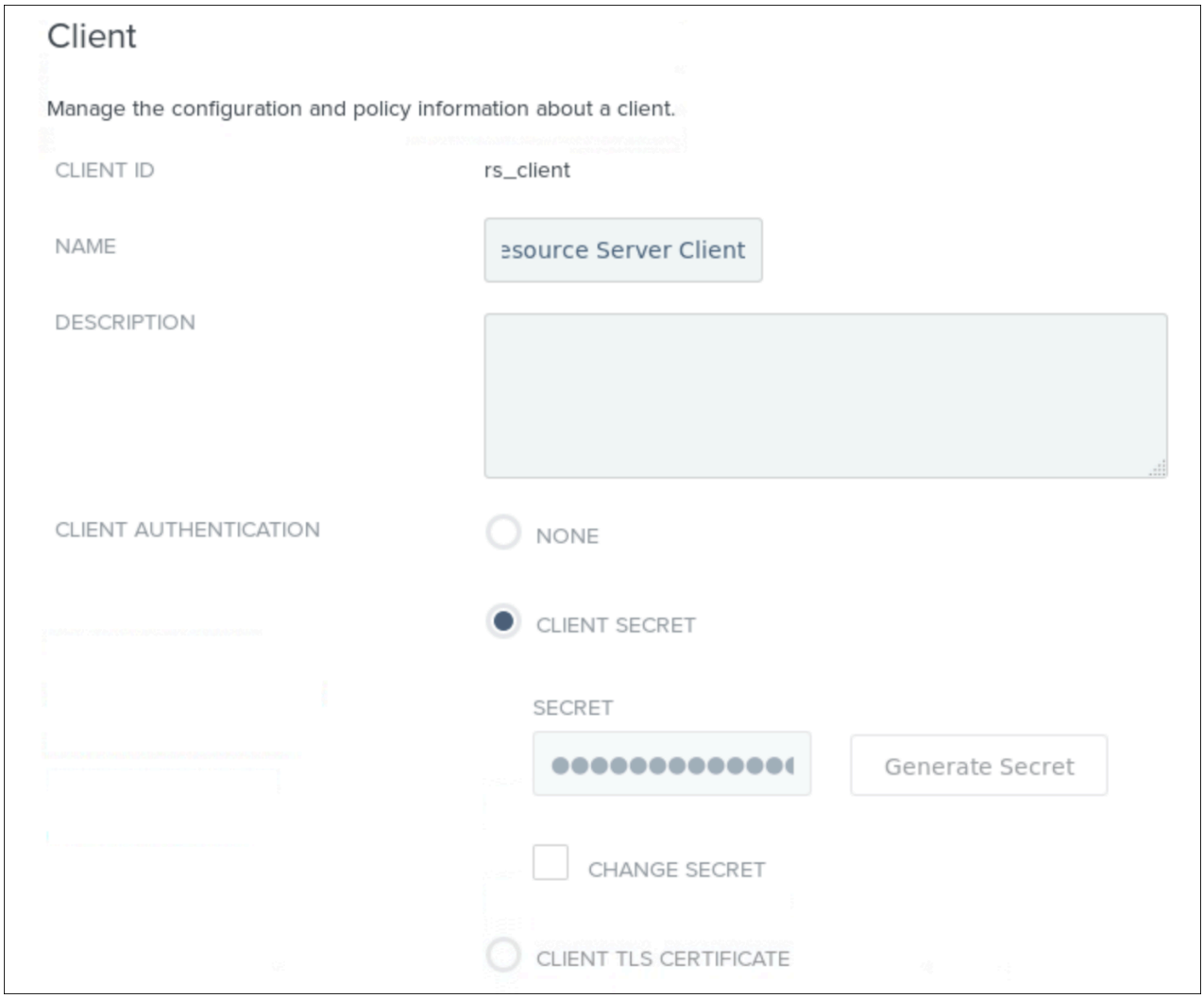
- Create OAuth Client using the Access Token Validation Grant Type for Resource Servers.
- Create an Access Token Management Instance using the Client Credentials Context.

Create the Resource Server OAuth Client

Create the OAuth Client

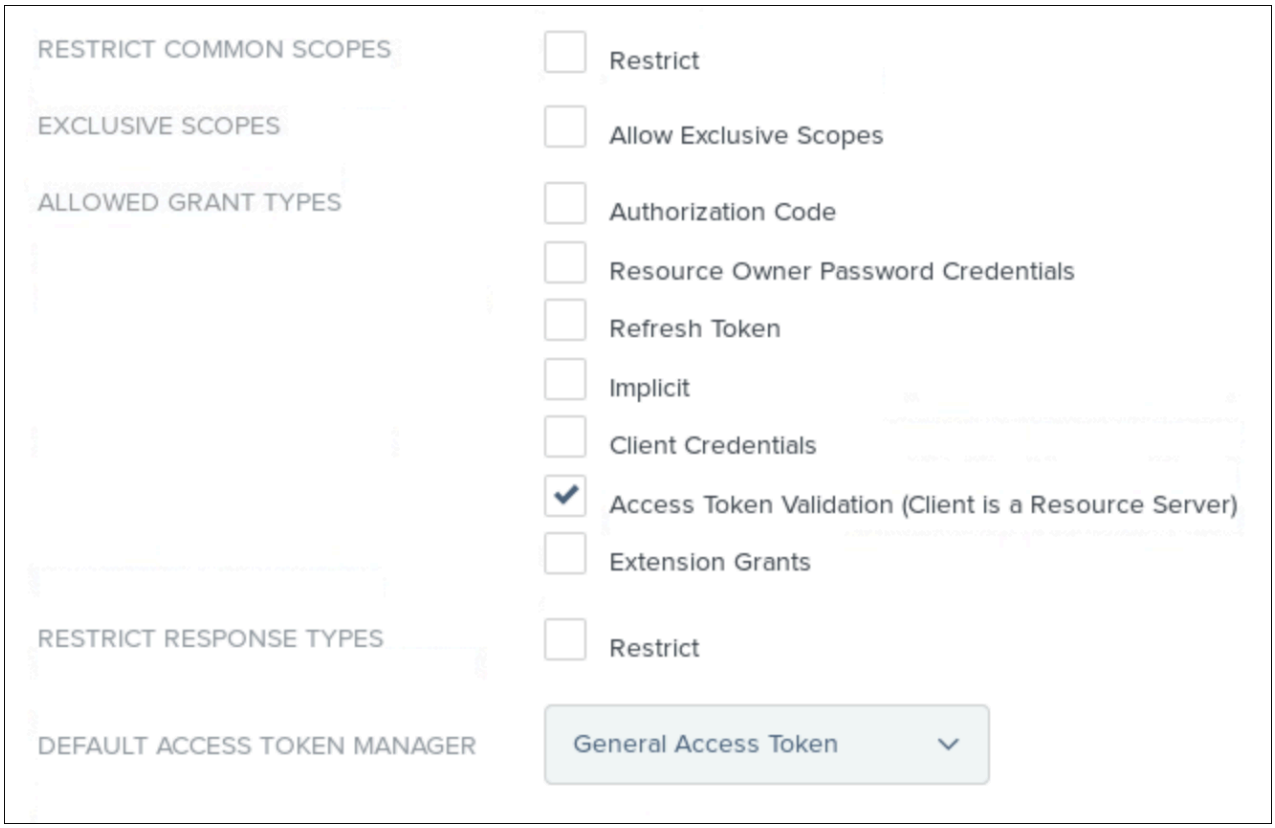
1. Navigate to the **Applications > OAuth > Clients** page.
2. Click the **Add Client** button, an empty **Client** page will display.
3. Configure the client:
   - Client ID: **rs_client** {case sensitive}
   - Name: **Resource Server Client**
   - Client Authentication:
     - Select **Client Secret**,
     - Enter **2Federate** (Remember to check the **CHANGE SECRET** checkbox)

   Note: Don't click **Generate Secret**. This generates a new random client secret, and is recommended for production use.)



4. Scroll down to ALLOWED GRANT TYPES and select **Access Token Validation (Client is a Resource Server)**.

5. Scroll down to DEFAULT ACCESS TOKEN MANAGER. Select **General Access Token** from the list.



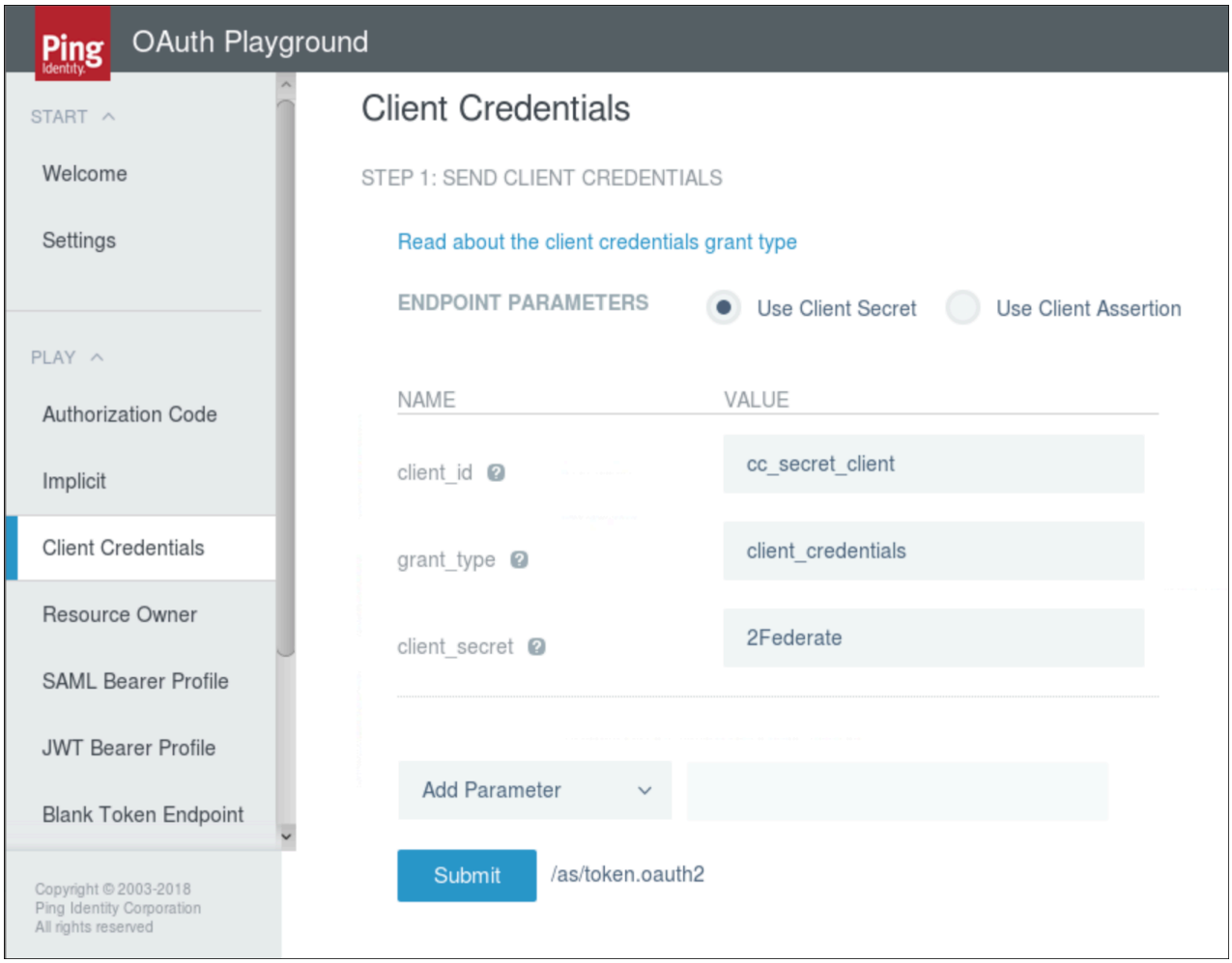
Note: We could leave it as Default, as the default will use the General Access Token we created previously. Here we are explicitly choosing General Access Token and that will not change if we were to change the Default Access Token later.

6. Click **Save**.

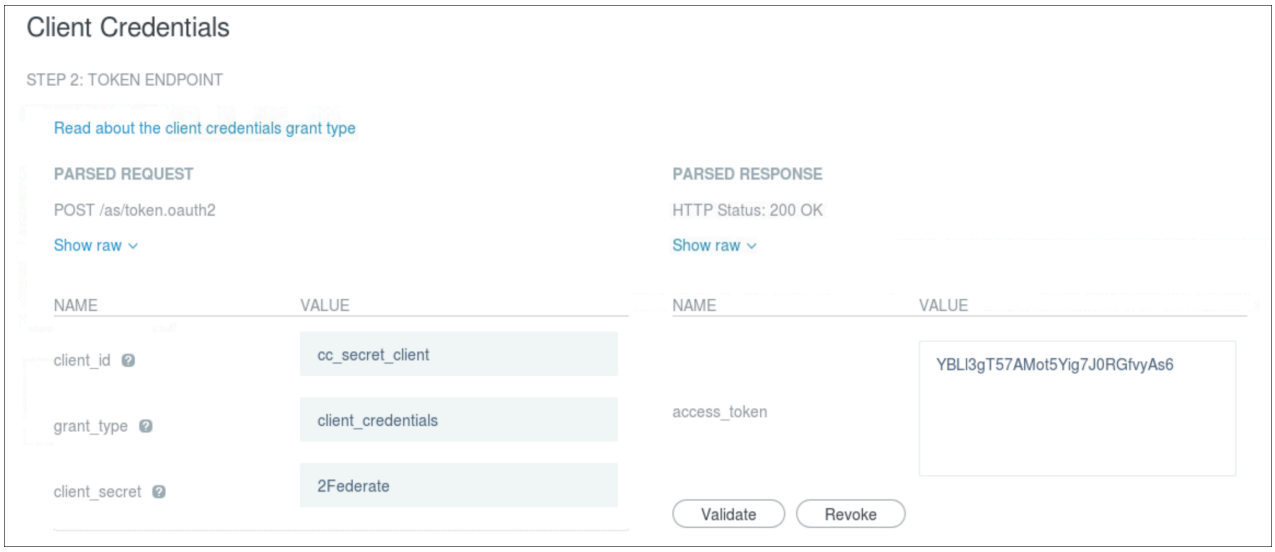
Re-test the Client Credentials Workflow

1. In a new tab, click the **Outh2 Playground** bookmark.

2. In the PLAY section of the left-hand navigation bar click **Client Credentials**.



3. Click **Submit**.
4. OAuth 2.0 Playground sends the request to PingFederate and will receive an Access Token in return with an HTTP Status: 200 OK, which is summarized on the STEP 2: TOKEN ENDPOINT page.



5. Now to validate the Access Token: click on **Validate**.

6. Success! We received a 200 HTTP Response. The payload values are what a Resource Server would see.

7. Look closer at the attributes. The **UserName** and **SecretAcctNumber** we defined in the Access Token are not displayed!
8. This is because the Client Credentials Grant Type has no Resource User from which the Default Access Token Attribute Mapping can map attributes to the General Access Token.
9. To fix this for clients using the Client Credentials Grant Type we need to define another Access Token Attribute Mapping specifically for Client Credentials usage.

Access Token Attribute Mapping for Client Credentials

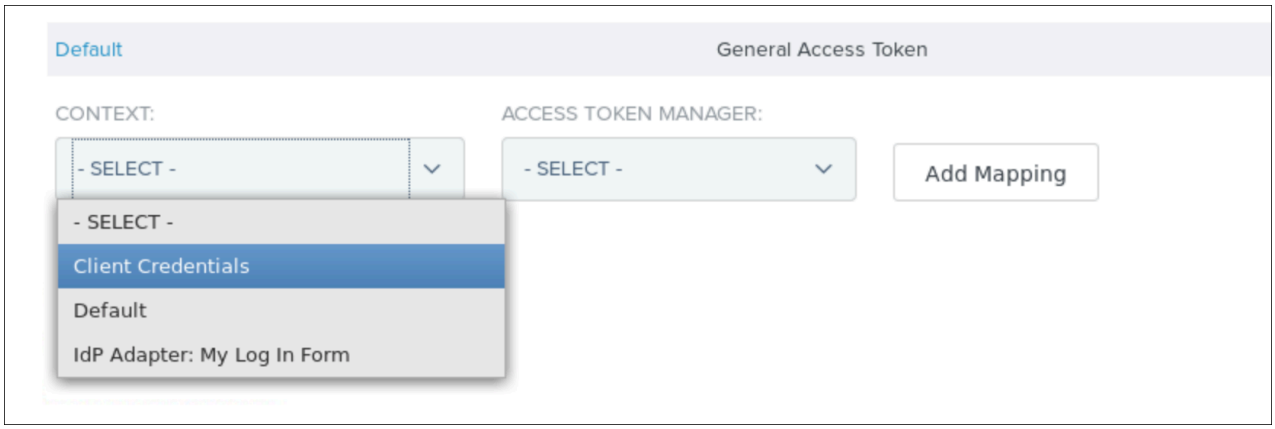
Previously, we created a default Access Token Attribute Mapping for our PingFederate, and this is mapping attributes for a Resource Owner, which we don't currently have.

To map attributes into the Access Token for clients using the Client Credentials Grant Type we need to create a new mapping specifically for Client Credentials. We will do that now, and re-test the client in Playground.

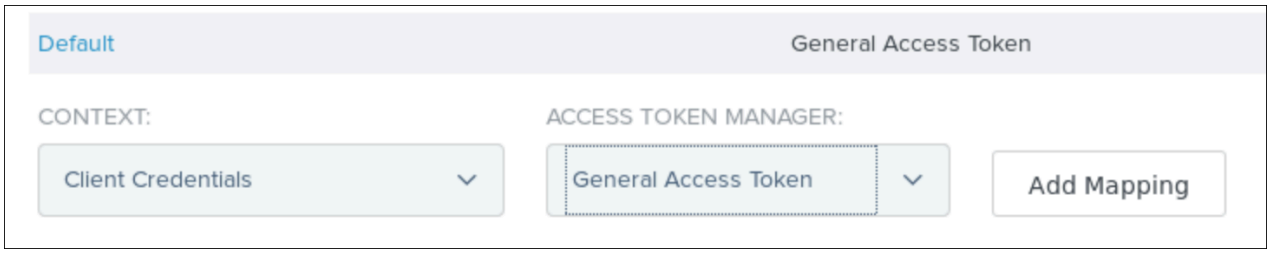
Create a New Access Token Attribute Mapping

1. Navigate to the **Applications > OAuth > Access Token Mappings** page.

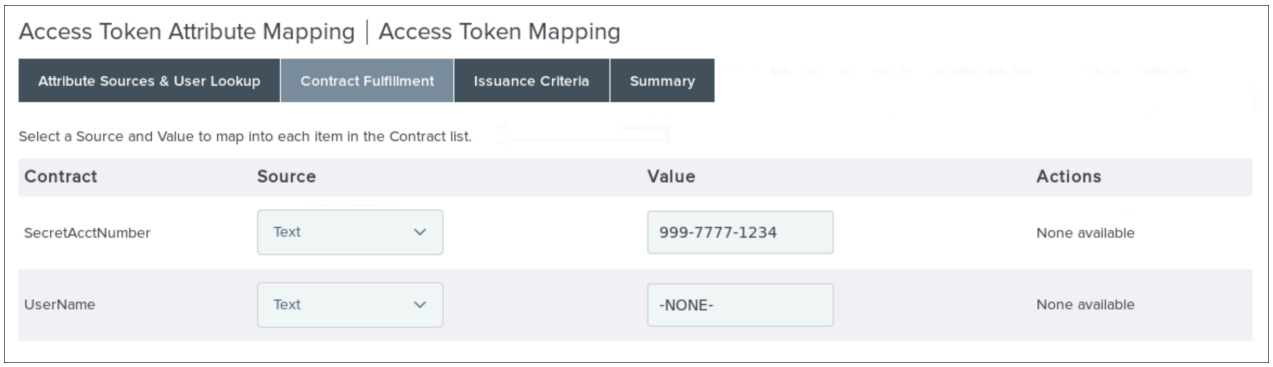
2. Under Context, select **Client Credentials** from the dropdown list.



3. For Access Token Manager, select **General Access Token** from the dropdown list.
4. Click **Add Mapping**.



5. On the Attribute Sources & User Lookup tab we are not mapping from a data store, so click **Next**.
6. On the Contract Fulfillment tab:
   - Set the UserName to map a **Text** source **-NONE-**.
   - Set the SecretAcctNumber to map from a **Text** source **999-7777-1234**.

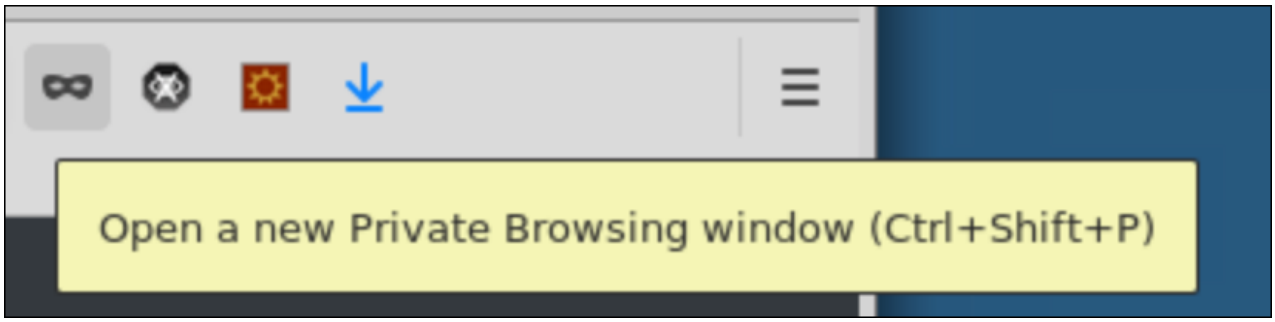


Note: Did you notice that there is no Persistent Grant source? That is because this Access Token Attribute Mapping is specifically for Client Credentials Grant Type, which has no Resource Owner, and therefore no Persistent Grant. We can only use Text, Context, or a Data Store if one is configured on the previous Attribute Sources & User Lookup tab.

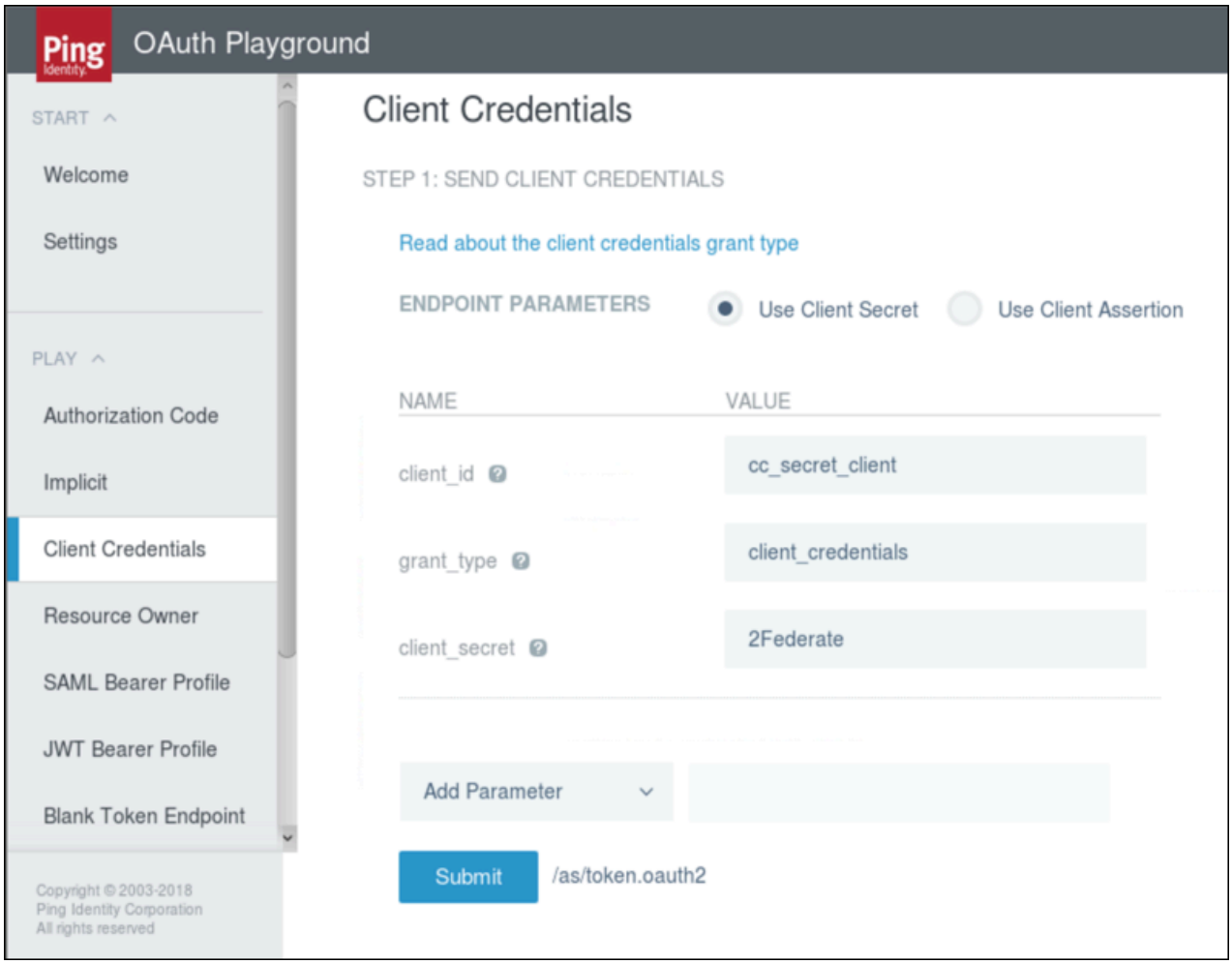
7. Click **Next**.
8. Click **Next**, the **Issuance Criteria** tab will display.
9. You will not be configuring any issuance criteria, leave everything at the defaults.
10. Click **Next**, the **Summary** tab will display.
11. Review the settings, then **Save**.

Re-test the Client Credentials Workflow

1. Open a new Firefox Private Browsing window.

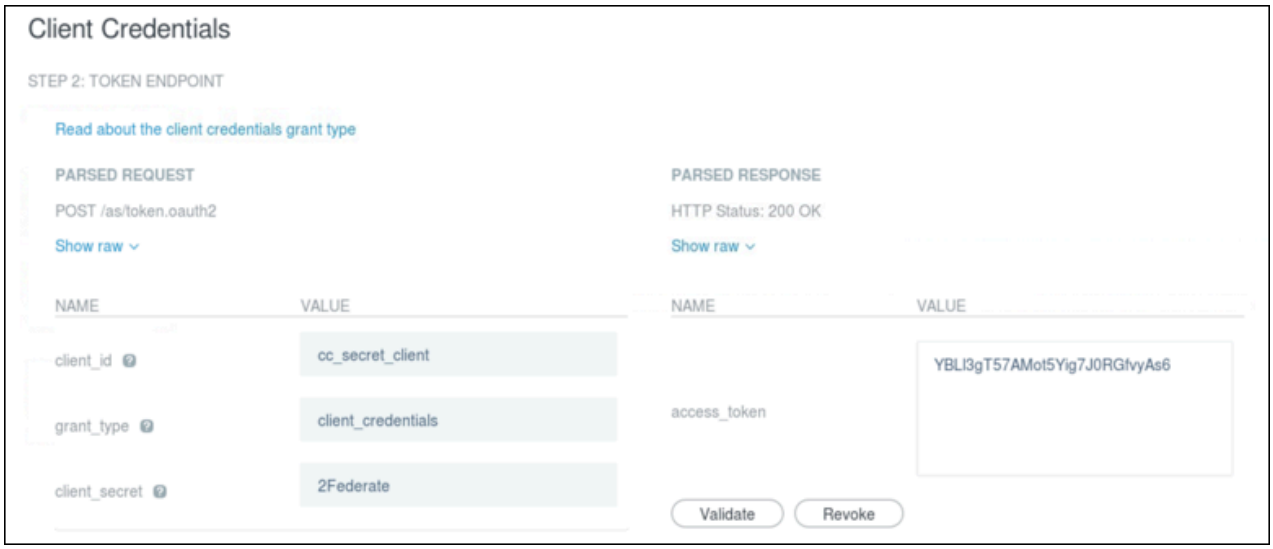


2. In a new tab, click the **OAuth 2 Playground** bookmark.
3. In the PLAY section of the left-hand navigation bar click **Client Credentials**.



4. Click **Submit**.

5. OAuth 2.0 Playground sends the request to PingFederate and will receive an Access Token in return with an HTTP Status: 200 OK, which is summarized on the STEP 2: TOKEN ENDPOINT page.



6. Now to validate the Access Token: Click on **Validate**.
7. Success! We received a 200 HTTP Response and the payload includes the attributes we were expecting to see!

Summary

Congratulations, in this lab you learned how to setup a Resource Server Client in PingFederate, and how to create an Access Token Attribute Mapping specifically for clients requesting an Access Token for the Client Credentials Grant Type.

Lab 11 - Create an OAuth Client for Authorization Code Grant Type

Introduction

In this exercise, you will practice how to define an OAuth Client in PingFederate that uses the Authorization Code Grant Type and test it in OAuth Playground.

This lab should take 20 minutes to complete.

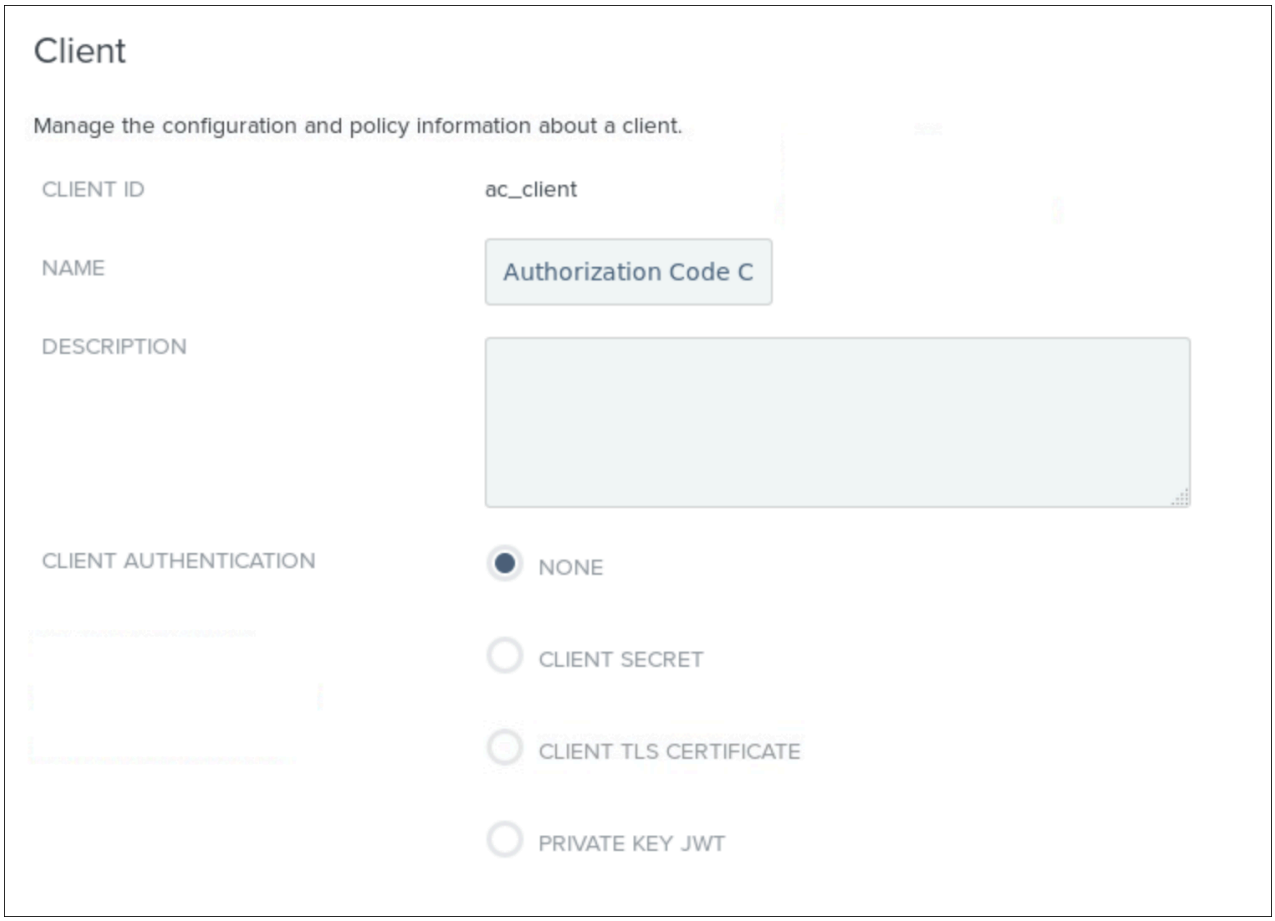
Objectives

- Create OAuth Client using the Authorization Code Grant Type.
- Test a client requesting a specific OAuth Scope.

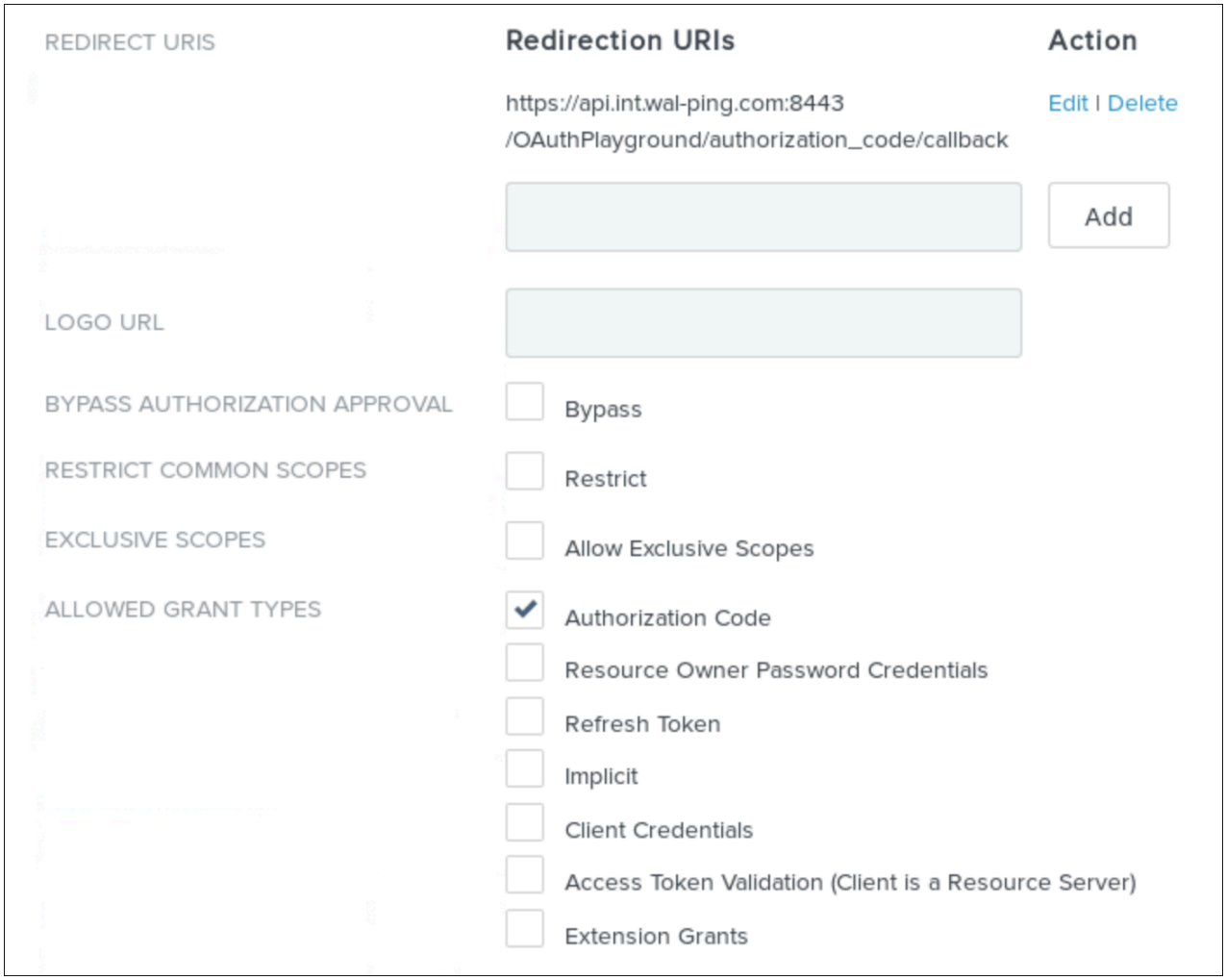
Create the OAuth Client using Authorization Code Grant Type

1. Navigate to the Applications > OAuth > Clients page.
2. Click the Add Client button, an empty Client page will display.

3. Configure the client:
   - Client ID: **ac_client** {case sensitive}
   - Name: **Authorization Code Client**
   - Client Authentication: **None**



   - Redirection URIs: **https://api.int.wal-ping.com:8443/OAuthPlayground/authorization_code/callback** (case sensitive) then click **Add**.
   - Scroll down to ALLOWED GRANT TYPES and select **Authorization Code**



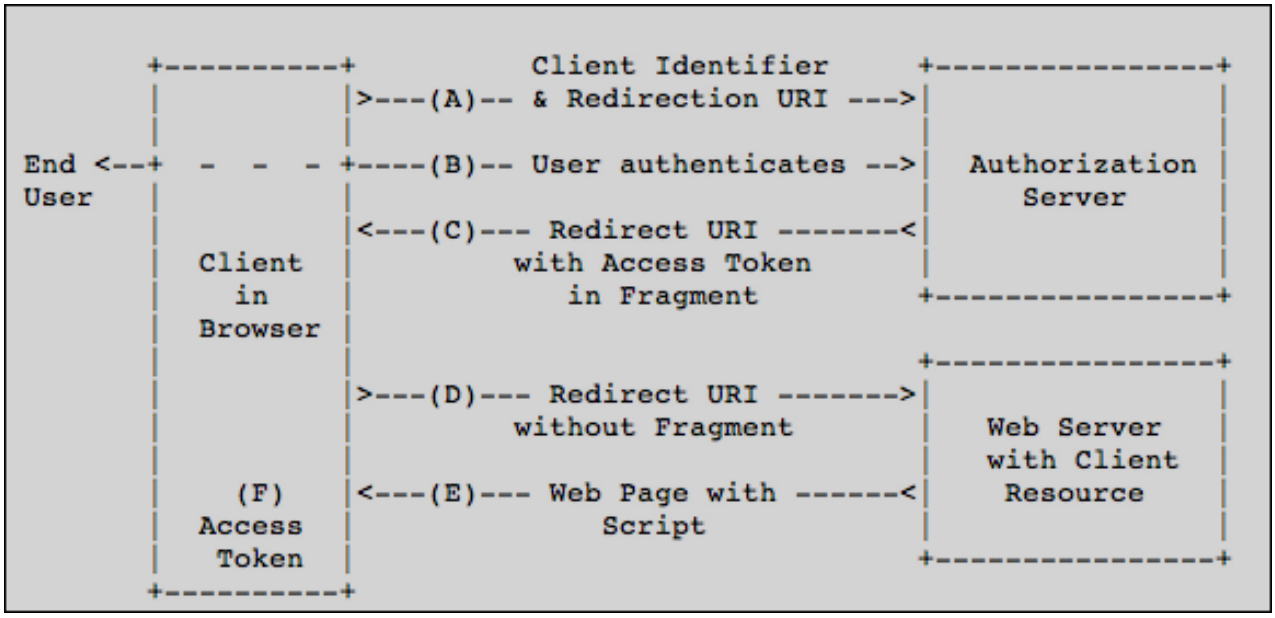
- Scroll down to DEFAULT ACCESS TOKEN MANAGER. Select **General Access Token** from the list.
- Click **Save**.

Test the Authorization Code Work Flow

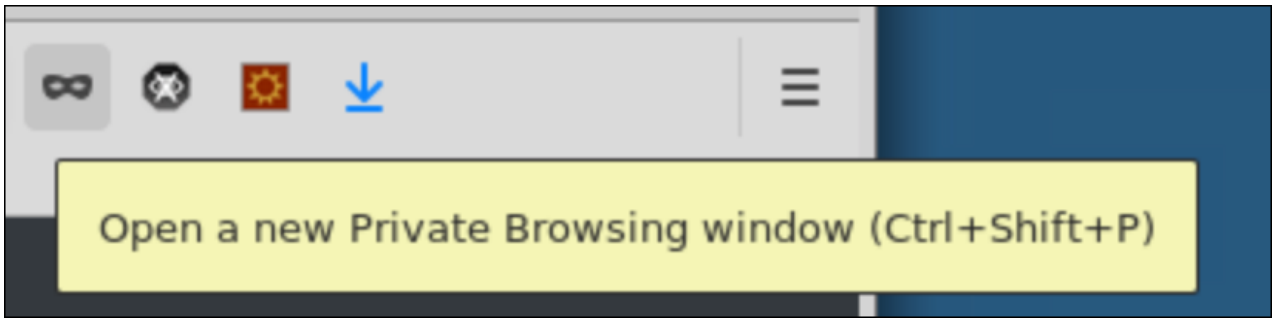
Congratulations! PingFederate is configured with an OAuth Client using the Authorization Code Grant Type. In this section you will complete the exercise by testing your new Client.

An example of this use case is when users are granting access to a 3rd party website to access the user's protected resources stored on another site, such as if you purchase a product off a merchant website and authorize that merchant website to charge your web payment service account for the purchase.

Below is the workflow for this grant type:

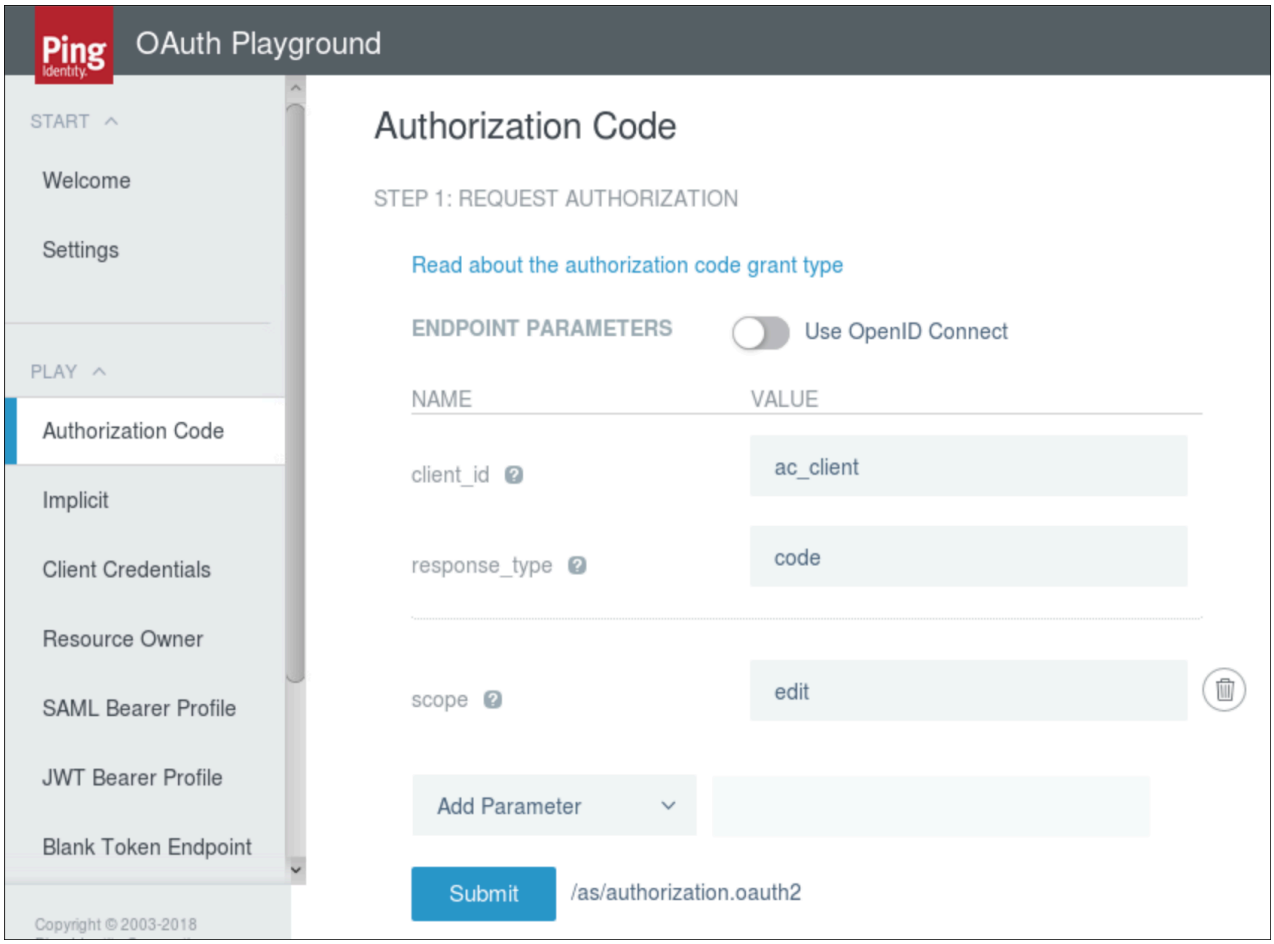


1. Open a new Firefox Private Browsing window.



2. In a new tab, click the **OAuth 2 Playground** bookmark.
3. In the PLAY section of the left-hand navigation bar click **Authorization Code**.
4. Click **Add Parameter** drop-down and choose **Scope** from the list. A new Scope field will be added.

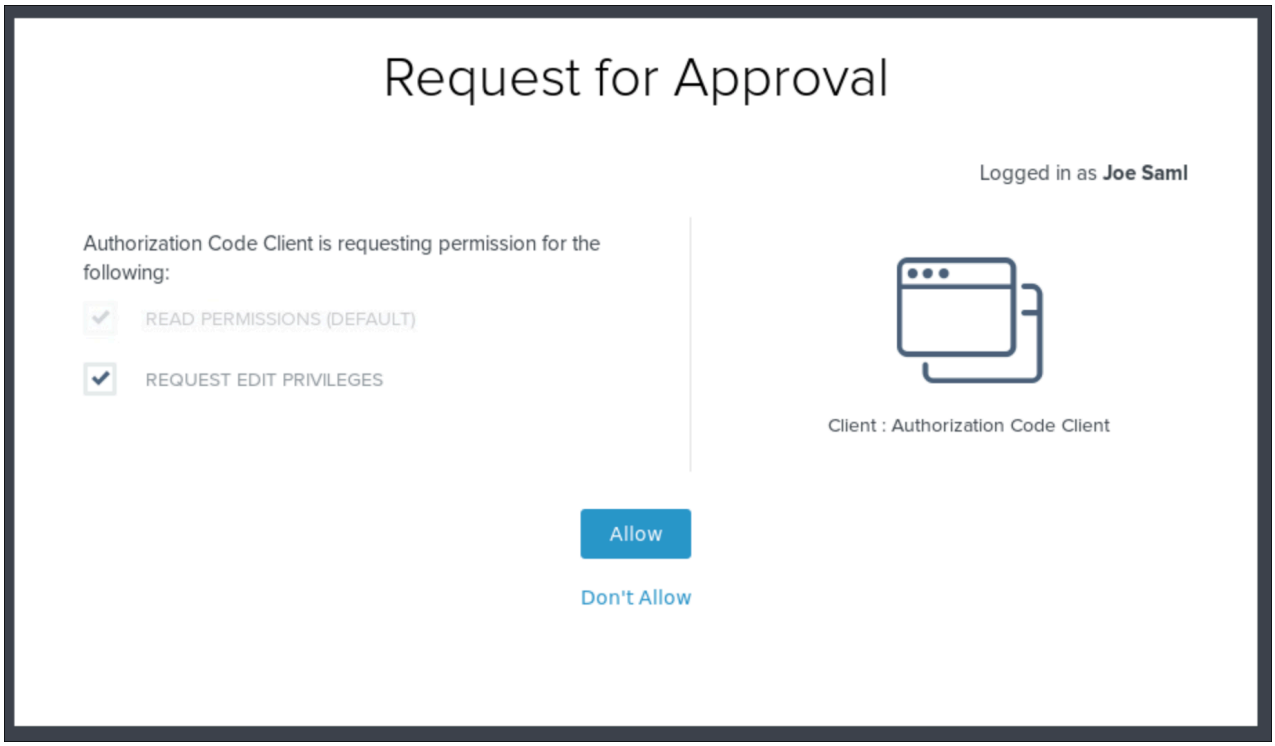
5. Enter **edit** in the scope field. This is the edit scope we defined previously on the Authorization Server.



6. Click **Submit**.
7. Your browser is redirected to the Authorization Server and you are prompted to sign in.

   Note: In Firefox, look at the address bar and you will see `https://sso.int.wal-ping.com:9031/as/authorization.oauth2?client_id=ac_client&response_type=code&scope=edit`. The Request parameters are `client_id=ac_client` which is our Authorization Code Client, and `response_type=code` that is telling the AS that we are using the Authorization Code Grant Type, and `scope=edit` tells the Authorization Server the client is requesting the “edit” scope.

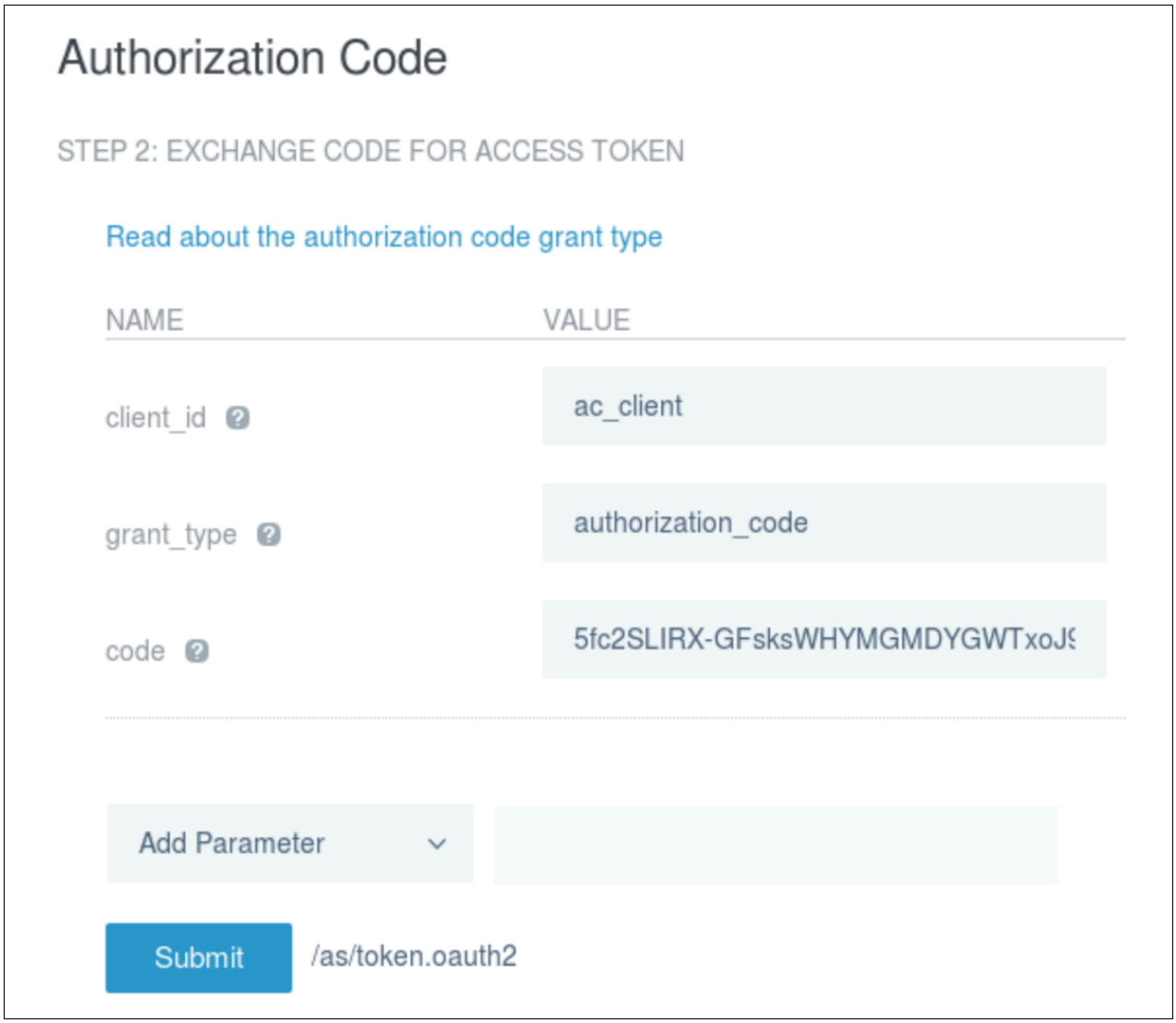
8. Sign On using the credentials of a Resource Owner: **jsaml/2Federate** The Authorization page is displayed.



Note: The client requested the edit scope, so the Authorization Server displays the text description associated with the edit scope: **REQUEST EDIT PRIVILEGES**. The default scope is also selected as mandatory (greyed out).

9. Click **Allow**.

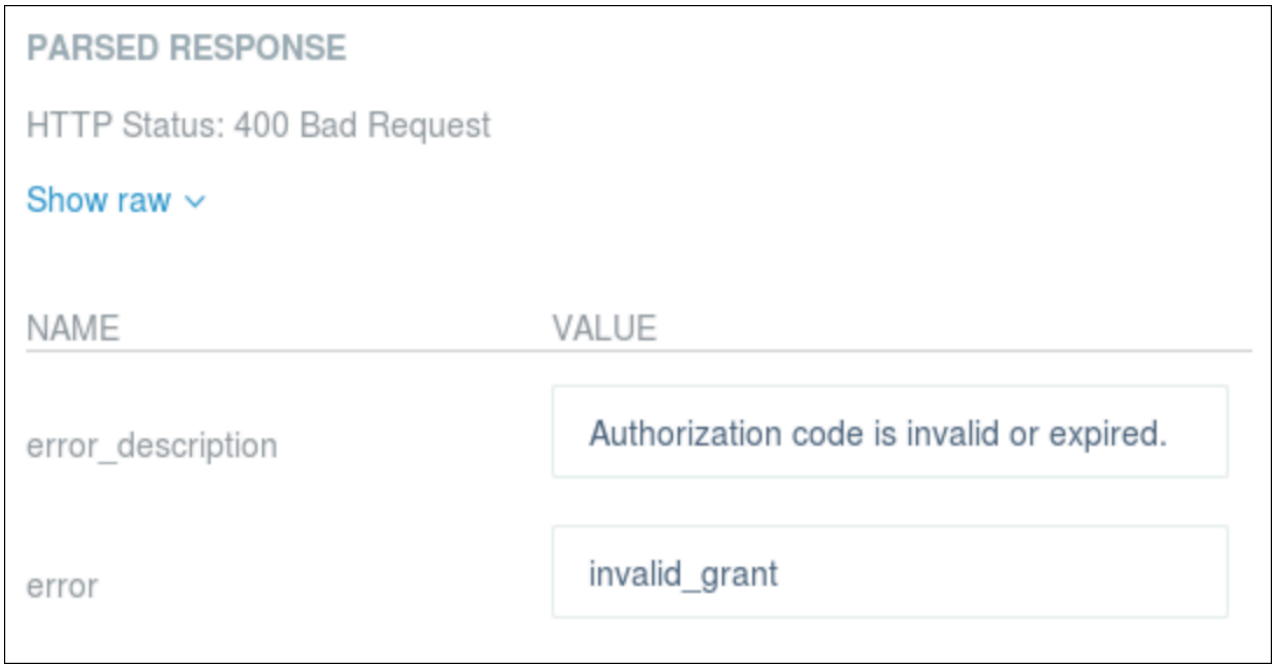
10. The browser is redirected back to Playground using the Redirection URI specified in the PingFederate Client configuration for this client, along with the Authorization Code (not the Access Token.)



11. Now the client must exchange the Authorization Code for an Access Token. Click **Submit**.

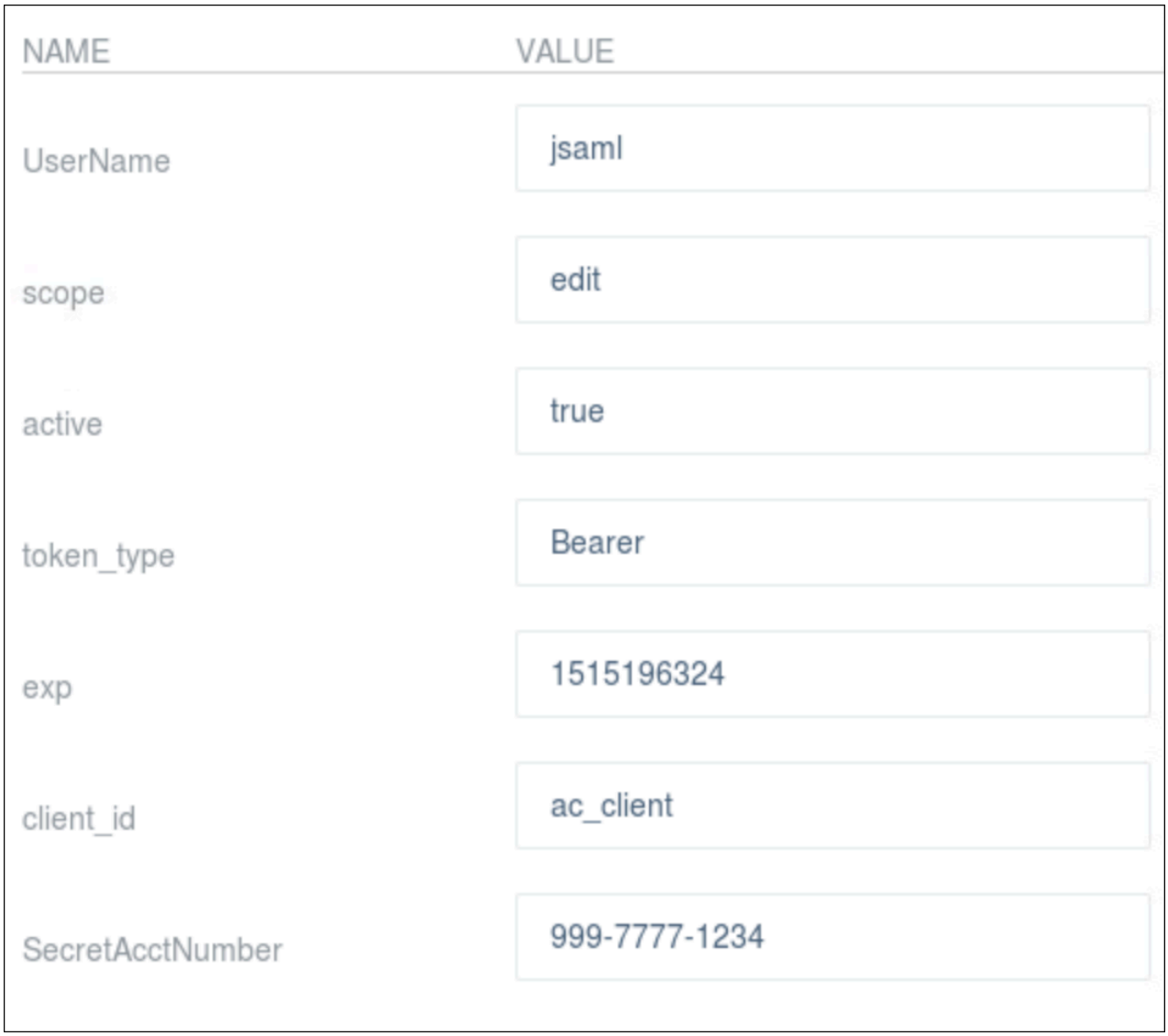
Note: Did you get an HTTP Request: 400 Bad Request status? Remember, Authorization Codes has a short life time, and if you take too long between getting the Auth Code, and using it to get an Access Token, you will see this error: The error_description states the “Authorization code is invalid or expired”,

possibly due to taking more than 60 seconds (the default life time of an Authorization Code) before clicking Submit



12. Now to validate the Access Token click **Validate**.

13. Success! We received a 200 HTTP Response. The payload values are what a Resource Server would see.



NAME	VALUE
UserName	jsaml
scope	edit
active	true
token_type	Bearer
exp	1515196324
client_id	ac_client
SecretAcctNumber	999-7777-1234

Note: Along with the UserName and SecretAcctNumber attributes, notice the Resource Server also gets a scope attribute containing the scope name the client requested, and the Resource Owner authorized.

A Resource Server could use the scope attribute to run some business process or logic that has been associated with the "edit" scope. What that process or logic is would not be PingFederate's responsibility, but a decision for the developers or administrators of the Resource Server itself.

Summary

Congratulations, you have created a client with an Authorization Code Grant Type, and successfully tested in in Playground using a scope you defined in your PingFederate Authorization Server.

Lab 12 - Administrator API In PingFederate

Introduction

PingFederate includes a REST-based Application Programming Interface (API) for administrative functions. The administrative API provides a programmatic way to make configuration changes to PingFederate as an alternative to using the administration console. The configuration changes that you can make through the administrative API include, but are not limited to:

- Server settings
- Connections
- Keys and Certificates
- OAuth settings
- Cluster Management

PingFederate provides an interactive tool for use by both developers and non-developers to explore the API endpoints, view documentation for the API, and experiment with API calls.

This lab should take 20 minutes to complete.

Objectives

- Locate the administrator API
- Use the administrator API to OAuth Clients
- Use the administrator API to change a specific OAuth client

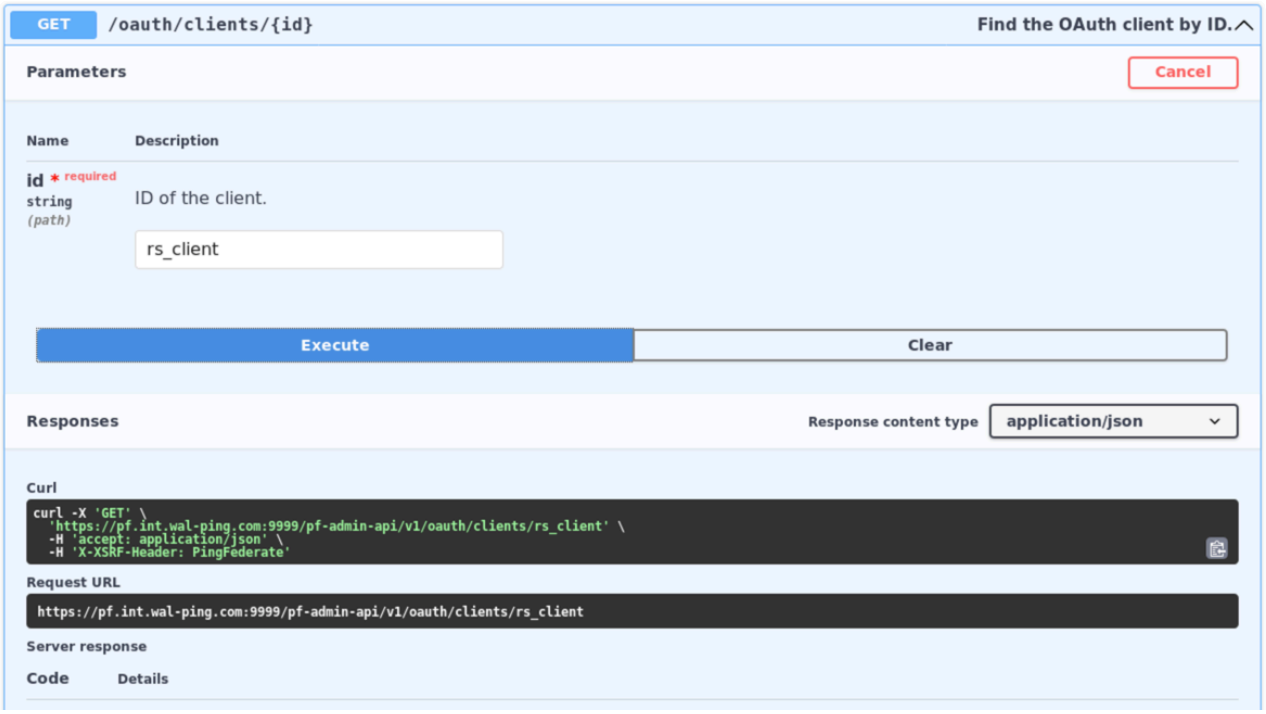
Description of tasks to perform

In this lab, you will use the administrator API to view and change an OAuth client on your PingFederate IdP instance.

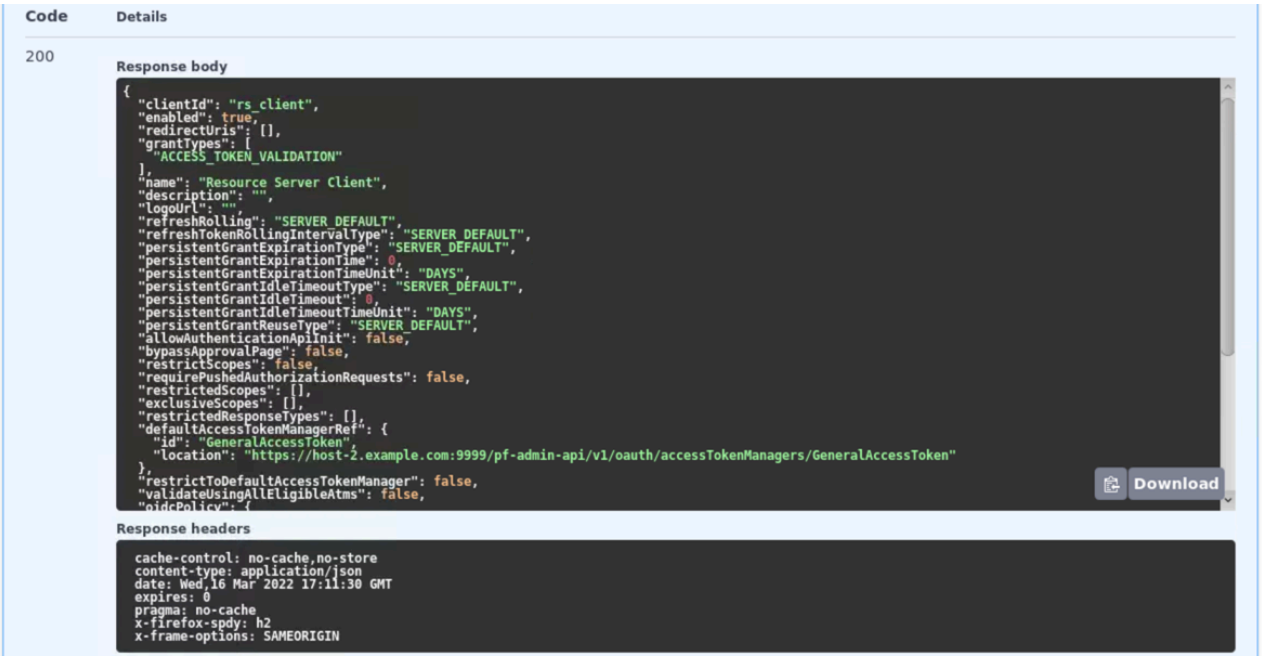
Disable an OAuth Client Using the Admin API

1. Open Firefox by double-clicking the **Firefox** on your desktop.
2. In the bookmark toolbar click **Ping API Docs > PF Admin API Docs**, the **Administrative API Documentation** page will display.
3. Scroll down and click **/oauth/clients**.
4. Click **GET /oauth/clients/{ID}**.
5. Click the **Try it out** button to activate the input fields.
6. In the **id** field enter **rs_client**.

7. Click the **Execute** button, the **Authentication Required** window will display.

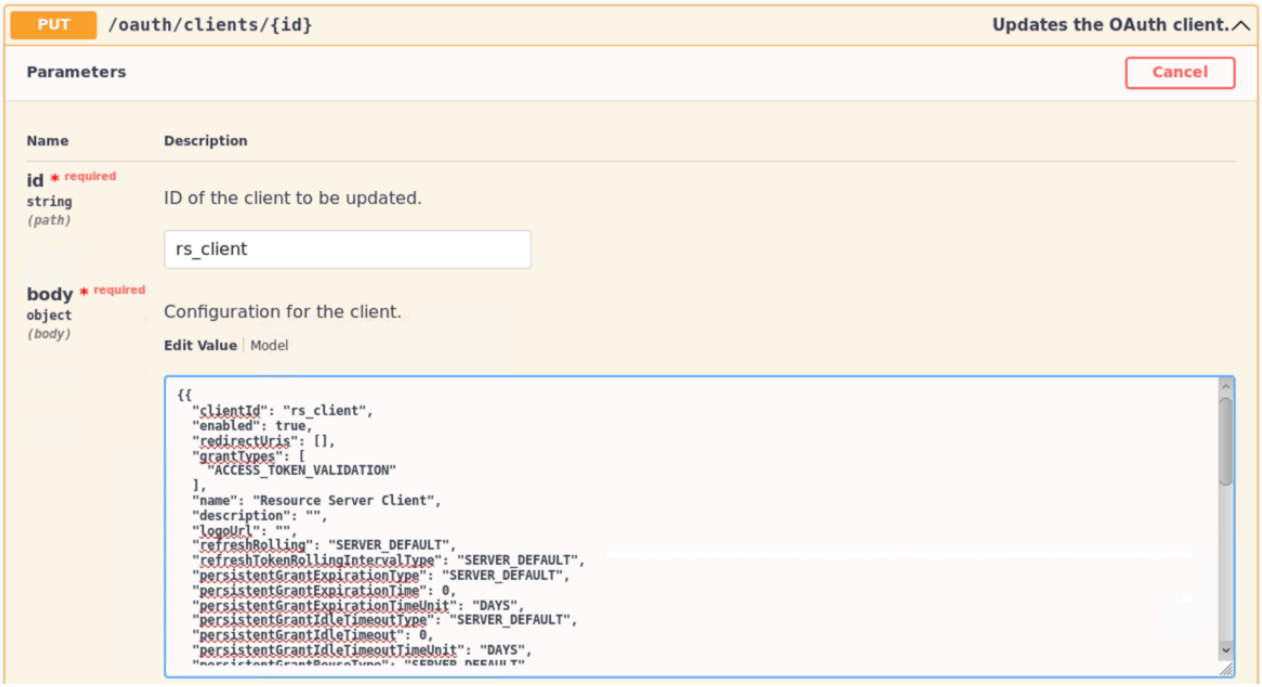


8. Authenticate with the **Administrator/2Federate** credentials. The API returns the settings for the rs_client in JSON format.

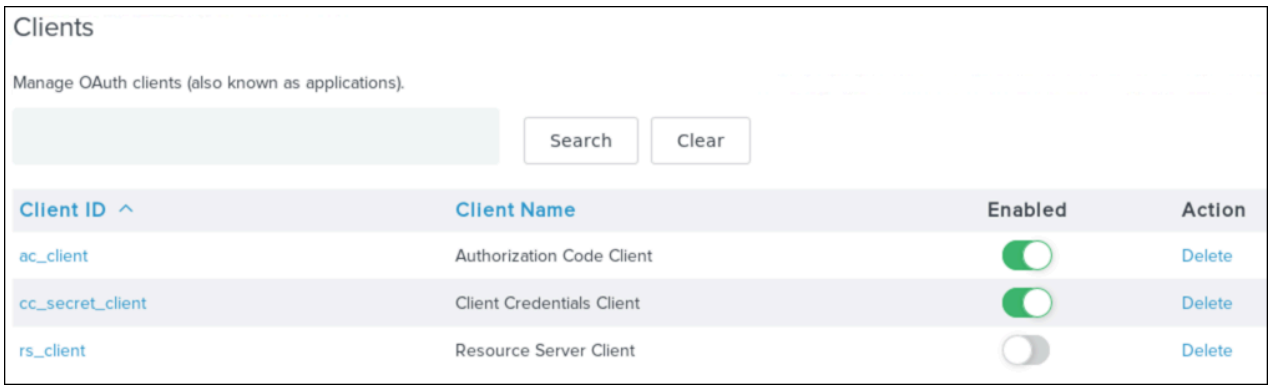


9. Select the entire **Response Body** and copy it to the clipboard. You can use **right-click > Copy** or use the clipboard icon in the bottom right corner of the response field.
10. Scroll down and click **PUT /oauth/clients{id}** to expand that section.
11. Click the **Try it out** button to enable the input fields.
12. Paste all of this into the **body** textbox for **PUT /oauth/clients/{id}**.

13. In the **id** field enter **rs_client**.



14. In the **body** section change `"enabled": true` to `"enabled": false`.
15. Click the **Execute** button.
16. Check the **Response Code** field, if everything was entered correctly you will have a response code of **200**.
17. Log into your PingFederate IdP admin. console and navigate to the **Applications > OAuth > Clients** page. You will see that the Resource Server Client has been disabled.



Summary

Congratulations, you have completed the Administrator API lab! In this lab you have seen how the administrator API can be used to automate tasks outside of the PingFederate admin console.

Lab 13 - Cluster Deployment

Introduction

In this exercise, you change a standalone instance of PingFederate into a clustered deployment.

The Clustered Runtime Engine will consist only of one functional node. Adding additional nodes (for failover and load balancing) would require only basic configuration changes to the new instance of PingFederate.

This lab should take 45 minutes to complete.

Objectives

- Configure PingFederate as a clustered console
- Configure PingFederate as a clustered runtime engine
- Clone an existing PingFederate instance
- Verify cluster functionality

Description of tasks to perform

The steps are as follow:

1. Convert an existing standalone PingFederate instance into a Clustered Console
2. Clone the Clustered Console
3. Reconfigure clone into a Clustered Runtime Engine
4. Verify the configuration of the cluster

Prepare a Clustered Console

In this section you edit run.properties on an existing standalone PingFederate server to configure it as a Clustered Console.

1. On your desktop double-click Terminal, the Mate Terminal window will display.
2. Stop your PingFederate IdP instance: **`systemctl stop pingfed-wal.service`**
3. First, you will verify your private IP address so you know what address the server is listening on. Use the `ifconfig` command and make note of the `inet` parameter for `ens160` in the output:

   Note: The output below has been formatted for this guide. Your output will look slightly different.

   `/sbin/ifconfig`

   ```
   ens160 Link encap:Ethernet  HWaddr 00:50:56:29:0A:9D
    inet addr:192.168.122.1 Bcast:192.168.122.255
    Mask:255.255.255.0
    inet6 addr: fe80::250:56ff:fe29:a9d/64
   ...
   ```

4. You can see the IP address is `192.168.122.1`, this is the IP you will bind your cluster to.
5. From your desktop double-click the Computer icon to open a file system browser.
6. Navigate to the /opt/PingIdentity/pingfederate-12.0-wal/pingfederate/bin directory.
7. Double-click the run.properties file to open it in Pluma.
8. This is going to be your admin console machine. Make the following changes each listed parameter:

   - `pf.operational.mode=`**`CLUSTERED_CONSOLE`**
   - `pf.cluster.node.index=`**`100`**
   - `pf.cluster.auth.pwd=`**`2Federate`**
   - `pf.cluster.bind.address=`**`192.168.122.1`**
   - `pf.cluster.tcp.discovery.initial.hosts=`**`192.168.122.1[7601],192.168.122.1[7600]`**

   Note: Each node in the cluster must have a unique index, so you have set this to 100. The password is optional; all nodes must share the same password so the nodes can authenticate to the cluster. The bind address is the network interface to which the group communication will bind. You set it to the IP address

of the VM, which you discovered before. For the discovery hosts, the runtime and console engine need to find other nodes in the cluster and join using this

9. Click **Save** to save your changes.

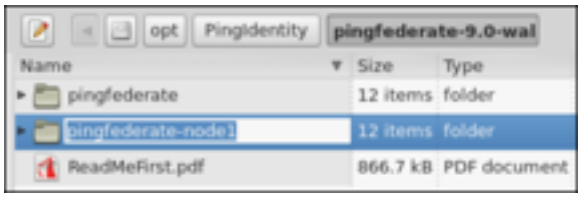
Prepare a Clustered Runtime Engine

In a clustering configuration, you must have at least one runtime engine. In this section, you're cloning the PingFederate IdP installation that you have converted in the previous section and turning it into a Clustered Runtime Engine. The following procedures may be repeated to create additional Clustered Runtime Engines.

Create a Runtime Engine

In this section, you clone the instance of PingFederate which you just configured as a Clustered Console.

1. Return to the file system browser window and navigate to the **/opt/PingIdentity/pingfederate-12.0-wal** directory.
2. Right-click the **pingfederate** folder and click **Copy**.
3. Right-click in the empty white space of the folder and click **Paste**.
4. When the copy has completed change the folder name to **pingfederate-node1**.



Edit run.properties of the Node1 Instance

Now you edit run.properties on the new instance to turn it into a Clustered Runtime Engine.

In the file system browser window navigate to the **/opt/PingIdentity/pingfederate-12.0-wal/pingfederate-node1/bin** directory.

Double-click the **run.properties** file to open it in Pluma.

Edit the following parameters:

- `pf.operational.mode=`**`CLUSTERED_ENGINE`**
- `pf.cluster.node.index=`**`200`**
- `pf.cluster.bind.port=`**`7601`**
- `pf.cluster.failure.detection.bind.port=`**`7701`**

Click **Save** to save your changes.

Close the Pluma window.

Verify Your Configuration

Now that your cluster console and engine have been configured, it is time to start the nodes.

Start the Clustered Console

First start the original node that will run the Administration Console for this cluster.

1. In the Mate Terminal window start PingFederate again **`systemctl start pingfed-wal.service`**.
2. Don't continue until this is started. This may take a minute or two. You can open Firefox and try the IdP Admin bookmark until it loads, then continue. Though this doesn't apply in a real-world environment, in this lab, the Admin Console must be started before the runtime engine.

Start the Clustered Runtime Engine

Next you will start the runtime engine for your cluster. Your cluster will have just one node, which is sufficient for this classroom example. In a typical production environment, however, you would have several runtime engines that would be placed behind a network load balancer.

1. In the Mate Terminal window, change to the bin folder of your node1 install: **`cd /opt/PingIdentity/pingfederate-12.0-wal/pingfederate-node1/bin`**.
2. Enter the following command to start PingFederate server: `# ./run.sh`. After a short time you should see `PingFederate running...`.
3. After approximately 20 seconds, your PingFederate Clustered Runtime Engine node is ready. You can tell when the engine is on because there will be text at (or near) the bottom of the screen informing you that it's started.
4. Minimize the terminal, to leave the server running, and continue with the next section.

   Note: Now you have three instances of PingFederate running: The IdP CLUSTERED_CONSOLE, the IdP CLUSTERED_ENGINE, and the SP. Performance of your training VM will very likely be slow! Even though you can run multiple instances of PingFederate on one server or VM, this is not a recommended production scenario!

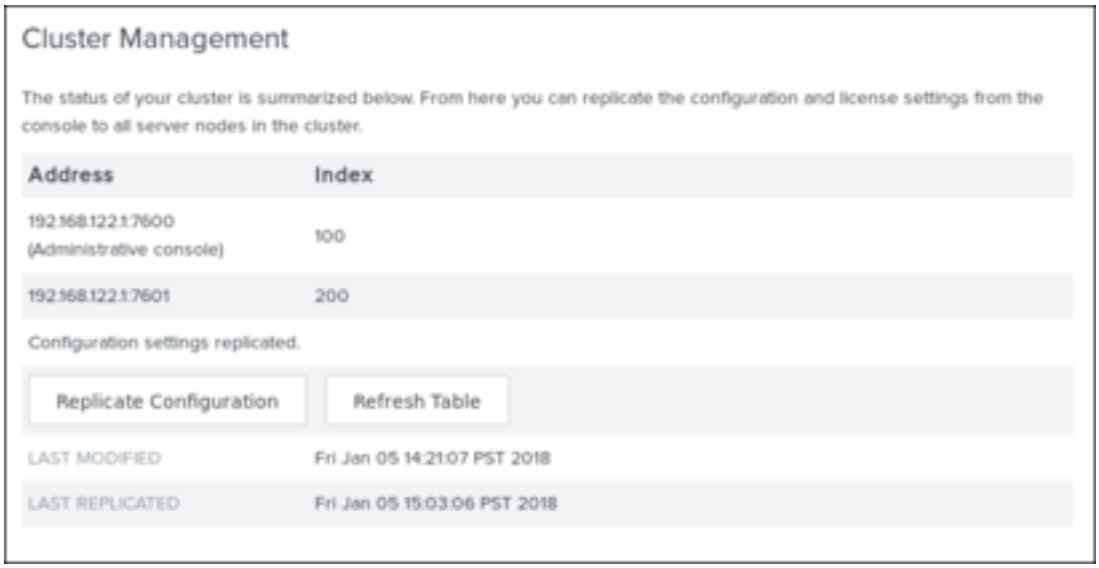
Verify the Clustering Configuration

1. Open Firefox and navigate to the IdP Admin Console.
2. Login using the **administrator**/**2Federate** credentials.
3. Navigate to **System > Server > Cluster Management**. You will see an orange warning message regarding replication of the configuration of the cluster.

   Note: This is a new option, which is now visible after configuring this node as a Clustered Console.

4. Verify you have an **Administrative console** with index **100** and below it a **Runtime Engine** with index **200**.

5. Click the Replicate Configuration button to force an initial replication. This will remove the warning messages you see at the top of the screen.



Test SSO on the Clustered Runtime Engine

Click the bookmark SSO to SP > Selector - LDAP in the bookmark toolbar.

Authenticate using the smanager/2Federate credentials.

If SSO works, then success! You can verify this as well by looking at the node1's server.log.

Open server.log file for the node1 instance.

At the bottom of the log, you can see the last transaction was this PingFederate engine creating and sending the assertion.

Close the server.log.

Now you will reset everything so you can easily continue with future labs.

Stop the engine console by going to the Terminal window where it's running and pressing ctrl-c. This will take a moment. It's stopped when the prompt reappears.

Then, stop the admin console by entering the command **`systemctl stop pingfed-wal.service`**.

Open the run.properties file for your PingFederate console in the /opt/PingIdentity/pingfederate-12.0-wal/pingfederate/bin directory

Find the `pf.operational.mode=CLUSTERED_CONSOLE` line, and change it back to `pf.operational.mode=`**`STANDALONE`**.

Click Save.

Start PingFederate again by going back to the Terminal prompt and entering **`systemctl start pingfed-wal.service`**.

Check that it's reset by opening Firefox and clearing any SSO cookies, then clicking the SSO to SP > Selector - LDAP bookmark and verifying it works (it may take a few moments for PF to turn back on).

Challenge: You can look at the server logs for the runtime and console installs and see that only the runtime node saw the request for SSO and received the assertion.

Summary

Congratulations, you have finished the PingFederate clustering lab. In this lab you learned how to clone an existing PingFederate instance and convert it into a cluster.