A picture containing background pattern

Description automatically generated

**Shape, arrow

Description automatically generated**A picture containing background pattern

Description automatically generated

**Semarchy External SSO Integration**

**Semarchy setup with Okta SSO and AWS Secrets Manager.**

**9th September 2022**

www.amplifiuk.com

+44 (0) 1926 911820

Third Floor, Marlborough House

48 Holly Walk, Leamington Spa

CV32 4XP, United Kingdom

Contents

[1 Introduction 3](#_Toc114042306)

[2 Okta Setup 3](#_Toc114042307)

[2.1 Create an application mapping 3](#_Toc114042308)

[2.2 Create group(s) for the roles 5](#_Toc114042309)

[2.3 Update user profile configuration 7](#_Toc114042310)

[3 AWS Setup 8](#_Toc114042311)

[3.1 Create a user with access to Secrets Manager 8](#_Toc114042312)

[3.2 Create the secrets. 9](#_Toc114042313)

[4 Semarchy Setup 10](#_Toc114042314)

[4.1 Okta Configuration 10](#_Toc114042315)

[Creating an Identity Provider 10](#_Toc114042316)

[Map Okta Groups to Semarchy Roles 12](#_Toc114042317)

[4.2 AWS Configuration 13](#_Toc114042318)

# Introduction

This document describes the set up required to integrate Semarchy xDM with external providers of both Identity Management for Single Sign On (SSO) and Secrets Management. Semarchy provides its own internal mechanisms for user authentication and “secure” storage of secrets but can be configured to use external providers which provide greater degree of security and conform to various industry standard protocols.

The document describes the particular use of Okta for SSO and AWS Secrets Manager for storage of Okta connection secrets.

# Okta Setup

Since Okta is security enforcing for client users, it is therefore the responsibility of the client to set up Okta to their own specific needs. The following is a simple example that demonstrates some of the basics that could be set up to support a basic scenario of using Okta for multi-factor authenticated access to Semarchy.

## Create an application mapping

Use the *Create App Integration* button to create an application mapping that represents Semarchy and which provides the connection details that will allow Semarchy Identity Management configuration to connect to Okta. Ensure that the *OIDC - OpenID Connect* sign-in method and the *Web Application* application type options are selected during application creation.

Graphical user interface, application

Description automatically generated

The Application Sign-on behaviour/rules that will be used to activate the required level of Okta authentication, such as multi-factor authentication using Okta Verify, needs to be configured. Note the *Group Claims Filter* which specifies the “semarchy\_user” group as the claim, the *Client Credentials* and *Client Secrets* in the *Open ID Connect ID Token* section (the values of which will be stored in AWS Secrets Management and retrieved form there by Semarchy xDM *Identity Management*), and the *Sign On Policy* “Semarchy SSO Rule” that specifies multi-factor authentication be used. This policy may or may not be required for the specific application, depending on the required use case.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

## Create group(s) for the roles

A group or groups will need adding to Okta which will be used to represent “roles” that can be mapped in Semarchy. The Okta users which require access to Semarchy will need to be added to these groups. In the example below, the group name is “semarchy\_users”.

Graphical user interface, application

Description automatically generated

The group has a single user assigned in this example.

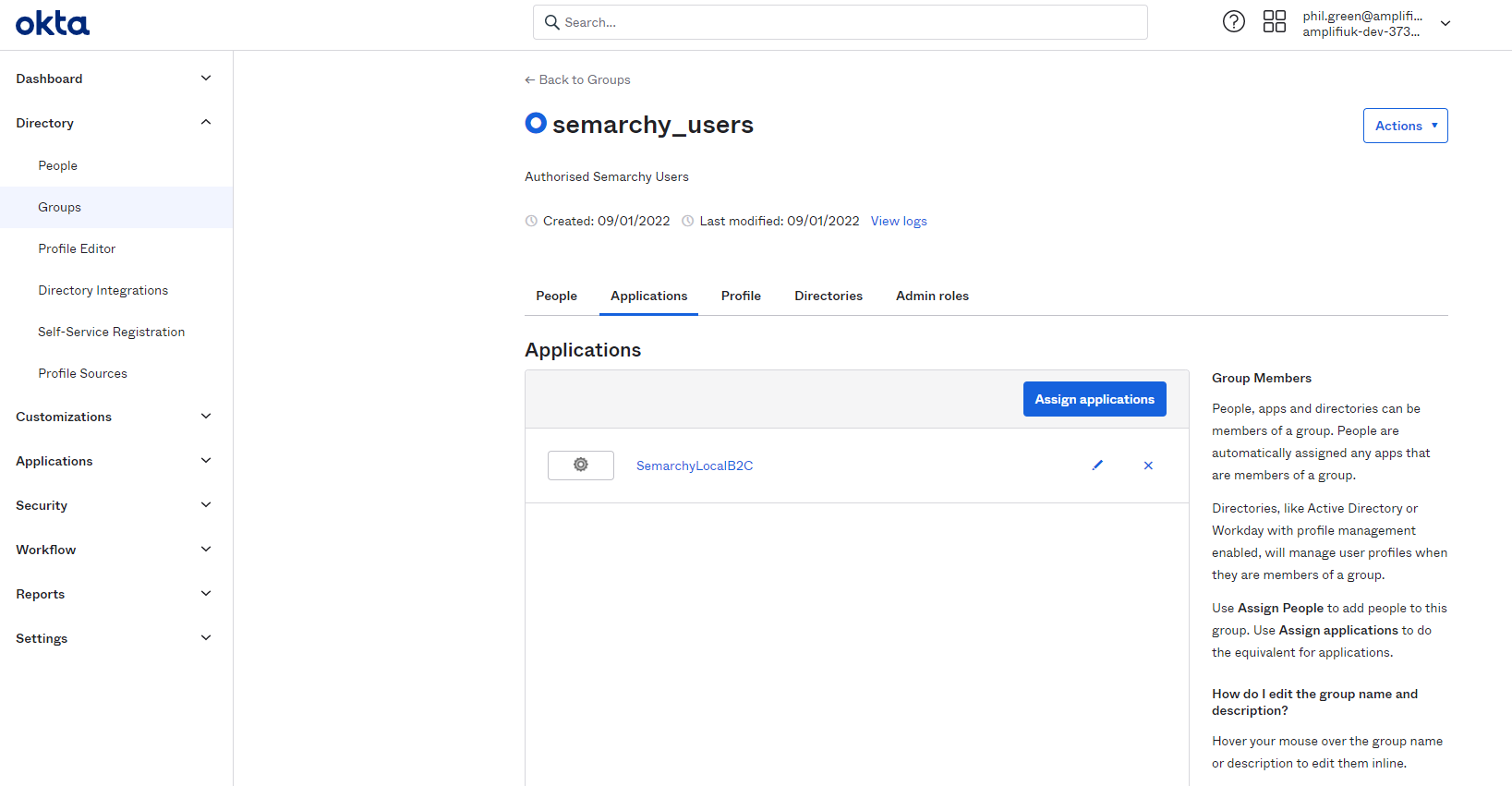
Graphical user interface, text, application

Description automatically generated

The above user was added via the *People* page in Okta. All users that are going to require access to Semarchy with SSO via Okta will need to be added to Okta and assigned to an Okta application mapping for Semarchy, via Okta group membership.

Graphical user interface, application

Description automatically generated



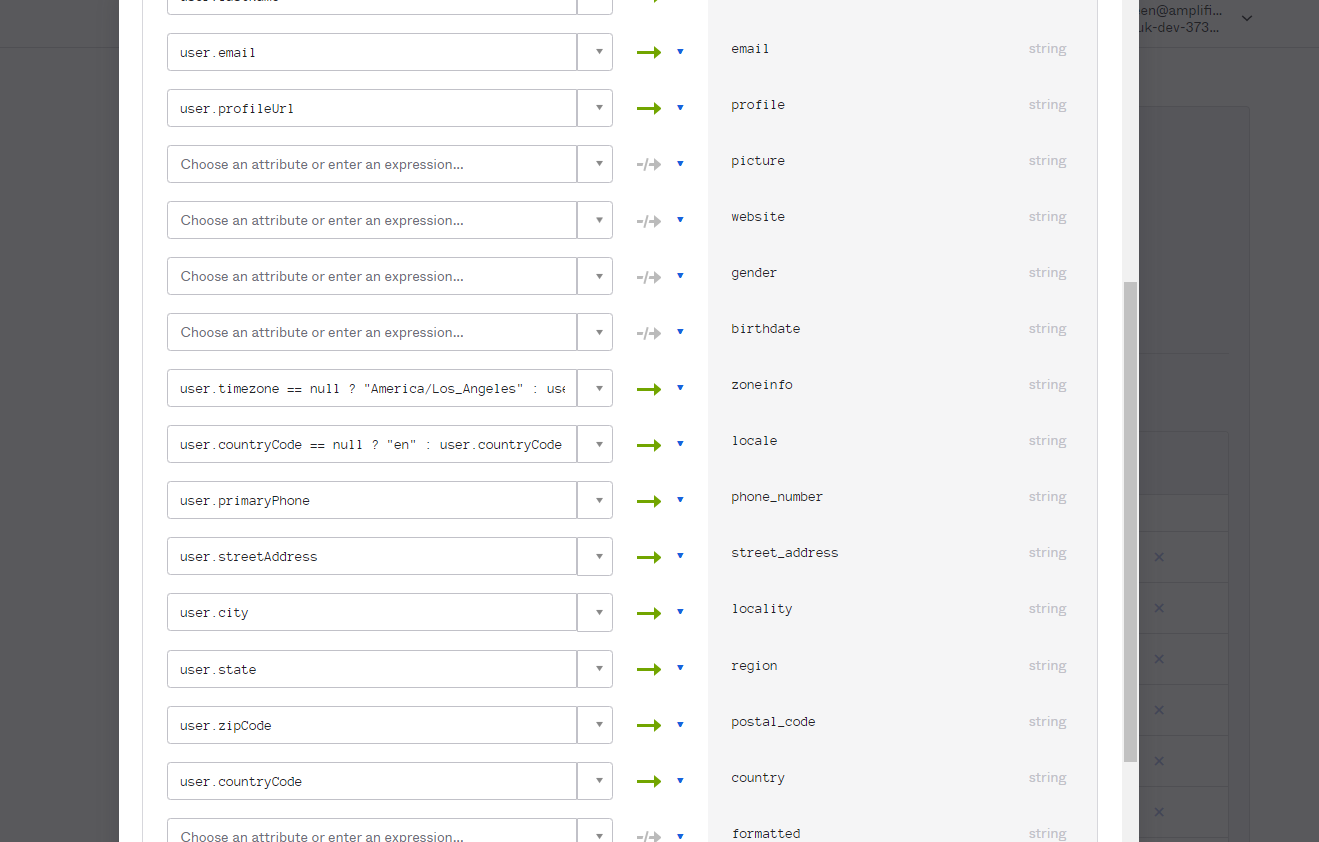
## Update user profile configuration

It may be necessary to edit the user profile mappings in the application to make sure compatible values are passed between Okta and Semarchy. This is done by clicking the “pencil” symbol (i.e., edit) for the application that represents your Semarchy deployment.

Graphical user interface, application

Description automatically generated

By default, the locale is setup in Okta as “en-US”. This does not seem to map correctly to Semarchy’s locale, and an error is generated when testing the SSO connection. Mapping simply to the language code alone i.e., “en” solves this issue. This mapping rule makes sure “en” is used, if not otherwise supplied by the Okta user account. This rule is set up in the *Profile Editor* for the application user under *Directory*, via the *Mappings* button.



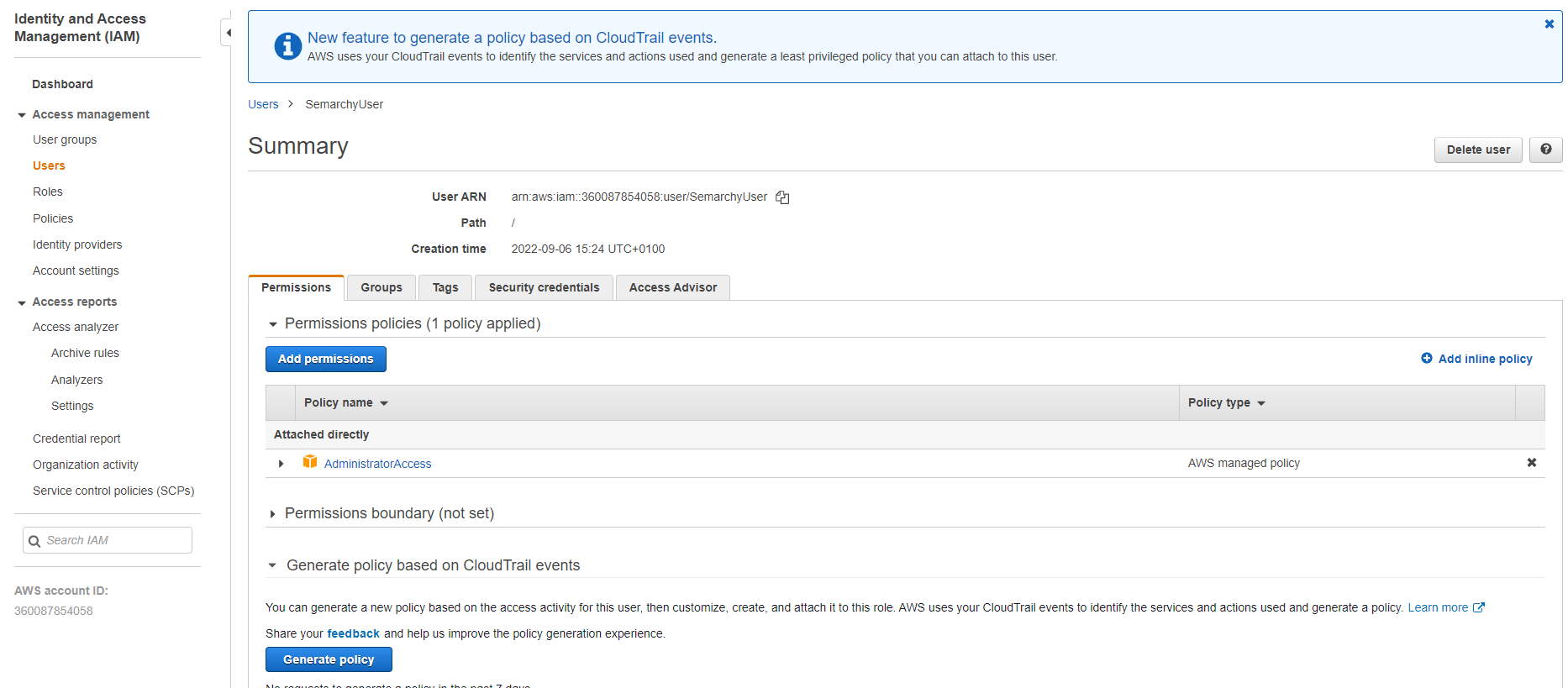


# AWS Setup

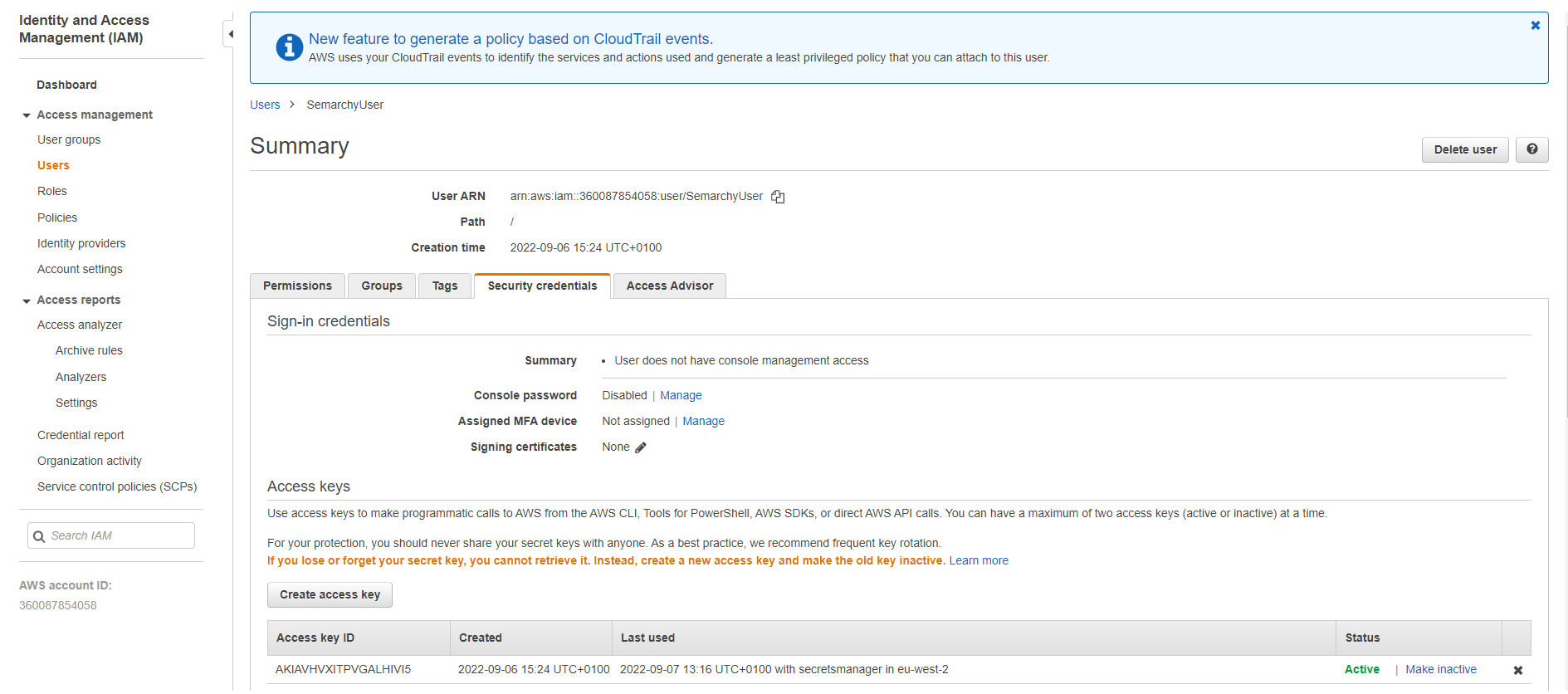
An AWS account must be available, in a tier where Secrets Manager is an available service. At present this includes the free tiers, which has been used successfully to prove the integration with Semarchy xDM. Secrets Manager provides the option to store individual, discrete secret key-value pairs or use JSON to store multiple key-value pairs as a single secret. Since the secrets we want to store are all related to Okta connectivity, this example uses the latter (JSON).

## Create a user with access to Secrets Manager

This user will be the account with which we access AWS Secrets Manager from Semarchy xDM.



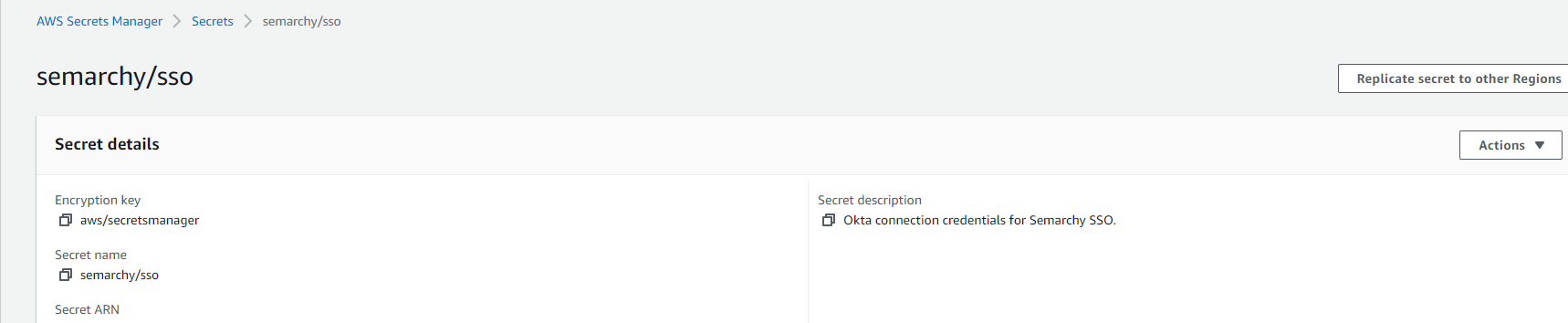
The security credentials section contains the Access Keys (as shown below) that will be used to set up Semarchy xDM via its config.properties file, described in [AWS Configuration](#_AWS_Configuration) later in this document.





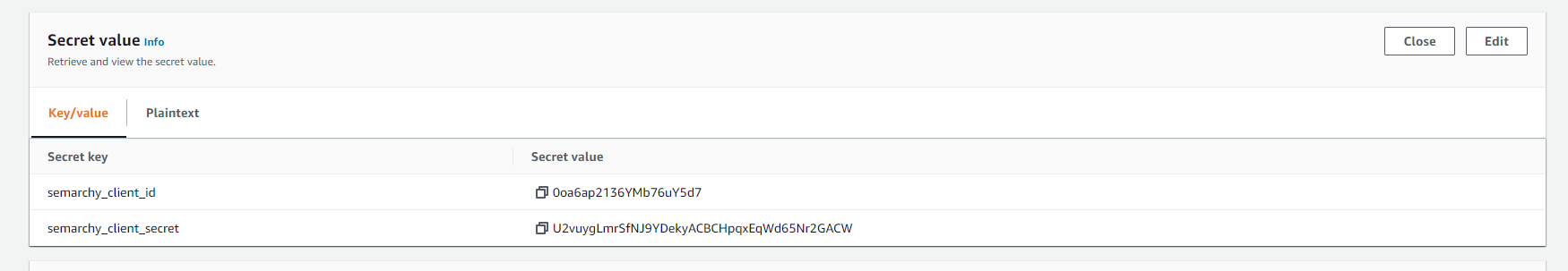
## Create the secrets.

In AWS Secrets Manager, create a secret that represents the two secret values that represent the Client ID and Client Secret used to connect to your Okta instance.



In the above example, the secret name is “semarchy/sso”. This is the value used to name the secret in the “Secret containing” field when creating the secret in Semarchy xDM.

The values stored in the secret can be a single entry or in this example’s case, it is the two related Okta connection values we want to protect as a secret.





As can be seen above, there are two key-value pairs stored in the “semarchy/sso” secret, one for Client ID and one for the Client Secret (defined by Okta’s application set up). When accessing from Semarchy xDM, the same secret will be used to reference each actual key-value pair as required.

# Semarchy Setup

## Okta Configuration

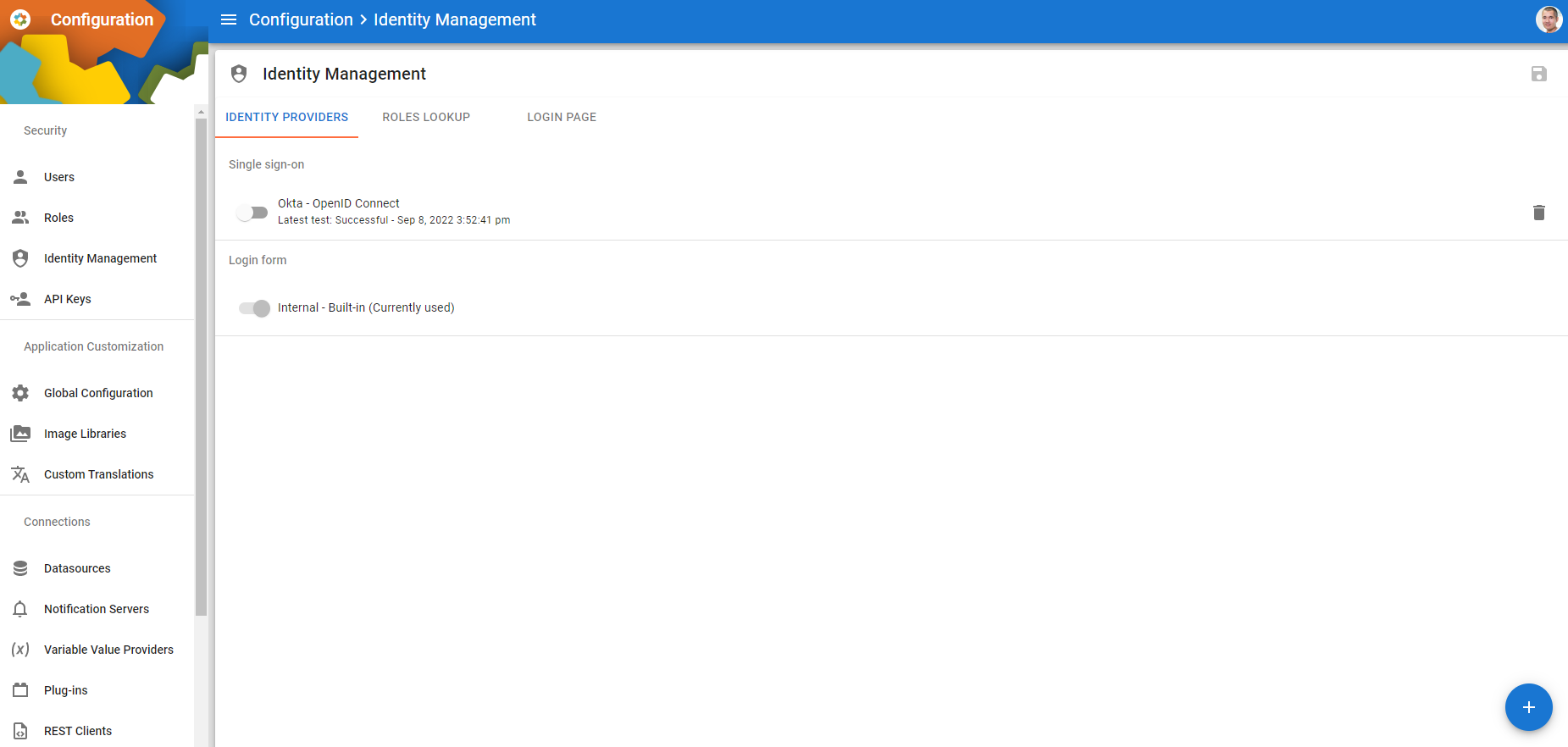
Although it has its own internal authentication mechanism, Semarchy can integrate with external identity providers (IDPs) using various protocols:

* OpenID Connect
* SAML
* LDAP
* Active Directory
* Windows Authentication

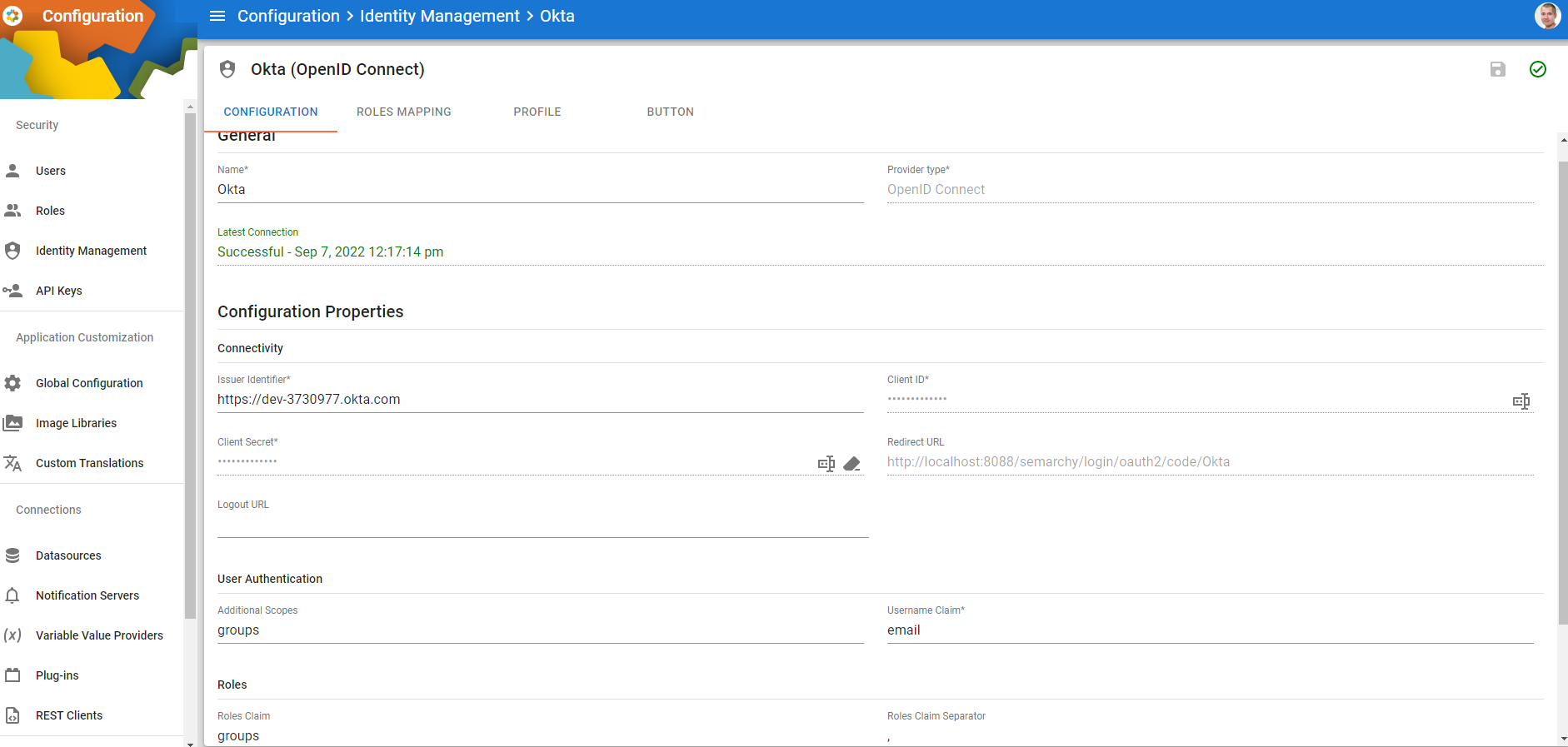
Okta uses OpenID Connect and so this is the protocol we need to set up in Semarchy. In the Semarchy xDM administration console, set up of external IDPs is carried via the Identity Management option, by doing the following:

### Creating an Identity Provider

In Identity Management, click on the + button to add a new Identity Provider (in this screenshot we have already created one called “Okta – Open ID Connect”)



Whether creating a new Identity Provider or editing an existing one, the following Identity Provider creation page allows configuration of the connection to the external IDP.



Under *Configuration Properties* in the *Configuration* tab:

* ***Issuer Identifier*** will be the Okta URL representing your Okta account, in this example’s case, the account is “dev-3730977.okta.com”.
* ***Redirect URL*** is the URL on your deployment of Semarchy xDM to which Okta will redirect to/return to once the user has authenticated with Okta SSO. In this example, Semarchy is running on a local PC, so the hostname is “localhost:8088”. This should be the only section of the URL to change, unless your Semarchy instance has a different name (depending upon your deployment environment).
* ***Logout URL*** is the URL back to the Identity Provider (Okta) to log the user out of both Semarchy and Okta. In this example this has not been set up so only local Semarchy logout will occur.
* ***Client ID*** will be the ID of the application as set up in Okta itself but using the path to the secret held in AWS Secrets Manger, where the actual Client ID value is held. This is shown in the following section [AWS Configuration](#_AWS_Configuration).
* ***Client Secret*** will be the secret application key set up in Okta itself but using the path to the secret held in AWS Secrets Manger, where the actual Client Secret value is held. This is shown in the following section [AWS Configuration](#_AWS_Configuration).
* ***Additional Scopes*** defines extra scopes that may be requested from the IDP (in addition to the default *openid*, *email*, *profile* scopes always requested), which define different data to return. Such a scope is “groups” which returns from the IDP’s endpoint with the groups to which a user is assigned (in the example, this will be “semarchy\_users”).
* ***Roles*** defines the claim that will contain the “roles” for the user authenticated by Okta, as set up within Okta. In this example, it is the “groups” claim in the IDP’s response that contains the “roles”, which happen to be the groups themselves!

### Map Okta Groups to Semarchy Roles

In *Identity Management* under the *Role Mappings* tab, we can map incoming claims to actual Semarchy roles, to give externally governed users access to Semarchy xDM.

So far only “group” claims have been successfully passed from Okta and mapped in Semarchy. In the example below, the “semarchy\_users” group defined earlier in Okta has been mapped to both *semarchyConnect* (a role required to allow a user to connect to Semarchy xDM) and *semarchyAdmin* (a role required to enable administrator rights to this Semarchy xDM instance). If the response from the IDP (Okta in this case) contains a role or group that matches an entry here, it is mapped to the Semarchy role. New mappings may be added simply by clicking “Add Role Mapping”. Note that the *Provider Role* can be a role or a group!

Many more mappings may be added depending upon the requirements of the specific application.

So far, we have not been able to get individual roles passed from Okta to Semarchy via Open ID Connect payloads, only group claims (as requested during the above configuration).

Graphical user interface, text, application, email

Description automatically generated

## AWS Configuration

AWS Secrets Manager integration is enabled with changes to the config.properties file in the Semarchy applications “config” directory, for example “C:\Program Files\Semarchy\mdm-server\conf”. This may vary depending on the nature of the deployment (on-prem or cloud such as Azure, AWS, etc.).

Add the following to the properties in the config.properties file:

*xdm.secrets.external.****corporatesecretsmanager****.type=AWS*

*xdm.secrets.external.****corporatesecretsmanager****.allowed=false*

*xdm.secrets.external.****corporatesecretsmanager****.allowed.idm=true*

*xdm.secrets.external.****corporatesecretsmanager****.versionstage=AWSCURRENT*

*xdm.secrets.external.****corporatesecretsmanager****.region=eu-west-2*

*xdm.secrets.external.****corporatesecretsmanager****.credentials.basic.accesskey=[place access key here]*

*xdm.secrets.external.****corporatesecretsmanager****.credentials.basic.secretkey=[place secret key here]*

Replace the placeholders with your own values for the AWS user account that has access to AWS Secrets Manager.

These will enable AWS Secrets Manager as an External Secrets Manager whose secrets can then be referenced from with the Semarchy xDM Identity Management functionality, once the config.properties has been edited, saved and Apache restarted.

Note that “corporatesecretsmanager” highlighted in the above configuration file entries can be whatever you want it to be (provided it confirms to Semarchy’s naming convention – see their documentation for details) and is the name of the Secrets Manager as it will appear in Semarchy xDM when configuring Client ID and Client Secret in the Semarchy xDM Identity Management functionality.

In the following example, the Secure With Secrets Manager has selected “corporatesecretsmanager” as the reference to the external secrets manager defined in config.properties (i.e., AWS Secrets Manager).

The secret is the name of the secret within AWS that contains the value of interest. In this case the secret is called “semarchy/sso”. This was defined earlier in AWS Secrets Manager as a JSON formatted key-value pair structure, with “semarchy\_client\_id” as the tag containing the actual secret value of the Okta Client ID.

There is another secret stored in the same JSON structure called “semarchy\_client\_secret” which is set up/referenced in an identical manner.

*Graphical user interface, application

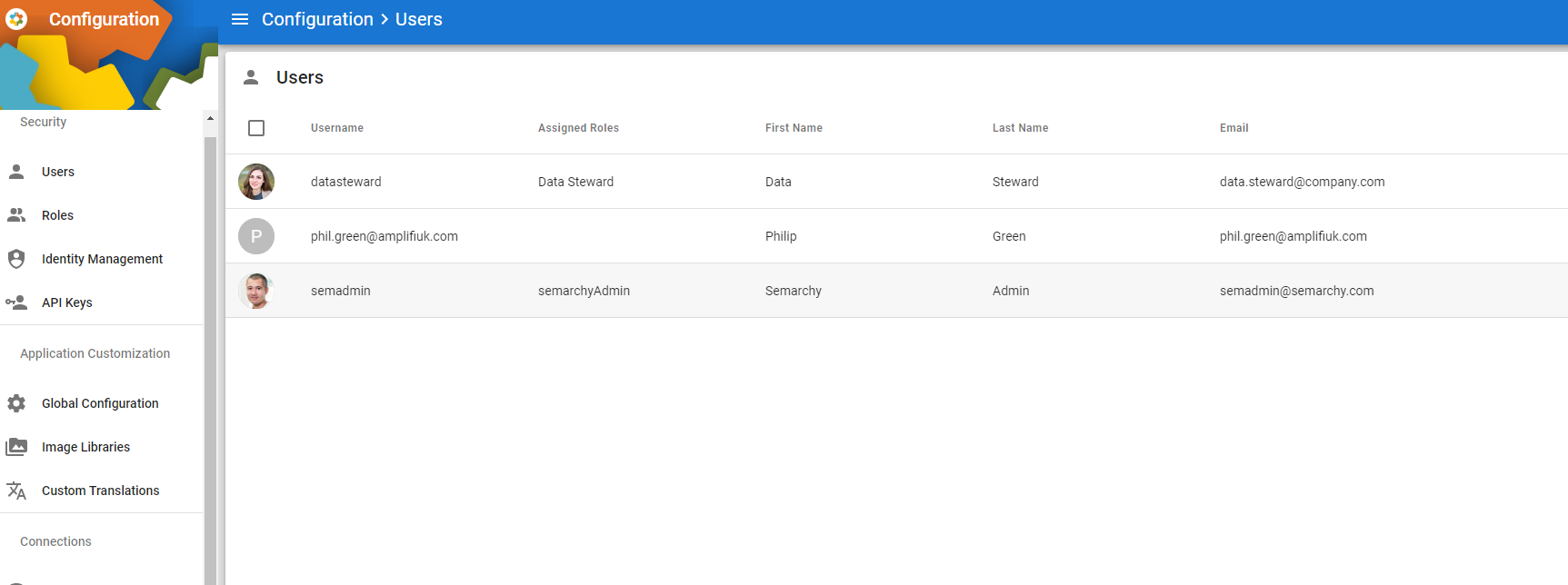
Description automatically generated*

Once the Identity Manager has been set up, it can be tested by clicking the “Test this identity provider” tick in the top right-hand corner of the screen.

This creates a small sidebar with a link to a URL which will exercise the connection. Be sure to follow the instructions and open the link using a private browser window, to make sure the current logon session is not used. The test should invoke Okta’s login mechanism and you should see the Okta SSO dialog window. Enter the details as requested following the instructions. This may include entry of a code from the Okta Verify app for multi-factor authentication, if that was configured earlier for this application within Okta.

If successful, the test sidebar in Semarchy’s Identity Manger displays the response from Okta, containing all the details received from Okta, including the group membership, and shows the mappings to Semarchy roles. A successful test will also attempt to login to Semarchy using the role mappings. If these do not confer sufficient access rights, you may see connection errors or minimal functionality in Semarchy xDM’s GUI.

Following a successful test, Semarchy automatically creates the externally authenticated user as a Semarchy user, with the username defined by the *Username Claim* field. In the example this is the email address of the user defined in Okta. The screenshot below shows that no permanent roles are set up for the highlighted user as these will always be the transient result of the mappings between Okta and Semarchy following successful external authentication with Okta.





There is no user image/photograph associated with this user, unlike the other two locally defined users shown above. It should be possible for an image to be supplied by the Identity Provider and passed to Semarchy as a field mapped to the *Avatar* property using the *Profile* tab within Identity Management, however this has not been tested yet. The *Profile* tab allows many properties to be mapped to incoming values from the external IDP and synchronised, if required, to whatever is supplied. These values should be returned by the external IDP as part of the default “profile” scope.

Once a successful test has been carried out and you are happy with the mappings of the authenticated users to roles, enable the *Identity Provider* so that it becomes active for future logins.