SECURING MULTI-VENDOR CLOUDS

LABS 1 - Deploying Azure AD for single sign-on to an individual AWS account

ABSTRACT

This lab document contains a step-by-step guide to configuring Azure AD as the identity provider for an AWS Account. It will provide attendees with instructions on how to deploy Azure AD for single sign-on to an individual AWS account.

Luciana Blanchard

Written for "Securing Multi-vendor Clouds" a series of events for **Microsoft Partners** created by Luciana Blanchard.
Luciana.blanchard@microsoft.com

Contents

1. Getting started

Azure AD supports single sign-on integration with AWS SSO. With AWS SSO you can connect Azure AD to AWS in one place and centrally govern access across hundreds of accounts and AWS SSO integrated applications. This enables seamless Azure AD sign-in experience for users to use the AWS Console.

The following Microsoft security solution procedure implements SSO for the example roles **AWS Administrators** and **AWS Developers**. Repeat this process for any additional roles you need.

This lab covers the following steps:

1. Create a new Azure AD enterprise application.
2. Configure Azure AD SSO for AWS.
3. Enable Azure AD to provision AWS IAM roles.
4. Update role mapping.
5. Test Azure AD SSO into AWS Management Console.

2. Pre-requisites

To get started, you need the following items:

- An Azure AD subscription. If you don't have a subscription, you can get a free account.
- An AWS single sign-on (SSO) enabled subscription.

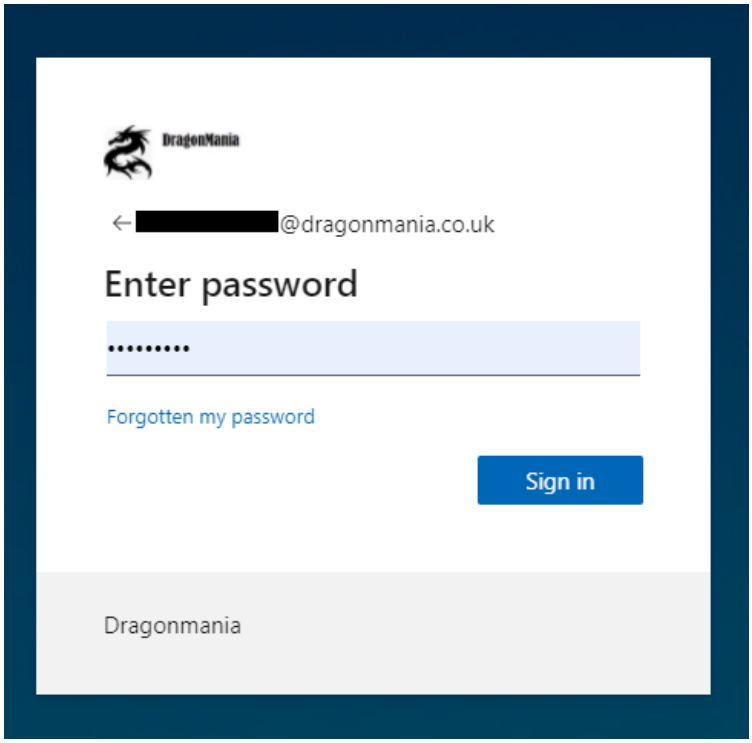
If you haven't done so yet, please read the "Part 0 - Getting Started - Lab Guide - Securing Multi-vendor Clouds" document. It will go through the steps required to setup the pre-requisites listed above.

3. Add AWS Single-Account Access from the gallery

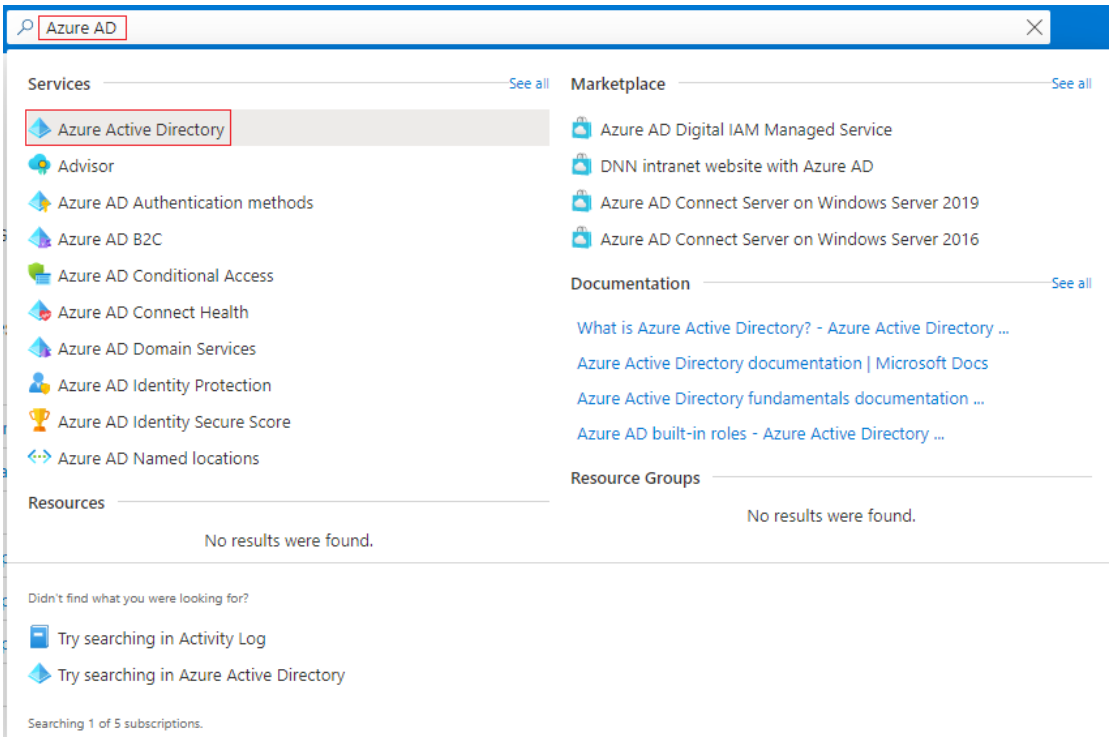
AWS administrators and developers use an enterprise application to sign in to Azure AD for authentication, then redirect to AWS for authorization and access to AWS resources. The simplest method to see the application is by signing in to https://myapps.microsoft.com.

To configure the integration of AWS Single-Account Access into Azure AD, you need to add AWS Single-Account Access from the gallery to your list of managed SaaS apps.

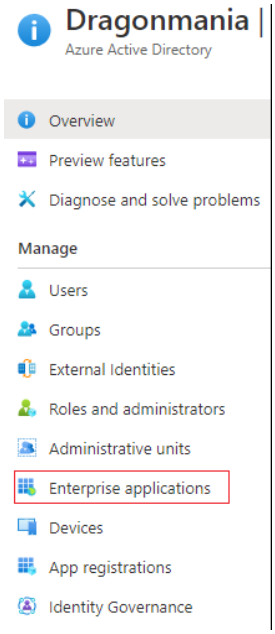
3.1 Sign in to the Azure portal using the admin username and password obtained during the creation of the tenant in "Part 0 - Getting Started - Lab Guide - Securing Multi-vendor Clouds", "How to create a new M365 Demo Tenant" section.



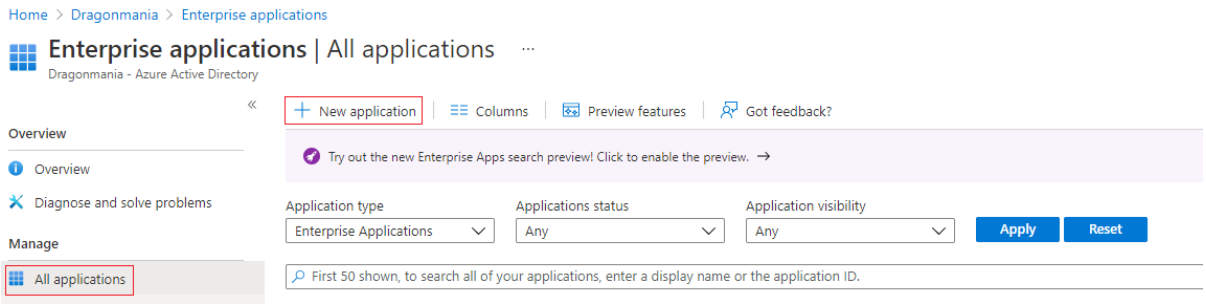
3.2 In the **Azure portal**, search for and select **Azure Active Directory**.



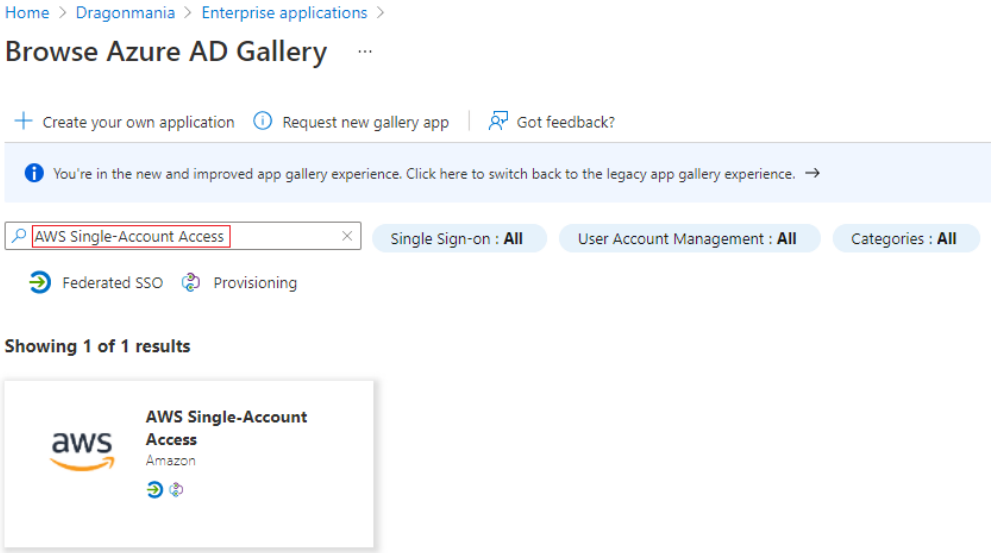
3.3 Within the **Azure Active Directory** overview menu, choose **Enterprise Applications**.



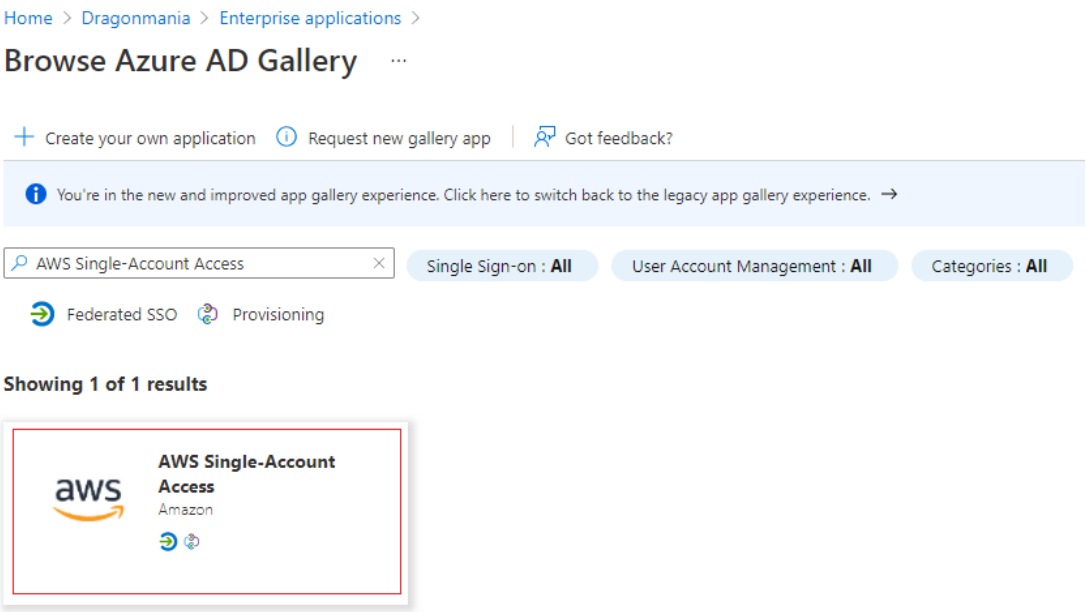
3.4 Select **New application** to add an application.



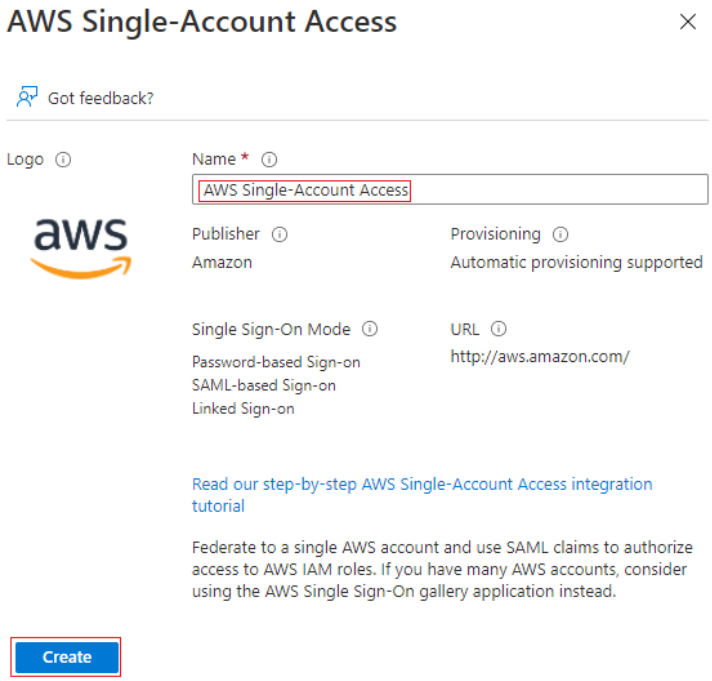
3.5 In the **Browse Azure AD Gallery** section, type **AWS Single-Account Access** in the search box.



3.6 Select **AWS Single-Account Access** from results panel.



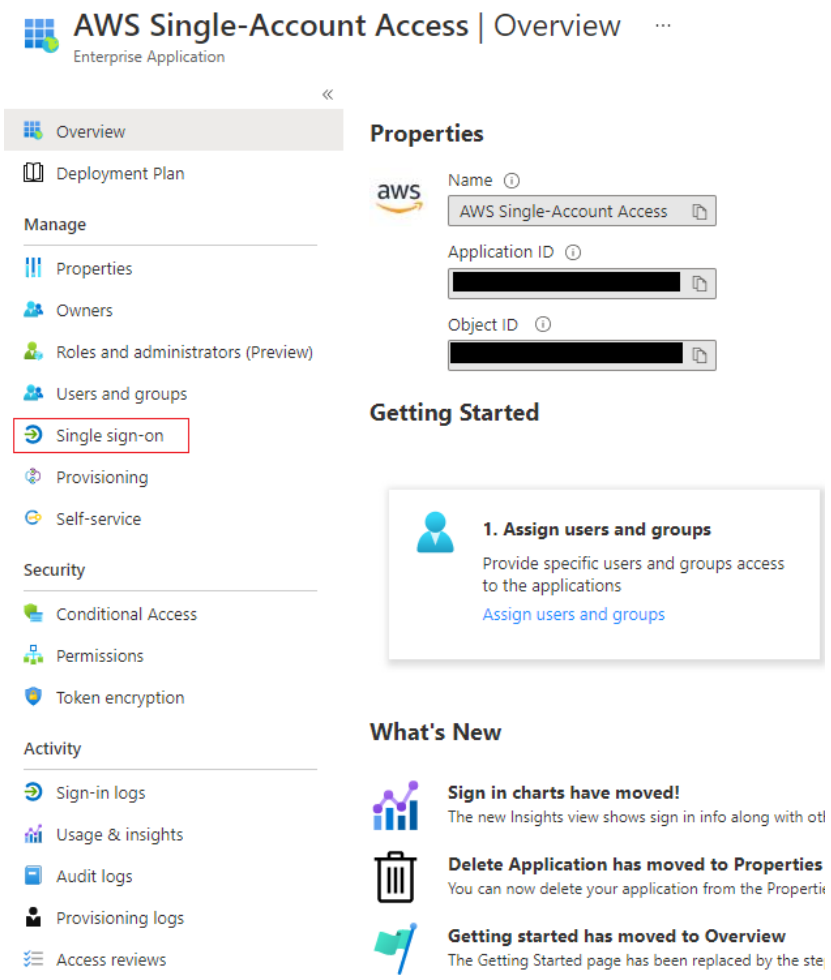
3.7 Leave the default name as is or give it a name that you can easily identity, then click **Create**.



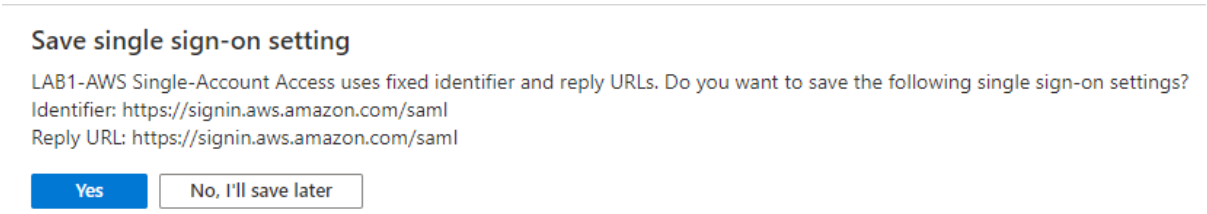
4. Configure Azure AD SSO

Follow these steps to enable Azure AD SSO in the Azure portal.

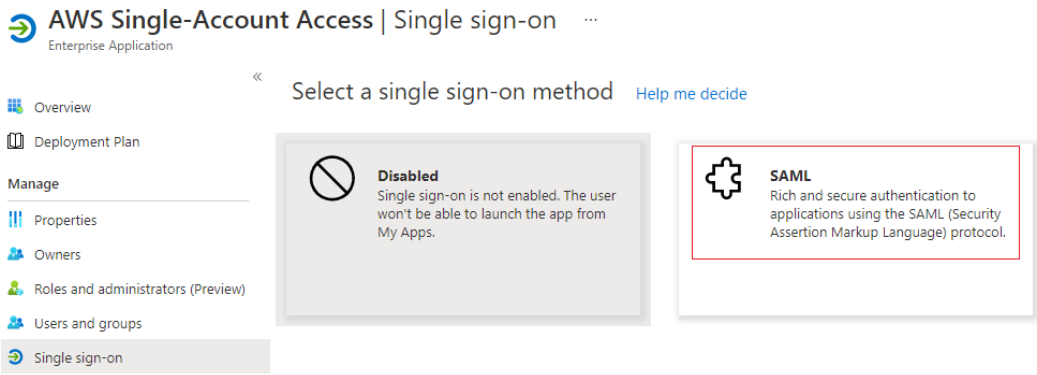
4.1 In the Azure portal, on the **AWS Single-Account Access** application integration page (the application you have created in the steps above), find the **Manage** section and select **single sign-on**.



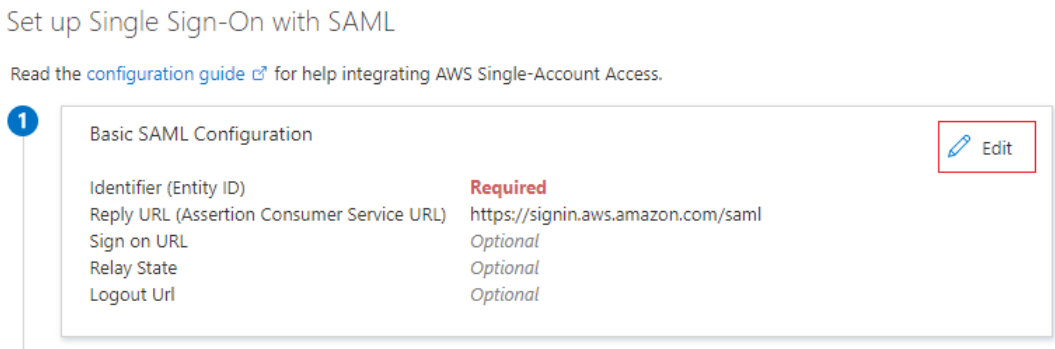
*If prompted with the dialog box below, select **Yes** to save the fixed settings and skip to section 4.12. Otherwise, follow the steps below.



4.2 On the Select a single sign-on method page, select SAML.

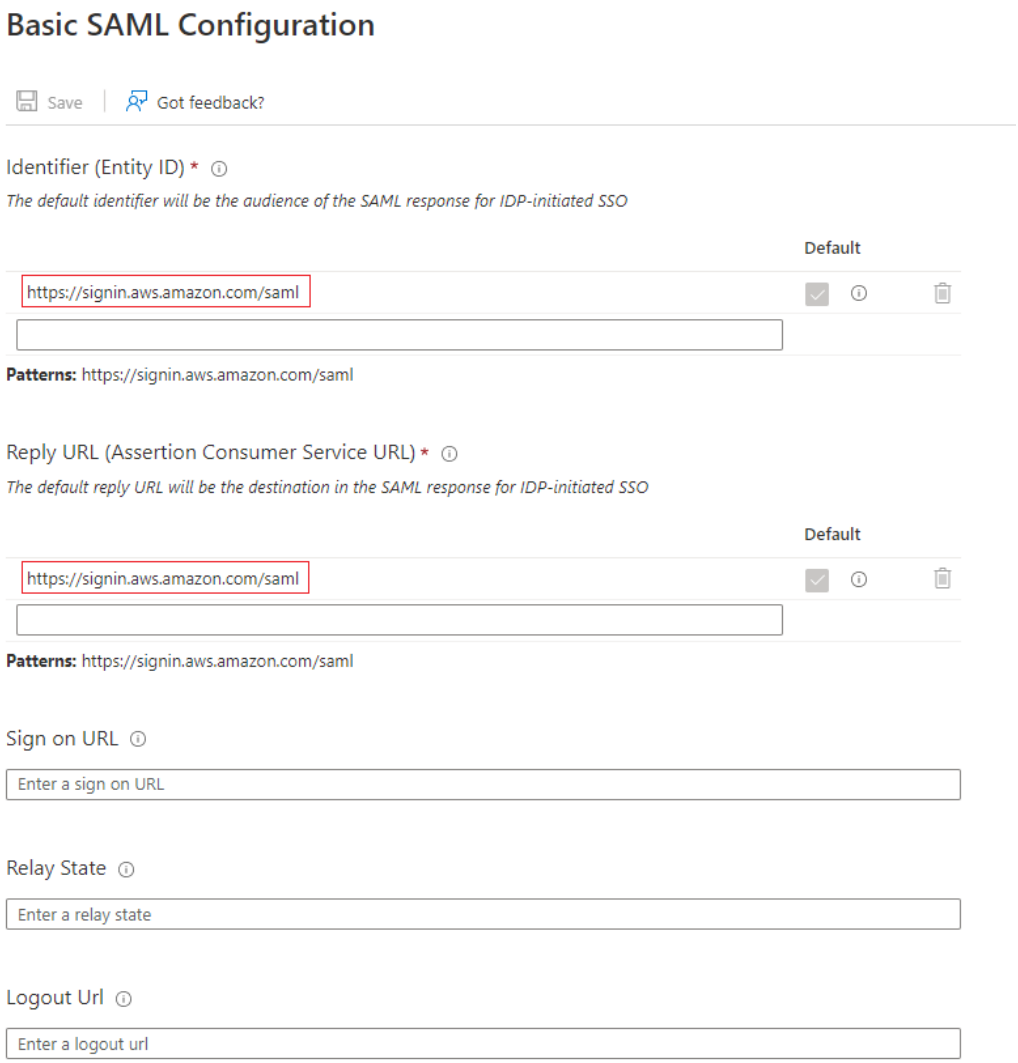


4.3 On the Set up single sign-on with SAML page, click the edit/pen icon for Basic SAML Configuration to edit the settings.

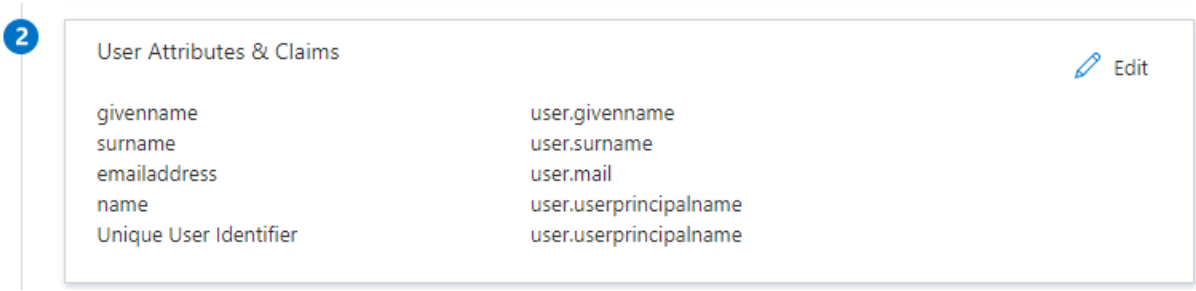


4.4 In the **Basic SAML Configuration** section, update both **Identifier** (Entity ID) and **Reply URL** with the same default value: https://signin.aws.amazon.com/saml. You must select Save to save the configuration changes.

* When you are configuring more than one instance (for example, you have multiple AWS accounts you want to integrate with Azure AD), provide an identifier value. From second instance onwards, use the following format, including a **#** sign to specify a unique SPN value. For example: **https://signin.aws.amazon.com/saml#2**.



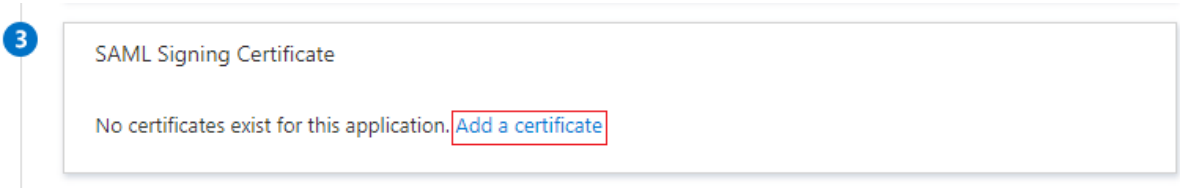
4.5 AWS application expects the SAML assertions in a specific format, which requires you to add custom attribute mappings to your SAML token attributes configuration. The following screenshot shows the list of default attributes.



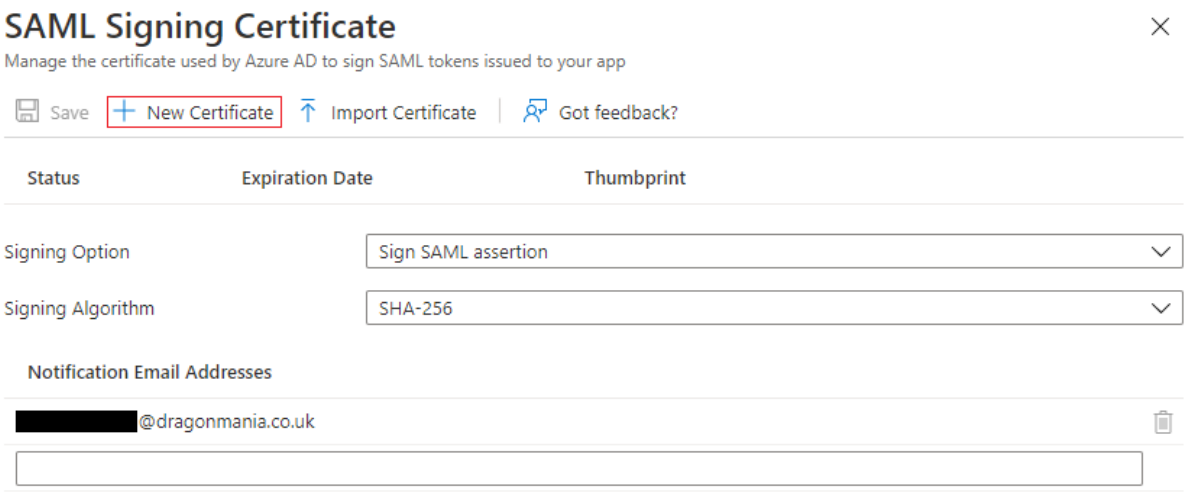
4.6 In addition to above, AWS application expects few more attributes to be passed back in SAML response which are shown below. These attributes are also pre populated but you can review them as per your requirements.

Name	Source attribute	Namespace
RoleSessionName	user.userprincipalname	https://aws.amazon.com/SAML/Attributes
Role	user.assignedroles	https://aws.amazon.com/SAML/Attributes
SessionDuration	"provide a value between 900 seconds (15 minutes) to 43200 seconds (12 hours)"	https://aws.amazon.com/SAML/Attributes

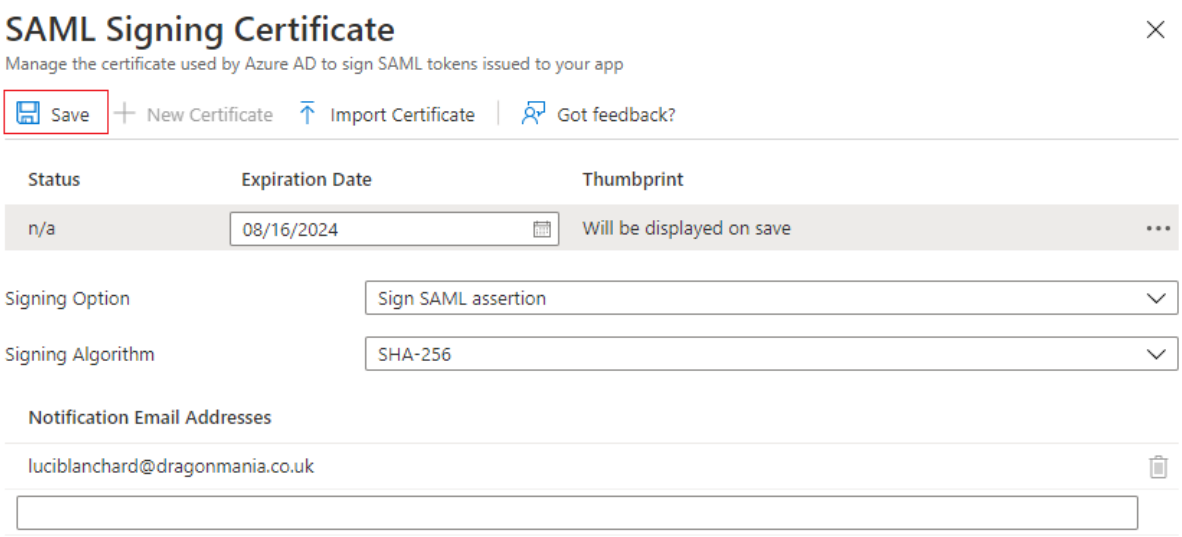
4.7 On the **Set up single sign-on with SAML** page, in the **SAML Signing Certificate** (Step 3) dialog box, select **Add a certificate**.



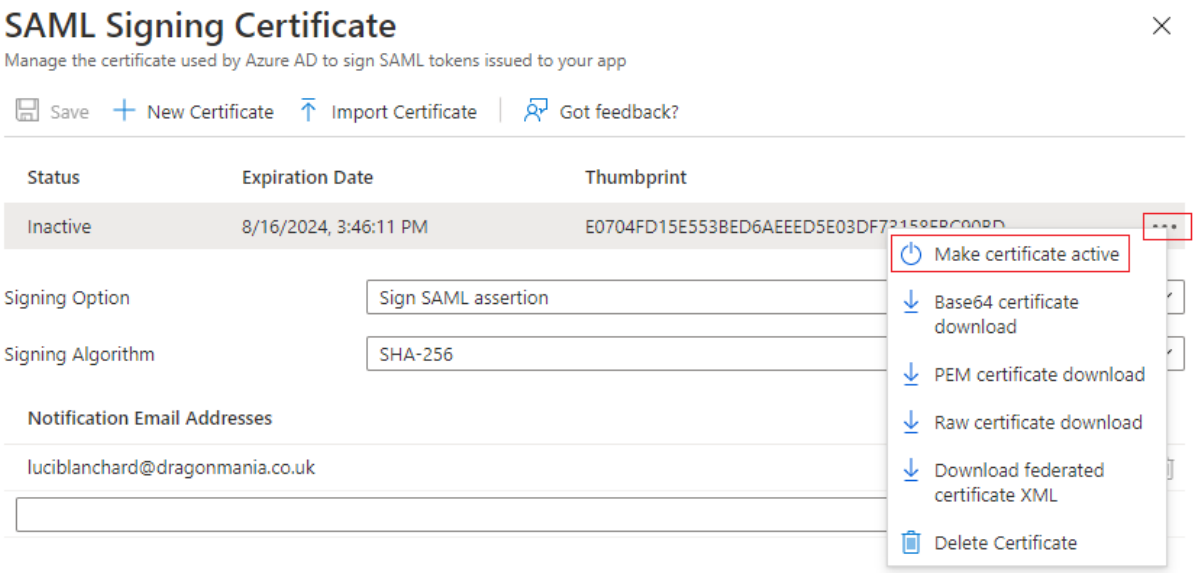
4.8 Generate a new SAML signing certificate by selecting **New Certificate**. Enter an email address for certificate notifications.



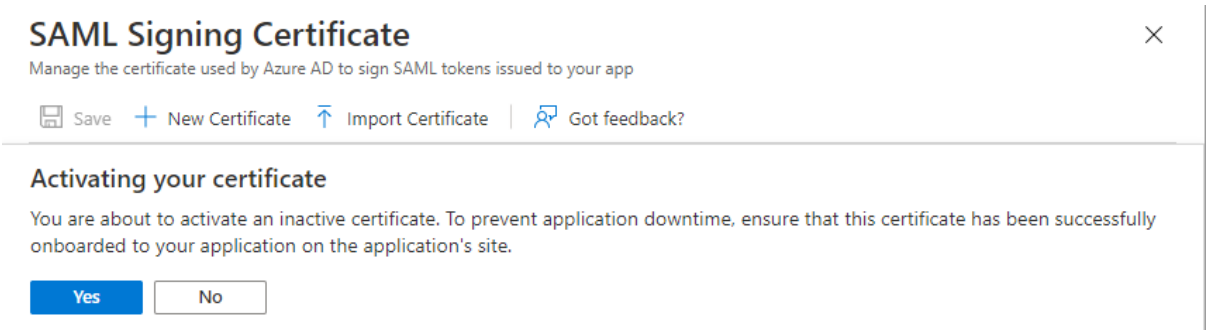
4.9 In the **SAML Signing Certificate**, click **Save**.



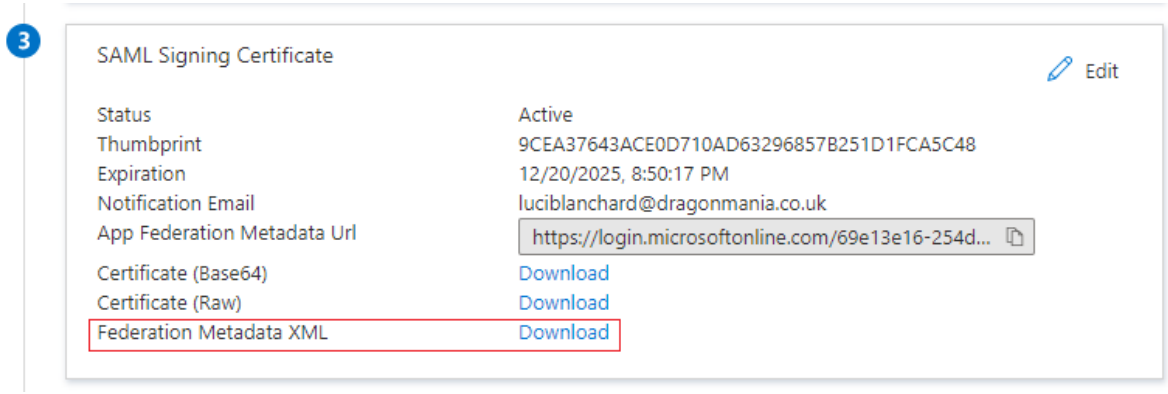
4.10 In the **SAML Signing Certificate**, click on the "…" and select **Mark certificate active**.



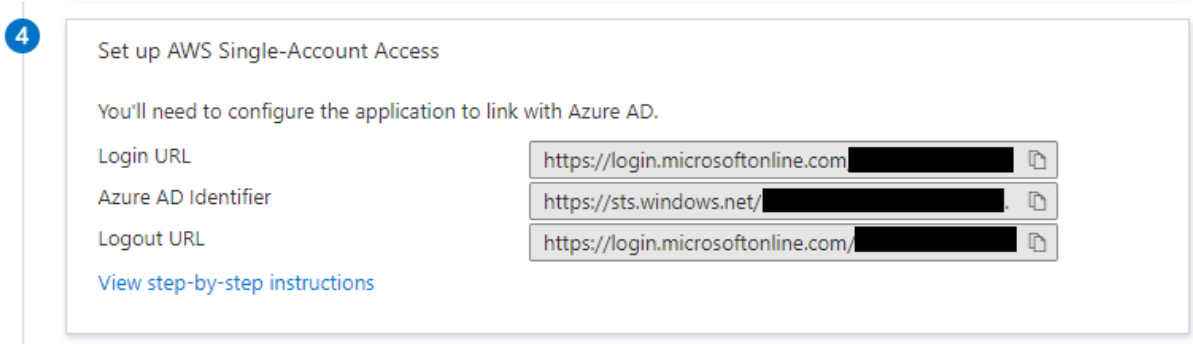
4.11 In the **Activating your certificate** prompt, select **Yes** to activate the certificate.



4.12 In the **SAML Signing Certificate** section, select **Download** to download the **Federated Metadata XML** and save it on your device.



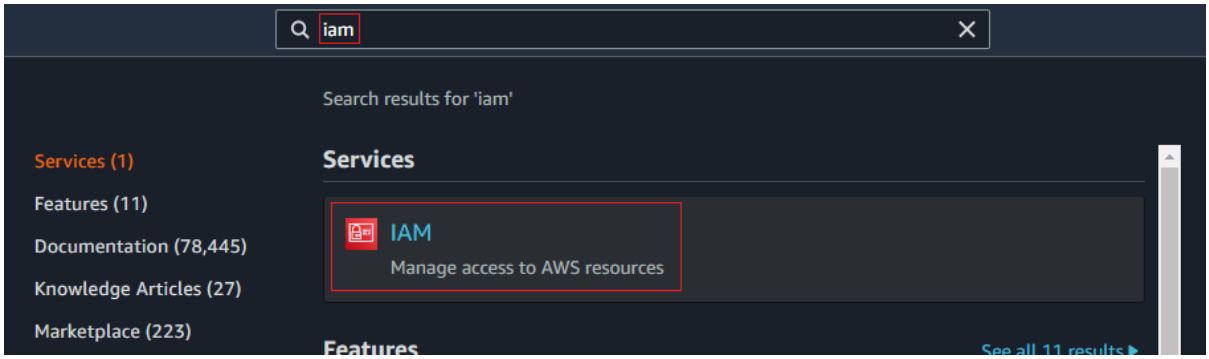
4.13 In the **Set up AWS Single-Account Access** section, copy the following URL(s):



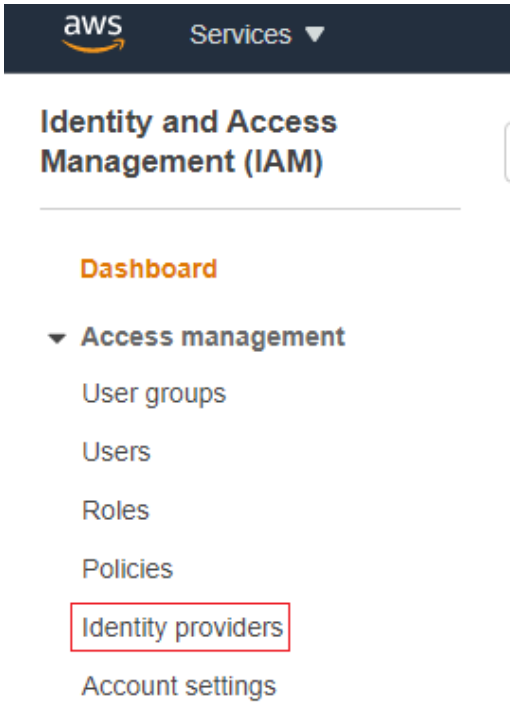
5. Configure AWS Single-Account Access SSO

5.1 In a different browser window, sign-on to your AWS company site as an administrator.

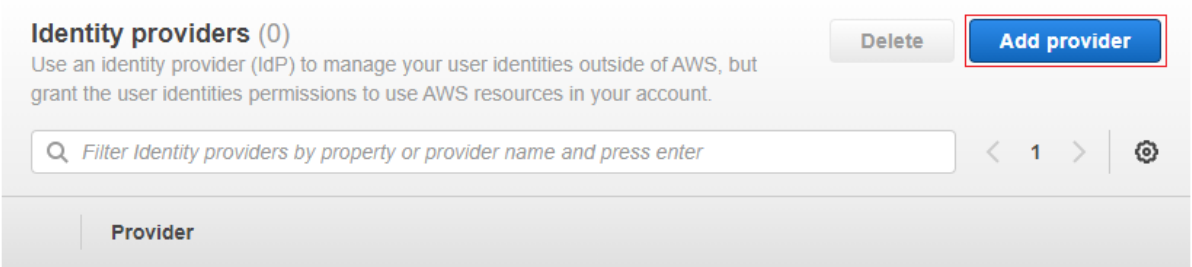
5.2 In the search box, type “IAM”, select “IAM – Manage access to AWS resources”.



5.3 Select **Identity Providers** > **Create Provider**.

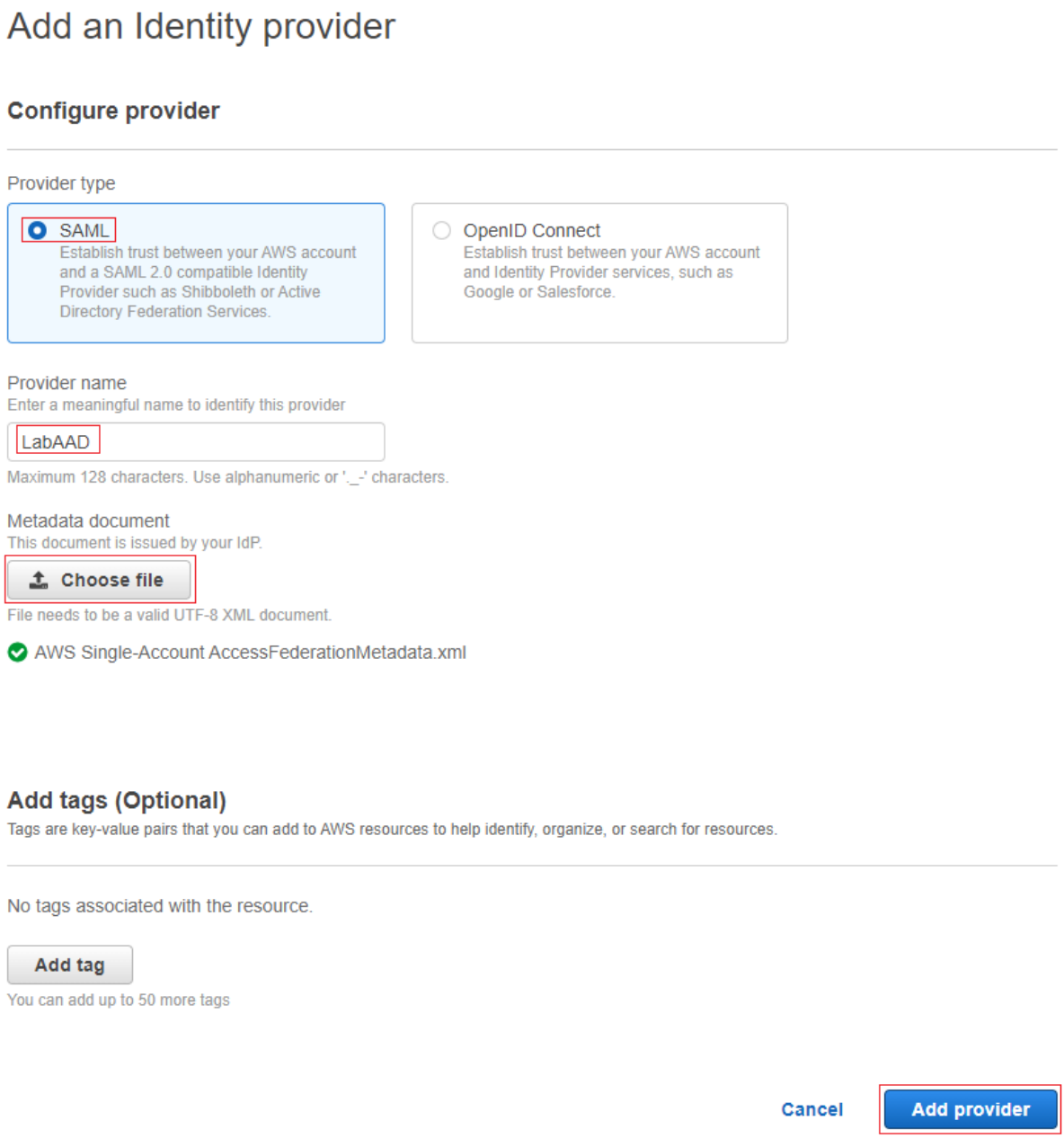


5.4 On the **Identity Providers** page, click **Add Provider.**



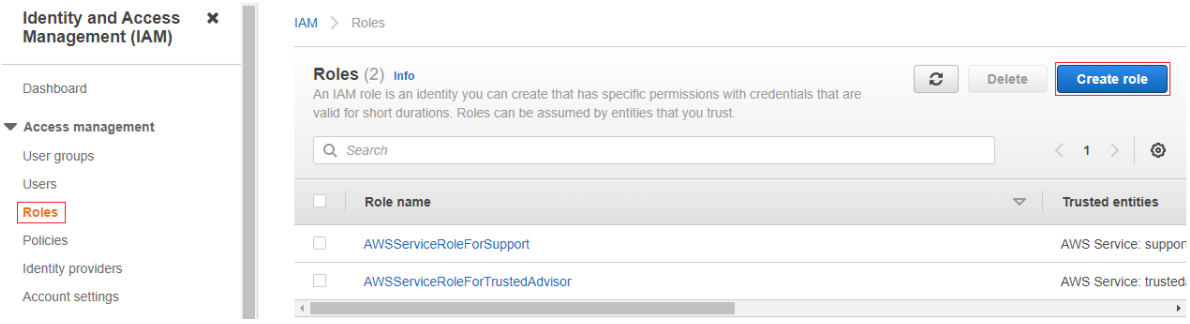
5.5 On the **Add Identity Provider** page, perform the steps below:

a) Select **SAML** for the **Provider Type**
b) Enter a name for the **Provider Name** (for example: LabAAD)
c) Click **Choose File** and upload the **metadata file** you download from Azure on step 4.12 earlier in this document.
d) Click **Add Provider**.



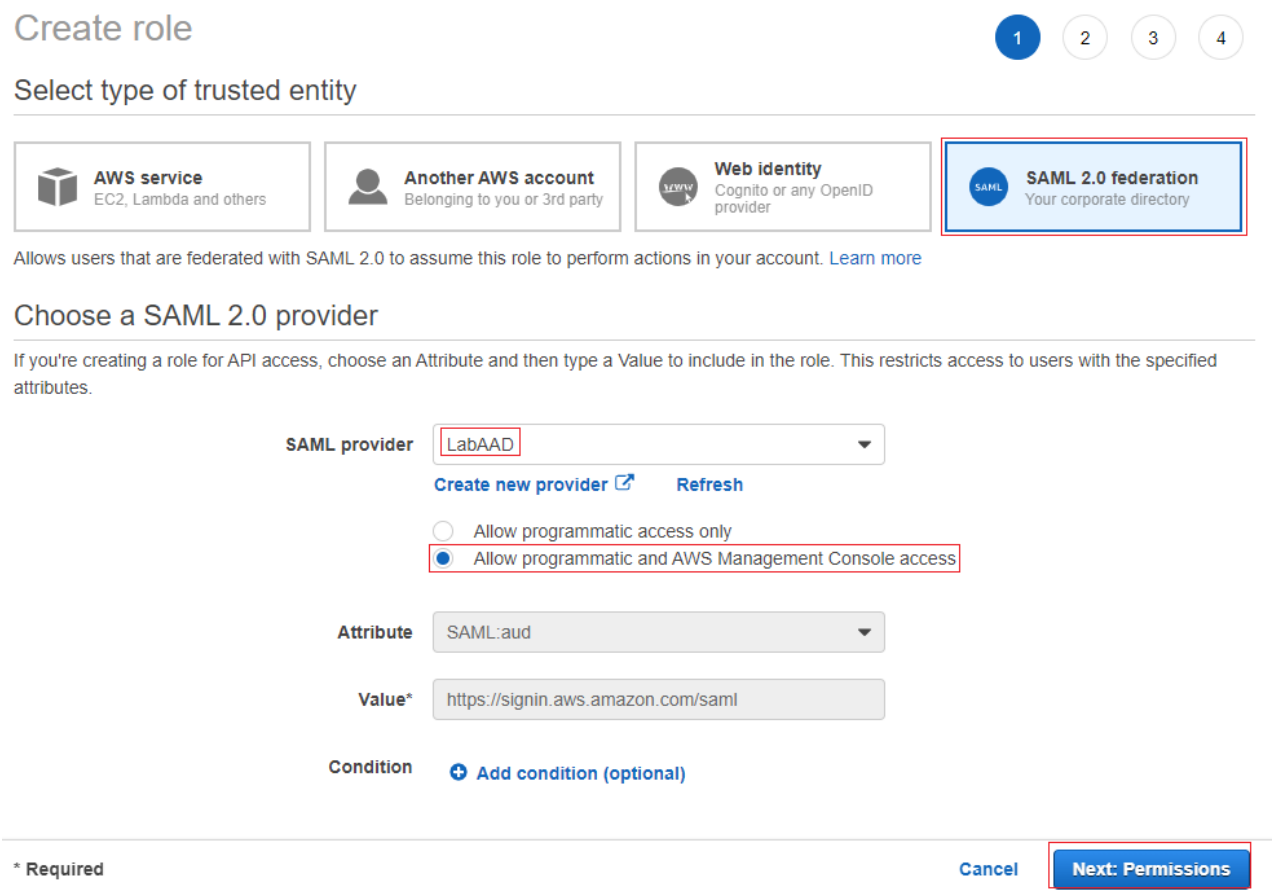
6. Create AWS roles

6.1 In **AWS IAM**, select **Roles** -> **Create Role**

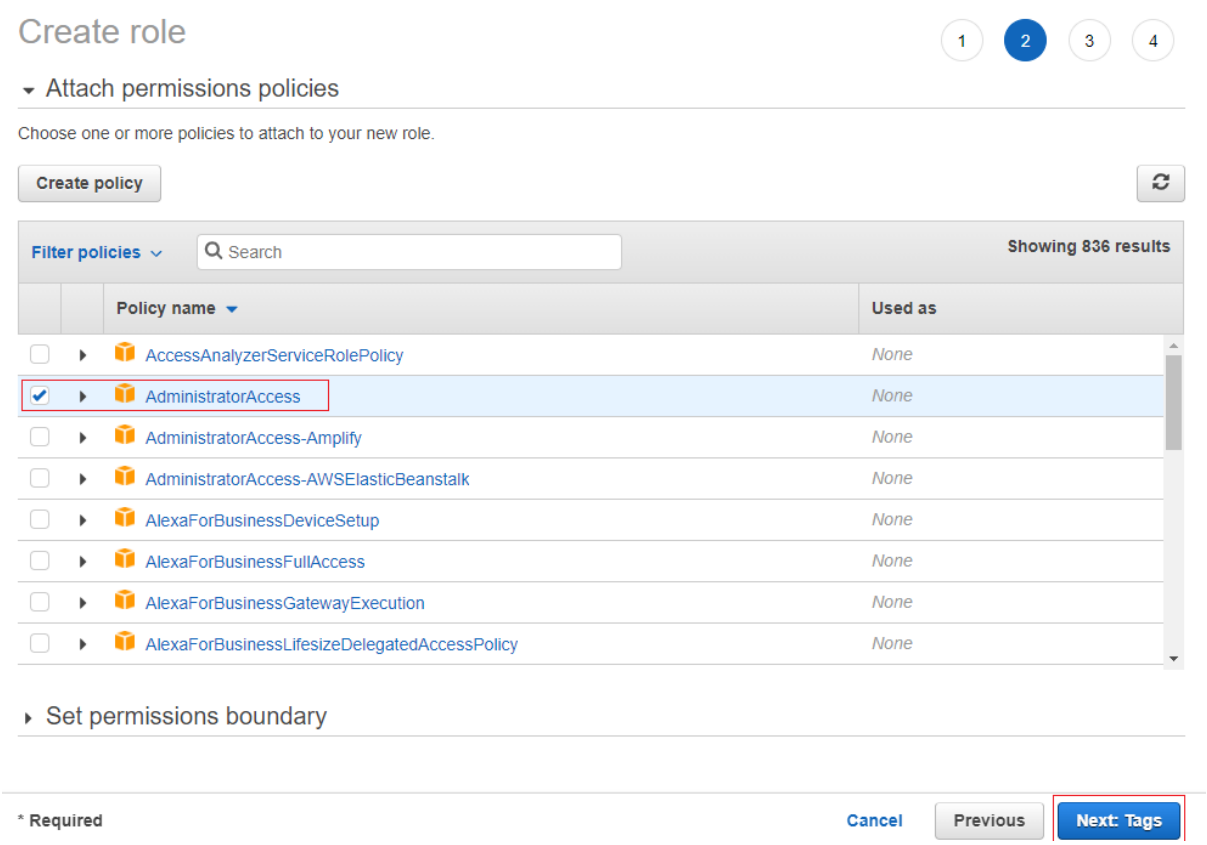


6.2 On the **Create role** page, perform the following steps:

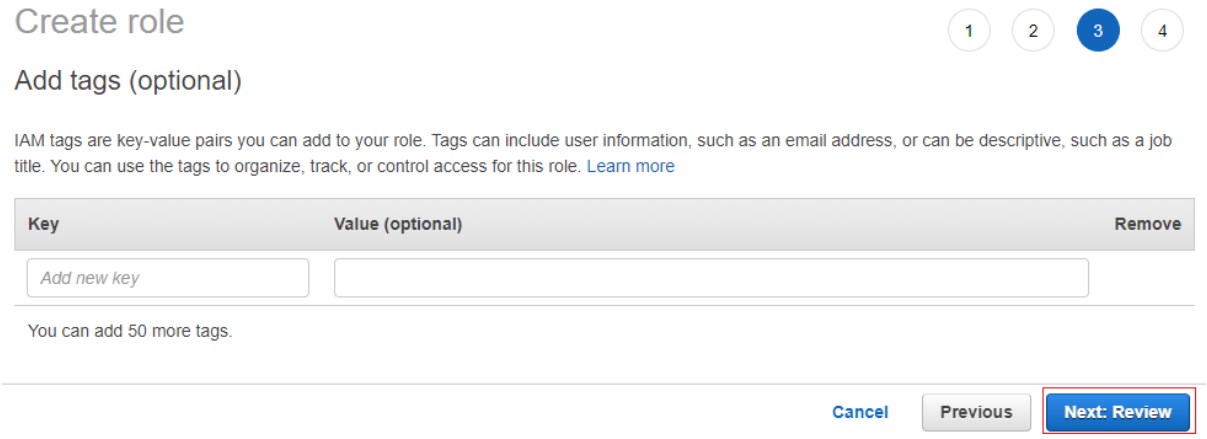
a) Under **Select type of trusted entity**, select **SAML 2.0** federation.
b) Under **Choose a SAML 2.0 Provider**, select the **SAML** provider you created previously (for example: *LabAAD*).
c) Select **Allow programmatic and AWS Management Console access**.
d) Select **Next: Permissions**.



6.3 On the **Attach permissions policies** dialog box, select **AdministratorAccess**. Then select **Next: Tags**.

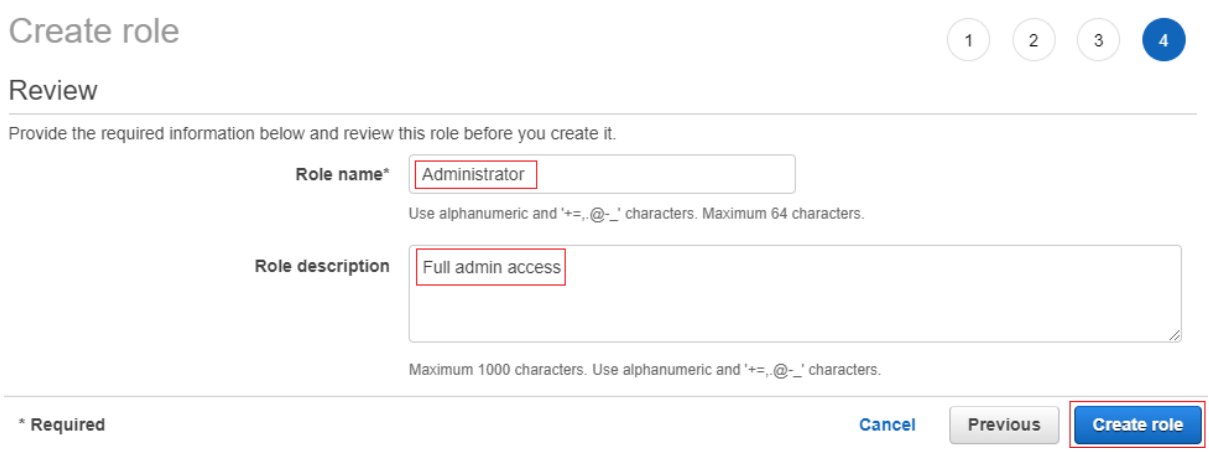


6.4 On the **Add Tags** dialog box, leave blank and select **Next: Review**.



6.5 On the **Review** dialog box, perform the following steps:

a) In **Role Name,** enter your role name (**Administrator**).
b) In **Role Description,** enter the description.
c) Select **Create Role.**

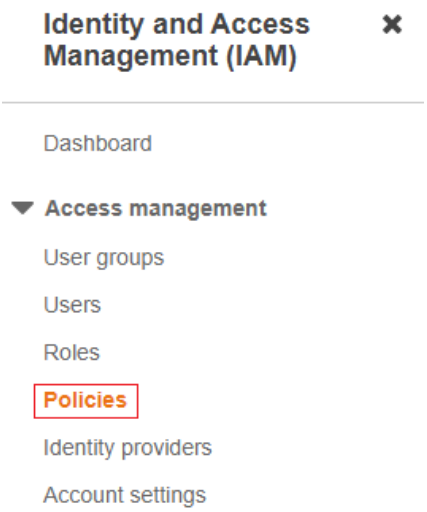


6.6 Create an additional role following the steps listed above, name it **Developer** and attach to it **AmazonS3FullAccess** permissions.

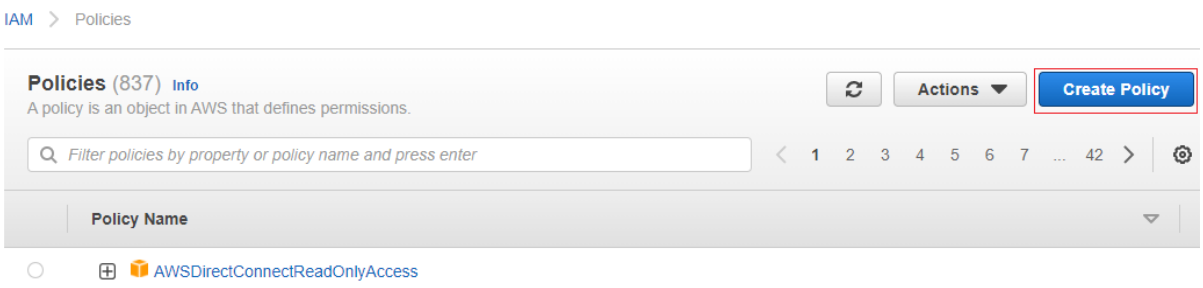
6.7 You have now successfully create an **Administrator** and a **Developer** role in AWS.

7. Create AWS Policy for fetching the roles

7.1 In **AWS IAM**, select **Policies**.



7.2 Create a new policy by selecting **Create policy** for fetching the roles from the AWS account in Azure AD user provisioning.

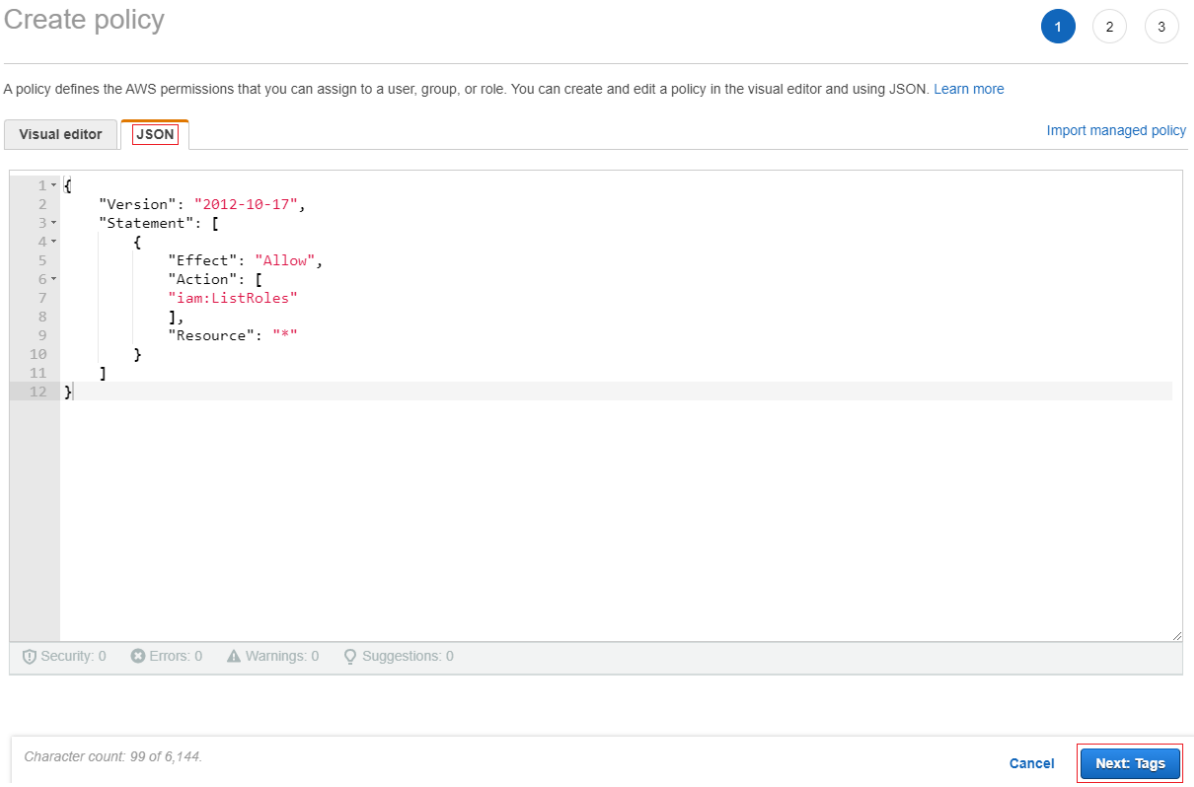


7.3 Create your own policy to fetch all the roles from AWS accounts.

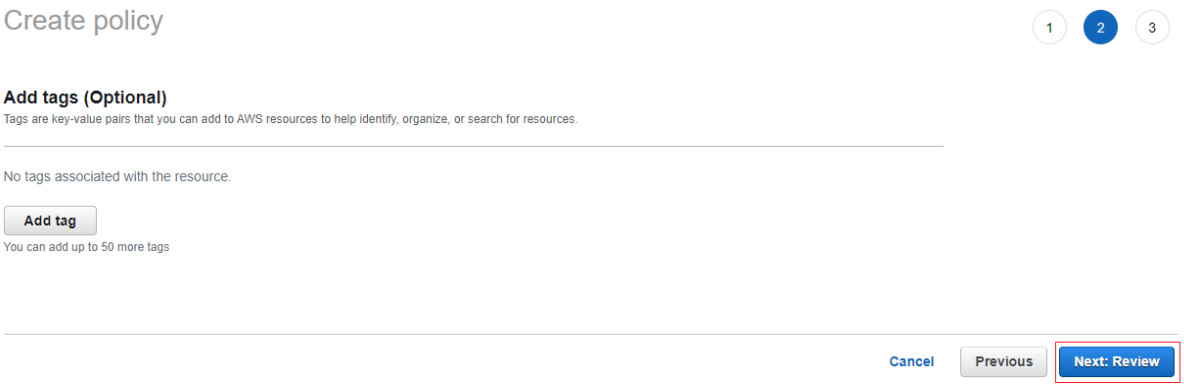
a) In **Create policy**, select the **JSON** tab.
b) In the policy document, add the following JSON:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
            "iam:ListRoles"
            ],
            "Resource": "*"
        }
    ]
}
```

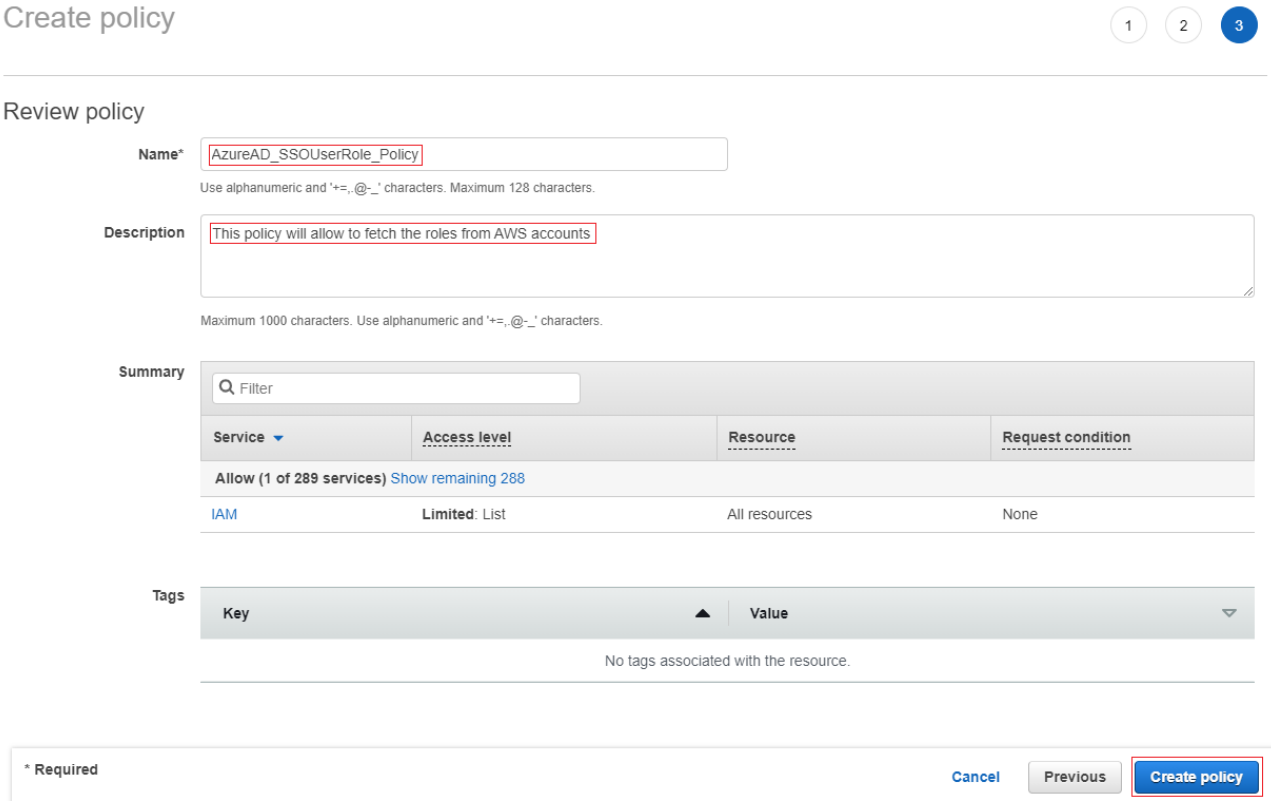


c) Select **Next: Tags** to continue.
d) On the **Add Tags** dialog box, leave empty and click **Next: Review.**



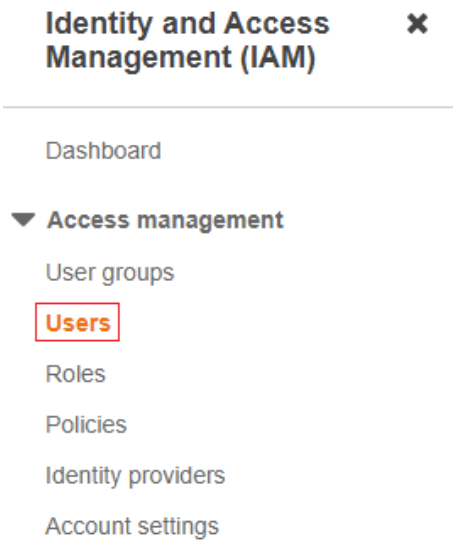
7.4 Define the new policy.

a) For **Name**, enter **AzureAD_SSOUserRole_Policy**.
b) For **Description**, enter **This policy will allow to fetch the roles from AWS accounts**.
c) Select **Create policy**.

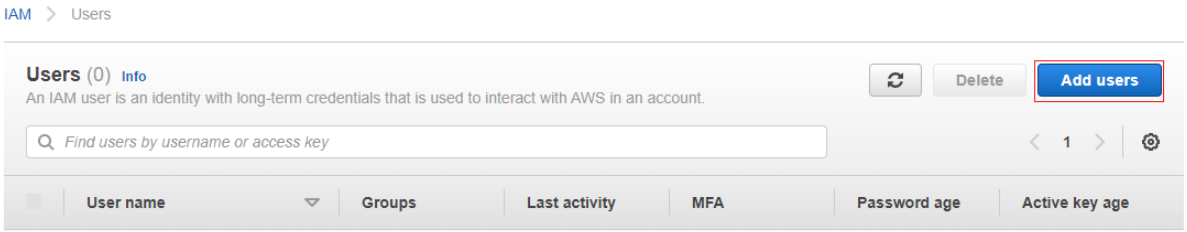


7.5 Create a new user account in the **AWS IAM** service.

a) In the **AWS IAM** console, select **Users**.

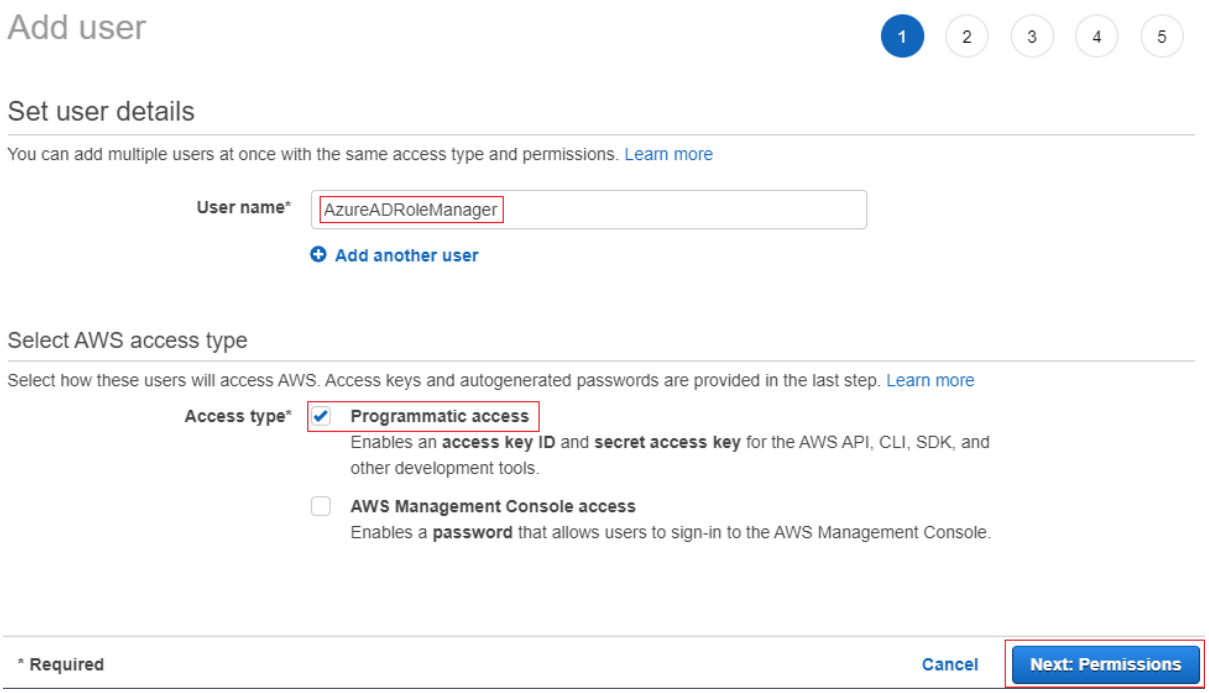


b) To create a new user, select **Add user**.



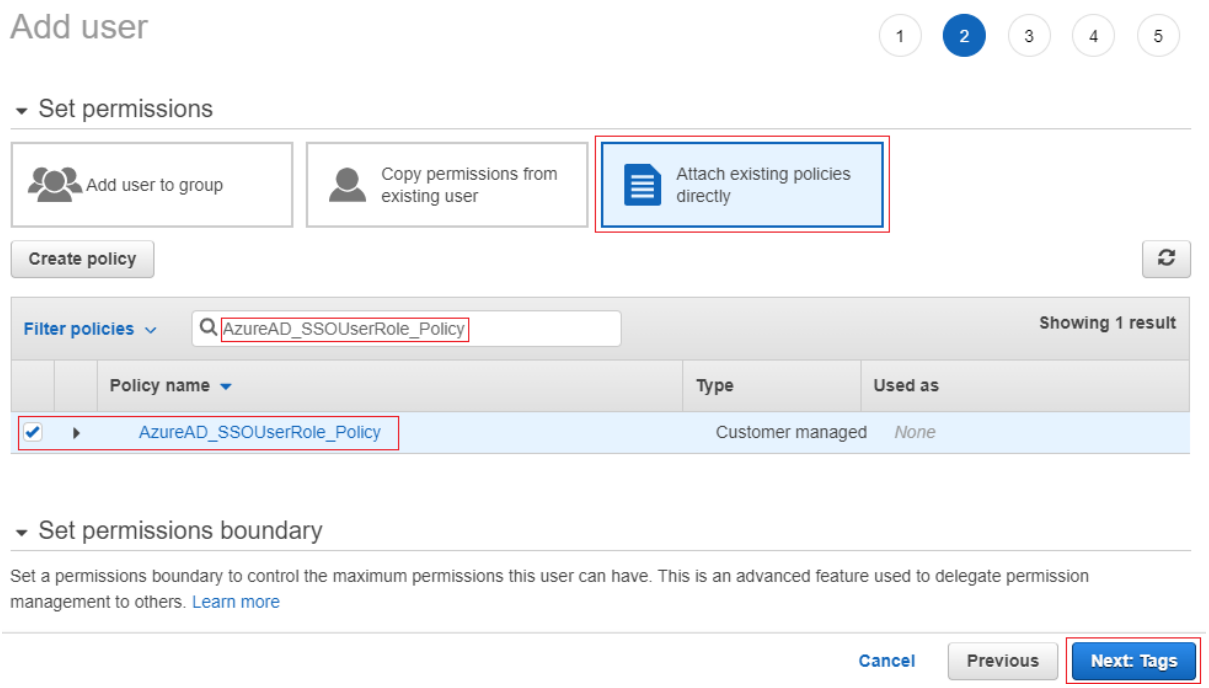
c) In the **Add user** section:

- Enter the user name as **AzureADRoleManager**.
- For the access type, select **Programmatic access**. This way, the user can invoke the APIs and fetch the roles from the AWS account.
- Select **Next Permissions**.

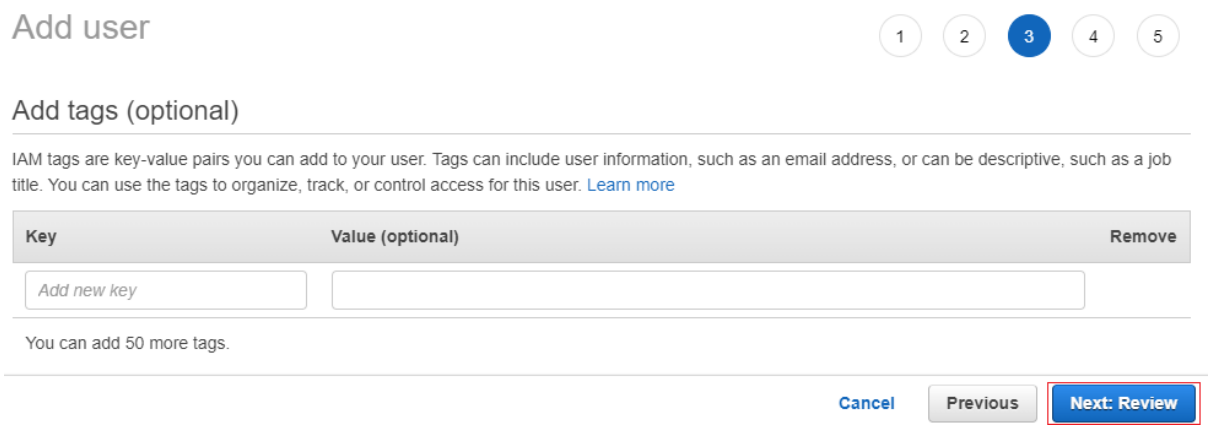


7.6 Create a new policy for this user.

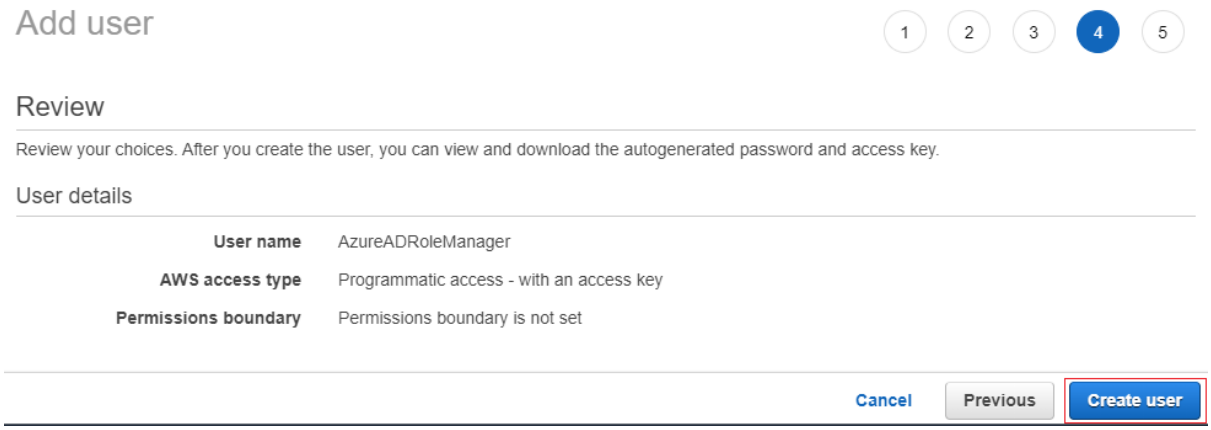
a) Select **Attach existing policies directly**.
b) Search for the newly created policy in the filter section **AzureAD_SSOUserRole_Policy**.
c) Select the policy, and then select **Next: Tags**.



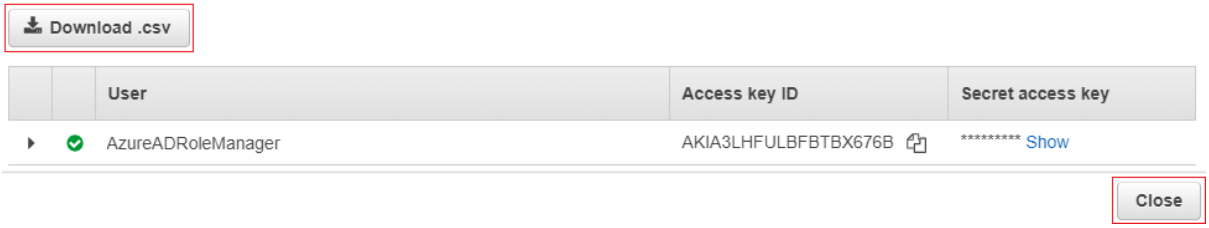
d) In the **Add Tags** dialog box, leave empty and select **Next: Review**.



7.7 Review the policy to the attached user, select **Create User.**

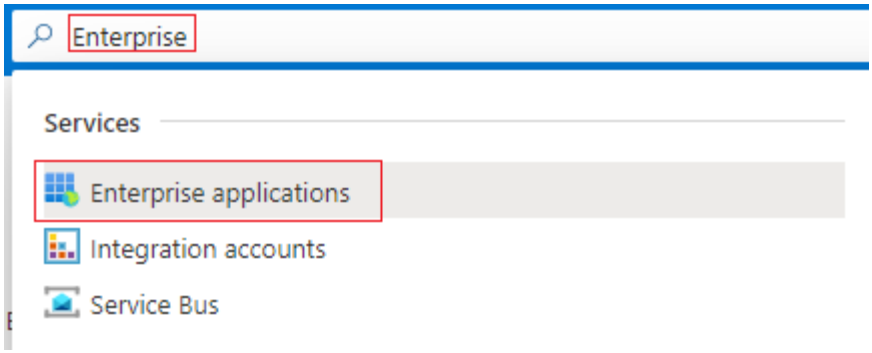


7.8 Download the user credentials by clicking on **download .csv**. Select **Close.**

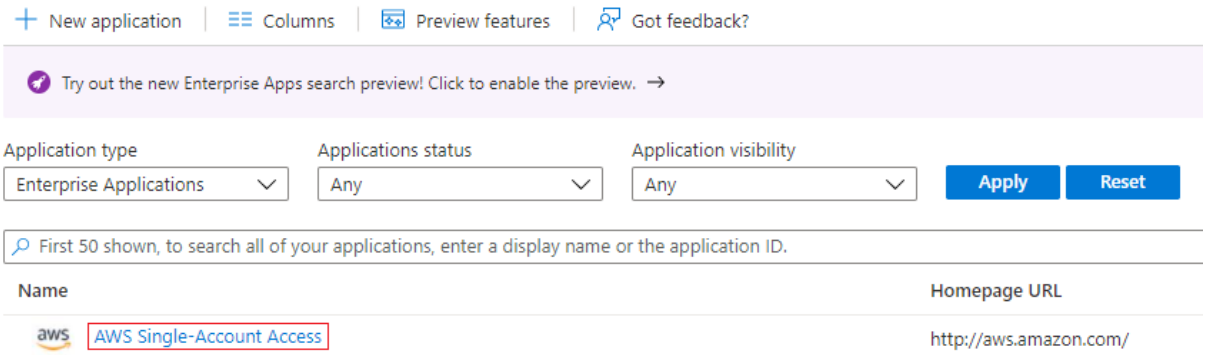


8. Configure role provisioning in AWS Single-Account Access

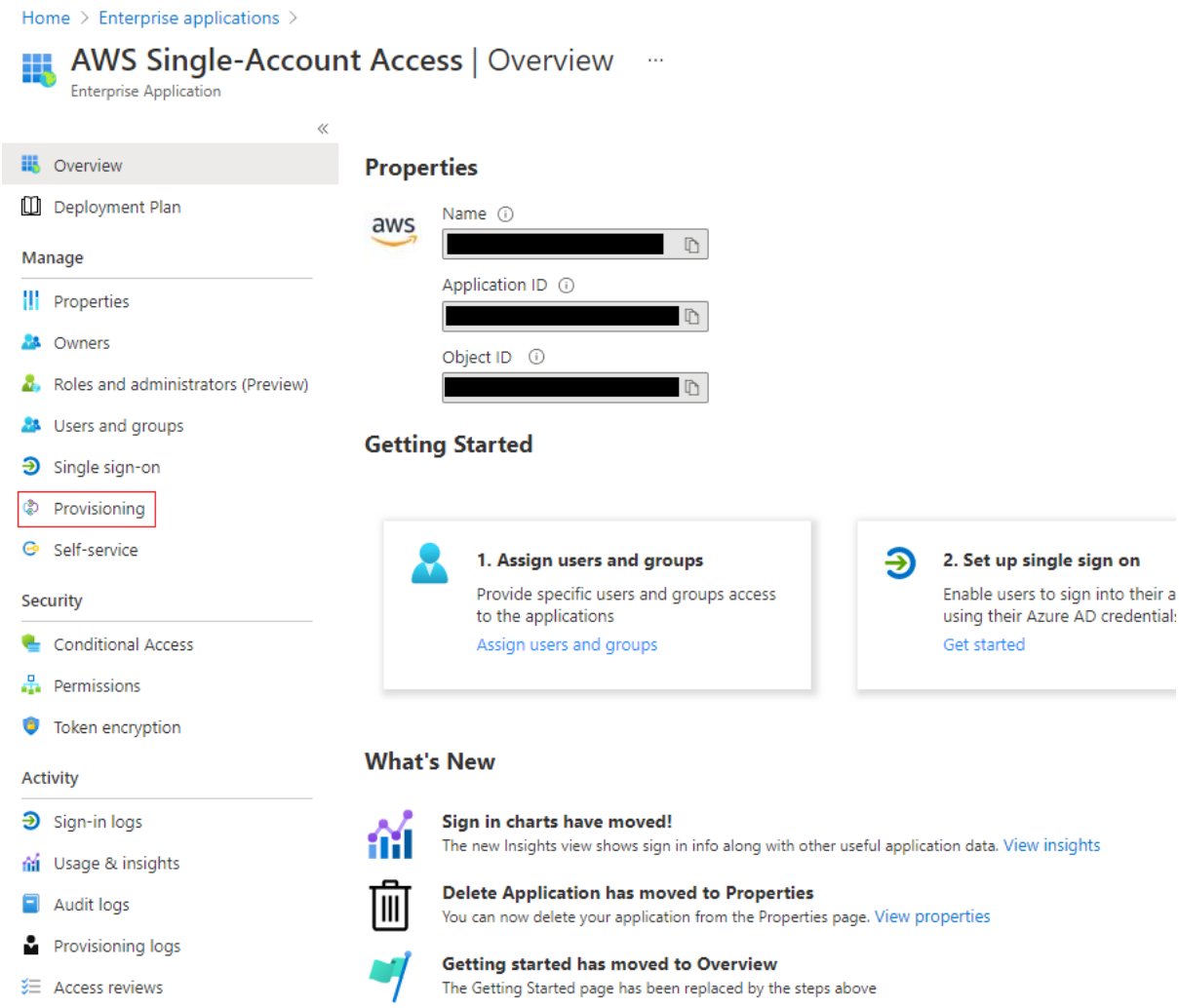
8.1 In **Azure Portal**, search for **Enterprise Applications**.



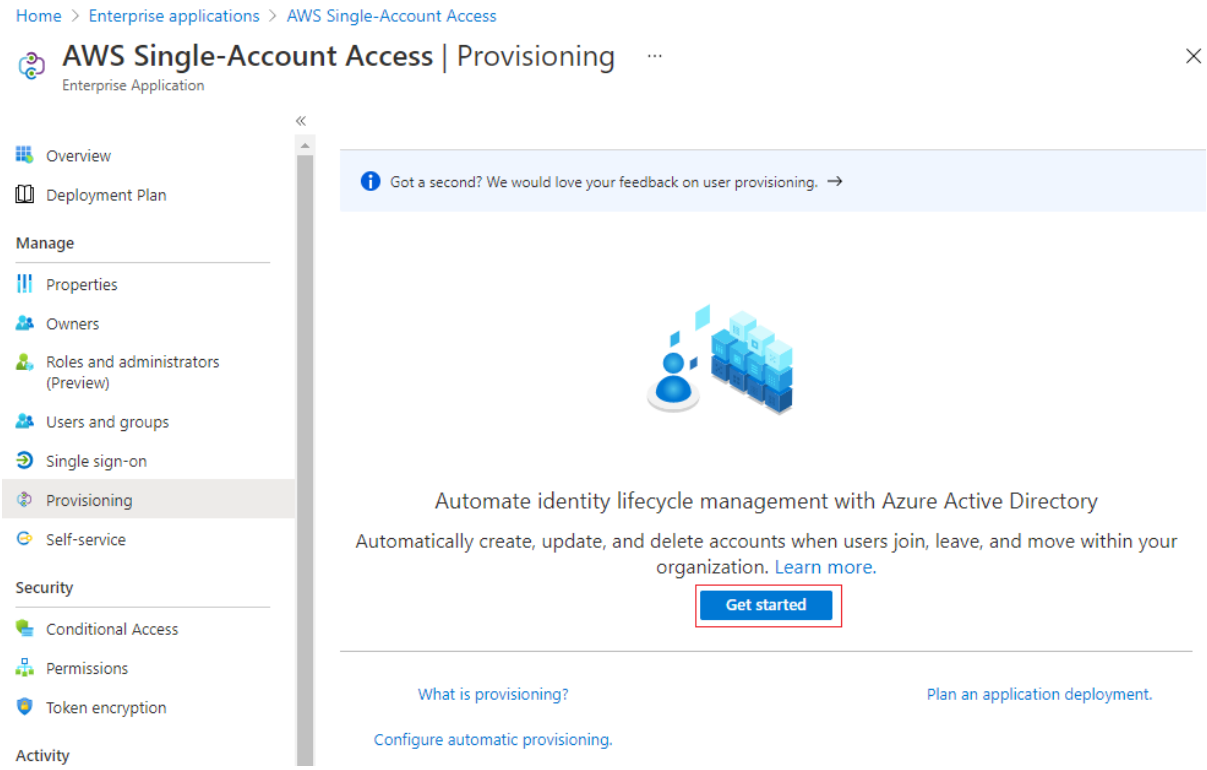
8.2 In **Enterprise Applications**, select the **AWS Single-Account Access** application you have created in step 3.4 earlier in this document.



8.3 On the application page, select **Provisioning.**

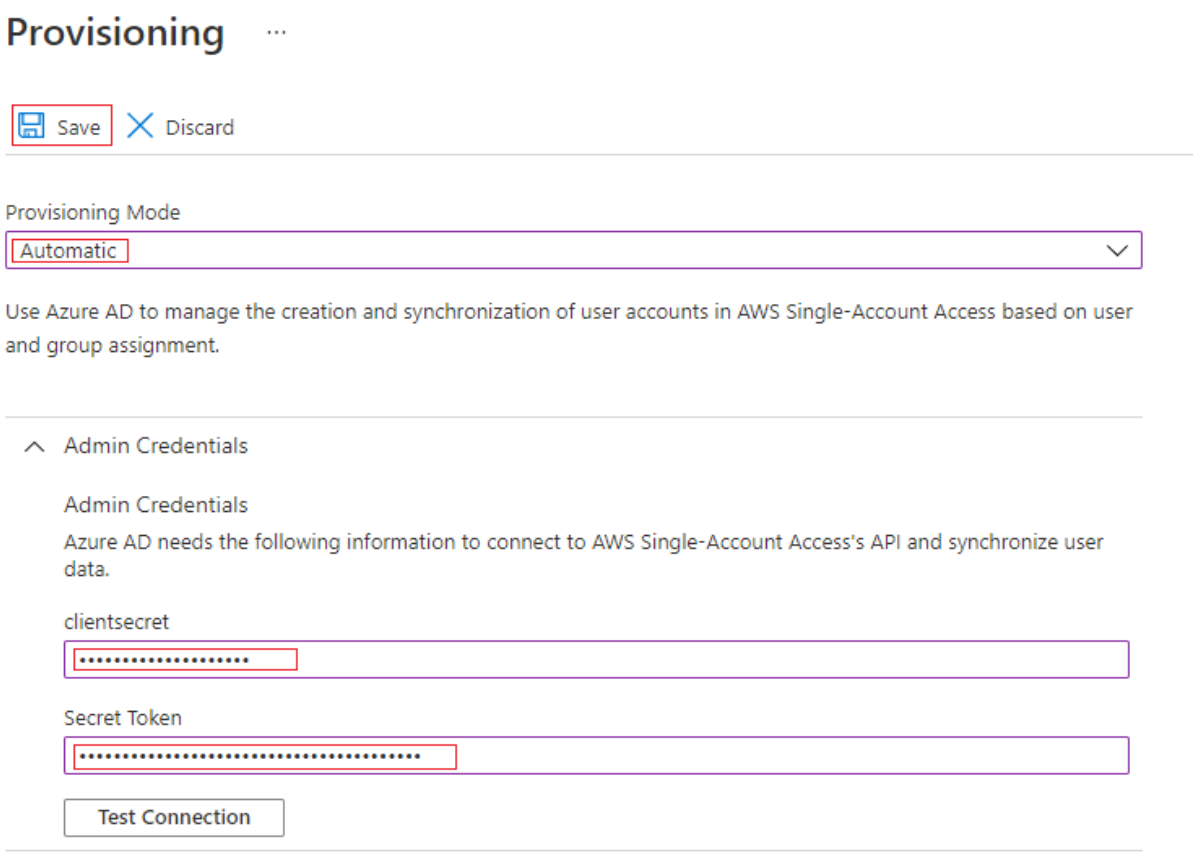


8.4 On the **Provisioning** page, select **Get Started.**

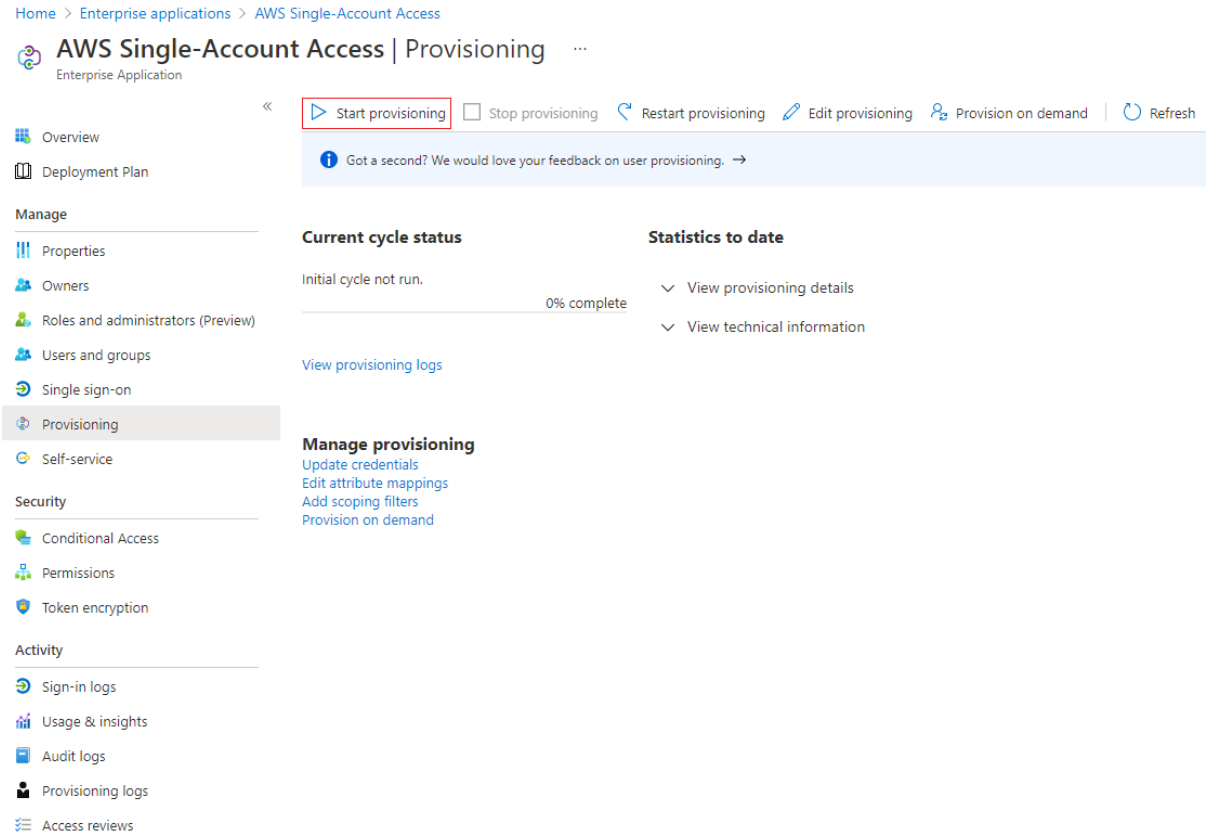


8.5 On the **Provisioning** page, perform the following:

a) In **Provisioning Mode,** select **Automatic.**
b) In **Admin Credentials,** copy and paste the **Client Secret** (Access Key ID) found in the .csv file you downloaded from AWS in step 4.6.9.
c) In **Admin Credentials,** copy and paste the **Secret Token** (Secret Access Key) found in the .csv file you downloaded from AWS in step 7.8 earlier in this document.
d) Select **Test Connection** to validate.
e) Select **Save.**



8.1 On the **Provisioning** page, select **Start Provisioning.**



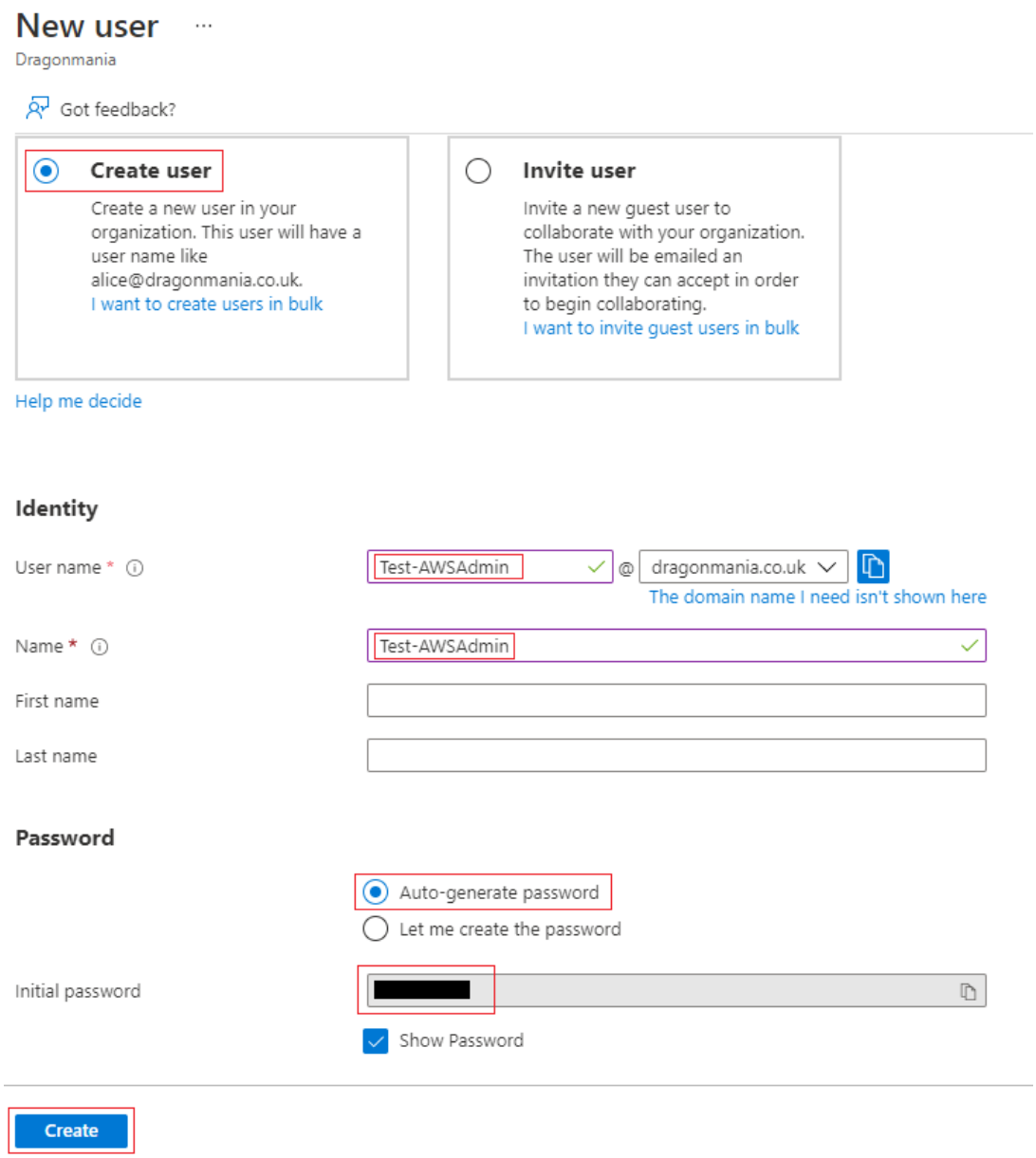
> ⓘ **Note**
>
> The provisioning service imports roles only from AWS to Azure AD. The service does not provision users and groups from Azure AD to AWS.

> ⓘ **Note**
>
> After you save the provisioning credentials, you must wait for the initial sync cycle to run. Sync usually takes around 40 minutes to finish. You can see the status at the bottom of the **Provisioning** page, under **Current Status**.

9. Create Azure AD test users

In this section, you'll create two test users in the Azure portal.

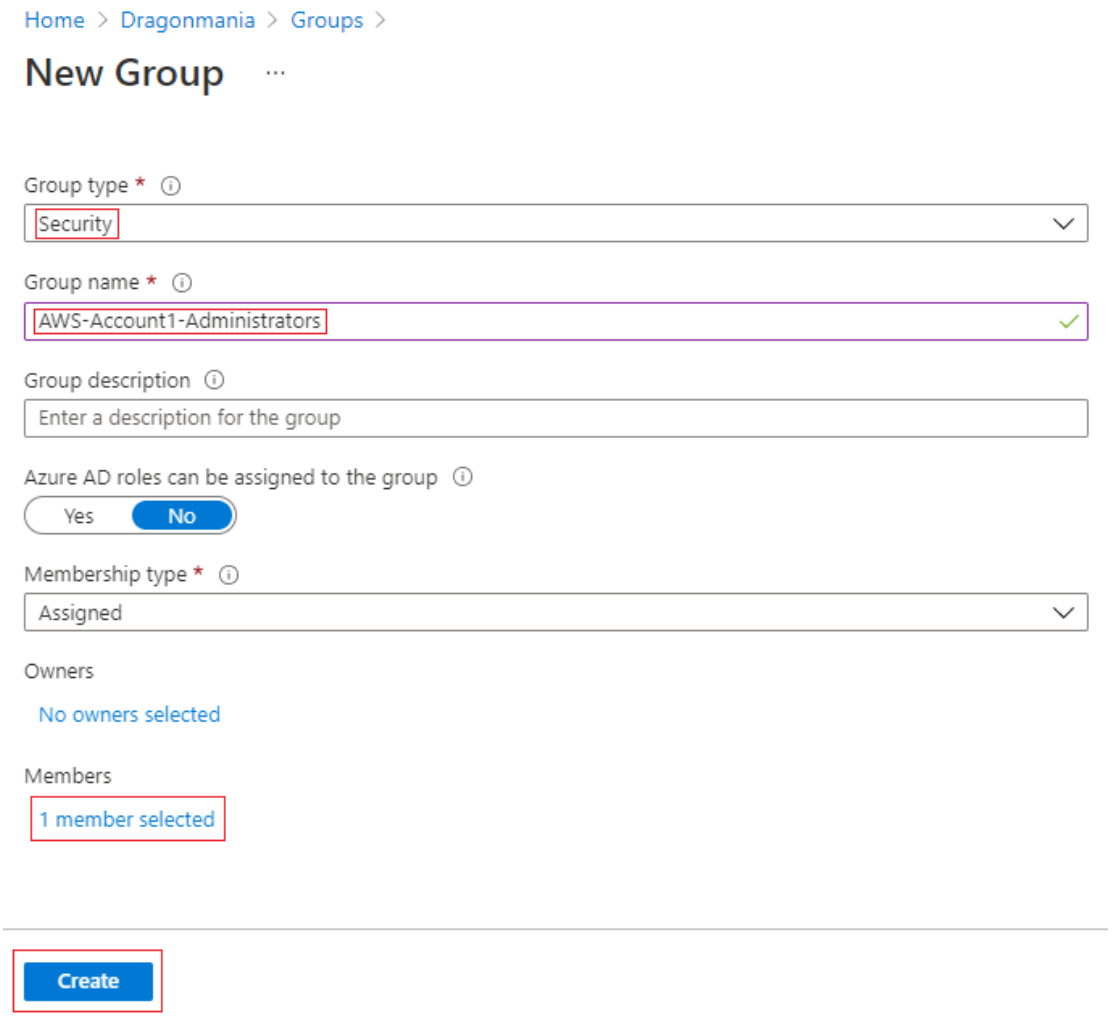
9.1 In the **Azure portal**, search for and select **Azure Active Directory**.
9.2 Within the **Azure Active Directory** overview menu, choose **Users** > **All users**.
9.3 Select **New user** at the top of the screen.
9.4 In the **User properties**, follow these steps:
9.5 In the **Name** field, enter **Test-AWSAdmin**.
9.6 In the **User name** field, enter the **Test-AWSAdmin**@companydomain.extension. For example, **Test-AWSAdmin**@contoso.com.
9.7 Select the **Show password** check box, and then write down the value that's displayed in the Password box.
9.8 Click **Create**.
9.9 Repeat the steps in this section to create a second user named **Test-AWSDeveloper**.



10. Create Azure AD test groups

In this section, you'll create two test groups in the Azure portal.

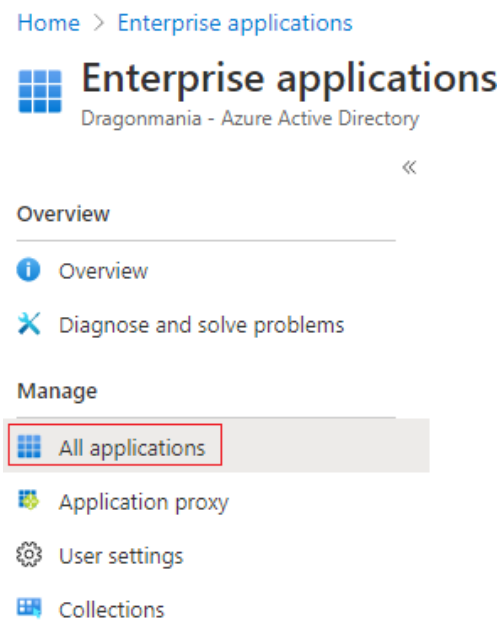
10.1 In the **Azure portal**, search for and select **Azure Active Directory**.
10.2 Within the **Azure Active Directory** overview menu, choose **Groups** > **All groups**.
10.3 In the **Group Type,** select **Security.**
10.4 In the **Group Name** field, enter **AWS-Account1-Administrators.**
10.5 Select **Members,** search for **Test-AWSAdmin** created in the previous section and click **Select**.
10.6 Select **Create** to create the new group in Azure AD**.**
10.7 Repeat the steps in this section to create another group called **AWS-Account1-Developers**, add **Test-AWSDeveloper** user to **AWS-Account1-Developers**.



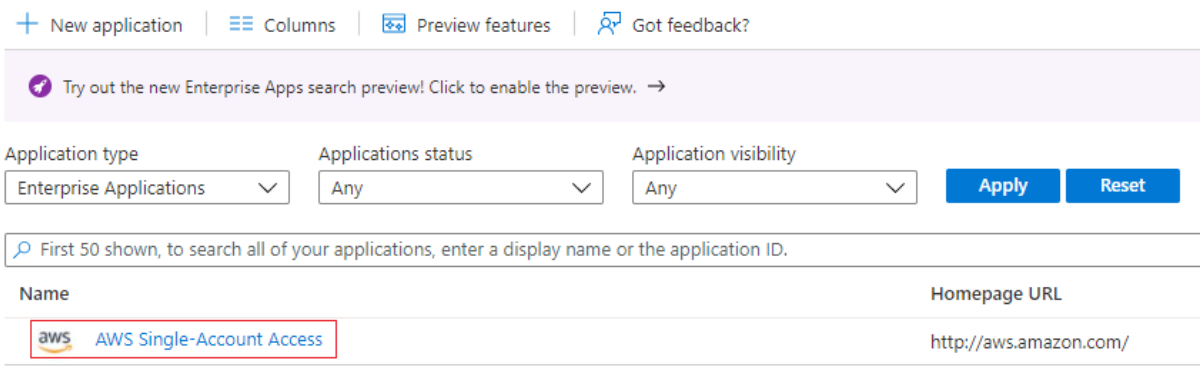
11. Assign the Azure AD test groups

In this section, you'll enable **Test-AWSDeveloper** and **Test-AWSAdmin** to use Azure single sign-on by granting access to **AWS Single-Account Access**.

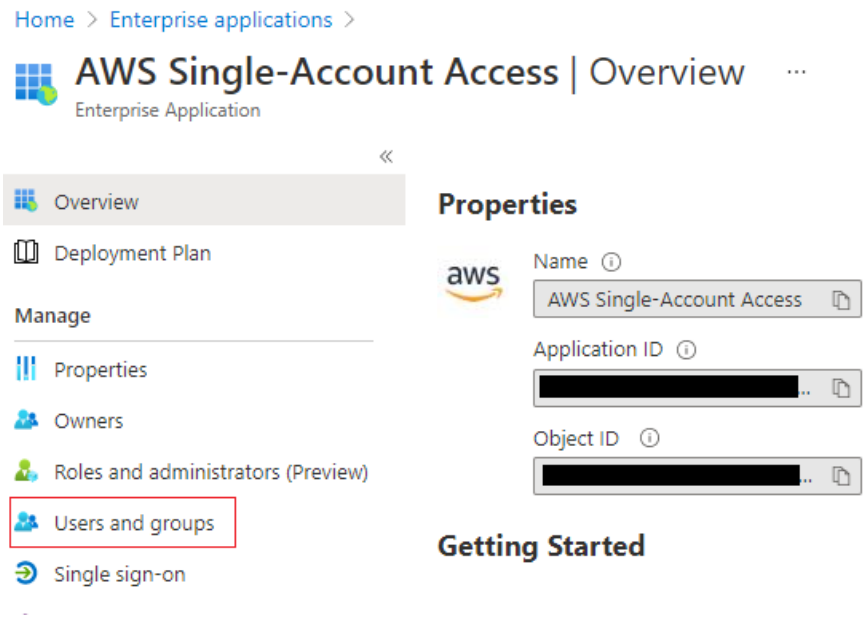
11.1 In the **Azure portal**, select **Enterprise Applications**, and then select **All applications**.



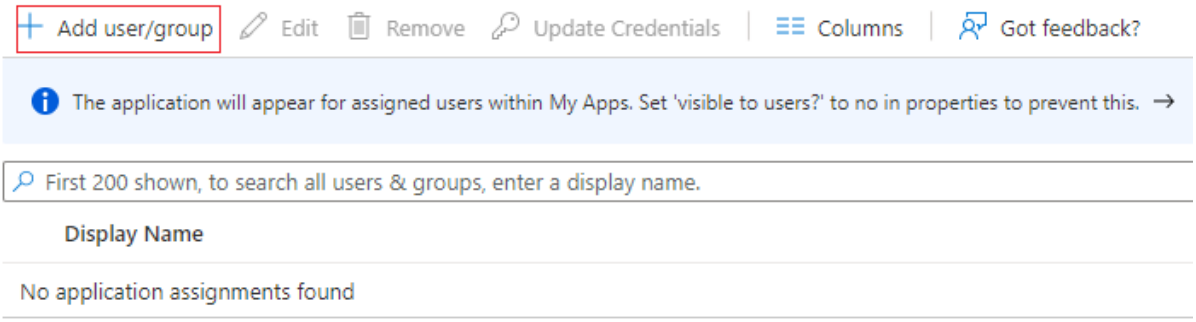
11.2 In the applications list, select **AWS Single-Account Access**.



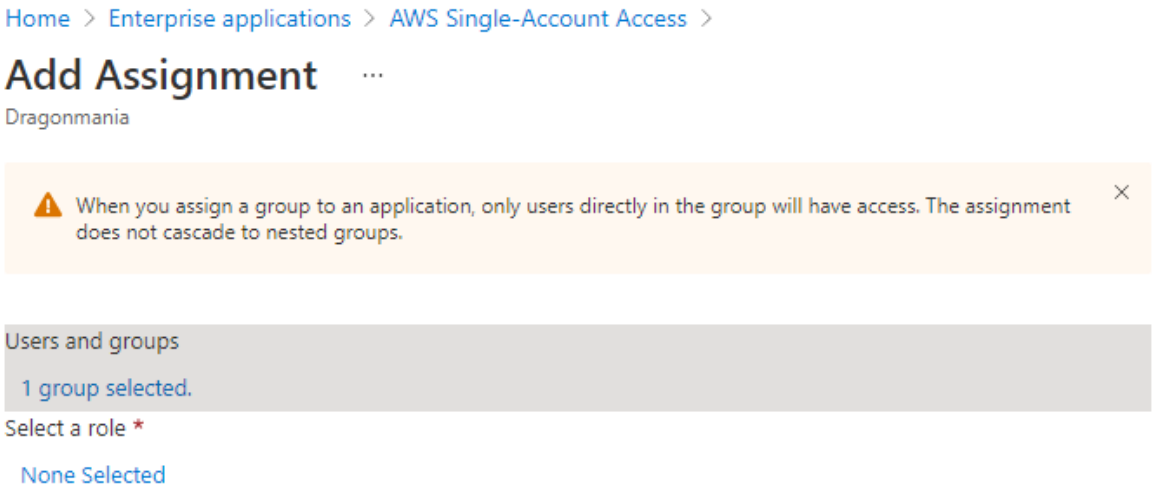
11.3 In the app's overview page, find the **Manage** section and select **Users and groups**.



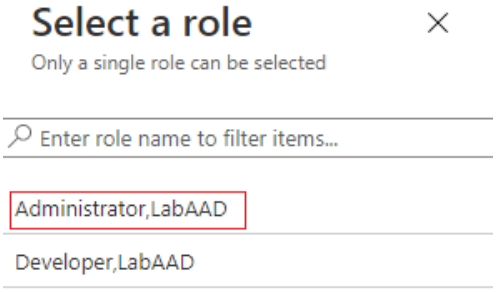
11.4 Select **Add user/group**, then select **None Selected** in the **Add Assignment** dialog.



11.5 In the **Users and groups** dialog, search for **AWS-Account1-Administrators,** then click the Select button at the bottom of the screen.

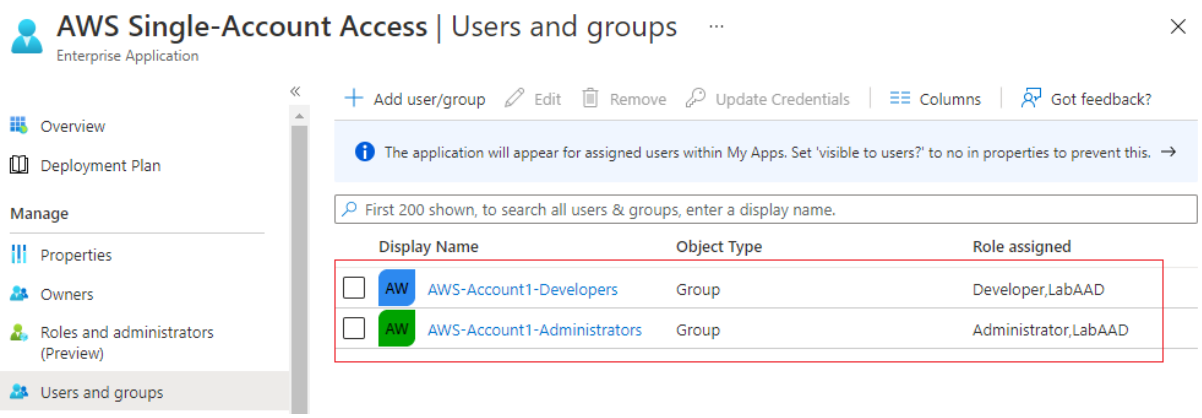


11.6 Under **Select a role**, select **None Selected,** then select **Administrator,LabAAD** (the administrator role you have created in AWS) and click **Select** at the bottom of the screen.



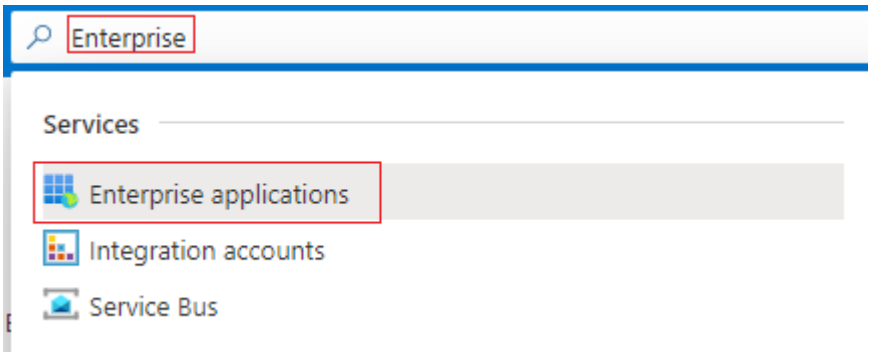
11.7 In the **Add Assignment** dialog, click the **Assign** button.

11.8 Repeat the steps in this section for the **AWS-Account1-Developers** group**,** assign it to the **Developer,LabAAD** role (the developer role you have created in AWS).

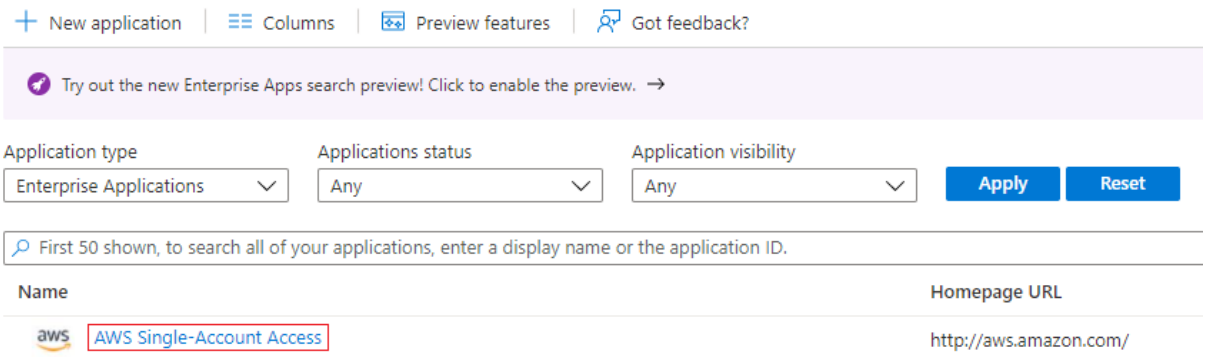


12. Test SSO

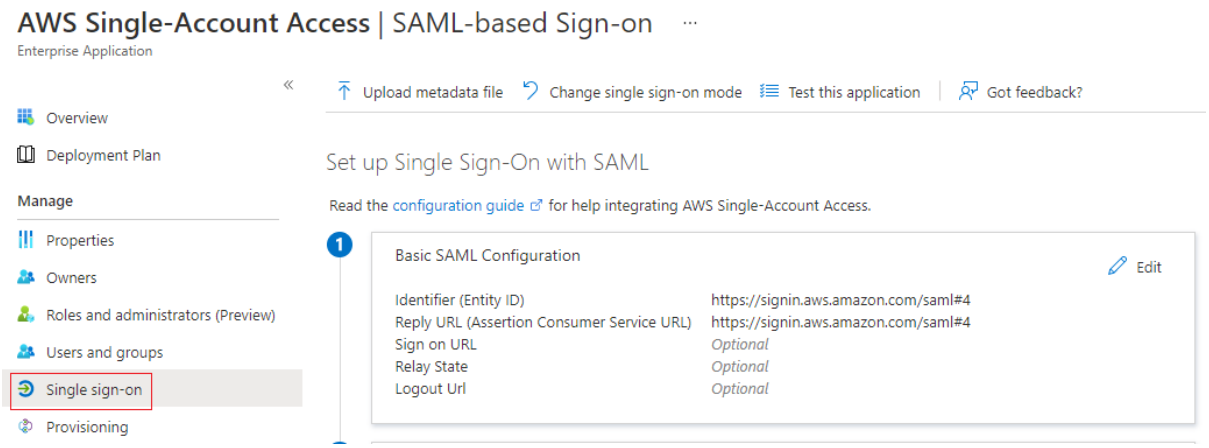
12.1 In Azure Portal, search for **Enterprise Applications**.



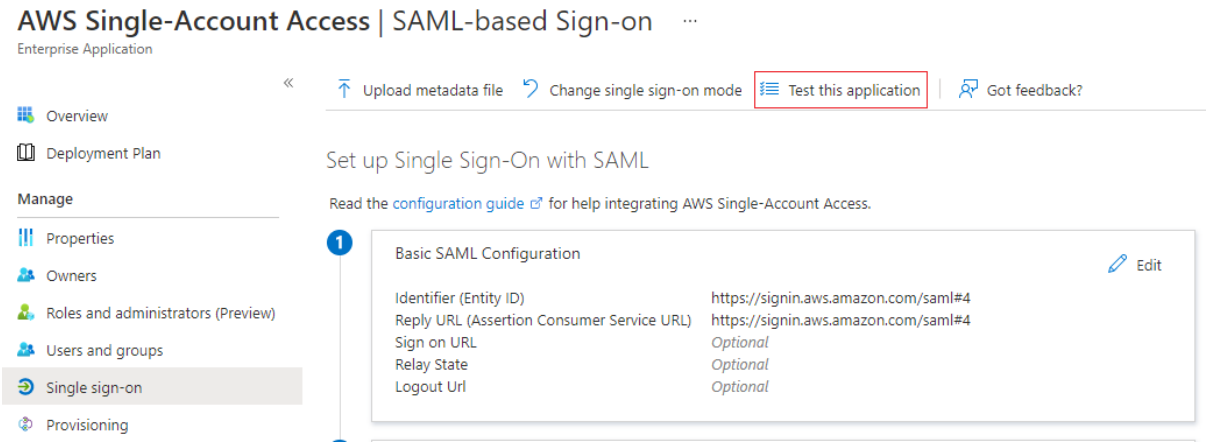
12.2 In **Enterprise Applications**, select the **AWS Single-Account Access** application you have created in step 3.4 earlier in this document.



12.3 On the application page, select **Single sign-on.**

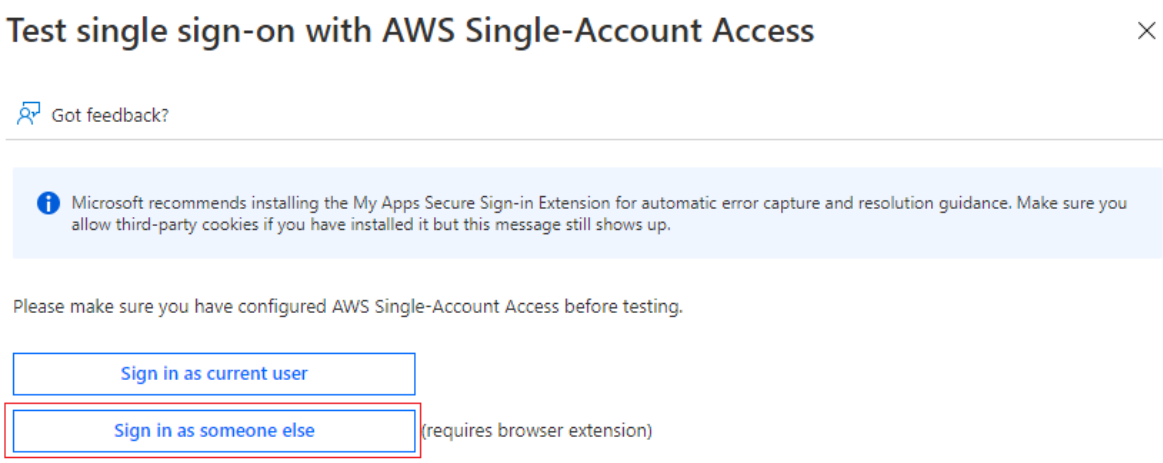


12.4 On the **Set up Single Sign-On with SAML** page, click on **Test this application**.

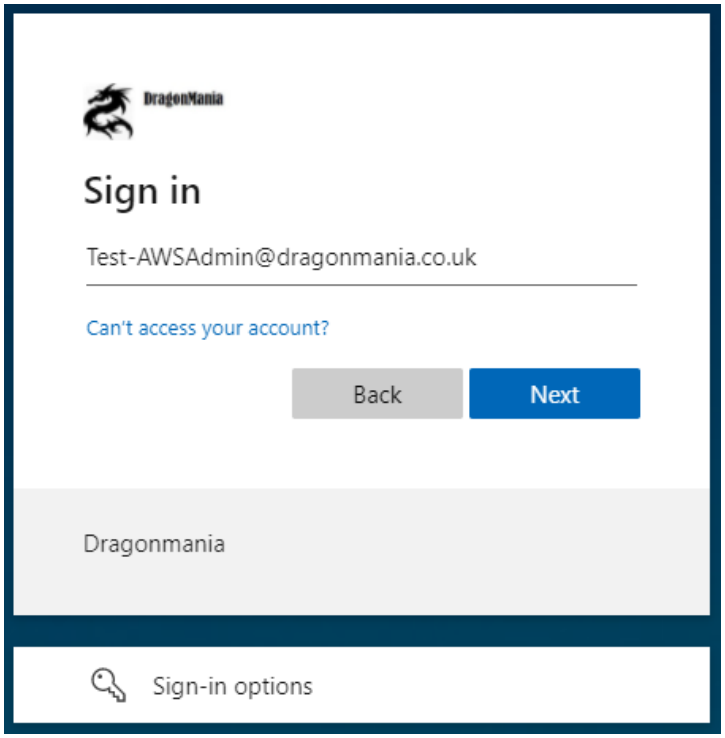


12.5 On the **Test single sign-on with AWS Single-Account Access** page, select **Sign in as someone else**.

***If you don't have the **My Apps Secure Sign-in Extension** installed, click on the prompt below to install it.

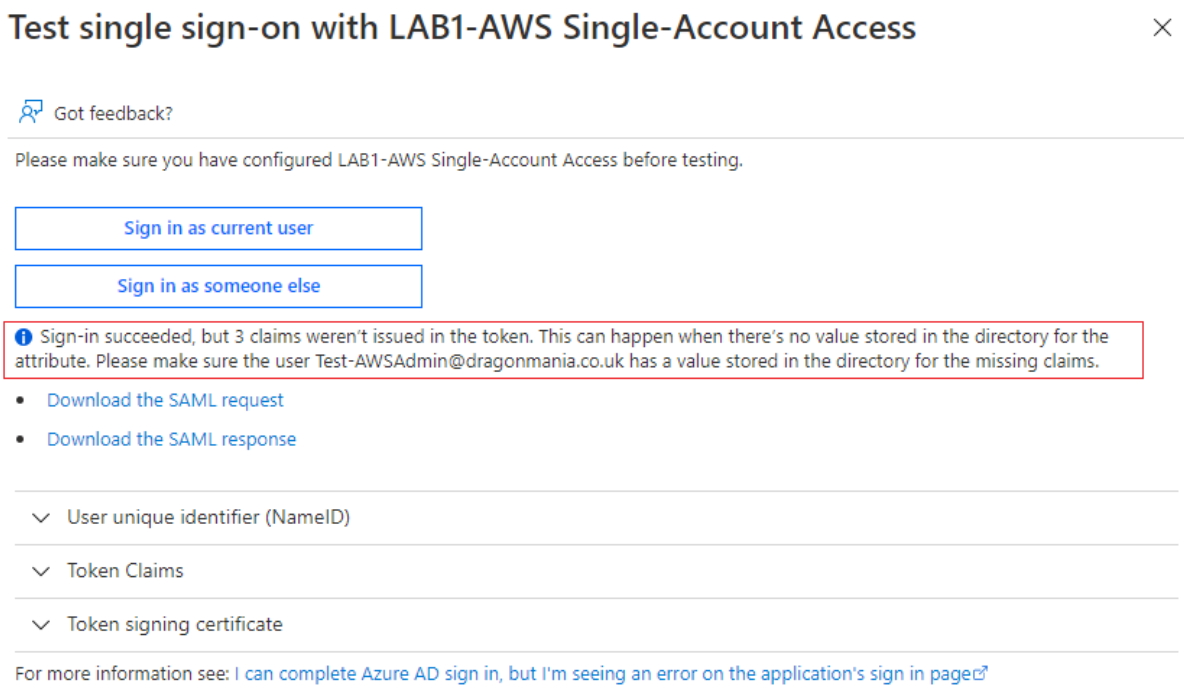


12.6 Sign in with the **Test-AWSAdmin** account you have created in section 9. If it's the first time you are signing in with this account, you will be prompted to change the password (and depending on your security settings to register for MFA). **Make a note of the new password**.

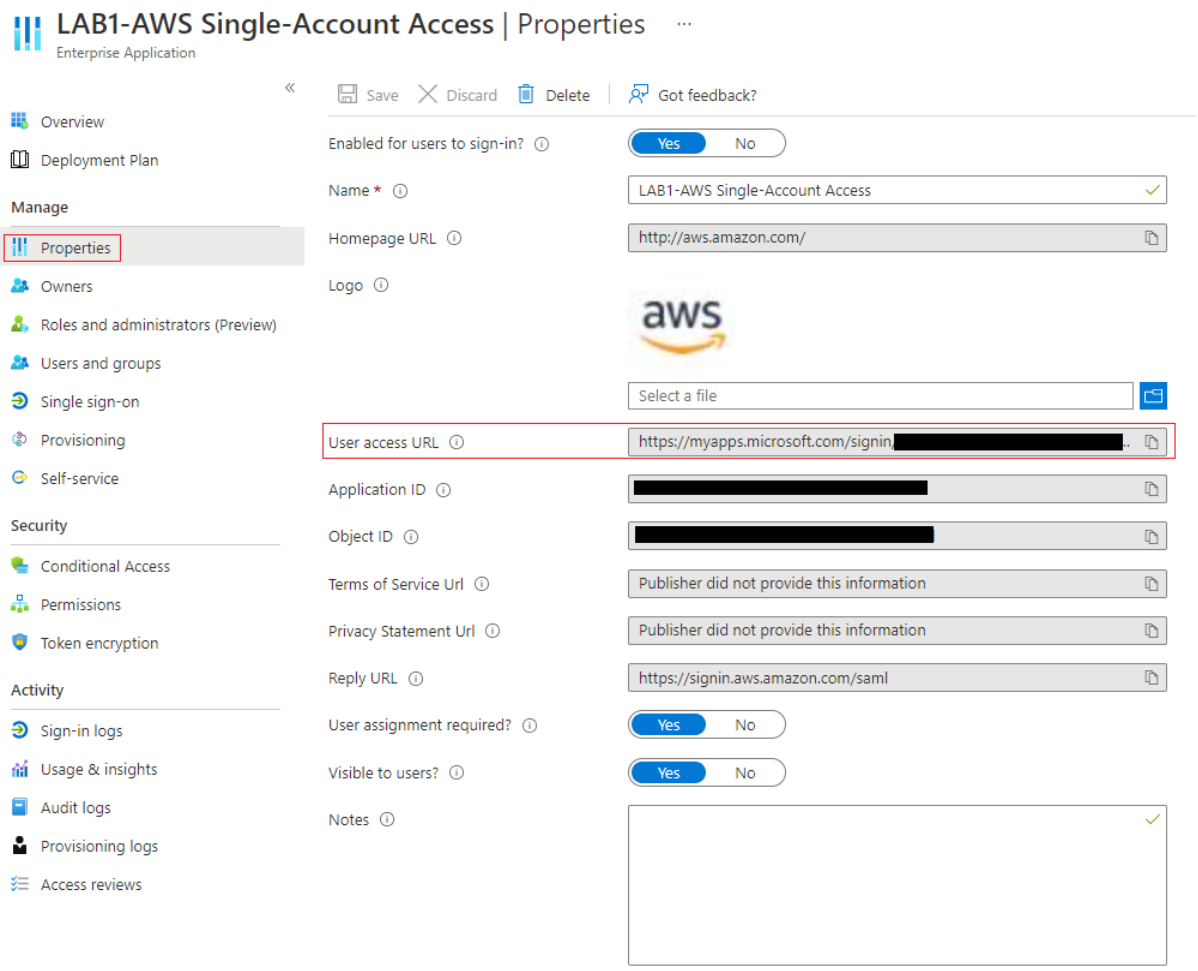


12.7 You will receive the a response as shown below that your sign in was successful. Any errors will be displayed in this page.

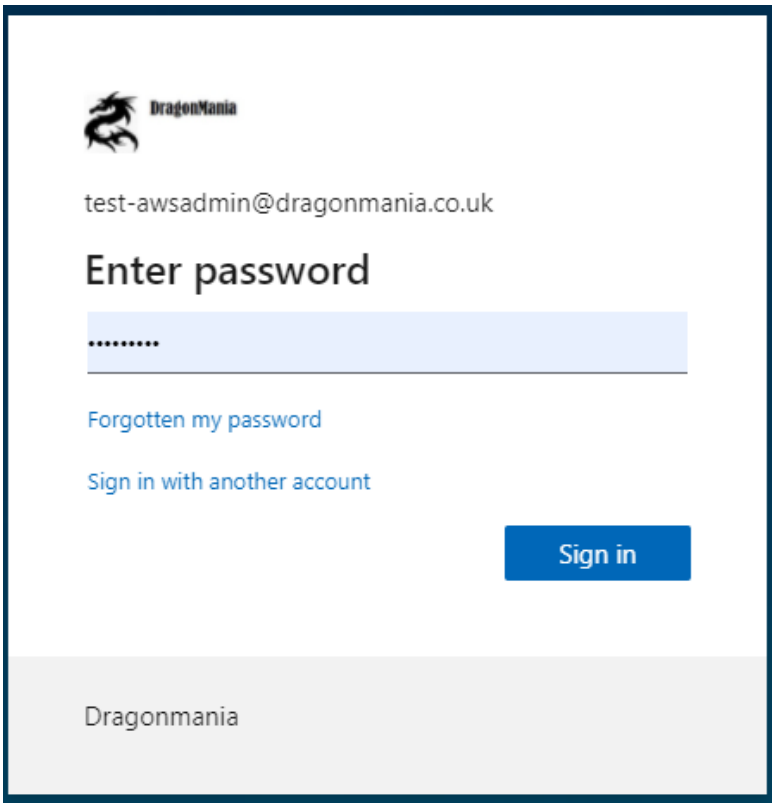
*If you haven't given the user account a first name, last name and an email address, then you may see a message such as "**Sign in succeed, but 3 claims weren't issued in the token**". This is because these attributes are listed in **Single Sign-on -> User Attributes & Claims**.



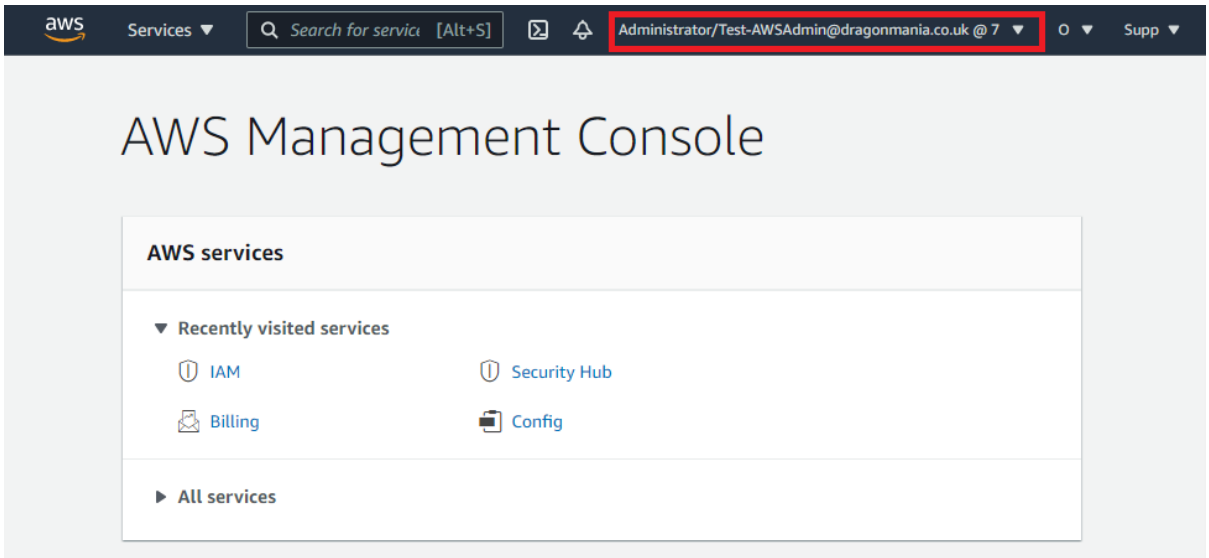
12.8 Alternatively, you can click on **Properties** and copy the **User Access URL** as shown below and paste it in another browser window.



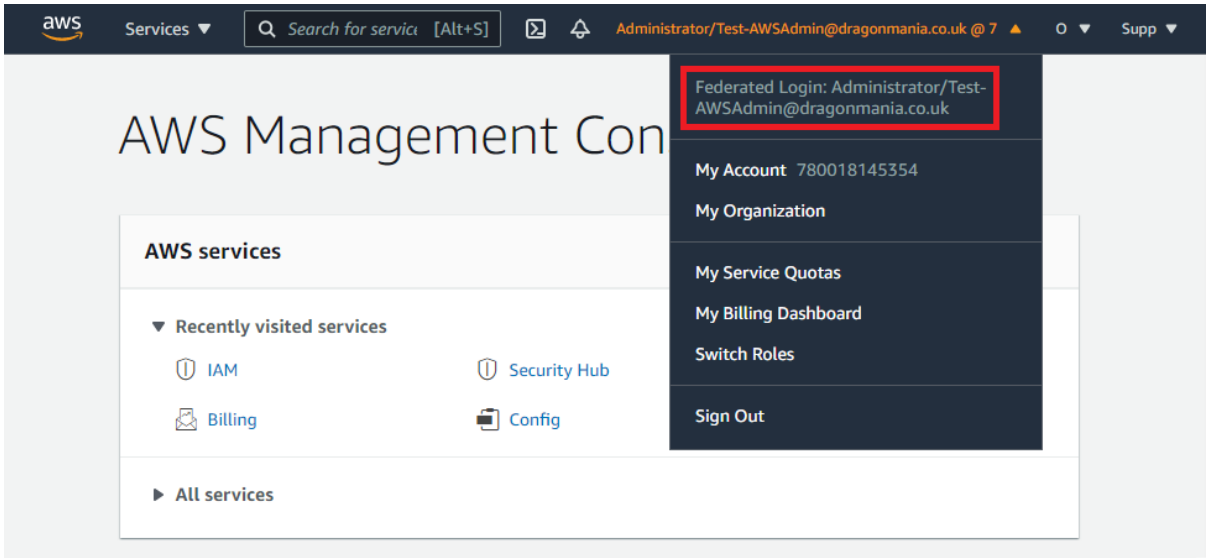
12.9 You will be prompted to sign-in, select the **Test-AWSAdmin** account you created earlier.



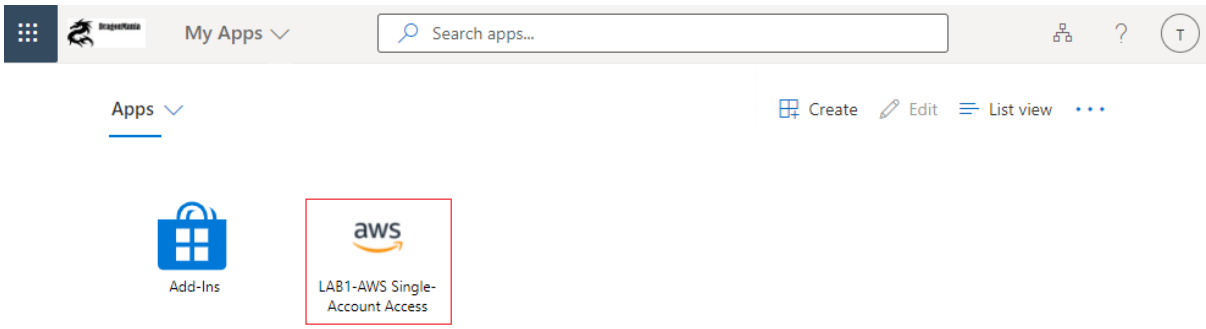
12.10 Once the sign-in is successful, you will be redirected to AWS Console as an Administrator.



12.11 If you select the arrow next to the user name to expand it, you will see the user is a federated user. This account is a full admin in AWS according to the role assignments configured earlier in this document, so play around and see what you can do (for example, create a new user, create a S3 bucket, etc.)



12.12 Another way of access the application is by going to **myapplications.microsoft.com** and signing in with the **Test-AWSAdmin** account, then select your AWS application from the list as shown below.



12.13 Make sure you also test with the Test-**AWSDeveloper** account you created and test the permissions you have assigned to the **Developer** role.

Congratulations! You have now successfully completed this lab. We hope you found this lab, and the associated lab materials useful. We look forward to seeing what you build as a result of attending this lab!

Lab Complete.