polaris

Polaris Authentication Integration Guide

8.1

innovative
Part of Clarivate

Innovative Interfaces, Inc.
789 E. Eisenhower Parkway
Ann Arbor, MI 48108
contract.admin@clarivate.com

Legal Notices

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

Contents

Introduction

Both Leap and Polaris System Administration (web-based) support the following types of authentication:

- Open ID Connect (OIDC) — OIDC is the identity layer that sits on top of OAuth 2.0. You can use Proof Key for Code Exchange (PKCE) for security with this configuration.
- Basic OAuth 2.0 — This configuration doesn't use OIDC or PKCE. In addition, basic OAuth 2.0 doesn't rely on claims found within a JSON web token (JWT). Instead, it exchanges an access token (typically, an opaque token) to retrieve claims from the OAuth /userinfo endpoint.
- Basic authentication — This configuration uses the Windows operating system's local security and/or its direct integration with Active Directory.

For detailed information about the types of authentication that are supported for each application and software version, see Minimum Requirements.

Staff authentication for each application is handled by an identity provider. We provide detailed instructions for configuring OIDC with Active Directory and Active Directory Federation Services (AD FS) or with Azure Active Directory (Azure AD). You can also use a different identity provider.

> **Note:**
> Azure AD has recently been rebranded as "Microsoft Entra ID". In this guide, references to Azure AD are interchangeable with Microsoft Entra ID.

To use this guide, review the Minimum Requirements for the configurations you plan to use:

- OIDC configurations:
  - OIDC with Active Directory and AD FS Authentication
  - OIDC with Azure AD Authentication
- Basic OAuth 2.0
- Basic Authentication

Then, continue to one of the following configuration procedures:

Minimum Requirements

This section discusses minimum requirements for the following types of configurations:

- OIDC configurations:
  - OIDC with Active Directory and AD FS Authentication
  - OIDC with Azure AZ Authentication
- Basic OAuth 2.0
- Basic Authentication

OIDC with Active Directory and AD FS Authentication

To use OIDC OAuth 2.0 with Azure AD (Microsoft Entra ID) authentication, you must have:

- Windows Server 2019 Standard
  - Polaris requires OAuth 2.0 with PKCE support
  - AD FS on Windows Server 2019 supports PKCE
- Active Directory Domain Services
- SSL Certificate
  - Publicly trusted CA signed certificate
- The Polaris 7.2 or later LeapWebApp installed
- The Polaris 7.1 or later PolarisAdmin installed

OIDC with Azure AD Authentication

To use OIDC OAuth 2.0 with Azure AD authentication, you must have:

- Access to Microsoft's Azure AD services
- The Polaris 7.3 or later LeapWebApp installed
- The Polaris 7.3 or later PolarisAdmin installed

Basic OAuth 2.0

To use basic OAuth 2.0, you must have:

- An identity provider configured for basic OAuth 2.0
- SSL Certificate
    - Publicly trusted CA signed certificate
- The Polaris 7.5 or later LeapWebApp installed
- The Polaris 7.6 or later PolarisAdmin installed

Basic Authentication

To use basic authentication, you must have the Polaris 7.7 or later PolarisAdmin installed.

Leap with Active Directory and AD FS Authentication

To use Leap with OIDC OAuth 2.0 and Active Directory and AD FS authentication, you must have the following installed:

- Windows Server 2019 Standard
    - Polaris requires OAuth 2.0 with PKCE support
    - AD FS on Windows Server 2019 supports PKCE
- Active Directory Domain Services
- SSL Certificate
    - Publicly trusted CA signed certificate
- Polaris 7.2 or later
- Polaris 7.2 or later LeapWebApp

Leap with Azure AD Authentication

To use Leap with OIDC OAuth 2.0 and Azure AD authentication, you must have:

- Access to Microsoft's Azure AD services
- The Polaris 7.3 or later LeapWebApp installed

Polaris System Administration (Web-Based) with Active Directory and AD FS Authentication

To use Polaris System Administration (web-based) with OIDC OAuth 2.0 and Active Directory and AD FS authentication, you must have the following installed:

- Windows Server 2019 Standard
    - Polaris requires OAuth 2.0 with PKCE support
    - AD FS on Windows Server 2019 supports PKCE
- Active Directory Domain Services
- SSL Certificate
    - Publicly trusted CA signed certificate
- Polaris 7.2 or later LeapWebApp
- Polaris 7.1 or later PolarisAdmin

Polaris System Administration (Web-Based) with Azure AD Authentication

To use Polaris System Administration (web-based) with OIDC OAuth 2.0 and Azure AD authentication, you must have:

- Access to Microsoft's Azure AD services
- The Polaris 7.3 or later PolarisAdmin installed

Leap with Basic OAuth 2.0

To use Leap with basic OAuth 2.0, you must have:

- An identity provider configured for basic OAuth 2.0
- SSL Certificate
    - Publicly trusted CA signed certificate
- The Polaris 7.5 or later LeapWebApp installed

Polaris System Administration (Web-Based) with Basic OAuth 2.0

To use Polaris System Administration (web-based) with basic OAuth 2.0, you must have:

- An identity provider configured for basic OAuth 2.0
- SSL Certificate
    - Publicly trusted CA signed certificate
- The Polaris 7.6 or later PolarisAdmin installed

Leap with Basic Authentication

To use Leap with basic authentication, you must have:

- The Polaris 5.0 or later LeapWebApp installed

Polaris System Administration (Web-Based) with Basic Authentication

To use Polaris System Administration (web-based) with basic authentication, you must have:

- The Polaris 7.7 or later PolarisAdmin installed

Upgrading to New Versions of Polaris

Upgrading to 7.7

In previous versions, you had to configure authentication separately for Leap and Polaris System Administration (web-based). Leap supported basic authentication, OIDC, and basic OAuth 2.0. Polaris System Administration (web-based) only supported OIDC and basic OAuth 2.0.

Version 7.7 includes a new centralized authentication service that's designed to be used with both Leap and Polaris System Administration (web-based).

This means that if your library currently uses basic authentication for Leap, you can now begin using Polaris System Administration (web-based) with basic authentication. For more information, see Configuring Basic Authentication.

If your library currently uses OIDC or basic OAuth 2.0, you can continue using it for authentication for both Leap and Polaris System Administration (web-based). However, when you upgrade to version 7.7, you must edit your configuration so that it uses the new centralized authentication service. To do this, you must configure a .json file for each of the following:

- Polaris.Authentication (the application that authenticates Polaris users)
- Polaris.AuthenticationServices (the API service that provides backend support for authentication)

For more information, see the appropriate instructions for your configuration:

- Set Up Web Services and Applications for OIDC with Active Directory and AD FS
- Set Up Web Services and Applications for OIDC with Azure AD
- Set Up Web Services and Applications for Basic OAuth 2.0

> **Important:**
> If your library currently uses single sign-on (SSO) authentication with Leap, you must update the allowed redirect URLs that are specified in your identity provider configuration.

Regardless of your configuration or identity provider, in version 7.7 you must also configure the following two lines in the appsettings.user.json file for LeapWebApp:

```
"Polaris.AuthAPI.URL": "https://[server-address]/Polaris.AuthenticationServices/",
"Polaris.AuthApp.URL": "https://[server-address]/polarisauth/",
```

Replace appsettings entries to match configured Polaris.Authentication and Polaris.AuthenticationServices URLs.

> **Note:**
> If you are upgrading to version 7.7 from a previous version, find the version 7.7 appsettings.user.json template file for LeapWebApp, copy these two lines, and paste them into your appsettings.user.json file.

You must perform additional configuration if one or both of the following is true for your library:

- You want to enable permission overrides in Leap.
- You want to enable reauthentication for Leap.

> **Note:**
> Permission overrides and reauthentication are not supported if your system uses multiple identity providers for authentication OR if the configured single authority is configured for Basic (Non-OIDC) flow.

If one or both of the above are true, refer to the instructions for your configuration in the 7.5 version of the Polaris and OAuth 2.0 with OpenID Connect Integration Guide:

- In the Configuring Active Directory with AD FS section, see the Set Up LeapWebApp instructions.
- In the Configuring Azure AD section, see the Configure LeapWebApp for Use with Azure AD instructions.

Upgrading to 7.8

Version 7.8 introduces new settings for authentication, session management, and Identity Provider (IDP) integration.

Cookie lifetime configuration is available through the following lines:

- **Polaris:AuthCookieLifetimeInMinutes**
    - Determines how long the authentication cookie remains valid before requiring the user to log in again.
    - Default: 1440 minutes (1 day; defaults to this value if set to 0)

    Note:
    When used for LeapWebApp, do not set this value higher than 1440. The PAS token associated with this cookie never lasts longer than one day.

- **Polaris:SessionIdleTimeoutInMinutes**
    - Determines how long a user session can remain idle before timing out.
    - Default: 1440 minutes (1 day; defaults to this value if set to 0)

When using an IDP for login and logout, additional options are available:

- **Polaris:OAuth:ForceIDPLogin**
    - When set to **false** (default), allows users with an active IDP session to log in to Polaris automatically. If set to **true**, it will force the user to provide IDP credentials every time when logging in.
- **Polaris:OAuth:ForceIDPLogout**
    - When set to **true** (default), logs the user out of their active IDP session when logging out of Polaris. If set to **false**, users are logged out of Polaris only, deleting the authentication cookie without affecting the IDP session.

Upgrading to 8.0

Version 8.0 removes Polaris.ApplicationServices OAuth server configuration from appsettings and delegates bearer token authentication to Polaris.AuthenticationServices.

In the Polaris.ApplicationServices appsettings file you will find the **AuthAPI** section:

```
"AuthAPI": {
   "Url": "[polaris_auth_api_base_url]",
   "OAuthEnabled": false,
   "DefaultOAuthAuthorityHeader": null
},
```

To enable bearer token validation, you need to populate the above section:

- "**AuthAPI.Url**" : add AuthService instance URL here.
- "**AuthAPI.OAuthEnabled**" : set to **true**.
- "**DefaultOAuthAuthorityHeader**" : leave as **null**.

Supported Identity Providers

These are the identity providers (IdPs) that are officially supported for single sign-on (SSO) with Polaris:

- Active Directory Federation Services (AD FS)
- Microsoft Entra ID (formerly Azure Active Directory)
- Okta Identity Cloud
- Google Cloud Identity

Using Multiple Identity Providers

As of version 7.7, if your system uses OIDC or basic OAuth 2.0, organizations in your system can use different identity providers to sign in to a single instance of Leap or Polaris Administration (web-based).

For example, a consortium system with three member libraries could use the following authentication configurations:

- Library A — OAuth 2.0 with Active Directory and Active Directory Federation Services (AD FS)
- Library B — OAuth 2.0 with Azure Active Directory
- Library C — Basic OAuth 2.0 with Akamai

Configuring OIDC with Active Directory and AD FS

> **Important:**
> The mechanism used to connect an Active Directory user to a Polaris user is the user principal name (UPN) in the format of an email address. For example, user@mydomain.com. During the account verification process, we request the UPN claim from Active Directory. This must return a UPN in the name@domain format. The Polaris.AdminServices (API) can then use that information to map the AD user to a Polaris user.

To configure OIDC with Active Directory and AD FS, perform the following tasks:

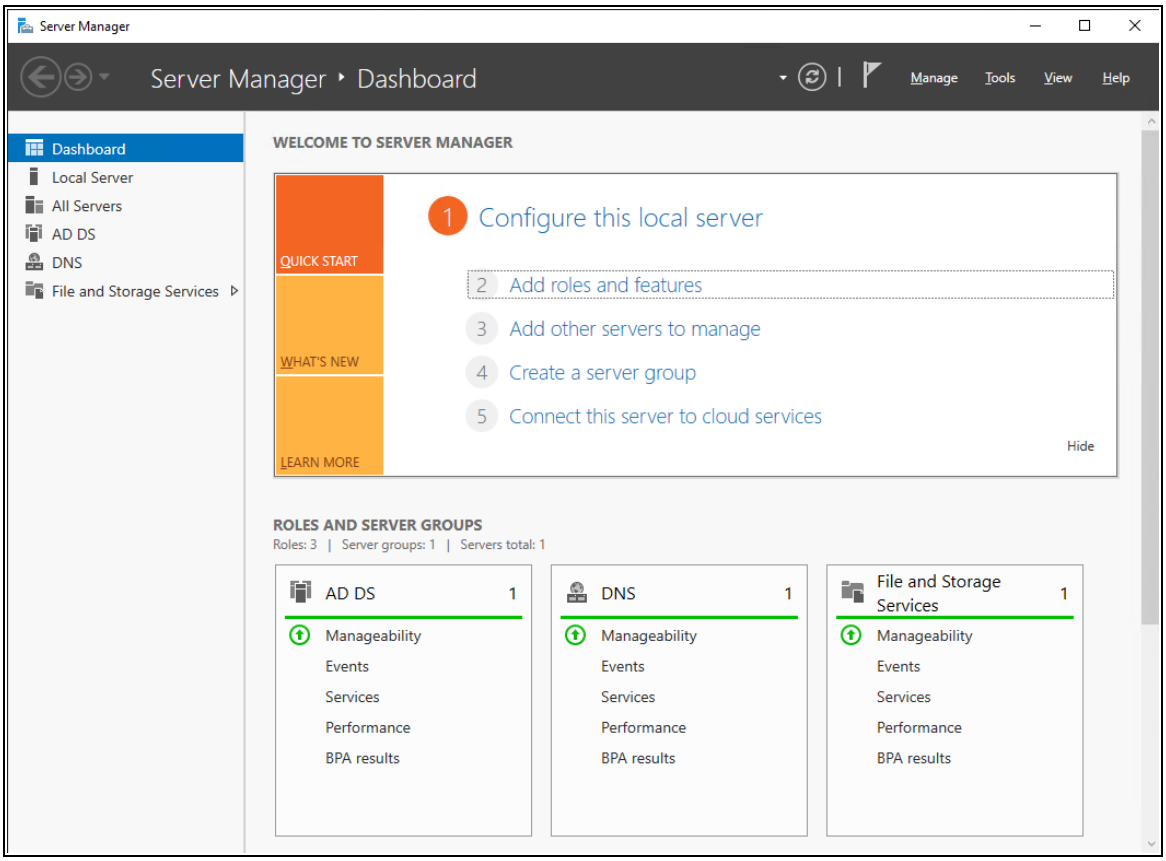
1. Install Active Directory Federation Services.
2. Configure Active Directory Federation Services.
3. Verify that Active Directory Federation Services is running.
4. Verify that OAuth 2.0 is Enabled.
5. Create an Application Group for Polaris LeapWebApp.
6. Configure the AD FS Web Application: Claims and Permitted Scopes.
7. Enable CORS on AD FS to accept requests from Polaris APIs.
8. Set up web services and applications.
9. Customize the AD FS pages.
10. Change the access token lifetime.
11. Bind a new SSL certificate.
12. Troubleshoot.

After you complete these tasks, Add a URL rewrite rule for LeapWebApp.

Install Active Directory Federation Services

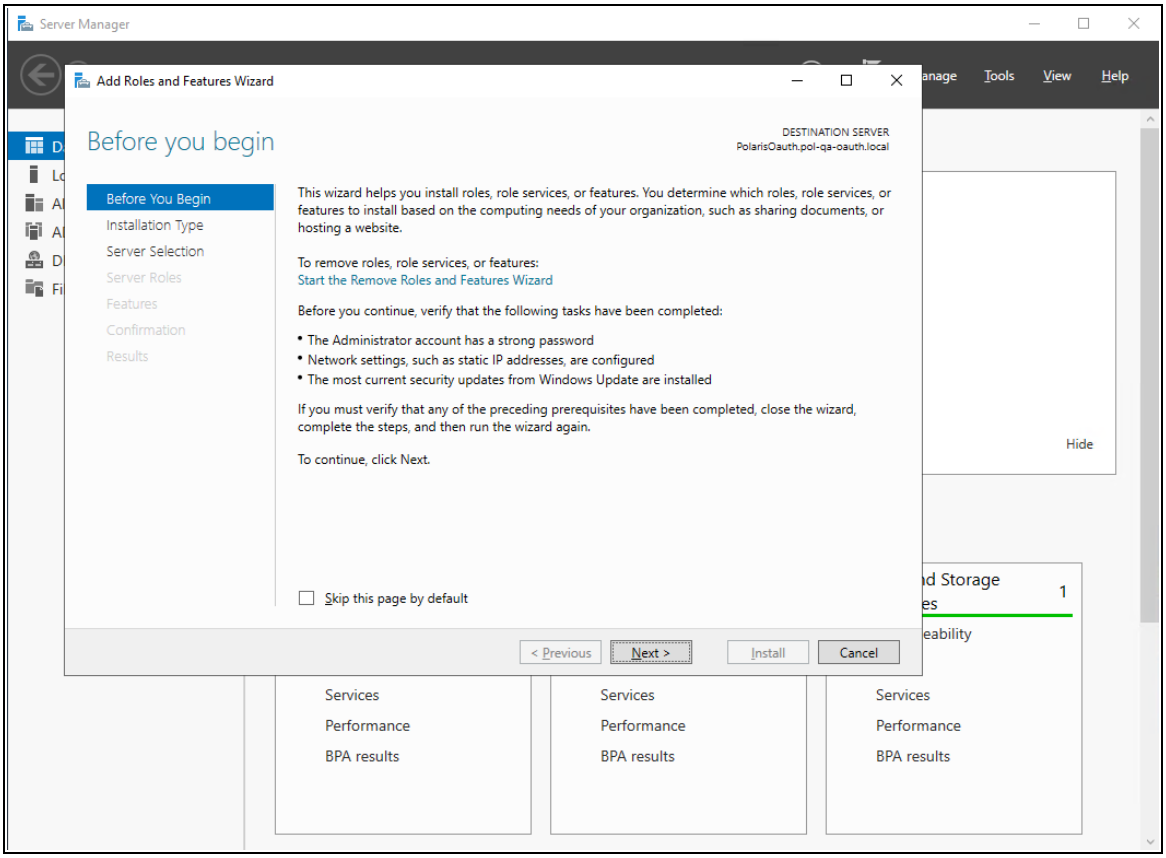
To install AD FS

1. Sign in to Windows Server 2019 with administrative privileges.
2. Start the Server Manager desktop application.

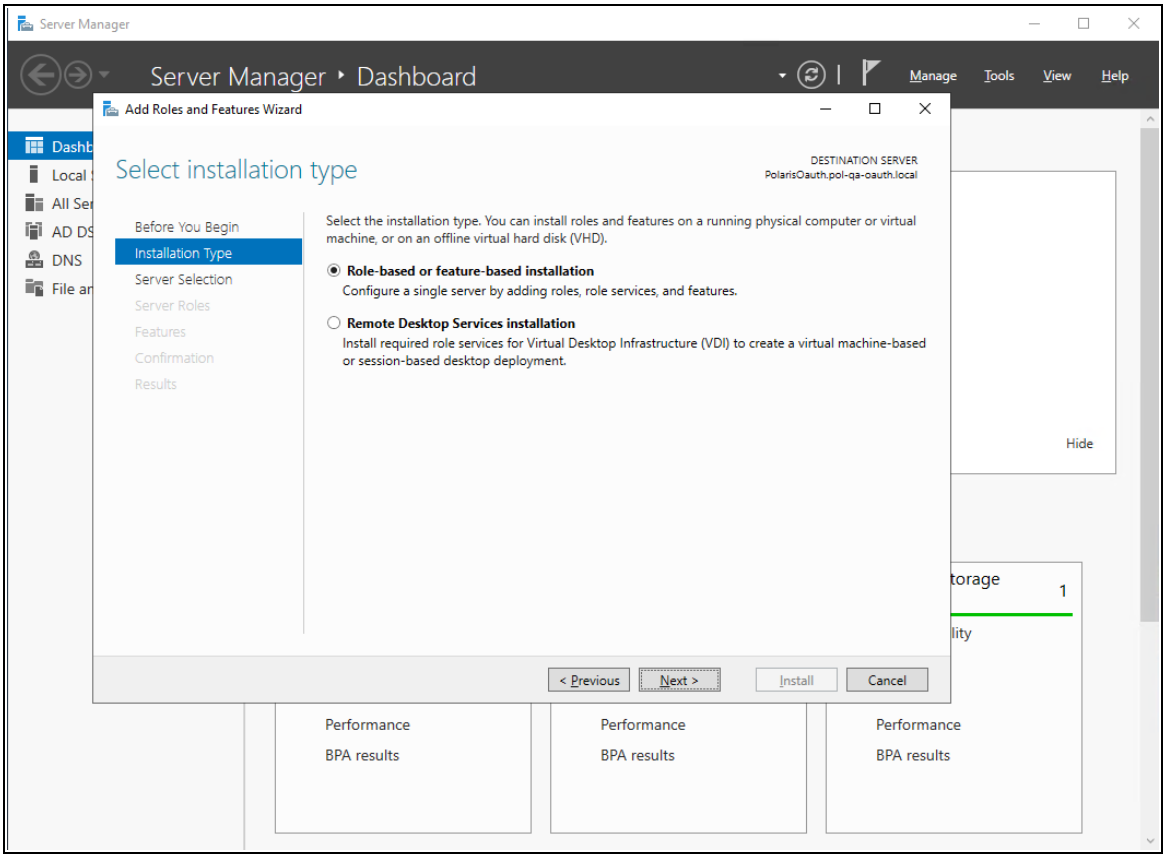


3. On the **Server Manager Dashboard** view, select **Add roles and features**.

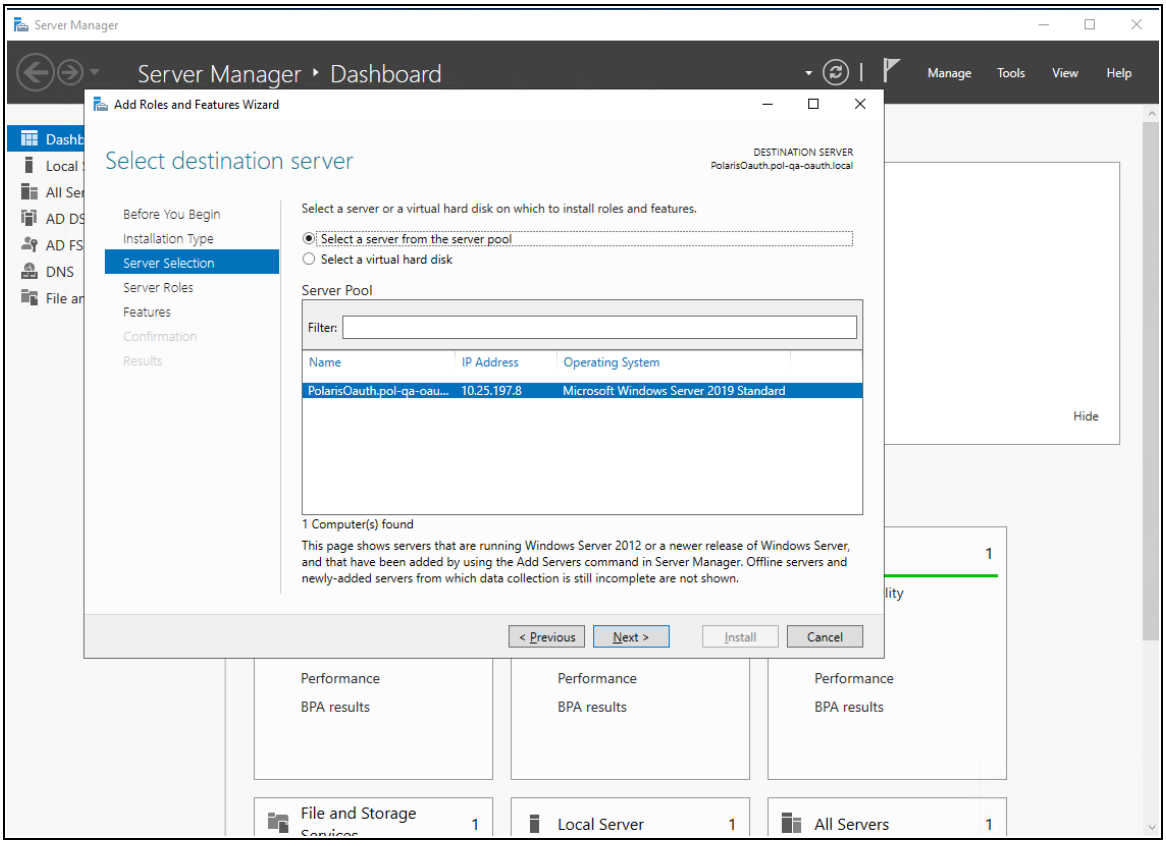
   The Add Roles and Features Wizard opens.



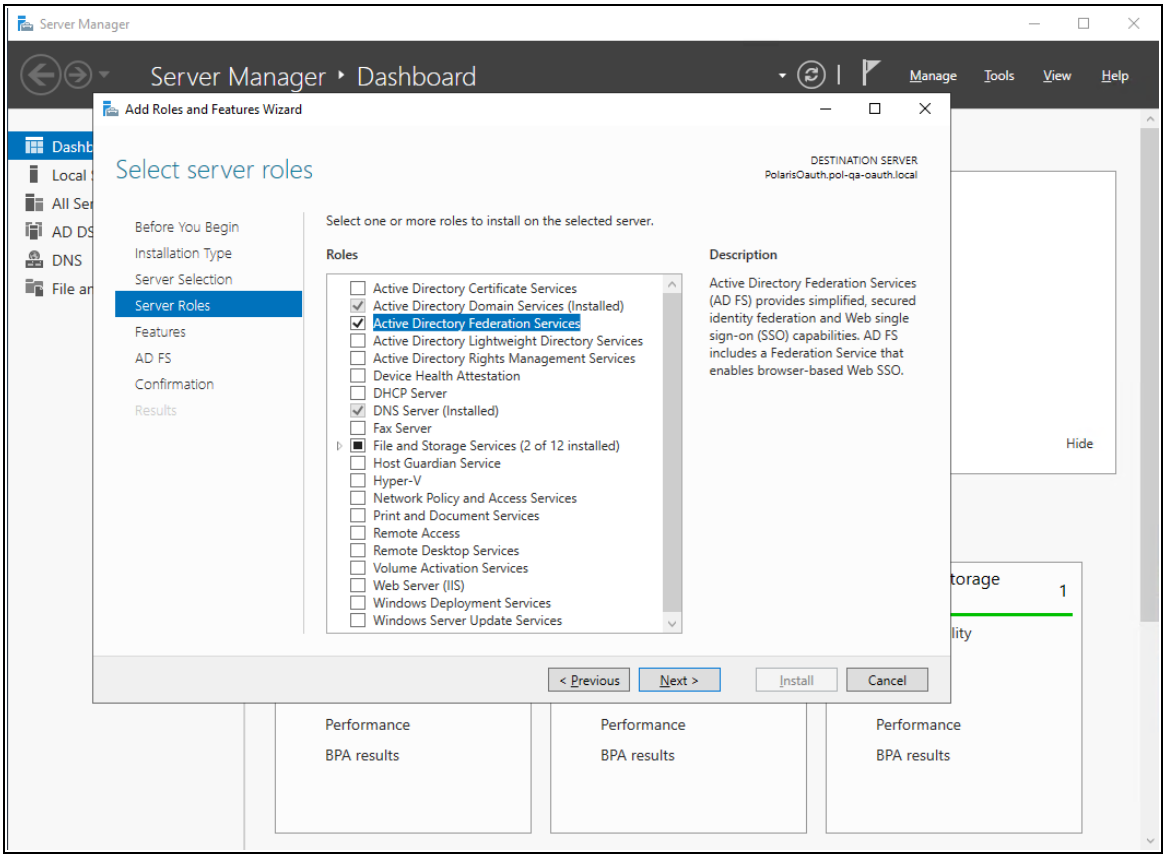
4. On the **Before You Begin** tab, select **Next**.



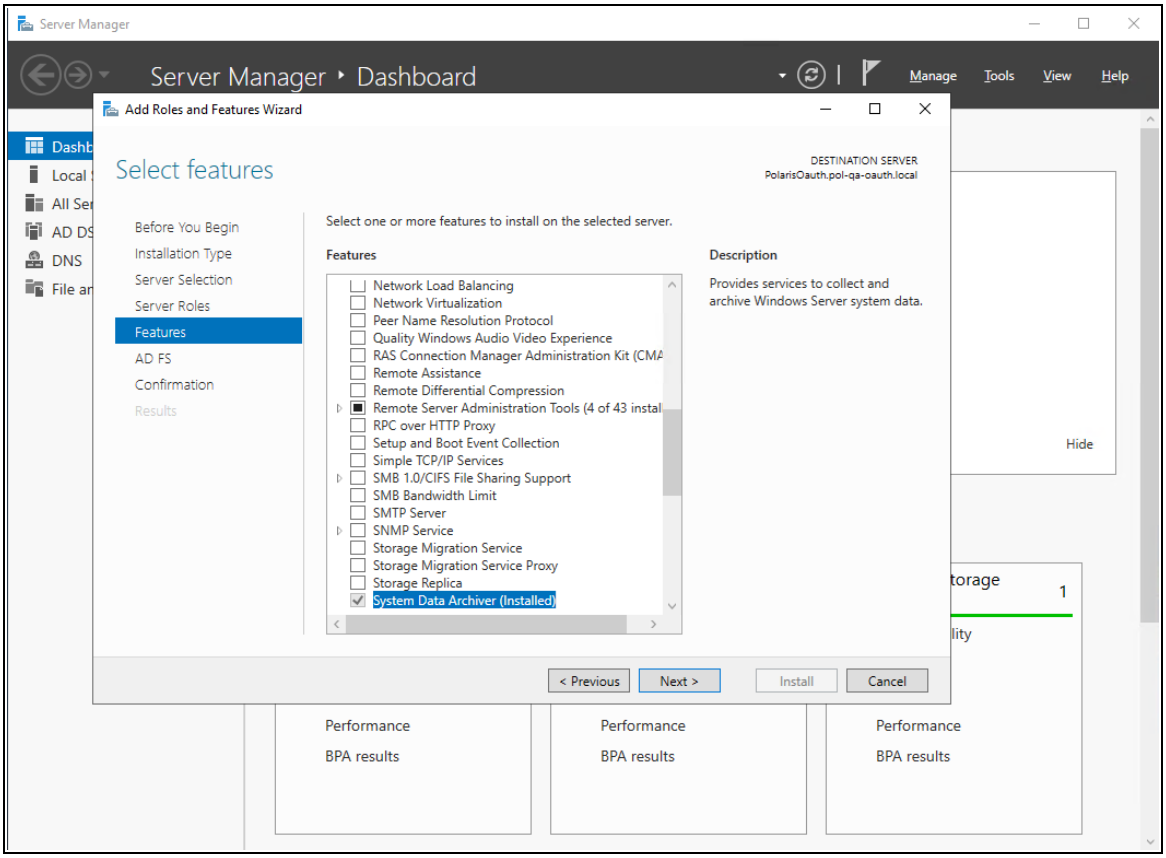
5. On the **Installation Type** tab, select **Role-based or feature-based installation**, and then select **Next**.



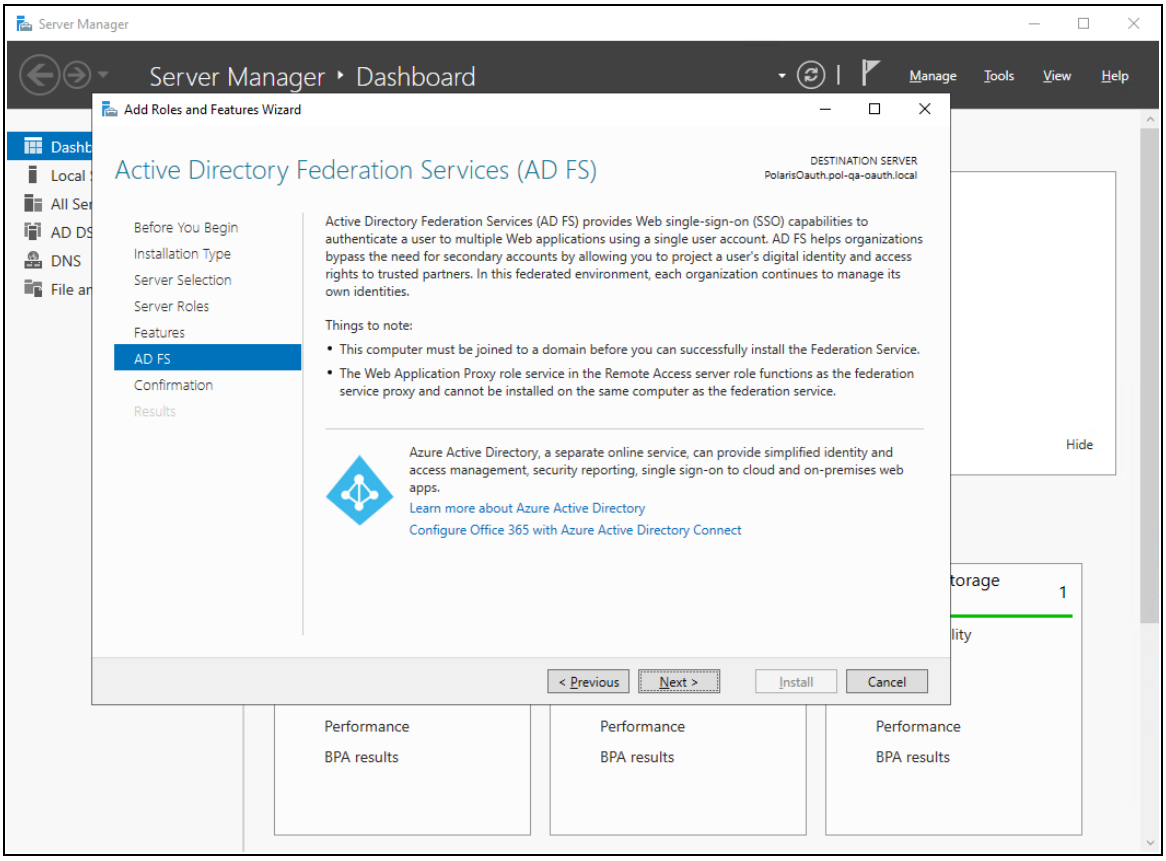
6. On the **Server Selection** tab, select the server, and then select **Next**.



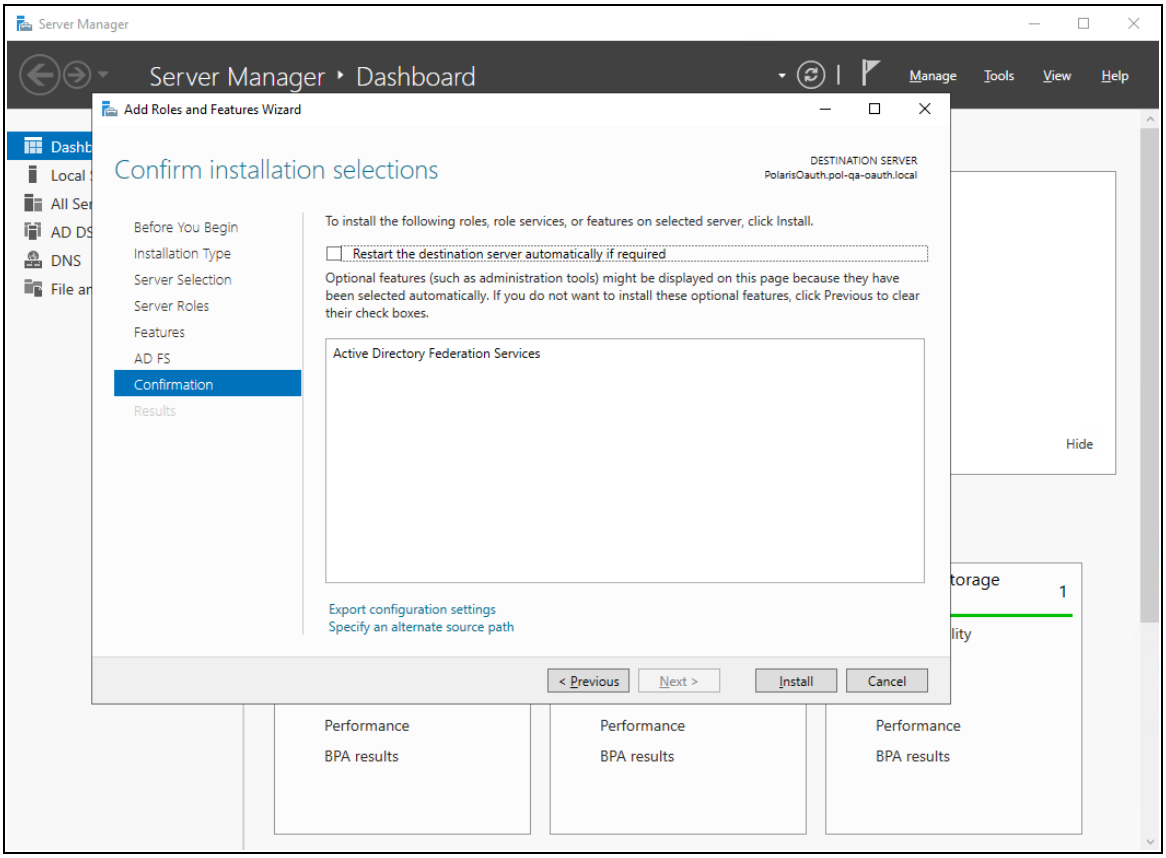
7. On the **Server Roles** tab, do the following:
    a. Verify that **Active Directory Domain Services** are installed.
    b. Select the **Active Directory Federation Services** role.
    c. Select **Next**.



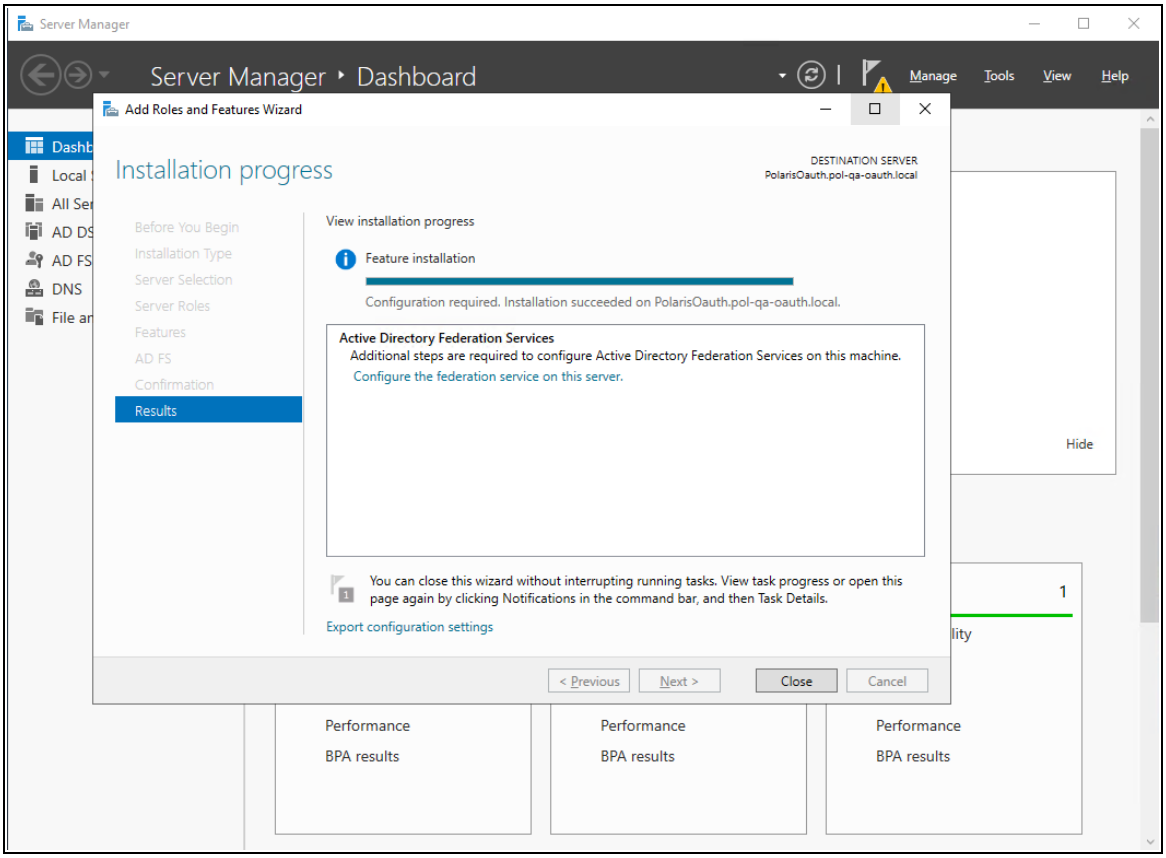
8. On the **Features** tab, select **Next**.



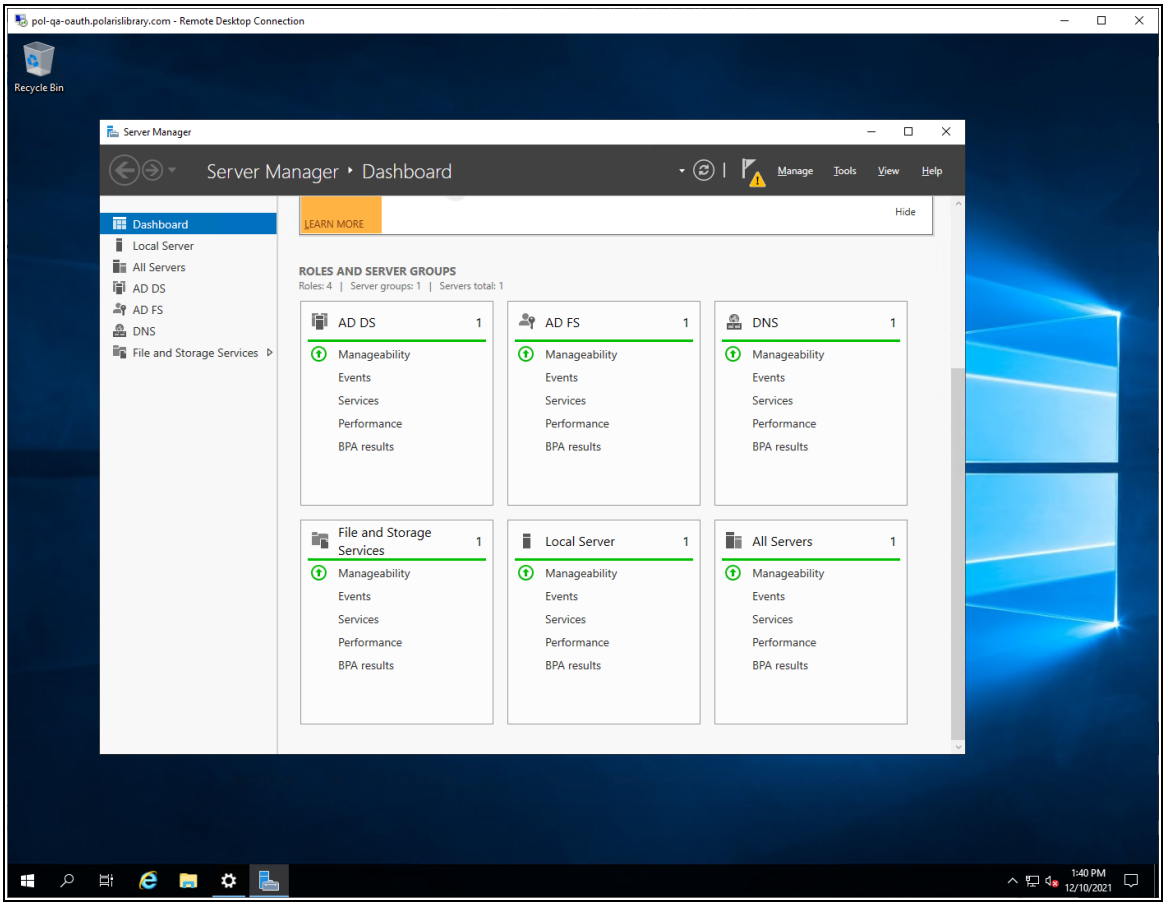
9. On the **AD FS** tab, read the Active Directory Federation Services (AD FS) information, and then select **Next**.



10. On the **Confirmation** tab, confirm your selections, and then select **Install**.



11. On the **Results** tab, select **Close** when the installation is complete.



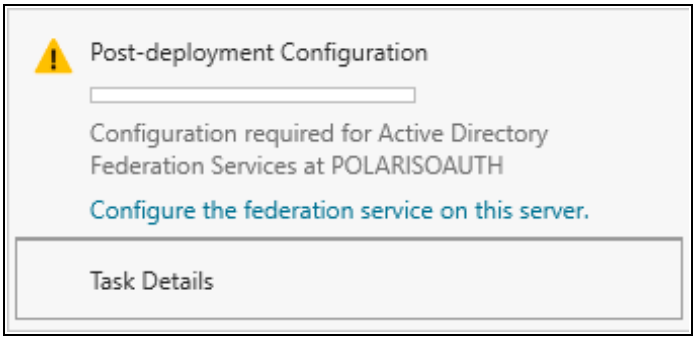
12. On the Server Manager dashboard, verify that AD FS is an installed role.
13. Restart the server.

Configure Active Directory Federation Services

To configure Active Directory Federation Services

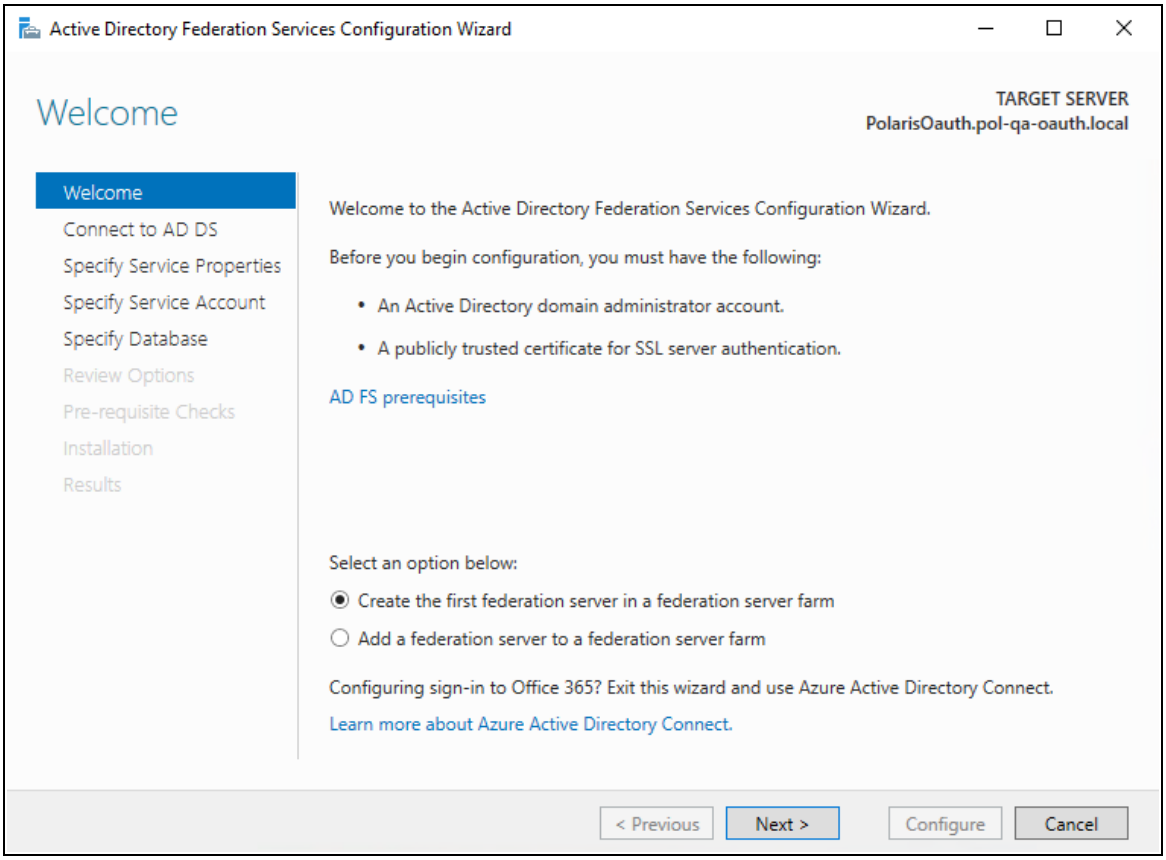
1. Start the Server Manager desktop application.

   The system generates a configuration notification.

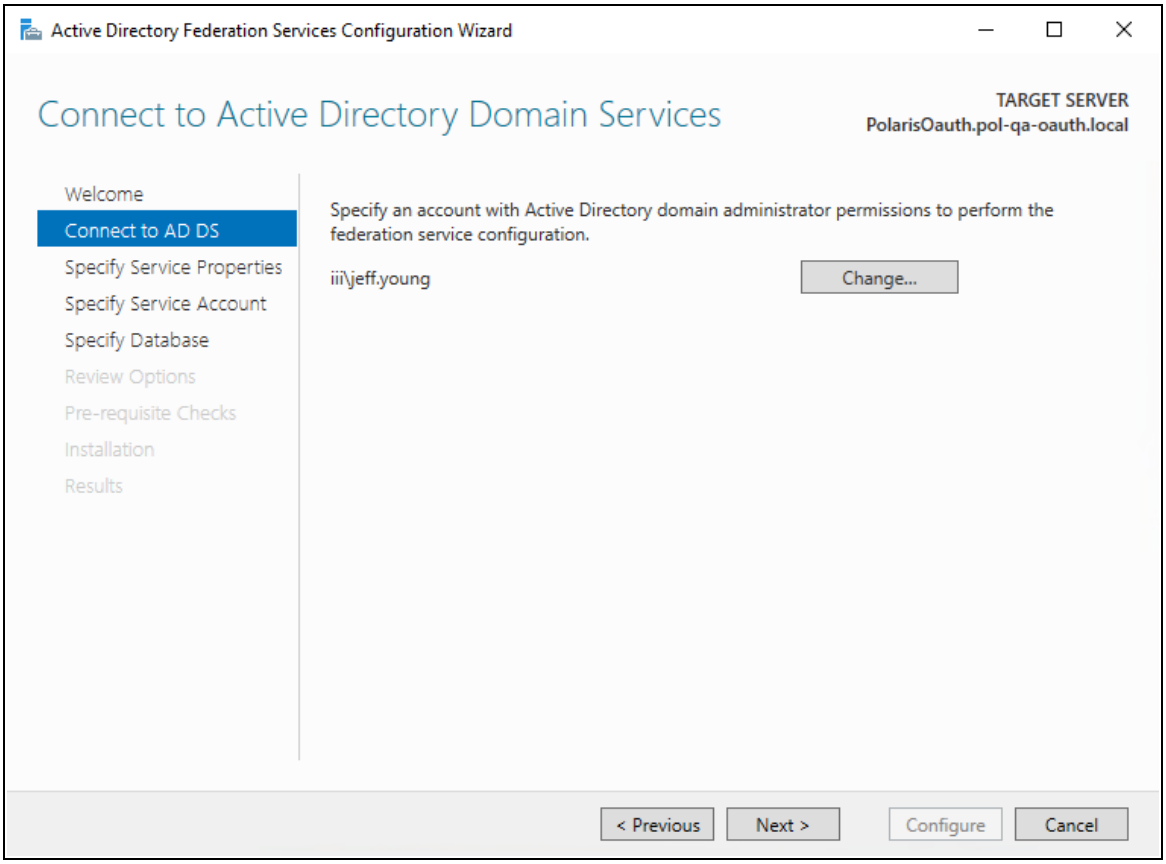


2. Open the notification, and select **Configure the federation service on this server**.

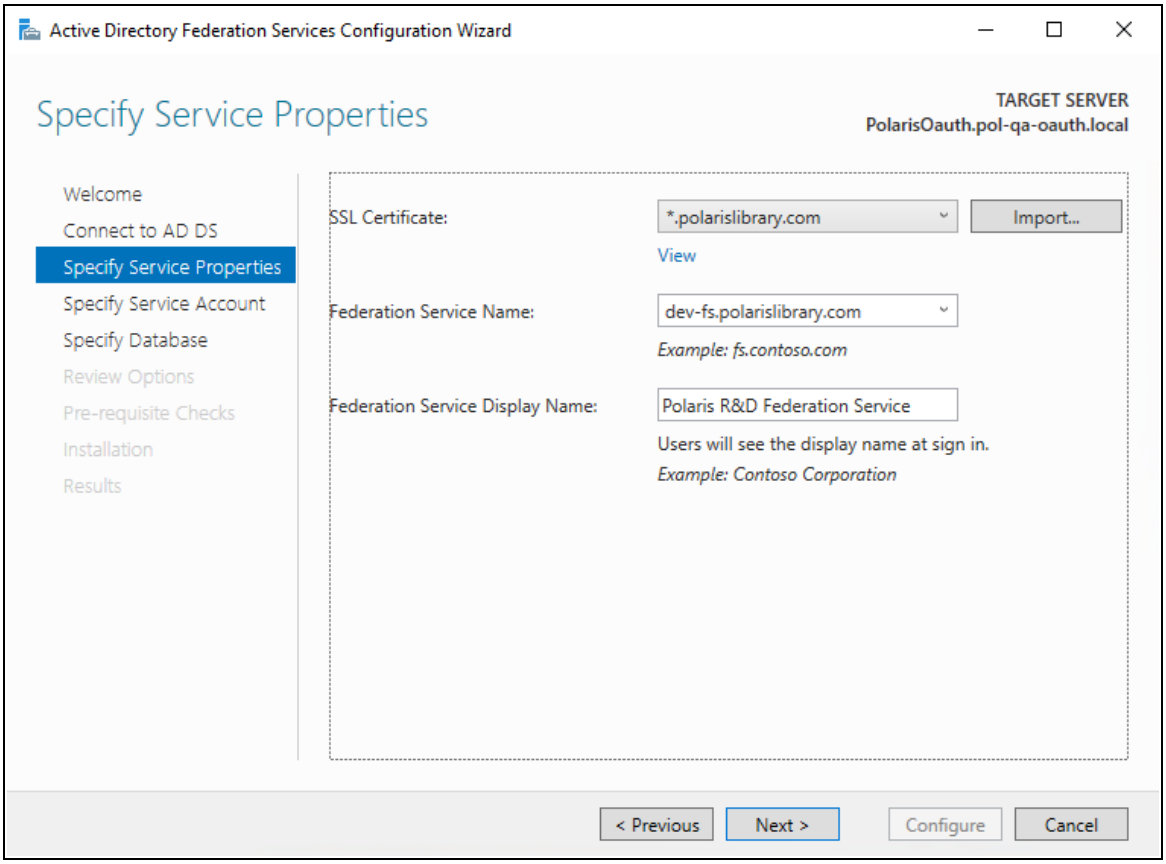
   The Active Directory Federation Services Configuration wizard opens.



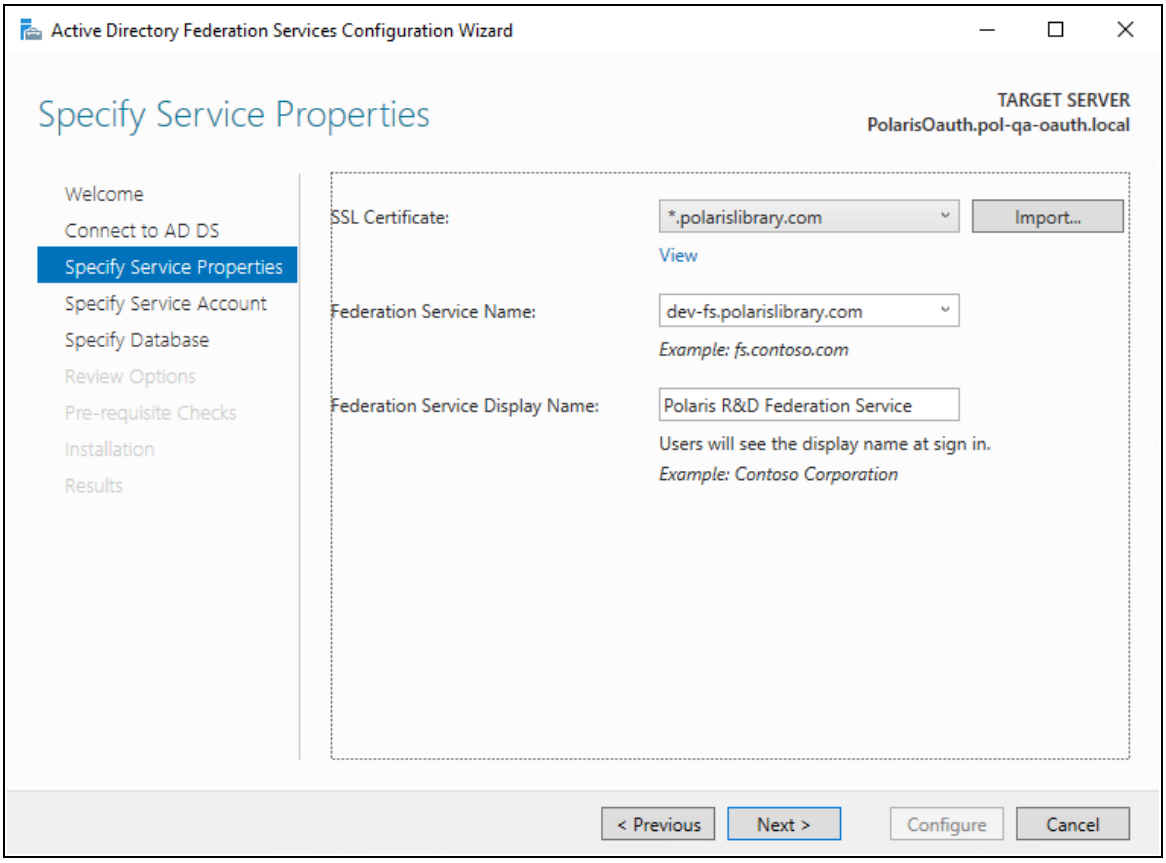
3. On the Welcome tab, select **Next**.



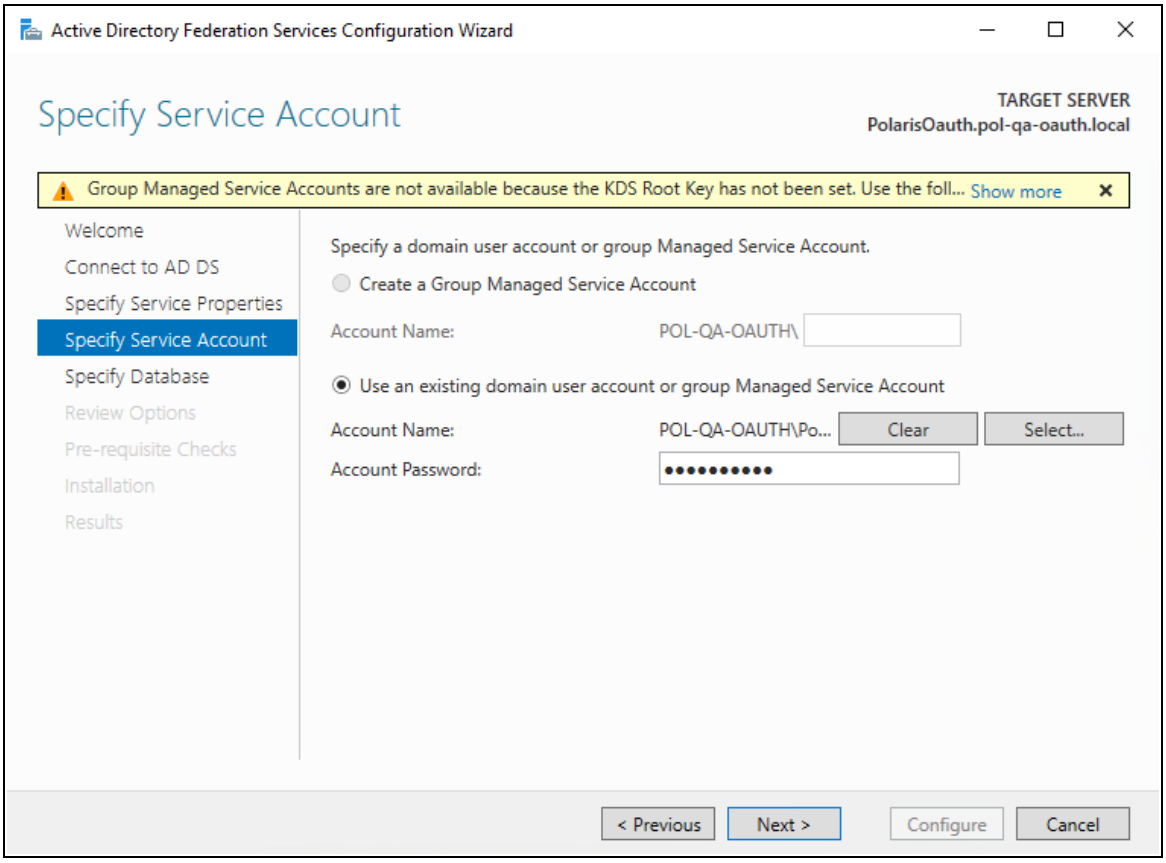
4. Select **Change**, and provide an administrator account. Then select **Next**.



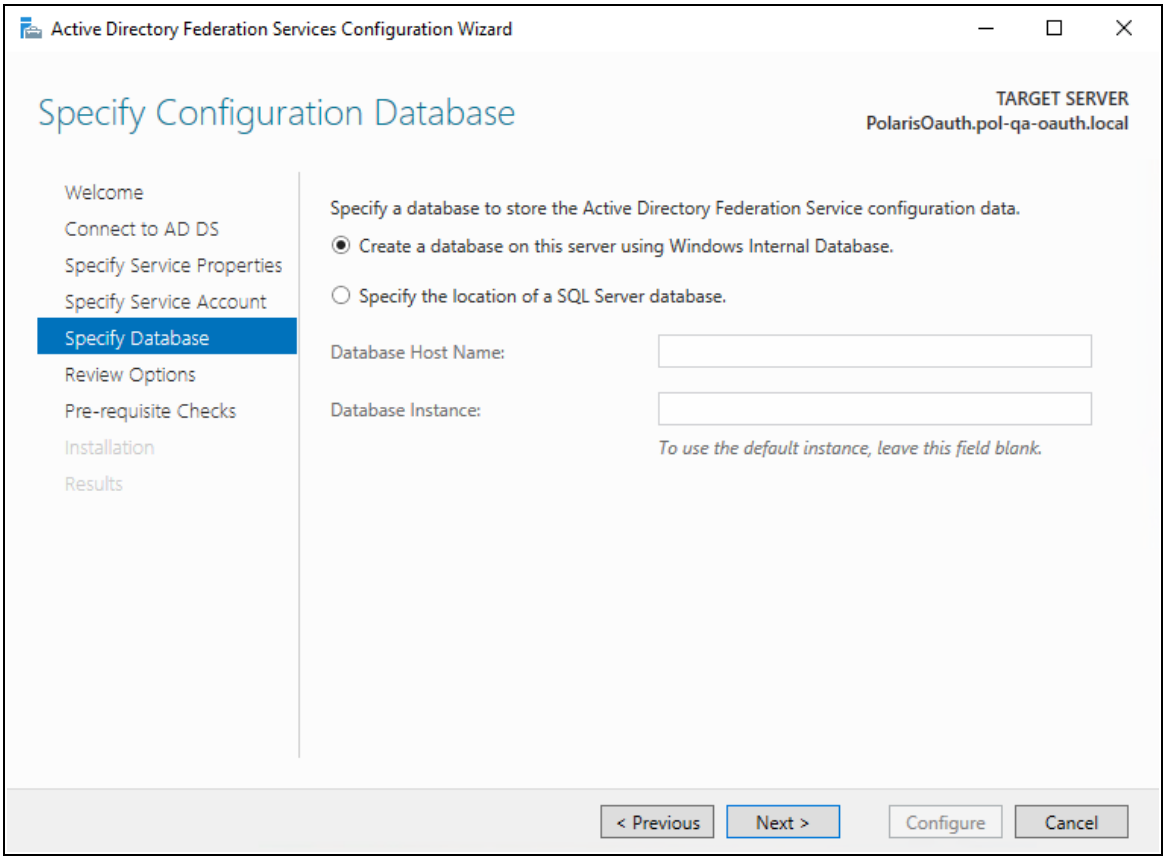
5. If not already installed on the server, select **Import** to install an SSL certificate.



6. Enter the following, and then select **Next**:
    - Federation Service Name
    - Federation Service Display Name

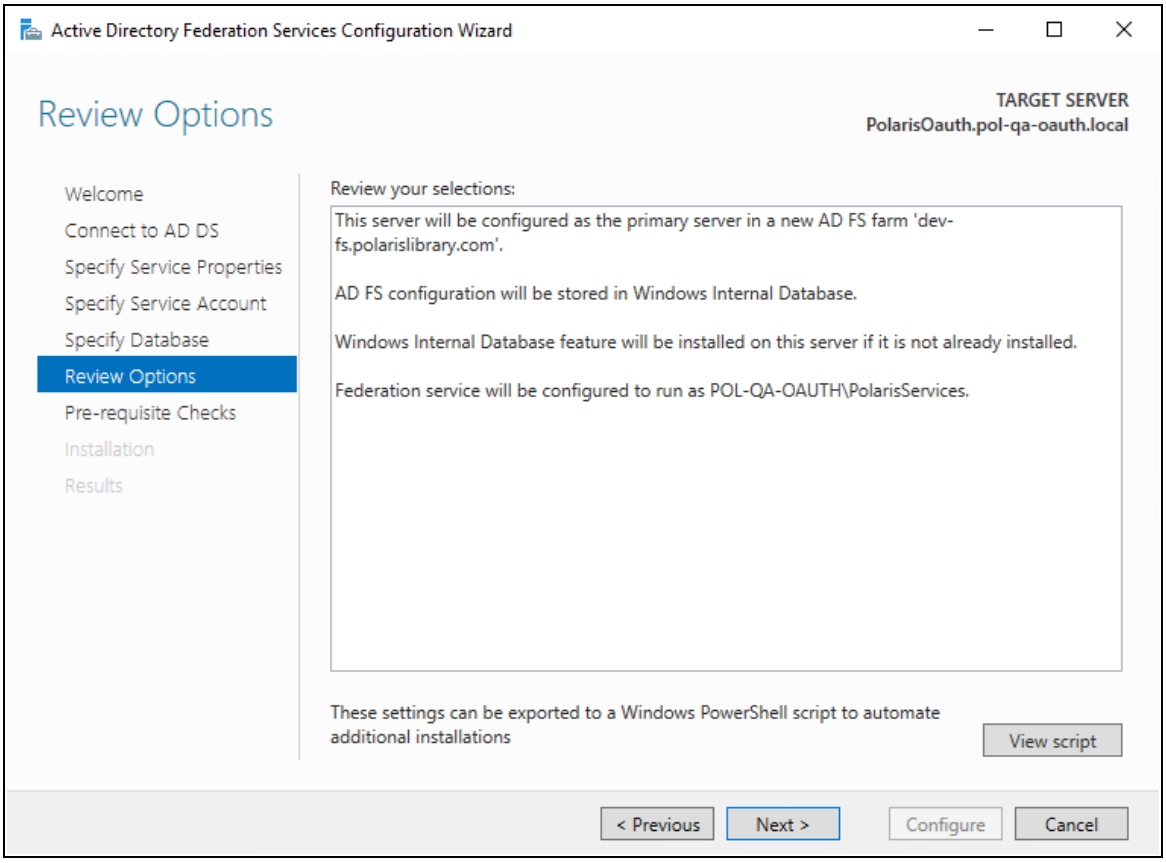


7. Specify a service account, and then select **Next**.

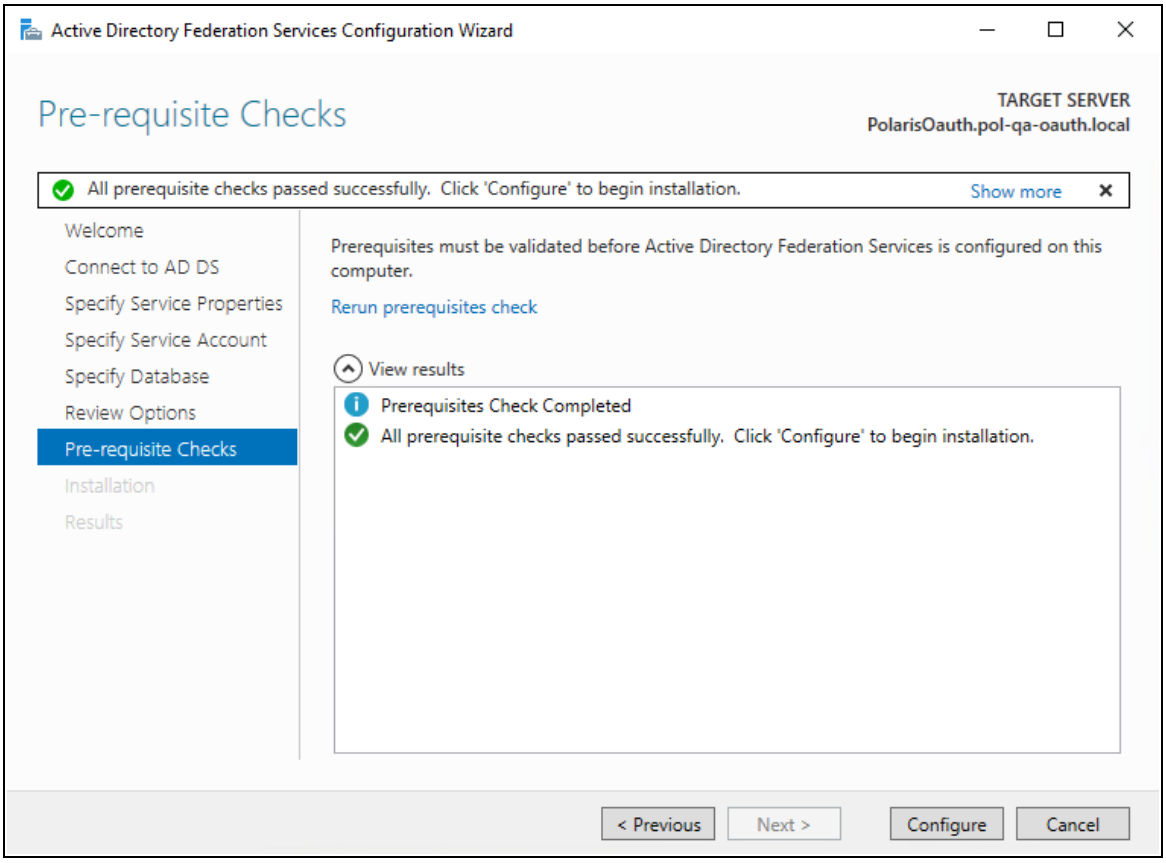


8. Specify the location of the AD FS configuration database, and then select **Next**.

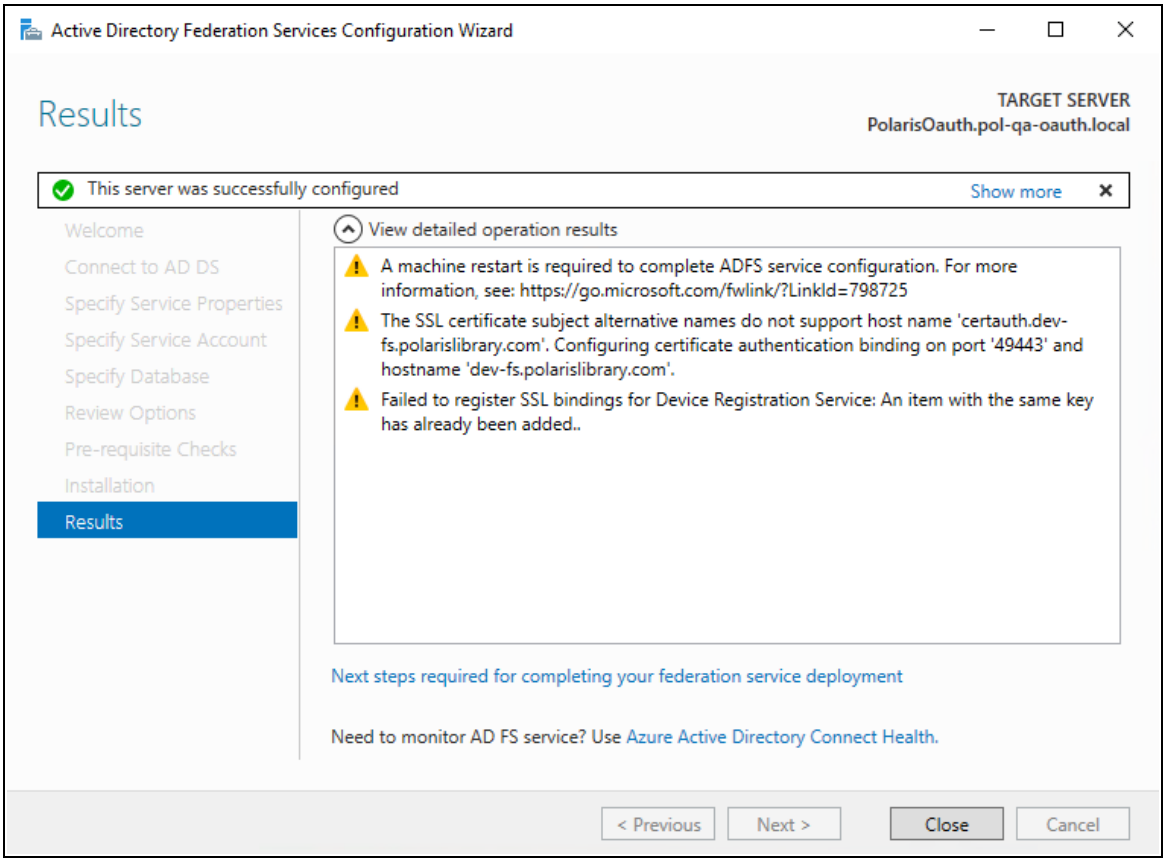
   For simple scenarios, creating the local database is acceptable.



9. Review your selections, and then select **Next**.



10. After you complete all pre-requisite checks, select **Configure**.



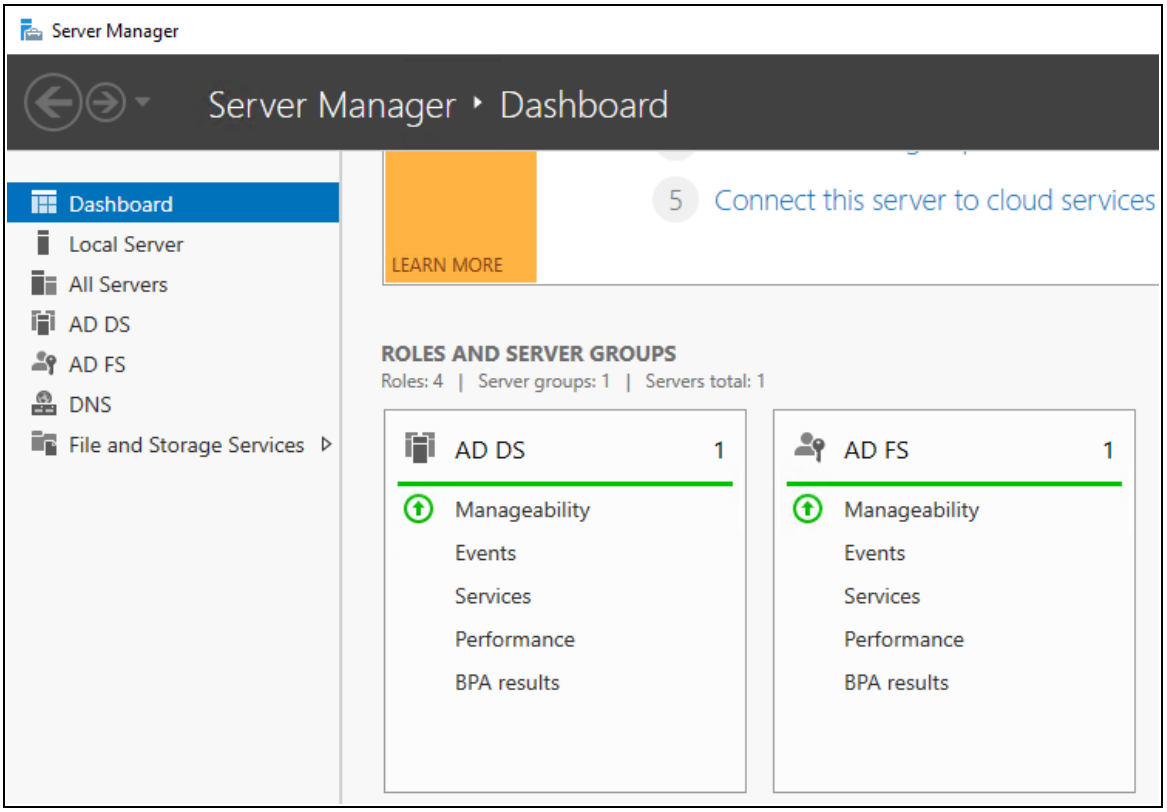
11. When the configuration wizard has completed successfully, select **Close**, and then restart the server.

Verify Active Directory Federation Services Is Running

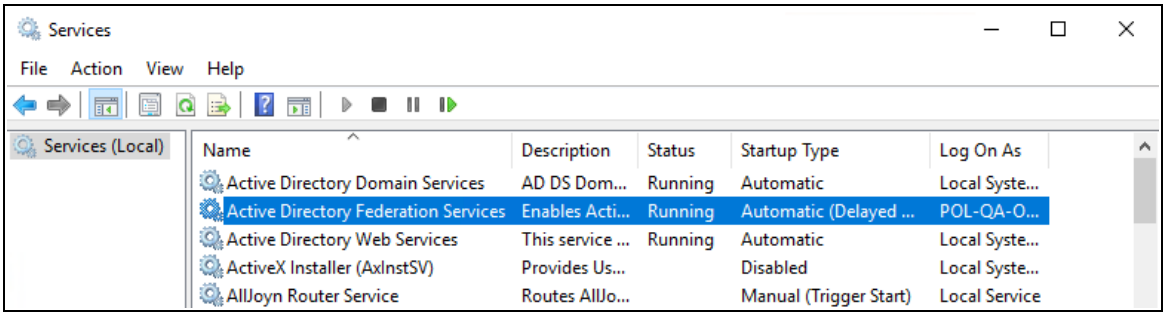
To verify that Active Directory Federation Services is running

1. Start the Server Manager desktop application.

   AD FS should be green.

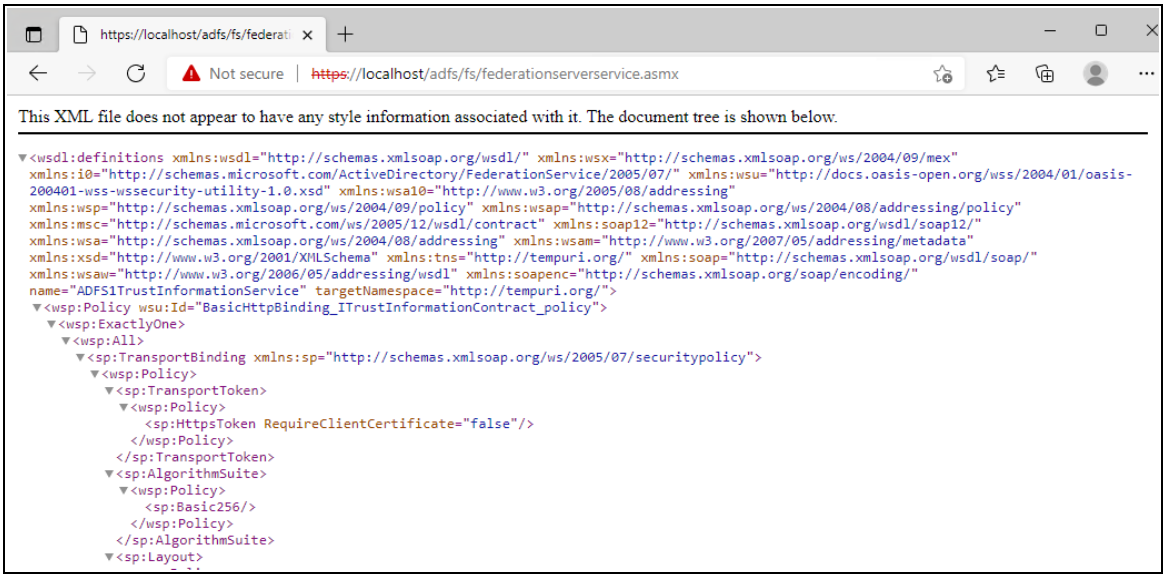


2. Start the Services application and check the status.



3. Open the Edge (or Chrome) web browser and go to https://localhost/adfs/fs/federationserverservice.asmx
   - If you want to ignore certificate errors, select **Advanced**.

   A page similar to the following image opens:

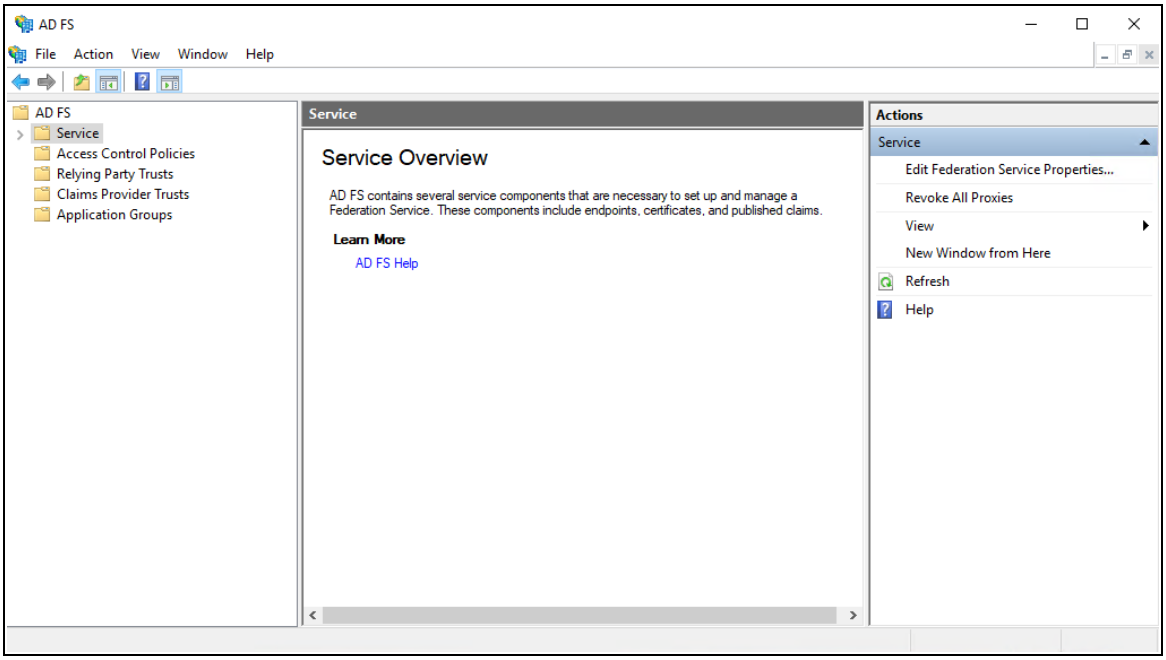


This indicates that Active Directory Federation Services is running.

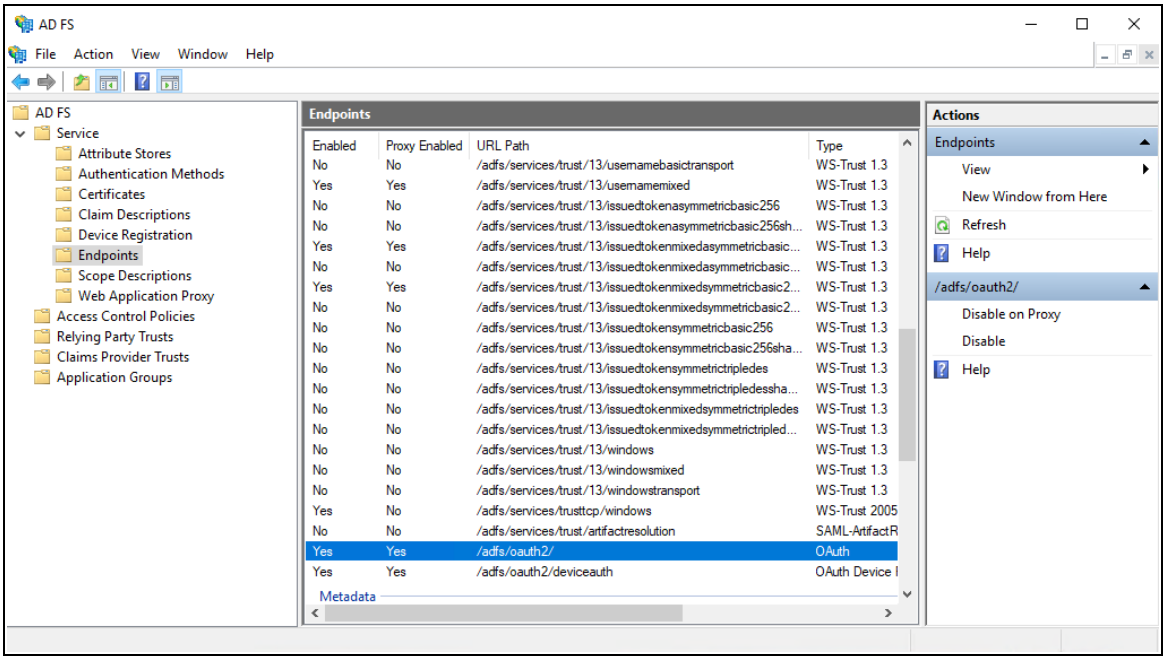
Verify that OAuth 2.0 is Enabled

To verify that OAuth 2.0 is enabled

1. Open the AD FS Management desktop application.



2. Open the **Service** folder, and then select the **Endpoint** folder.



3. Search for the oauth2 path.
4. In either the Edge or Chrome web browser, go to https://localhost/adfs/.well-known/openid-configuration
    - If you want to ignore certificate errors, select **Advanced**.

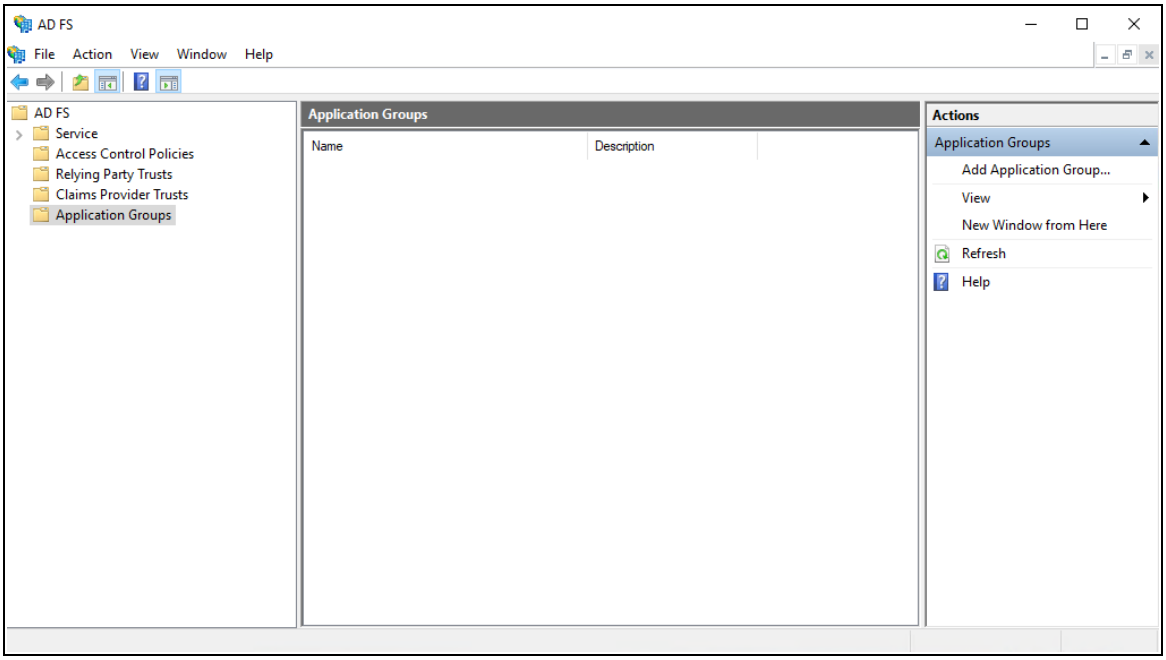
A page similar to the following image opens:

This indicates that OAuth 2.0 is available.

Create an Application Group

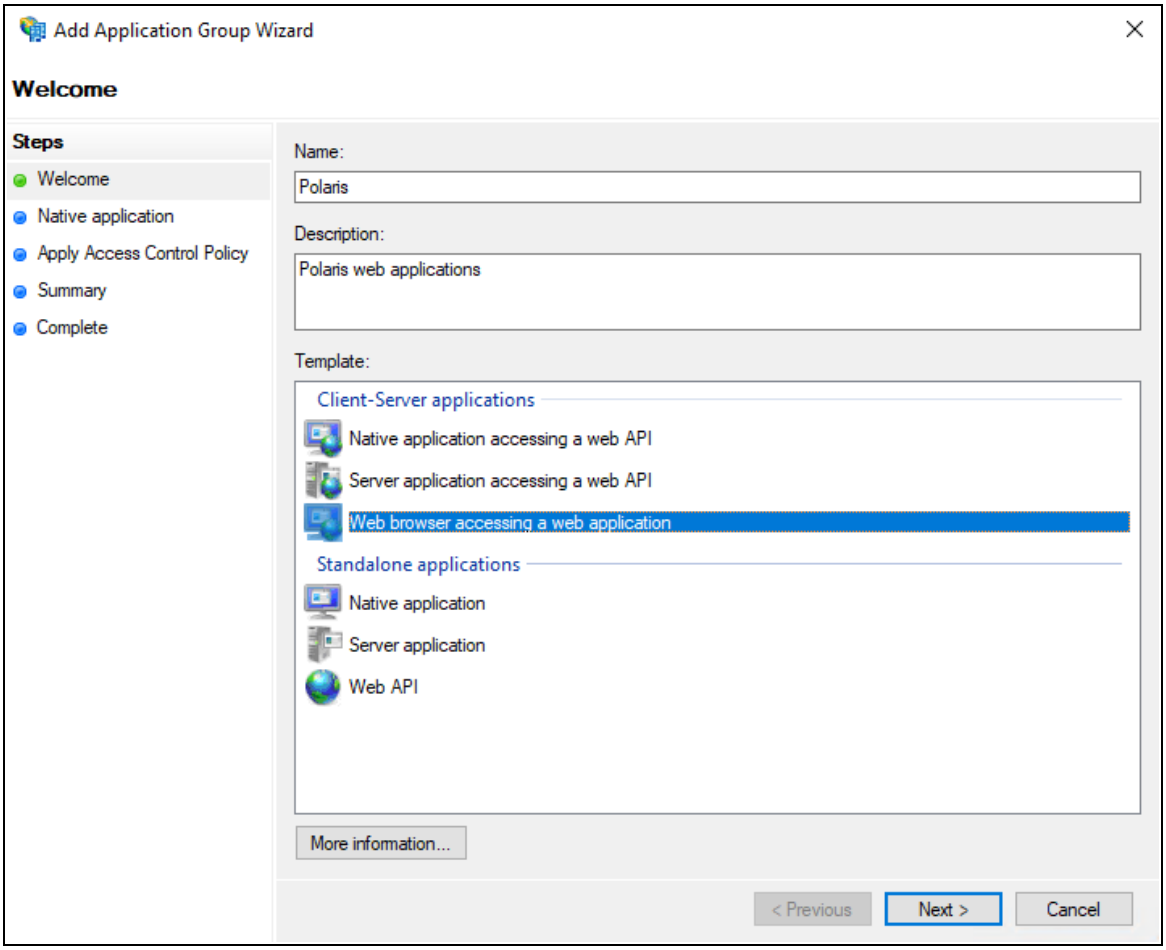
To create an application group for use with Polaris Admin and LeapWebApp

1. Open the AD FS Management desktop application.

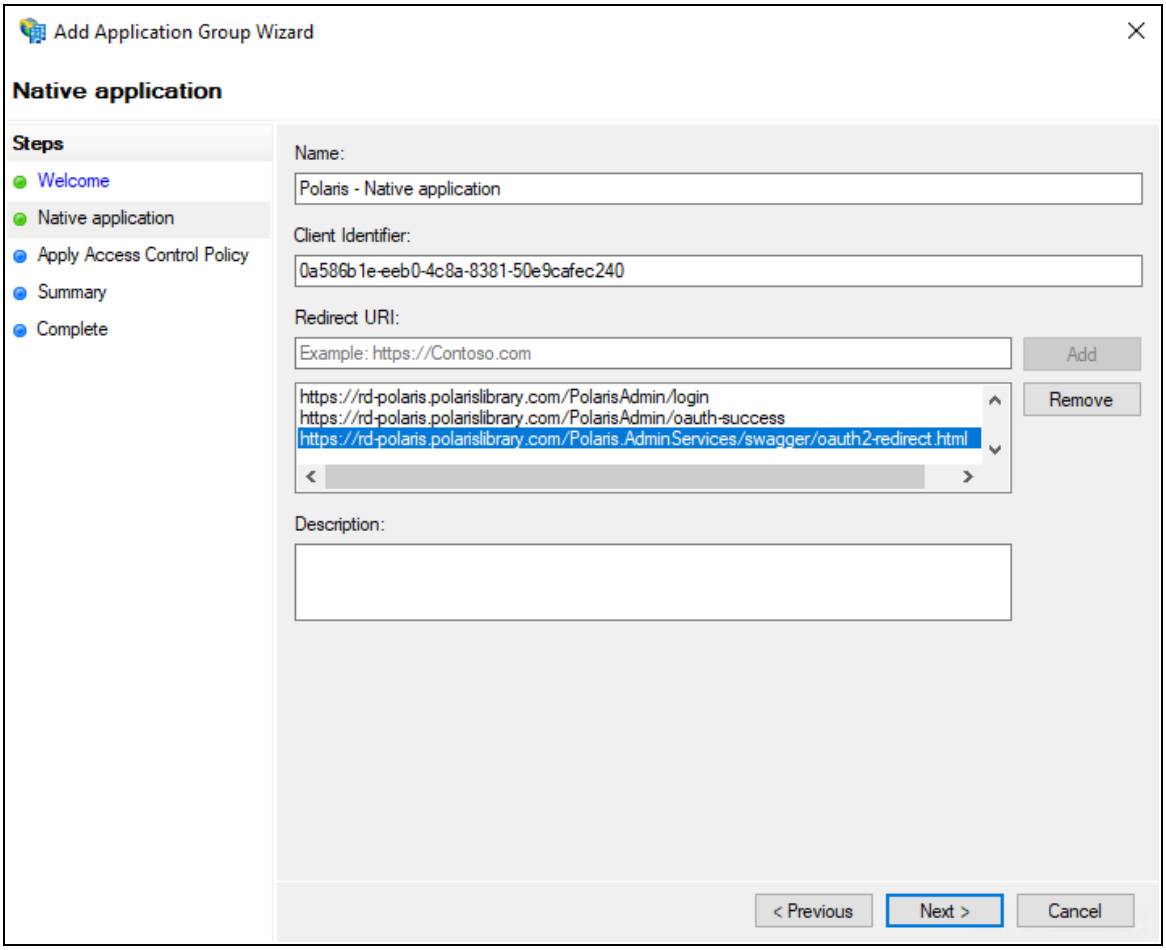


2. Select the **Application Groups** folder.
3. Under **Actions**, select **Add Application Group**.

   The Add Application Group wizard opens.



4. On the **Welcome** tab, do the following:
    a. In the **Name** box, enter **Polaris**.
    b. In the **Description** box, enter **Polaris web applications**.
    c. From the Template section, select **Web browser accessing a web application**.
5. Select **Next**.



6. On the **Native application** tab, in the **Redirect URI** box, enter the following URIs:
    - https://*server_address*/polarisauth/login
    - https://*server_address*/polarisauth/logout
    - https://*server_address*/polarisauth/signin-oidc
    - https://*server_address*/polarisauth/signout-callback-oidc
    - https://*server address*/leapwebapp/signin-oidc
    - https://*server address*/leapwebapp/signin-override-oidc
    - https://*server address*/leapwebapp/signout-callback-oidc

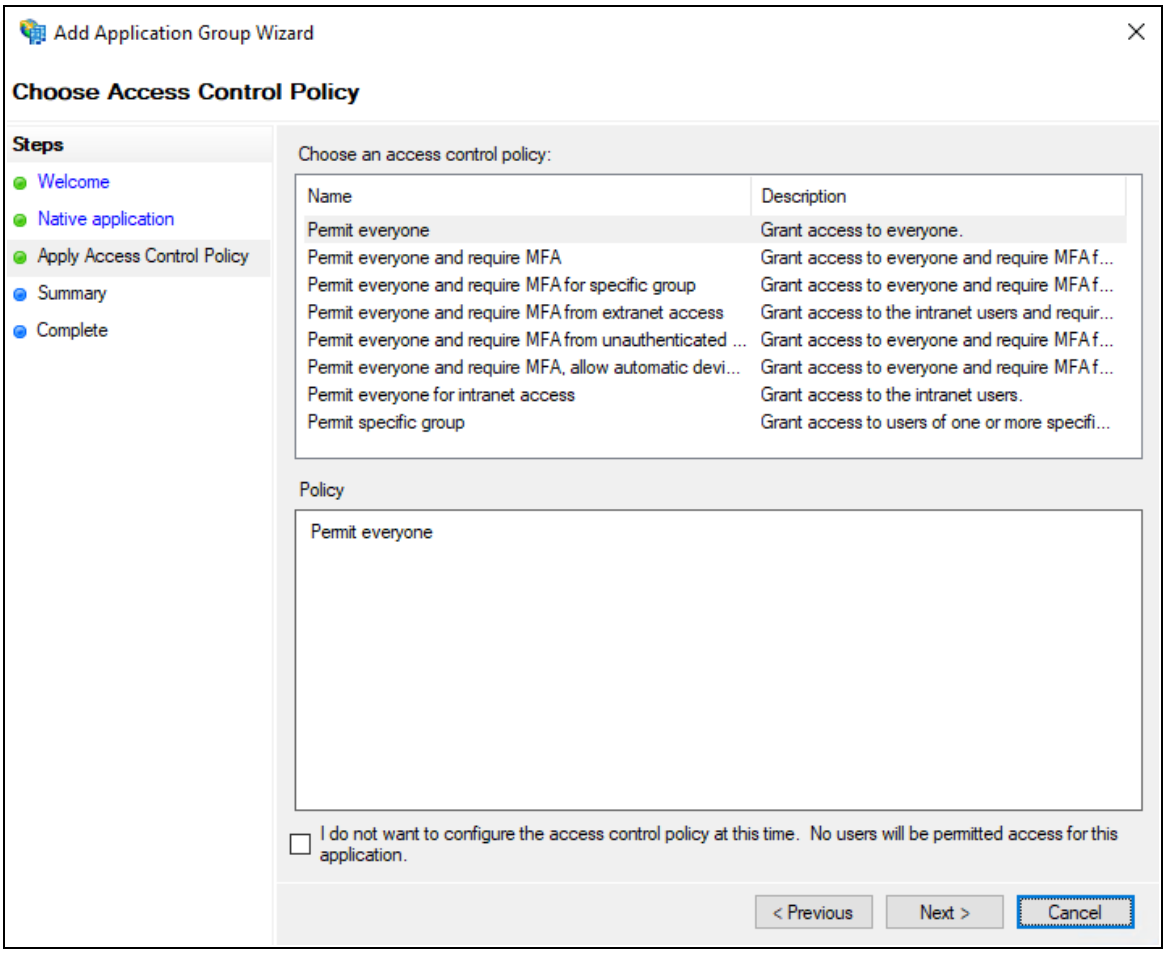
- https://*server address*/Polaris.ApplicationServices/swagger/oauth2-redirect.html

> **Note:**
> Replace *server address* with the FQDN that matches your Polaris System Administration (web-based) or Leap server name and certificate.

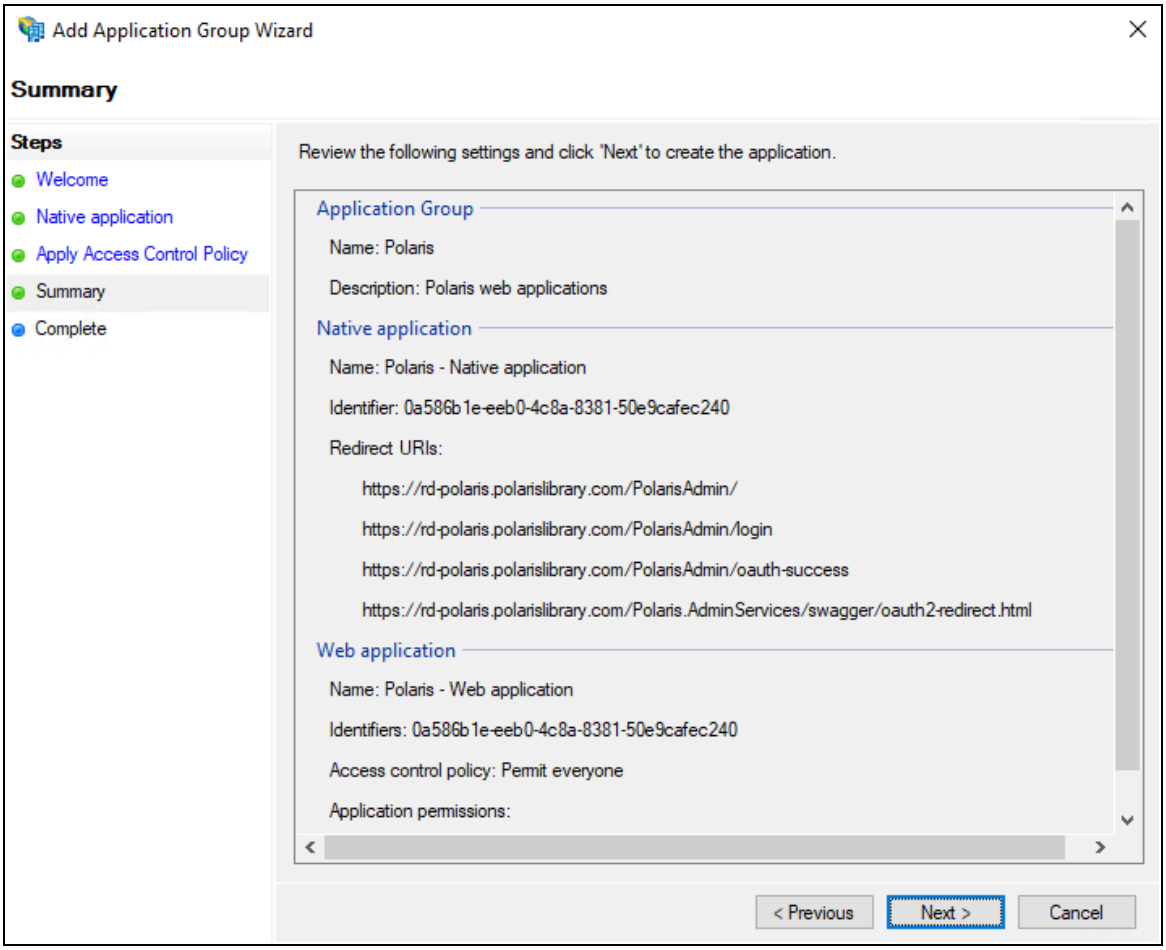
7. Copy the value in the **Client Identifier** box to Notepad.

   You'll need this when you set up PolarisAdmin's appsettings.user.json.

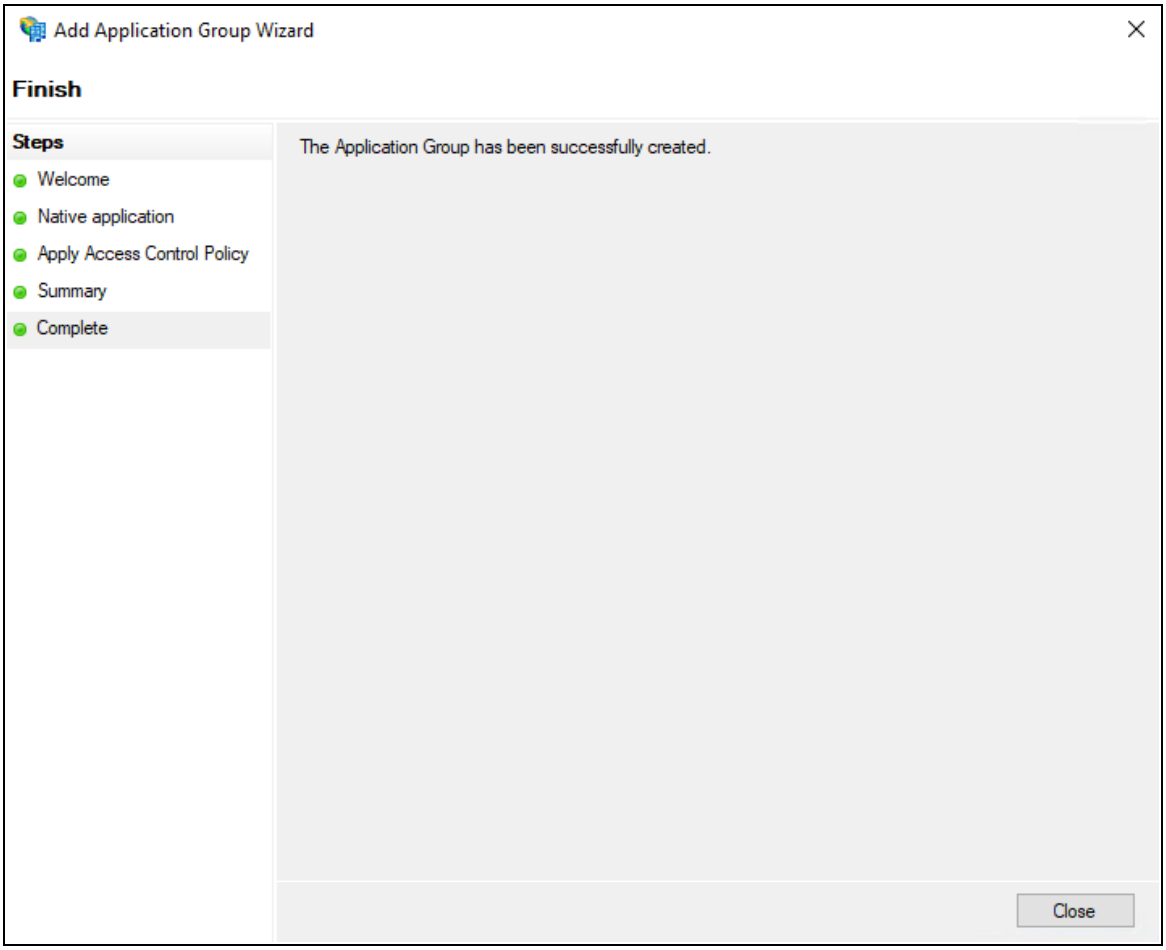
8. Select **Next**.



9. On the **Apply Access Control Policy** tab, select an access control policy, and then select **Next**.



10. On the **Summary** tab, review the settings and then select **Next**.

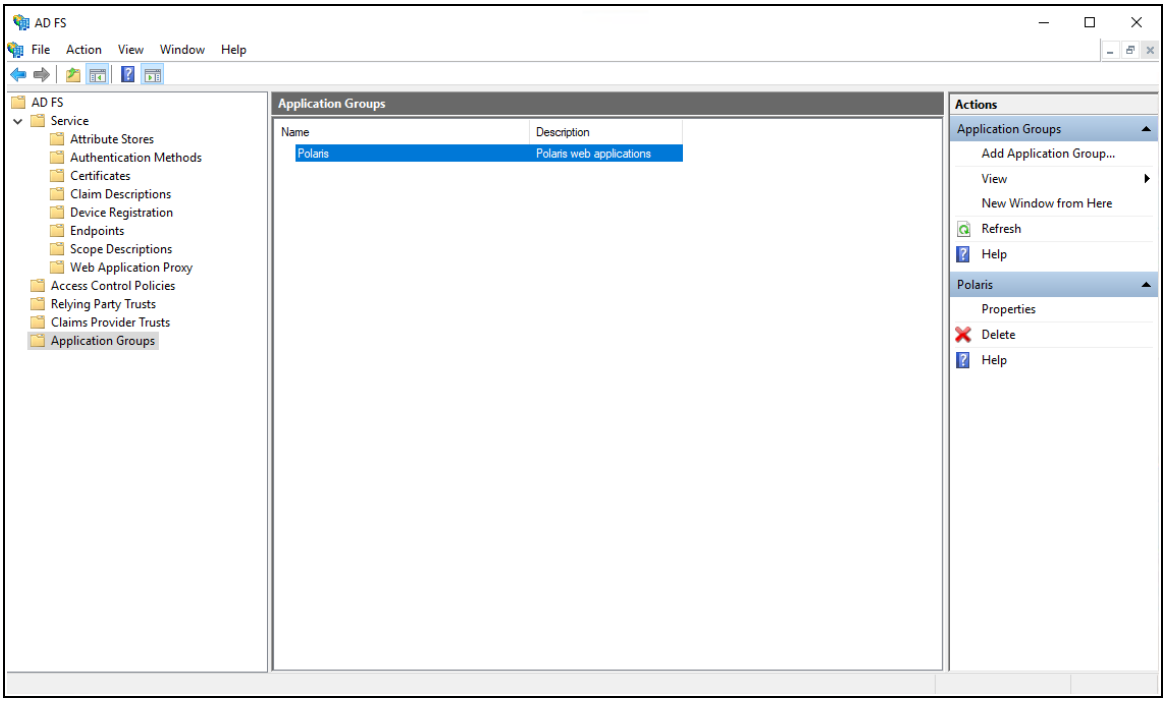


11. On the **Complete** tab, select **Close**.

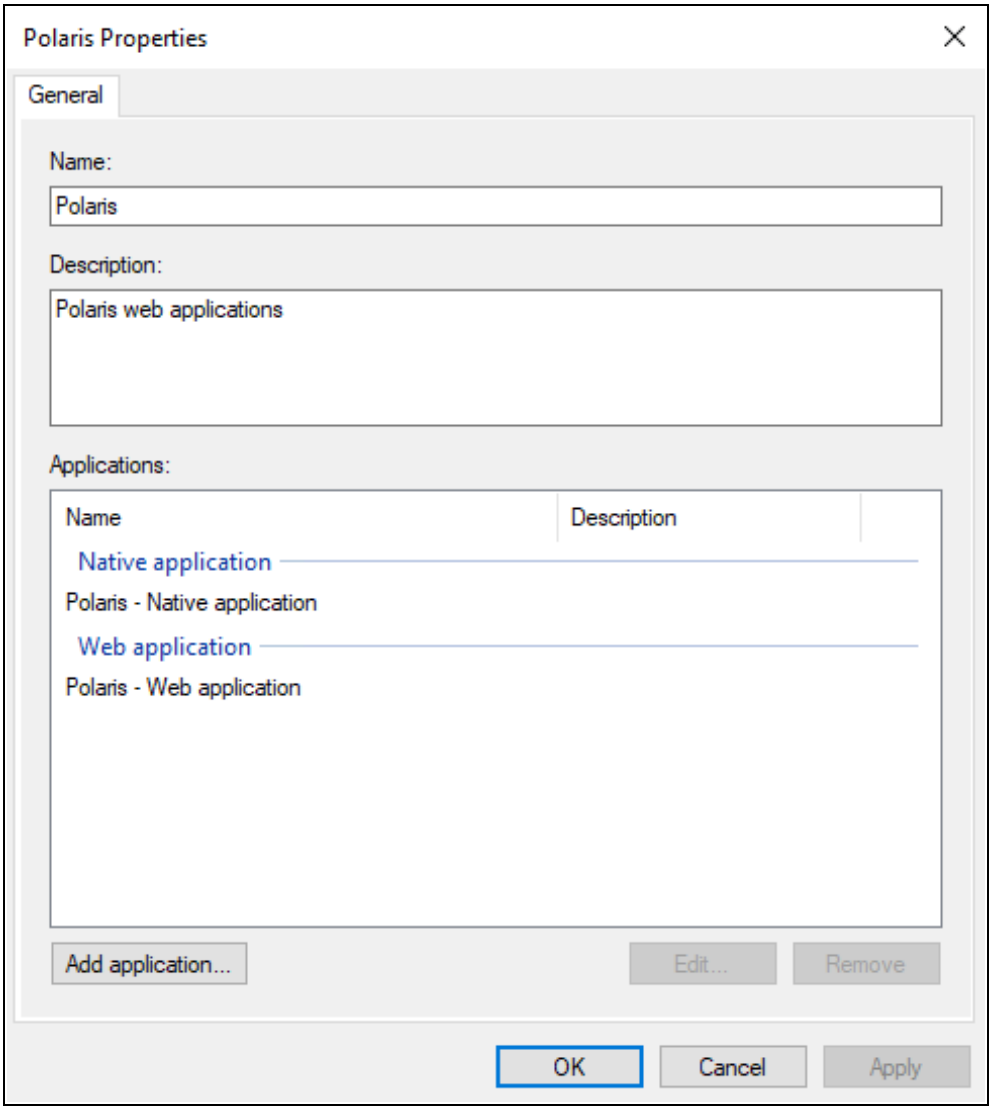
Configure the AD FS Web Application: Claims and Permitted Scopes

To configure the AD FS web application

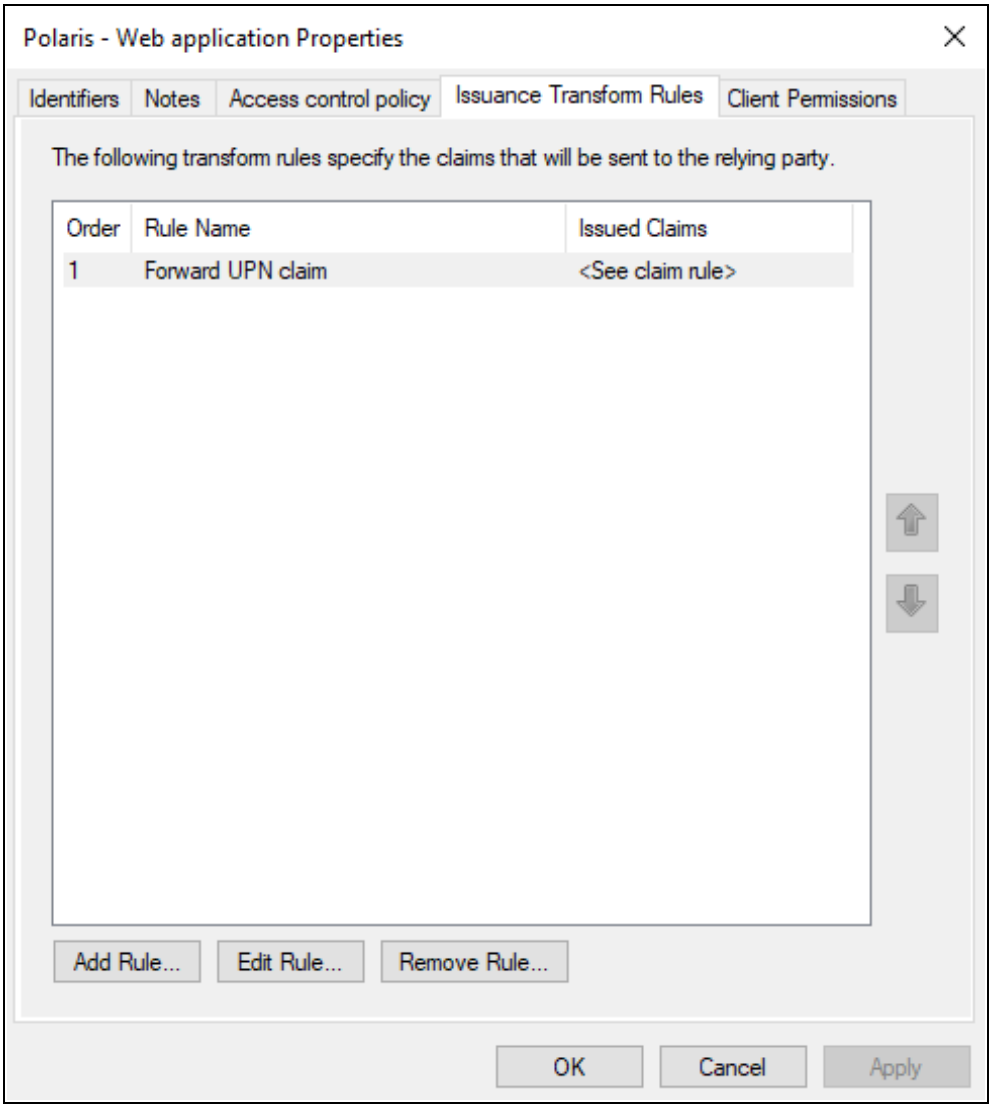
1. Open the AD FS Management desktop application.



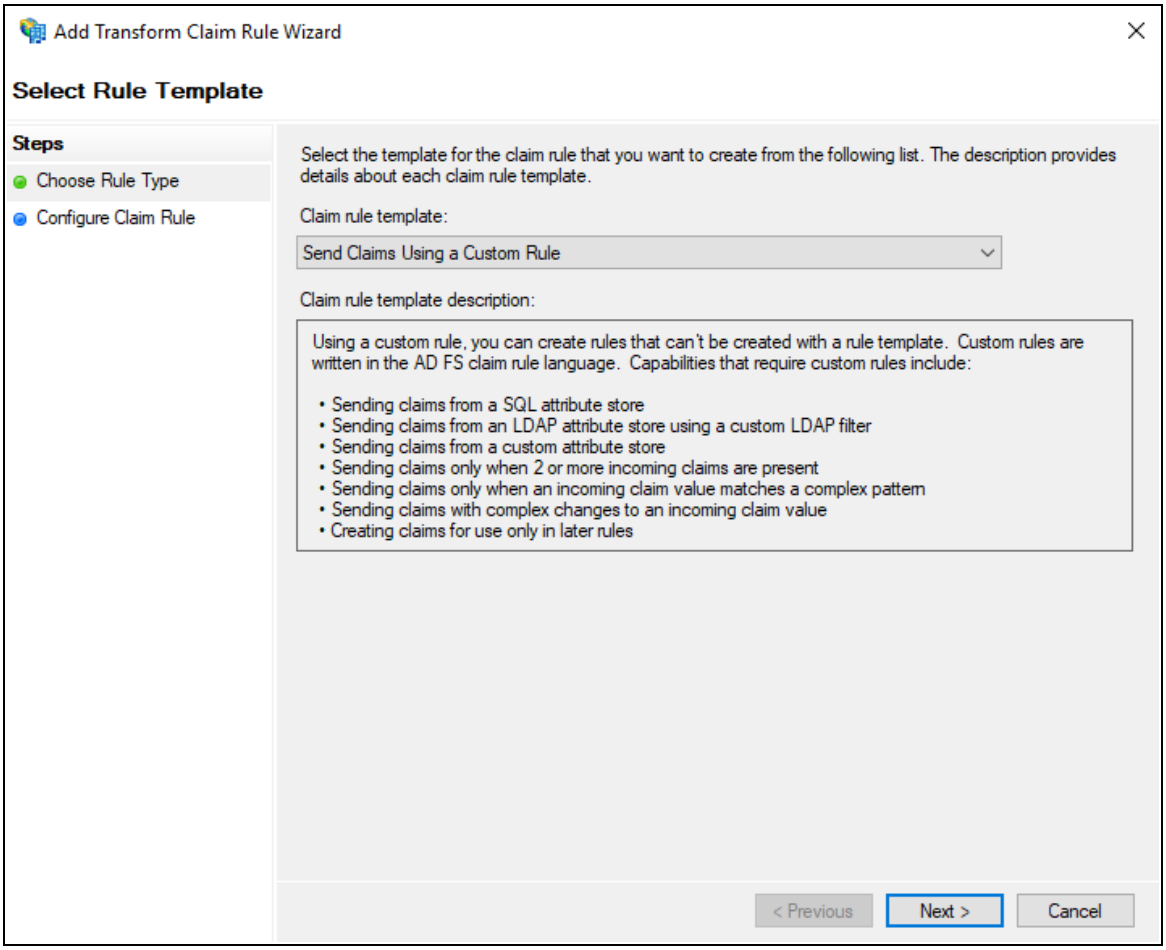
2. Select the **Application Groups** folder.
3. Select the **Polaris** application group, and then select **Properties**.



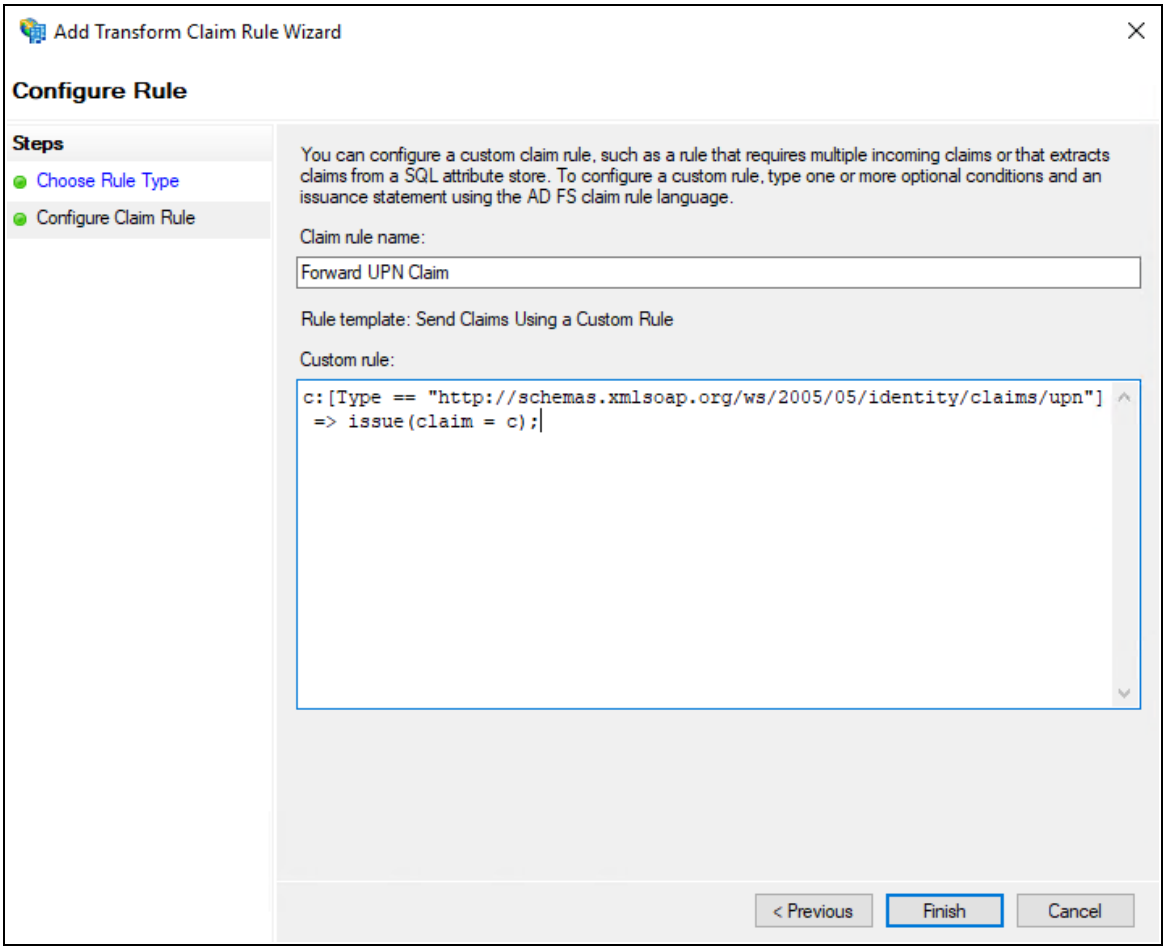
4. Select **Polaris** - **Web application**, and then select **Edit**.



5. Select the **Issuance Transform Rules** tab, and then select **Add Rule**.



6. On the Add Transform Claim Rule Wizard, select **Send Claims Using a Custom Rule** from the **Claim rule template** list, and then select **Next**.

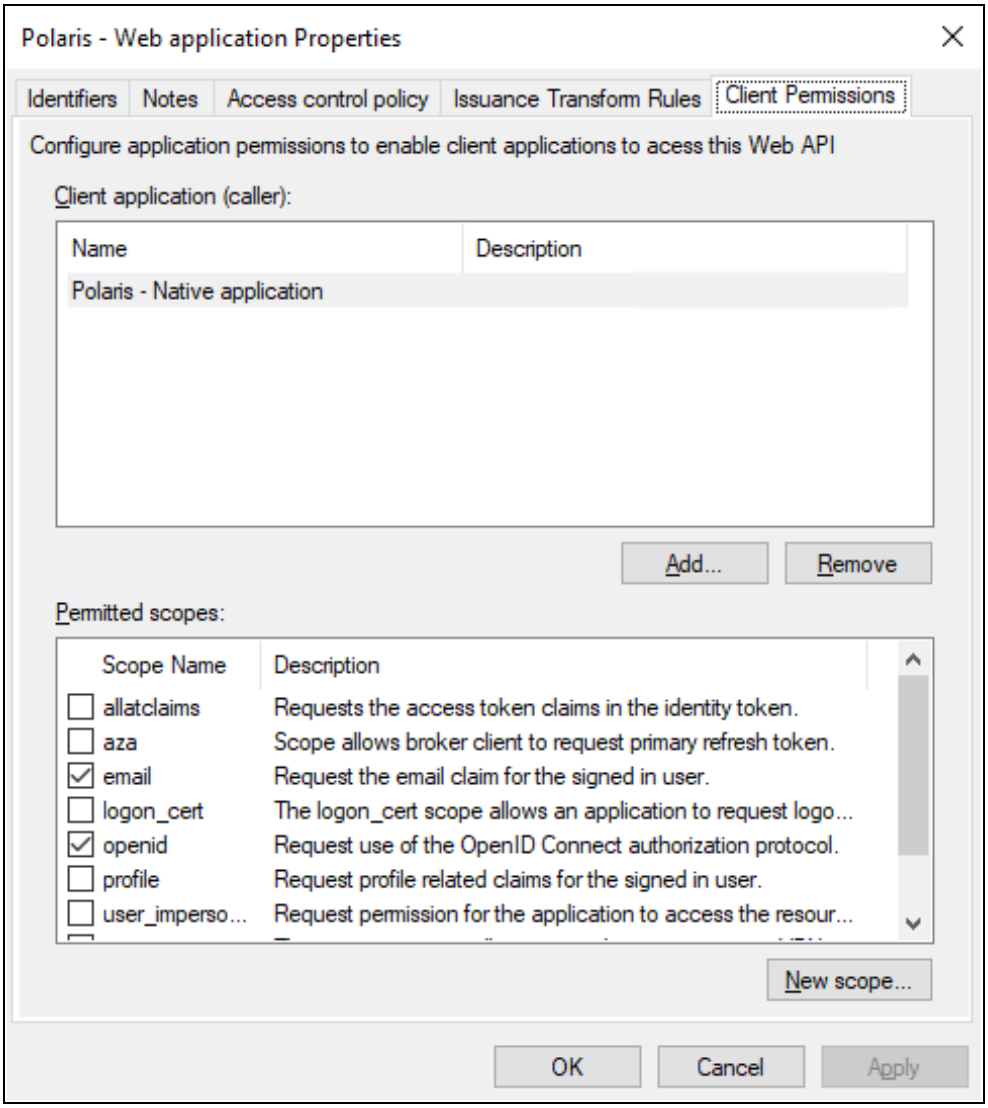


7. In the **Claim rule name** box, enter **Forward UPN Claim**.
8. In the **Custom rule** box, enter the following rule:

   ```
   c:[Type ==
   "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"]
   => issue(claim = c);
   ```
9. Select **Finish**.
10. On the **Issuance Transform Rules** tab, select **Add Rule**.
11. On the Add Transform Claim Rule Wizard, select **Send Claims Using a Custom Rule** from the **Claim rule template** list, and then select **Next**.
12. In the **Claim rule name** box, enter **Add TenantId**.
13. In the **Custom rule** box, enter the following rule:

```
=> issue(Type =
"http://schemas.microsoft.com/identity/claims/tenantid",
Value = "polaris");
```

14. Select **Finish**.



15. On the **Client Permissions** tab, verify that **email** and **openid** are selected.
16. Select **OK** to close the Web application Properties dialog.
17. Select **OK** to close the Polaris properties dialog.
18. Using the services applet, restart the Active Directory Federation Services service.

Enable CORS on AD FS To Accept Requests from Polaris APIs

To enable CORS on AD FS to accept requests from Polaris APIs

1. Refer to the information on the following page:
   - https://docs.microsoft.com/en-us/windows-server/identity/ad-fs/operations/customize-http-security-headers-ad-fs#cross-origin-resource-sharing-cors-headers
2. Use the following commands:
   - `Set-AdfsResponseHeaders -EnableCORS $true`
   - `Set-AdfsResponseHeaders -CORSTrustedOrigins` *`https://rd-polaris.polarislibrary.com`*`,`*`https://example2.com`*

     Note:
     Replace *`https://rd-polaris.polarislibrary.com`* and *`https://example2.com`* with your own URL or list of URLs.

Set Up Web Services and Applications for OIDC with Active Directory and AD FS

To set up each of the following web services and applications, you must configure a .json file for each of the following:

- Polaris.Authentication (the application that authenticates Polaris users)
- Polaris.AuthenticationServices (the API service that provides backend support for authentication)
- PolarisAdmin (the web-based Polaris System Administration application)
- LeapWebApp (Leap)
- Polaris.ApplicationServices (Leap's API service)

The five .json files are all named appsettings.user.json, but they reside in different directories:

- C:\Program Files\Polaris\8.1\Polaris.Authentication
- C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices
- C:\Program Files\Polaris\8.1\PolarisAdmin\assets
- C:\Program Files\Polaris\8.1\LeapWebApp
- C:\Program Files\Polaris\8.1\Polaris.ApplicationServices

This section contains the following topics:

Configure Polaris.Authentication for Use with AD FS

To configure Polaris.Authentication, update C:\Program Files\Polaris\8.1\Polaris.Authentication\appsettings.user.json. You will use several values copied from your identity provider.

To configure Polaris.Authentication

1. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
    - C:\Program Files\Polaris\8.1\Polaris.Authentication\appsettings.user.json
    - C:\Program Files\Polaris\8.1\Polaris.Authentication\RELEASE-NOTES.md
2. In the RELEASE-NOTES.md file, copy the settings in the ADFS Template section. The image below shows the settings to copy.

````
## Configuring OIDC OAuth 2.0

### ADFS Template

```
"OAuth": {
  "Enabled": true,
  "SendOAuthAuthorityHeader": false,
  "Authorities": [
    {
      "IsActive": true,
      "Name": "ADFS",
      "Authority": "https://[adfs-server-address]/adfs/",
      "UseOidc": true,
      "UsePkce": true,
      "EndSessionEndpoint": "https://[adfs-server-address]/adfs/oauth2/logout",
      "ClientId": "[client-id]",
      "ClientSecret": "[client-secret]",
      "CallbackPath": "/oauth/callback",
      "SignedOutCallbackPath": "/signout-callback-oidc",
      "SignedOutRedirectUri": "/login",
      "AlternateUpnClaimType": "email",
      "AlternateLogoutUri": null,
      "Scopes": [
        "profile",
        "openid"
      ],
      "SaveTokens": false,
      "MetaAddress": "https://[adfs-server-address]/adfs/.well-known/openid-configuration",
      "ResponseMode": "form_post",
      "ResponseType": "code"
    }
  ]
}
```
````

3. In the .json file, replace the entire contents of the file with the settings you copied from the RELEASE-NOTES.md file.
4. Verify that `Enabled` is set to `true`.
5. Verify that `IsActive` is set to `true`.
6. Verify that `UseOidc` is set to `true`.
7. Update the `Authority` and `EndSessionEndpoint` properties with values from your AD FS well-known configuration URL.

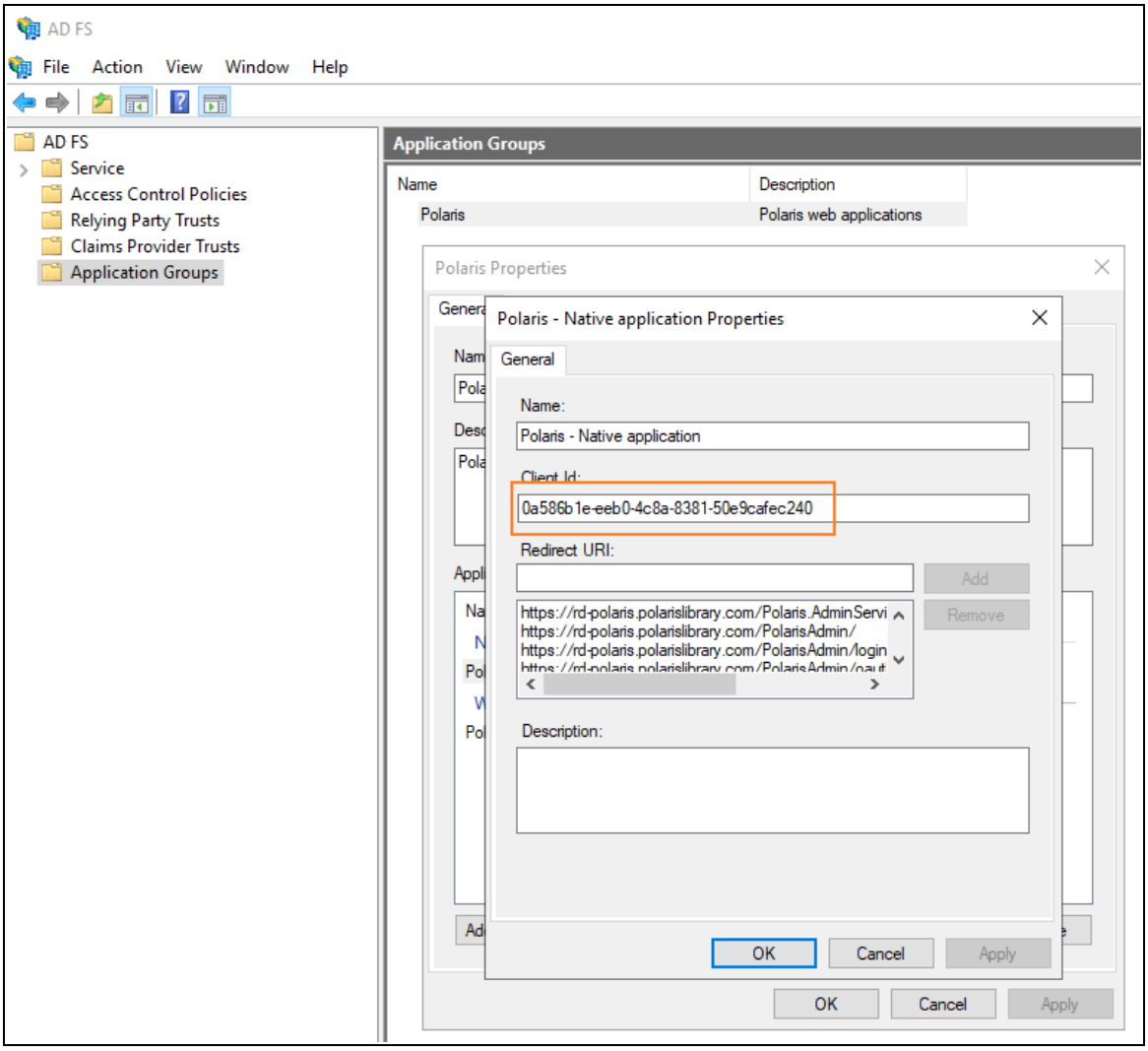
> **Note:**
>
> Sample well-known configuration URL:
>
> https://*adfs-server-address*/adfs/.well-known/openid-configuration

Replace *adfs-server-address* with the AD FS server address.

a. Replace the `Authority` property value with the `issuer` value from the well-known configuration URL.

b. Replace the `EndSessionEndpoint` property value with the `end_session_endpoint` value from the well-known configuration URL.

8. On the AD FS server, open the AD FS Management desktop application.



9. Copy the client ID from the Polaris - Native application properties dialog.

10. In the .json file, update the `ClientId` property. Replace *`[client-id]`* with the client ID copied from the AD FS Management desktop application.
11. Update the `Scopes` property if necessary for your AD FS configuration.

    Note:
    The default values provided for the `Scopes` property are appropriate for most AD FS configurations. However, it is possible that your configuration may use a different scope, for example, `email`.

12. In the `MetaAddress` property, replace *`[adfs-server-address]`* with the AD FS server address.
13. Save the .json file.

Configure Polaris.AuthenticationServices for Use with AD FS

To configure Polaris.AuthenticationServices, update C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices\appsettings.user.json. You will use several values copied from your identity provider.

Important:
Polaris 8.1 includes a RELEASE-NOTES.md file that contains template settings that apply to AD FS. See the OIDC/OAuth Setup for AD FS section of the release notes file.

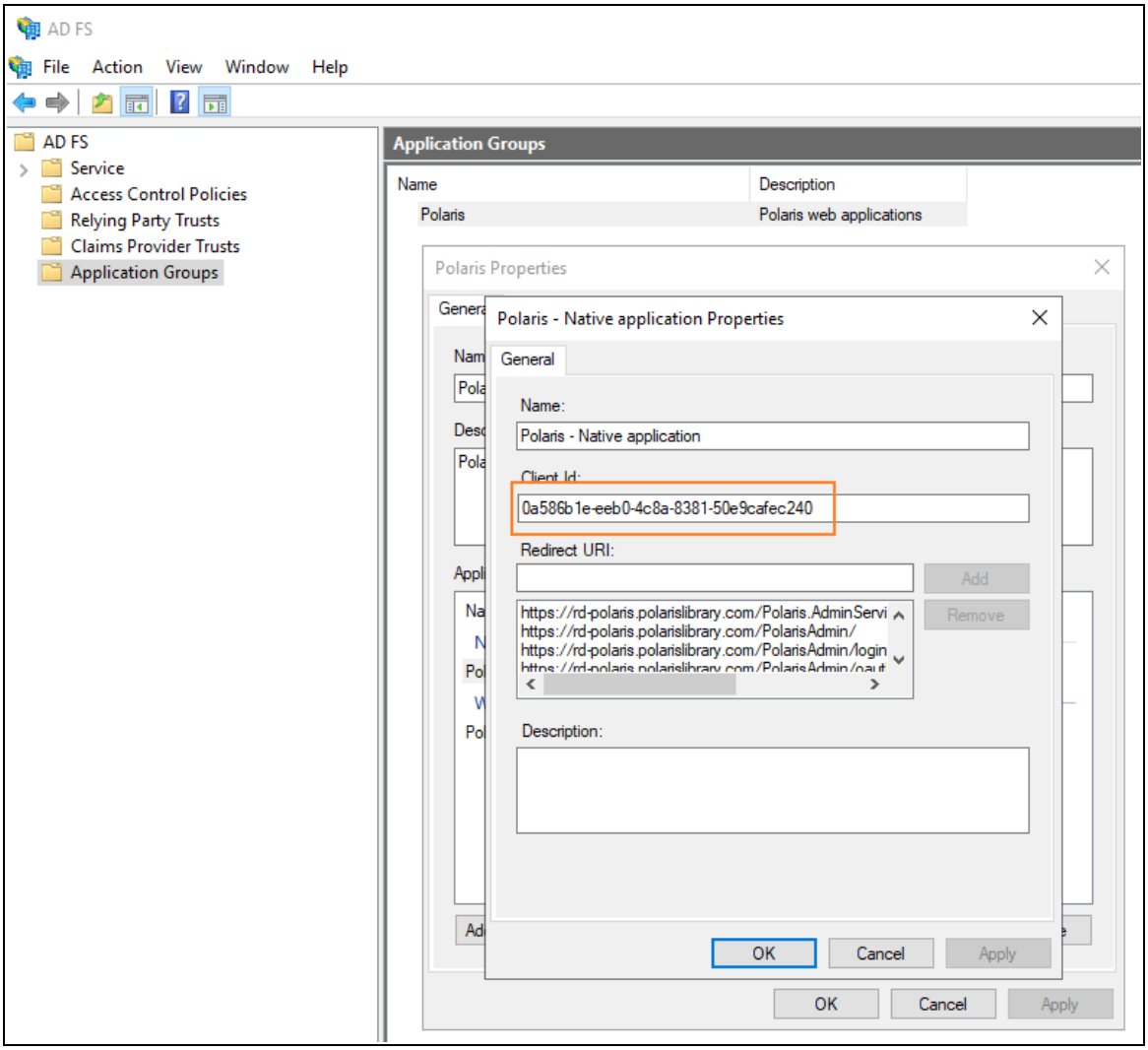
To configure Polaris.AuthenticationServices

1. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
   - C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices\appsettings.user.json
   - C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices\RELEASE-NOTES.md
2. In the RELEASE-NOTES.md file, copy the settings in the OIDC/OAuth Setup for AD FS section. The image below shows the settings to copy.

```
### OIDC/OAuth Setup for AD FS

```
"OAuth": {
  "Enabled": true,
  "Authorities": [
    {
      "Name": "ADFS",
      "UseOidc": true,
      "RequireHttpsMetadata": true,
      "RequireSignedTokens": true,
      "ValidateIssuer": true,
      "ValidateAudience": true,
      "Authority": "https://dev-fs.polarislibrary.com/adfs/",
      "Audience": "microsoft:identityserver:0a586b1e-eeb0-4c8a-8381-50e9cafec240",
      "MetaAddress": "https://dev-fs.polarislibrary.com/adfs/.well-known/openid-configuration",
      "ValidIssuers": [
        "https://dev-fs.polarislibrary.com/adfs",
        "http://dev-fs.polarislibrary.com/adfs/services/trust"
      ],
      "ValidAudiences": [
        "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
        "microsoft:identityserver:0a586b1e-eeb0-4c8a-8381-50e9cafec240"
      ],
      "UpnClaimTypes": [ "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn" ],
      "IsUpnExternalId": false
    }
```

3. In the .json file, replace the entire contents of the file with the settings you copied from the RELEASE-NOTES.md file.
4. Update the following properties. In each, replace *dev-fs.polarislibrary.com* with the AD FS server address:
    - `Authority`
    - `MetaAddress`
    - `ValidIssuers`
5. On the AD FS server, open the AD FS Management desktop application.
```



6. Copy the client ID from the Polaris - Native application properties dialog.
7. Update the following properties. In each, replace *`0a586b1e-eeb0-4c8a-8381-50e9cafec240`* with the client ID copied from the AD FS Management desktop application:
    - `Audience`
    - `ValidAudiences`
8. In the `UpnClaimTypes` property, update the default value if you want to specify a different claim to serve as the user identifier. The default value applies to most

configurations, but you can specify a different claim that exists in a JSON Web Token (JWT).

9. Save the .json file.

Configure PolarisAdmin for Use with AD FS

To configure PolarisAdmin, update C:\Program Files\Polaris\8.1\PolarisAdmin\assets\appsettings.user.json.

To configure PolarisAdmin

1. Open the C:\Program Files\Polaris\8.1\PolarisAdmin\assets\appsettings.user.json file in a text editor. You must run the editing application (for example, Notepad) as administrator.
2. In the .json file, update the server location in the `apiUrlRoot`. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *`[my-server-domain-name]`* with the FQDN of the server that hosts both the API service for Polaris System Administration (web-based) and the Authentication Application.
3. Update the server location in the `authAppUrl`. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *`[my-server-domain-name]`* with the FQDN of the server that hosts both the API service for Polaris System Administration (web-based) and the Authentication Application.
4. Save the .json file.

Configure LeapWebApp for Use with AD FS

Once you have made the changes described in the Upgrading to Polaris 7.7 section, you only need to do additional configuration for LeapWebApp if one or both of the following is true:

- You want to enable permission overrides in Leap.
- You want to set `ReAuthDisabled` to `false` for Leap.

If neither of the above conditions is true, skip to the next section. See Customize the AD FS Pages.

> **Note:**
> Permission overrides and reauthentication are not supported if your system uses multiple identity providers for authentication.

To configure LeapWebApp, update C:\Program Files\Polaris\8.1\LeapWebApp\appsettings.user.json. You will use several values copied from your identity provider.

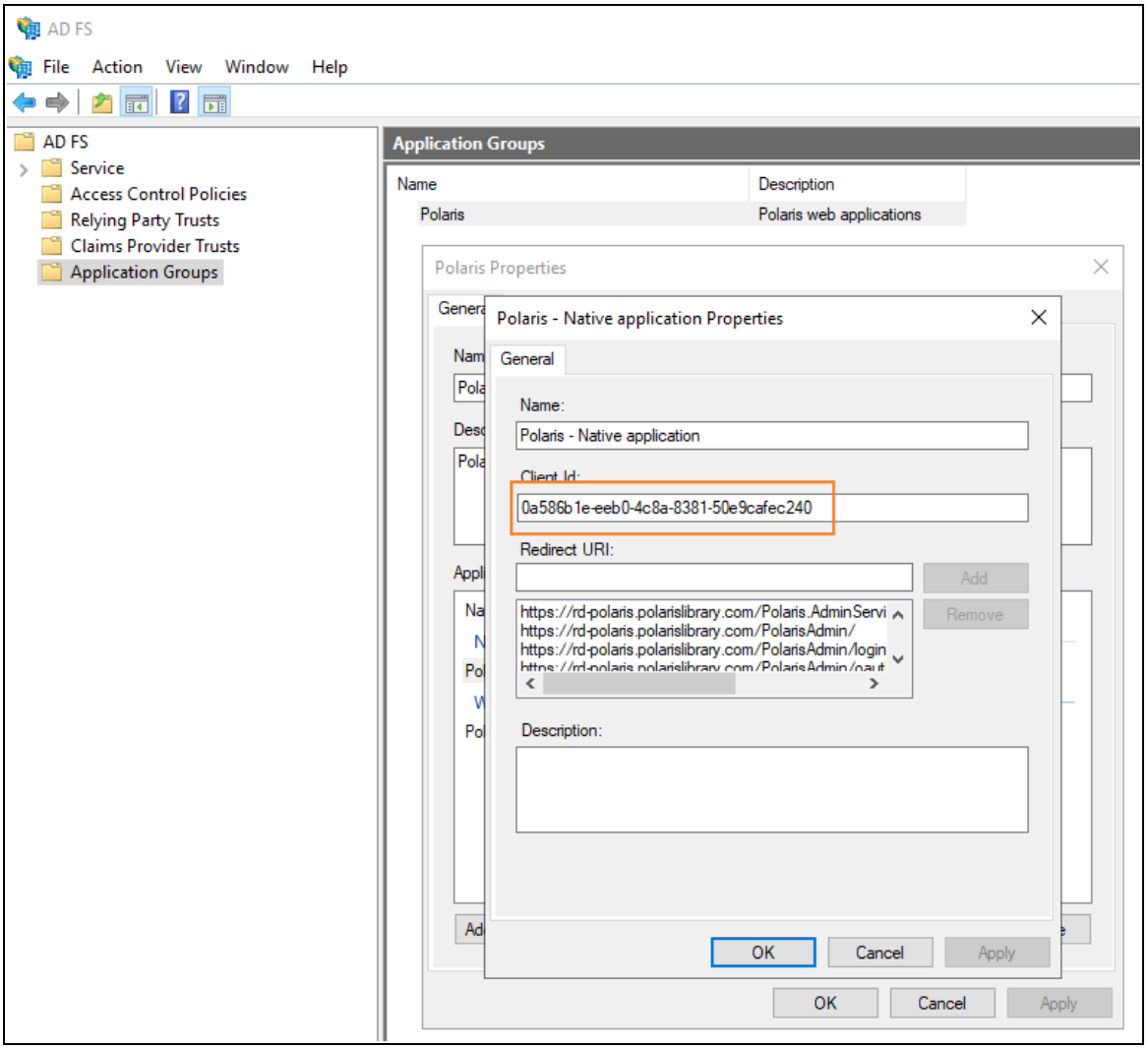
To configure LeapWebApp

1. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
    - C:\Program Files\Polaris\8.1\LeapWebApp\appsettings.user.json
    - C:\Program Files\Polaris\8.1\LeapWebApp\RELEASE-NOTES.md
2. In the RELEASE-NOTES.md file, copy the settings in the OIDC/OAuth Setup for AD FS section. The image below shows the settings to copy.

```
### OIDC/OAuth Setup for AD FS

```
  "OAuth": {
    "Authority": "https://dev-fs.polarislibrary.com/adfs/",
    "AuthorizationEndpoint": null,
    "TokenEndpoint": null,
    "UserInformationEndpoint": null,
    "ClientId": "3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
    "ClientSecret": null,
    "MetadataAddress": "https://dev-fs.polarislibrary.com/adfs/.well-known/openid-configuration",
    "KnownAuthorities": [ "dev-fs.polarislibrary.com" ],
    "CallbackPath": "/signin-oidc",
    "SignedOutCallbackPath": "/signout-callback-oidc",
    "SignedOutRedirectUri": "/login",
    "RemoteAuthenticationTimeout": 1,
    "RemoteFailureRedirectUri": "/leapwebapp/logout",
    "ResponseMode": "form_post",
    "ResponseType": "code",
    "SaveTokens": false,
    "Scopes": [ "openid", "profile" ],
    "UseOIDC": true,
    "UsePkce": true,
    "AlternateUpnClaimType": null,
    "AlternateLogoutUri": null,
    "OptionalAuthorizeParameters": null,
    "OptionalEndSessionParameters": null,
    "SendAccessTokenAsHeaderValue": false,
    "AccessTokenHeaderName": "",
    "BasicAuthorizationCredentials": {
      "Username": null,
      "Password": null
    }
  },
```

3. In the .json file, replace the OAuth contents of the file with the settings you copied from the RELEASE-NOTES.md file.
4. Update the following properties. In each, replace *dev-fs.polarislibrary.com* with the AD FS server address:
   - `Authority`
   - `MetadataAddress`
   - `KnownAuthorities`
5. On the AD FS server, open AD FS Management desktop application.
```



6. Copy the client ID from the Polaris - Native application properties dialog.
7. In the .json file, update the `ClientId` property. Replace *`3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a`* with the client ID copied from the AD FS Management desktop application.
8. Save the .json file.

Configure Polaris.ApplicationServices for Use with AD FS

To configure Polaris.ApplicationServices, update C:\Program Files\Polaris\8.1\Polaris.ApplicationServices\appsettings.user.json. You will use several values copied from your identity provider.

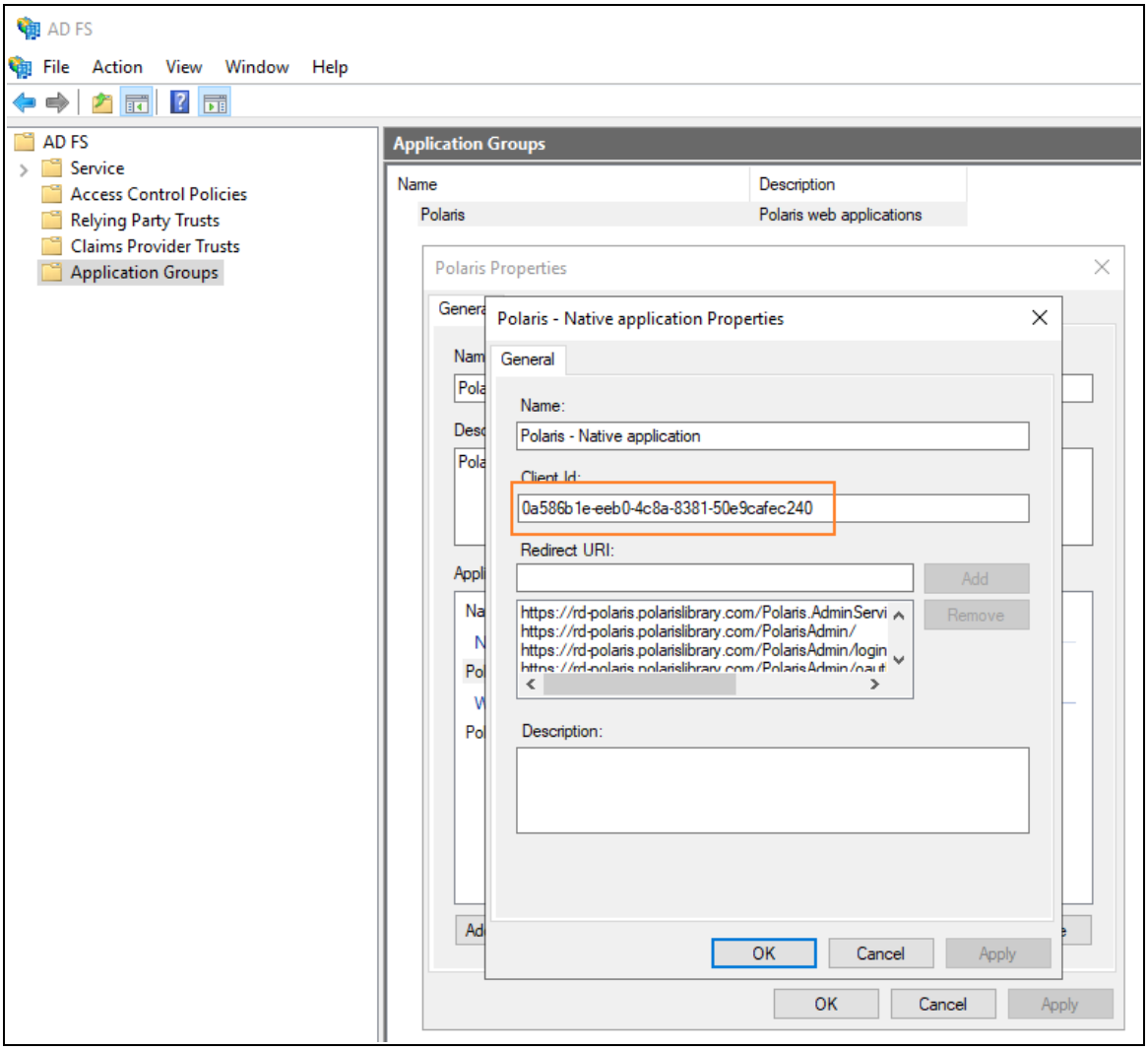
To configure Polaris.ApplicationServices

1. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
    - C:\Program Files\Polaris\8.1\Polaris.ApplicationServices\appsettings.user.json
    - C:\Program Files\Polaris\8.1\Polaris.ApplicationServices\RELEASE-NOTES.md
2. In the RELEASE-NOTES.md file, copy the settings in the **Example OAuth2 Settings** section. The image below shows the settings to copy.

Example OAuth2 Settings:

```
  "OAuth": {
      "Enabled": false,
      "Authorities": [
        {
          "Name": "ADFS",
          "Authority": "https://dev-fs.polarislibrary.com/adfs/",
          "Audience": "microsoft:identityserver:3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
          "MetaAddress": "https://dev-fs.polarislibrary.com/adfs/.well-known/openid-configuration",
          "RequireHttpsMetadata": true,
          "RequireSignedTokens": true,
          "ValidateIssuer": true,
          "ValidIssuers": [
            "https://dev-fs.polarislibrary.com/adfs",
            "http://dev-fs.polarislibrary.com/adfs/services/trust"
          ],
          "ValidateAudience": true,
          "ValidAudiences": [
            "3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
            "microsoft:identityserver:3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a"
          ],
          "UPNClaimTypes": [ "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn" ],
          "IsUPNExternalID": false,
          "OpaqueToken": false,
          "UserInformationEndpoint": null
        }
      ],
      "Swagger": {
        "ClientID": "3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
        "ClientSecret": "",
        "AppName": "Polaris.ApplicationServices",
        "AuthorizationUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/authorize",
        "TokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token",
        "RefreshTokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token",
        "LogoutUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/logout",
        "Scopes": [
          { "Name": "openid", "Description": "Use OIDC to verify the user's identity" },
          { "Name": "email", "Description": "Optional to return user's email address" },
          { "Name": "urn:microsoft:userinfo", "Description": "urn:microsoft:userinfo" }
        ]
      }
    },
```

3. On the AD FS server, open AD FS Management desktop application.



4. Copy the client ID from the Polaris - Native application properties dialog. You will use this value in the steps below.
5. In the `Authorities` property, update the following properties:
    a. In the `Authority` property, replace *`dev-fs.polarislibrary.com`* with the AD FS server address.
    b. In the `Audience` property, replace *`3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a`* with the client ID copied from the AD FS Management desktop application.

c. In the `MetaAddress` property, replace *`dev-fs.polarislibrary.com`* with the AD FS server address.

d. In both URLs in the `ValidIssuers` property, replace *`dev-fs.polarislibrary.com`* with the AD FS server address.

e. In both values in the `ValidAudiences` property, replace *`3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a`* with the client ID copied from the AD FS Management desktop application.

f. In the `UpnClaimTypes` property, update the default value if you want to specify a different claim to serve as the user identifier. The default value applies to most configurations, but you can specify a different claim that exists in a JSON Web Token (JWT).

6. In the `Swagger` property, update the following properties with information from your identity provider:

a. In the `ClientID` property, replace *`3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a`* with the client ID copied from the AD FS Management desktop application.

b. In the `AuthorizationUrl` property, replace *`dev-fs.polarislibrary.com`* with the AD FS server address.

c. In the `TokenUrl` property, replace *`dev-fs.polarislibrary.com`* with the AD FS server address.

d. In the `RefreshTokenUrl` property, replace *`dev-fs.polarislibrary.com`* with the AD FS server address.

e. In the `LogoutUrl` property, replace *`dev-fs.polarislibrary.com`* with the AD FS server address.

7. Save the .json file.

Customize the AD FS Pages

Use the following resources to customize AD FS pages:

- https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn280950(v=ws.11)

- Get-AdfsGlobalWebContent
- Set-AdfsGlobalWebContent

> **Examples:**
> Customize the examples below to suit your library's needs.

```
PS C:\Windows\system32> Set-AdfsGlobalWebContent -SignOutPageDescriptionText "You have successfully signed out.<br>If you have been directed here immediately after signing in, your session may have timed out."

PS C:\Windows\system32> Set-AdfsWebTheme -TargetName default -Logo @{path="c:\ADFS Custom\leap_logo.png"}

PS C:\Windows\system32> Set-AdfsGlobalWebContent -CompanyName "Polaris R&D"
```

- Advanced customization:
  - https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn636121(v=ws.11)

Change the Access Token Lifetime

The default token lifetime for both access and ID tokens is 60 minutes. Execute the following command to increase the expiration time to 24 hours:

```
Set-AdfsWebApiApplication -TokenLifetime 1440 -TargetIdentifier "0a586b1e-eeb0-4c8a-8381-50e9cafec240"
```

> **Note:**
> Replace *TargetIdentifier* with the Polaris Application Group native application client ID.

Bind a New SSL Certificate

If your web server certificate expires, use the instructions below to bind a new SSL certificate.

To bind a new SSL certificate

1. Install the certificate using Certificates Management.
2. Set the service communications certificate using the AD FS Management Console:
    a. Expand the Services folder.
    b. Select a new certificate.
    c. Restart the AD FS service.
3. Attach the certificate to AD FS using PowerShell:
    a. Get the certificate's thumbprint by viewing the certificate.

    ```
    c:\> Set-AdfsSslCertificate –Thumbprint
    e8fd5016542796214e94f72d76095f9fc587c731
    ```

    b. Restart the AD FS service.

Troubleshoot

Force a logout

- https://*AD FS server address*/adfs/oauth2/logout

> **Note:**
> Replace *AD FS server address* with your library's AD FS server address.

AD FS in one-way trust

Problem: Only local accounts are authenticating

Solution: Make sure the account running the AD FS service is a parent domain account and not a local account.

Receiving "User is not a valid Polaris user." error

- Check the setting Polaris.OAuth.ValidIssuer in the Polaris.AdminServices appsettings.user.json file.

  Example value: http://*AD FS server address*/adfs/services/trust

> **Note:**
> Replace *AD FS server address* with your library's AD FS server address.

- Verify a domain is attached to AD user accounts so the UPN claim can be added to the ID token's claims.

  The UPN claim should look like user@mydomain.com.

Troubleshoot Redirect URIs

Redirect URIs are case-sensitive.

Configuring OIDC with Azure AD

Important:

The mechanism used to connect an Azure AD user to a Polaris user is the user principal name (UPN) in the format of an email address. For example, user@mydomain.com.

During the account verification process, we use the `openid` and `profile` scopes, which triggers Azure AD to return the `upn` claim or the `preferred_username` claim (or both). These must be returned in the name@domain format. The Polaris.ApplicationServices (API) can then use that information to map the Azure AD user to a Polaris user. If the `preferred_username` is a generic name, phone number, or other value, you can choose to apply the `email` scope to return the email.

See Configure LeapWebApp for Use with Azure AD for more information.

Note:
Azure AD has recently been rebranded as "Microsoft Entra ID". In this guide, references to Azure AD are interchangeable with Microsoft Entra ID.

To configure OIDC with Azure AD, perform the following tasks:

1. Register LeapWebApp with Azure AD.
2. Create client credentials.
3. Add authentication redirect URIs.
4. Expose the Polaris.ApplicationServices API.
5. Configure an ID token.
6. Set up users and groups.
7. Control access to LeapWebApp using Azure AD.
8. Set up web services and applications.

After you complete these tasks, Add a URL rewrite rule for LeapWebApp.

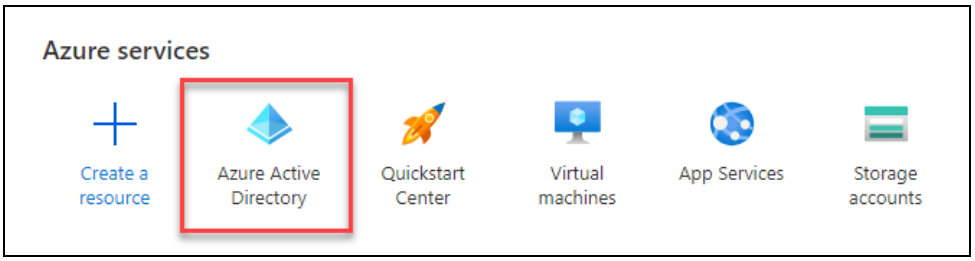
Register LeapWebApp with Azure AD

To register LeapWebApp with Azure AD

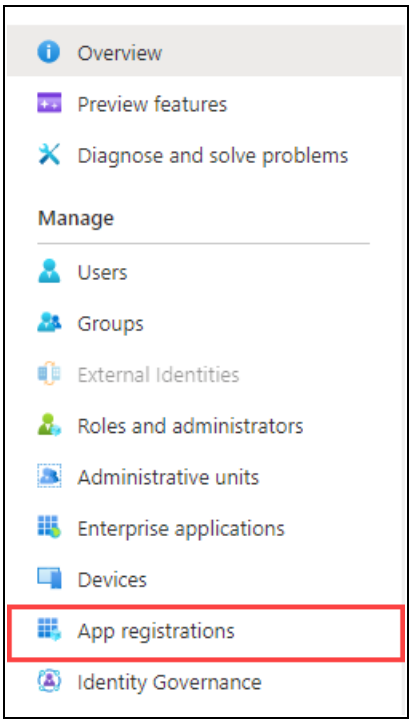
1. Sign in to the Azure portal:

   https://portal.azure.com/

2. In the **Azure services** list, select **Azure Active Directory**.



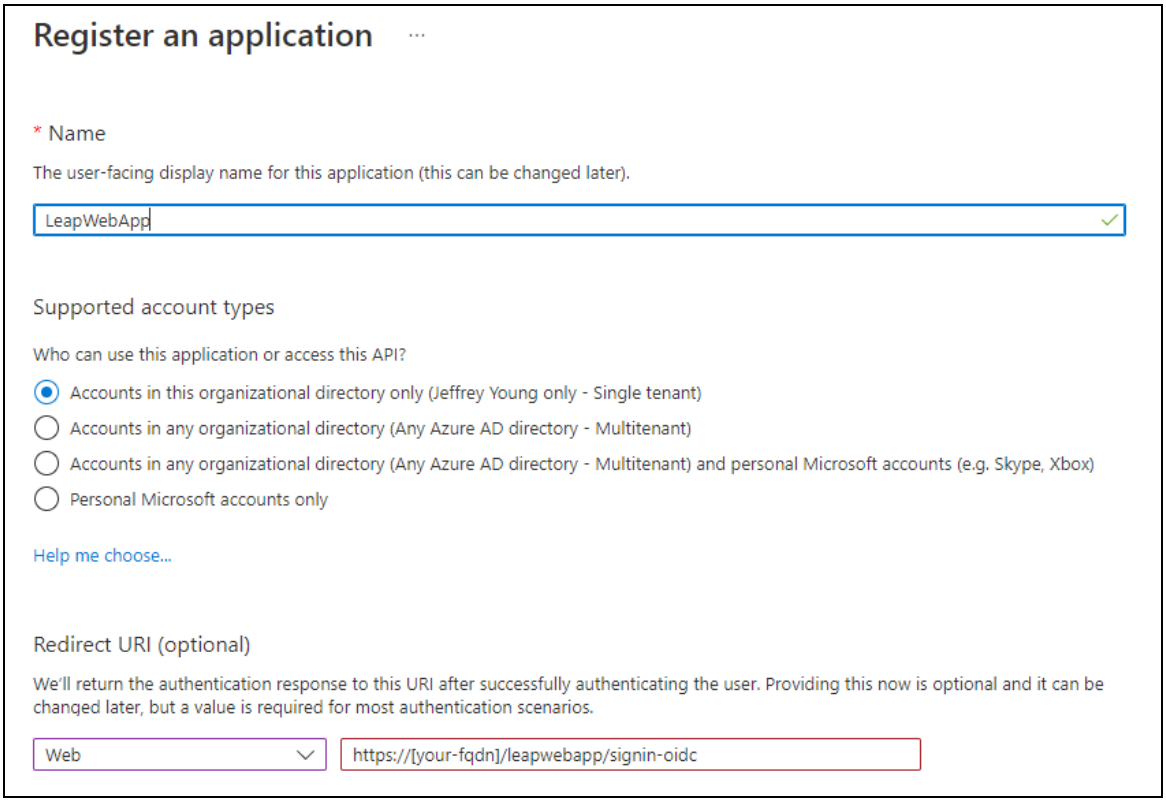
3. In the list of options at the left side of the screen, select **App registrations**.



   The App registrations page appears.

4. Select **New registration**.

   The Register an application dialog appears.



5. Enter "LeapWebApp" in the **Name** field.
6. Select an option from the **Supported account types** list.
7. Add a redirect URI:
    a. Select the **Web** URI type.
    b. Enter an address that uses the following format:

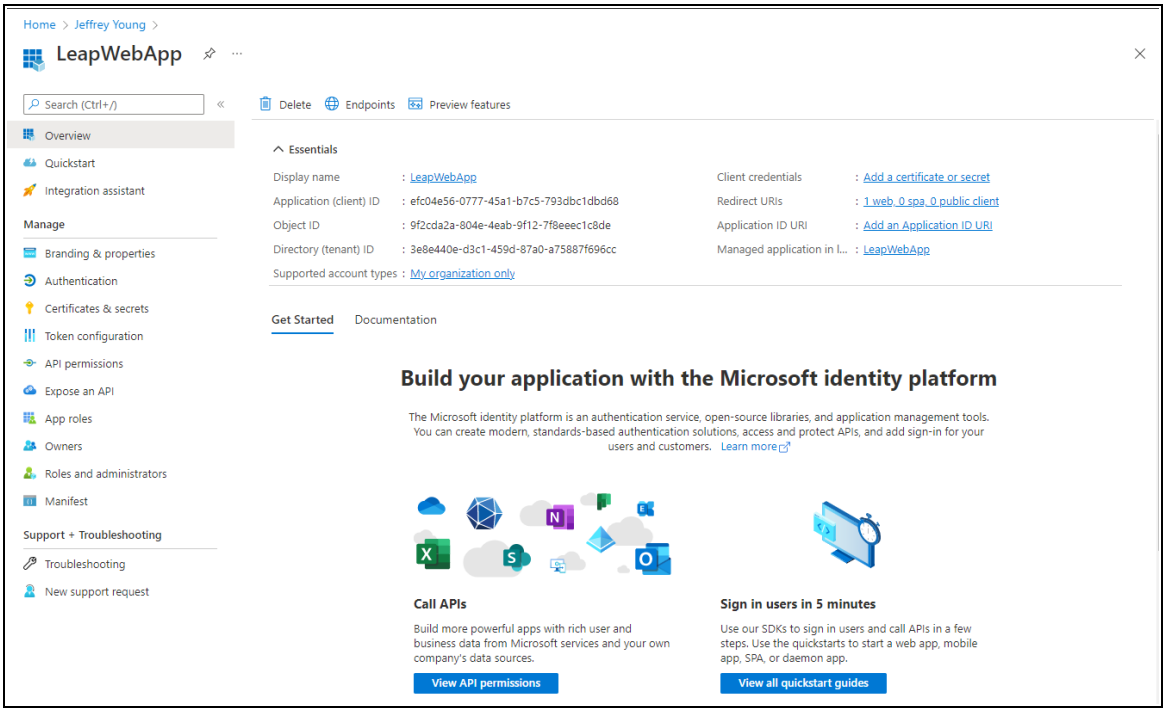
    https://*[FQDN]*/leapwebapp/signin-oidc

    Notes:

    - Replace *[FQDN]* with the fully-qualified domain name of your LeapWebApp server.
    - Example: https://leap.mylibrary.org/leapwebapp/signin-oidc

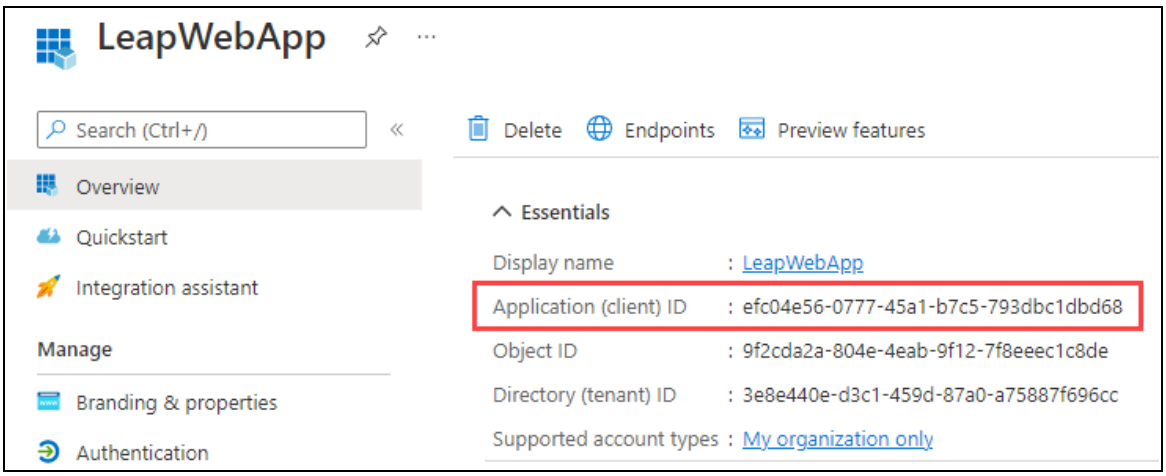
8. Select **Register**.

    The page for your new LeapWebApp application appears.



9. Copy the application (client) ID and paste it into Notepad (or a similar text editor) and save the file. You must have this value to complete several procedures later in the Azure AD configuration process.

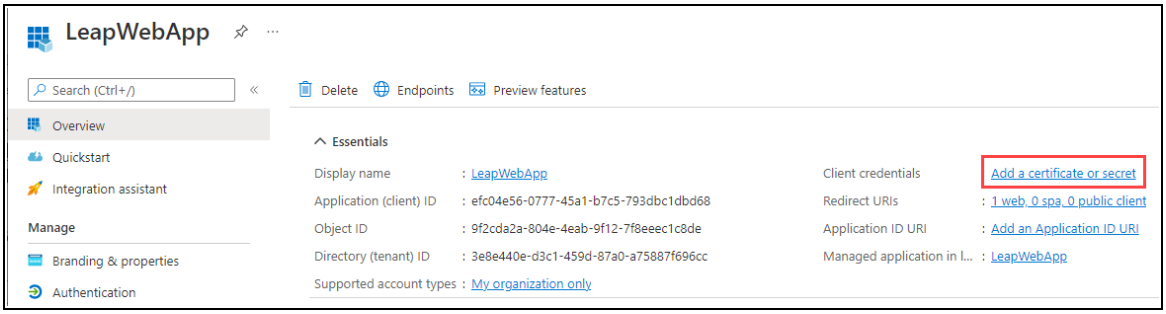
   In the example below, the application (client) ID is "efc04e56-0777-45a1-b7c5-793dbc1dbbd68".



Create Client Credentials

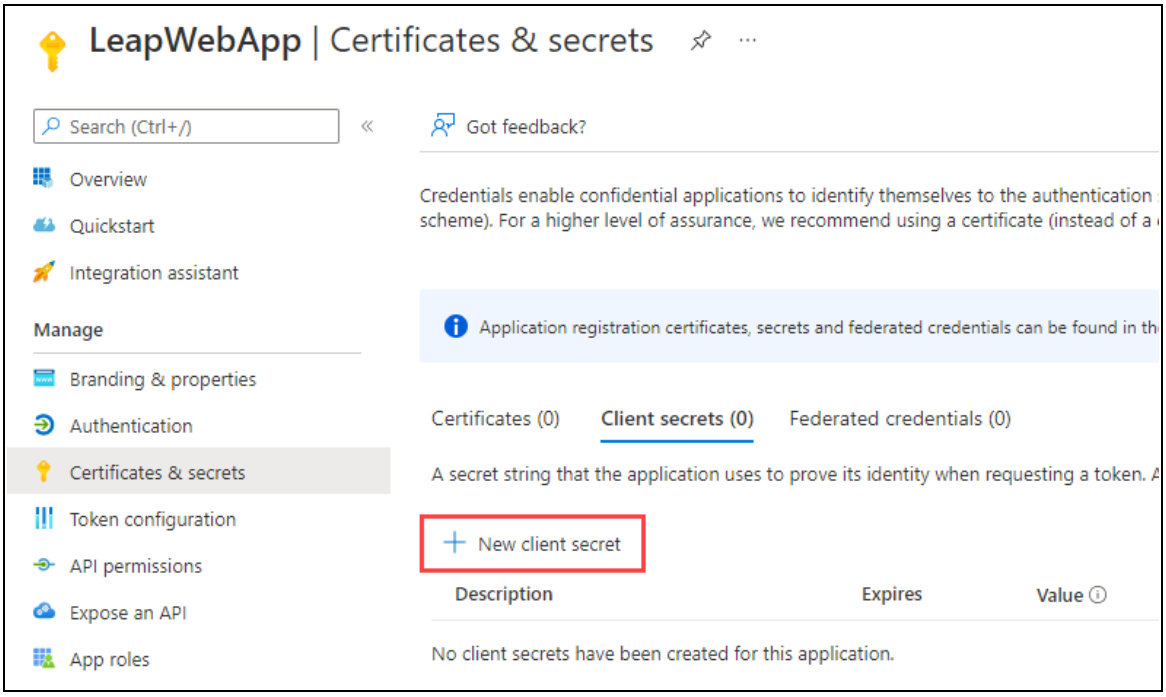
To create client credentials for LeapWebApp

1. On the LeapWebApp page, select **Add a certificate or secret**.



   The Certificates & secrets page appears.

2. Select the **Client secrets** tab.
3. Select **New client secret**.



   The Add a client secret dialog appears.

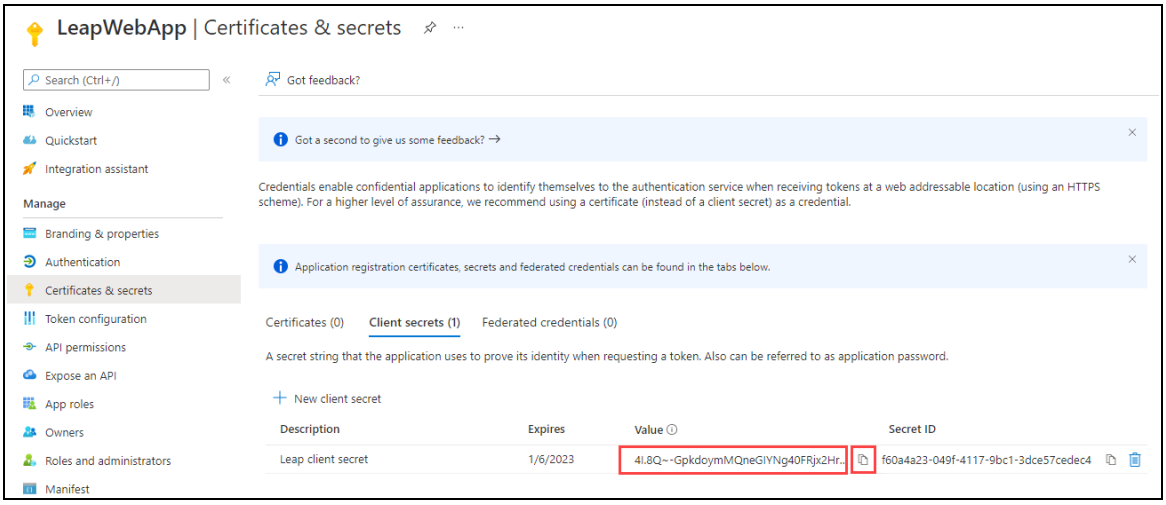
4. Enter a description in the **Description** field.
5. Select an option from the **Expires** list to specify when the client credentials expire.
6. Select **Add**.

   The Add a client secret dialog closes. The client secret for LeapWebApp appears on the **Client secrets** tab of the Certificates & secrets page.
7. Copy the text in the **Value** column, then paste it into Notepad (or a similar text editor) and save the file. You must have this value to complete the Configure LeapWebApp for Use with Azure AD procedure.

   Important:

   - You must save this value now. Only a portion of the value appears when you return to the page later.
   - Use the copy icon to be sure that you are copying the entire value.

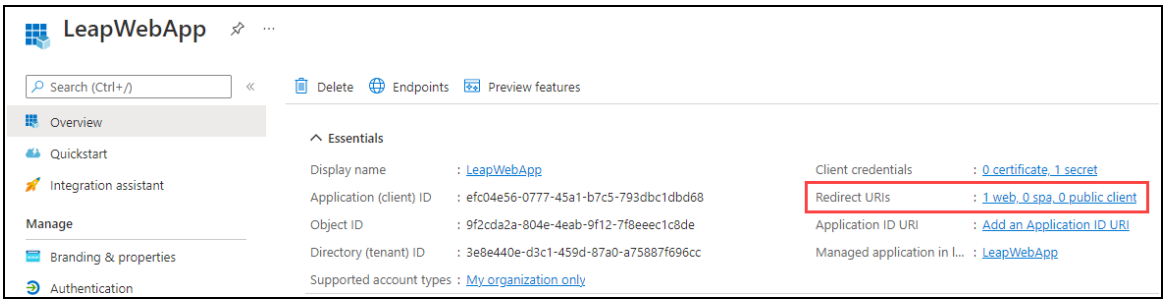
   In the example below, the value is "4I.8Q~-GpkdoymMQneGIYNg40FRjx2Hr1wWLDcbr".



Add Authentication Redirect URIs

To add authentication redirect URIs

1. On the LeapWebApp page, select the link beside **Redirect URIs**.



   The Authentication page appears.

2. Select **Add a platform**.

   The Configure platforms dialog appears.

3. Select the **Single-page application** tile.

   The Configure single-page application dialog appears.

4. In the **Redirect URIs** field, enter the following required redirect URIs:

   - https://*[FQDN]*/polarisauth/login
   - https://*[FQDN]*/polarisauth/logout
   - https://*[FQDN]*/polarisauth/signin-oidc

   Notes:

   - Replace *[FQDN]* with the fully-qualified domain name of your LeapWebApp server.
   - Example: https://leap.mylibrary.org/polarisauth/login

5. If you are using permission overrides or reauthentication in Leap, enter the following additional redirect URIs in the **Redirect URIs** field:

   - https://*[FQDN]*/leapwebapp/signin-override-oidc
   - https://*[FQDN]*/leapwebapp/silent-logout-msal

6. If you are using Swagger, enter the following additional redirect URIs in the **Redirect URIs** field:

- https://*[FQDN]*/Polaris.ApplicationServices/swagger/oauth2-redirect.html
- https://*[FQDN]*/Polaris.AdminServices/swagger/oauth2-redirect.html

7. Select **Save** to save the redirect URIs.
8. Return to the Configure platforms dialog.
9. Select the **Web** tile.

   The Configure web dialog appears.
10. In the **Redirect URIs** field, enter the following redirect URIs:

    - https://*[FQDN]*/polarisauth/signin-oidc

      https://*[FQDN]*/polarisauth/login

      Notes:

      - Replace *[FQDN]* with the fully-qualified domain name of your LeapWebApp server.
      - Example: https://leap.mylibrary.org/polarisauth/signin-oidc
11. Select **Configure**.

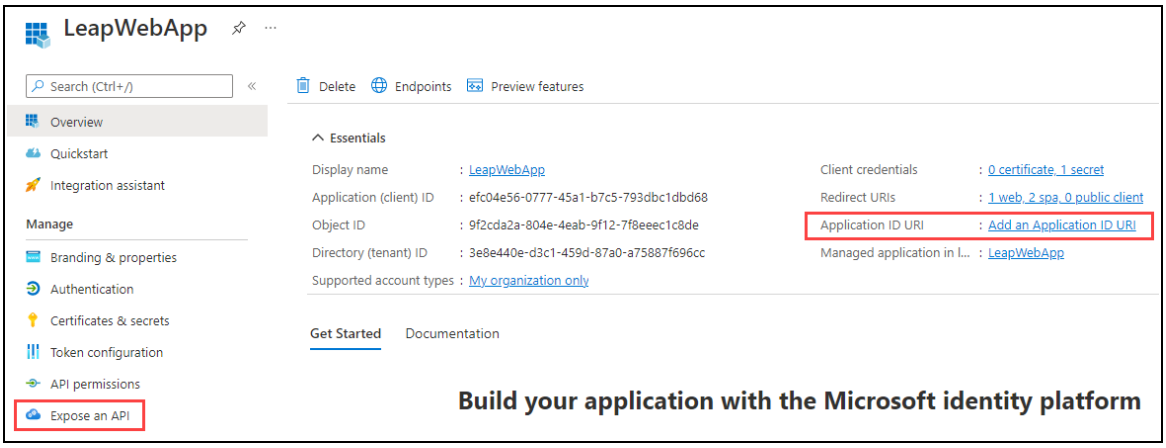
    The new redirect URI appears on the Authentication page in the **Web** list.
12. Select **Save** to save the web redirect URIs.

Expose the Polaris.ApplicationServices API

To expose the Polaris.ApplicationServices API

1. On the LeapWebApp page, select **Add an Application ID URI**.

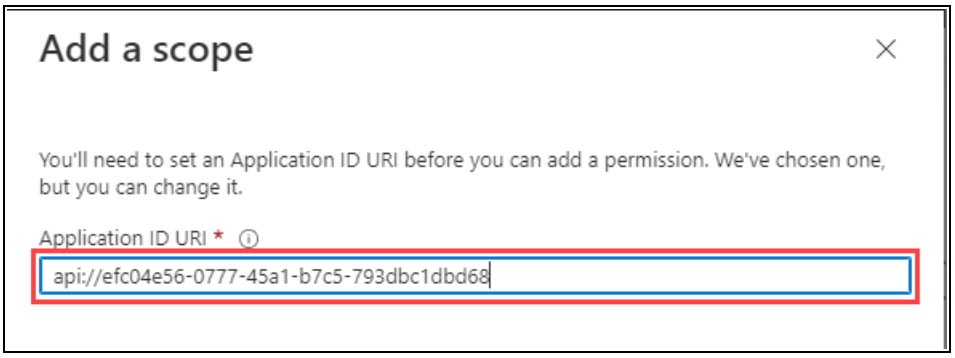
   > **Note:**
   > You can also select **Expose an API** in the list of options at the left side of the screen.



   The Expose an API page appears.

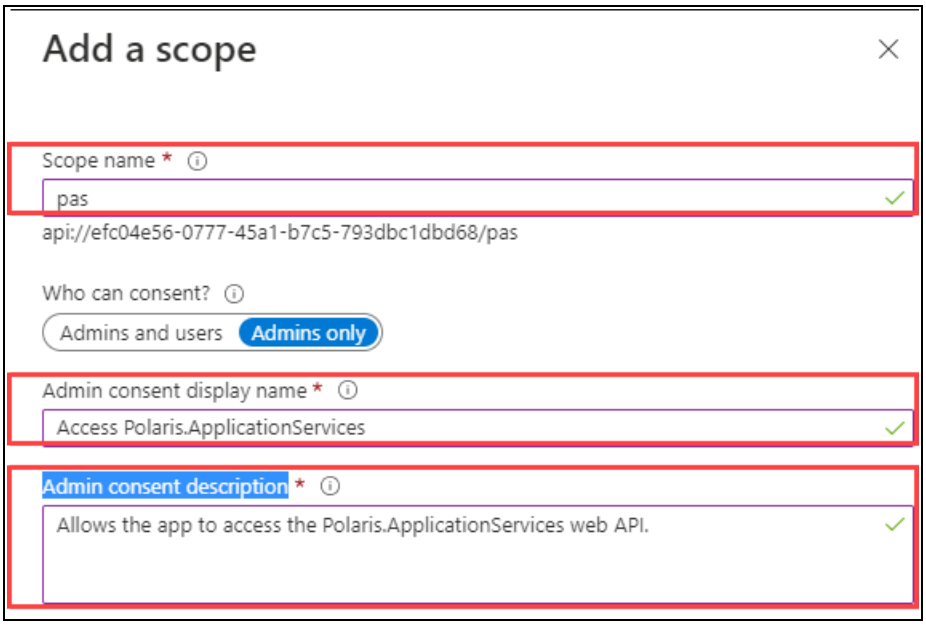
2. Select **Add a scope**.

   The Add a scope dialog appears. The **Application ID URI** field contains an automatically-generated URI.



3. Select **Save and continue**.

The Add a scope dialog refreshes.



4. Enter "pas" in the **Scope name** field.
5. Enter "Access Polaris.ApplicationServices" in the **Admin consent display name** field.
6. Enter "Allows the app to access the Polaris.ApplicationServices web API." in the **Admin consent description** field.
7. Select **Add scope**.

   The Azure portal saves the scope and closes the Add a scope dialog.
8. On the Expose an API page, select **Add a client application**.

   The Add a client application dialog appears.

9. In the **Client ID** field, enter the Application (client) ID that you copied and saved during the Register LeapWebApp with Azure AD procedure.
10. Select the **Authorized scopes** checkbox.
11. Select **Add application**.

    The Azure portal saves your changes and closes the Add a client application dialog.
12. On the Expose an API page, copy the new scope, then paste it into Notepad (or a similar text editor) and save the file. Your value will be similar to this one:

    api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas

    You must have this value to complete two procedures later in the Azure AD configuration process.

Configure an ID Token

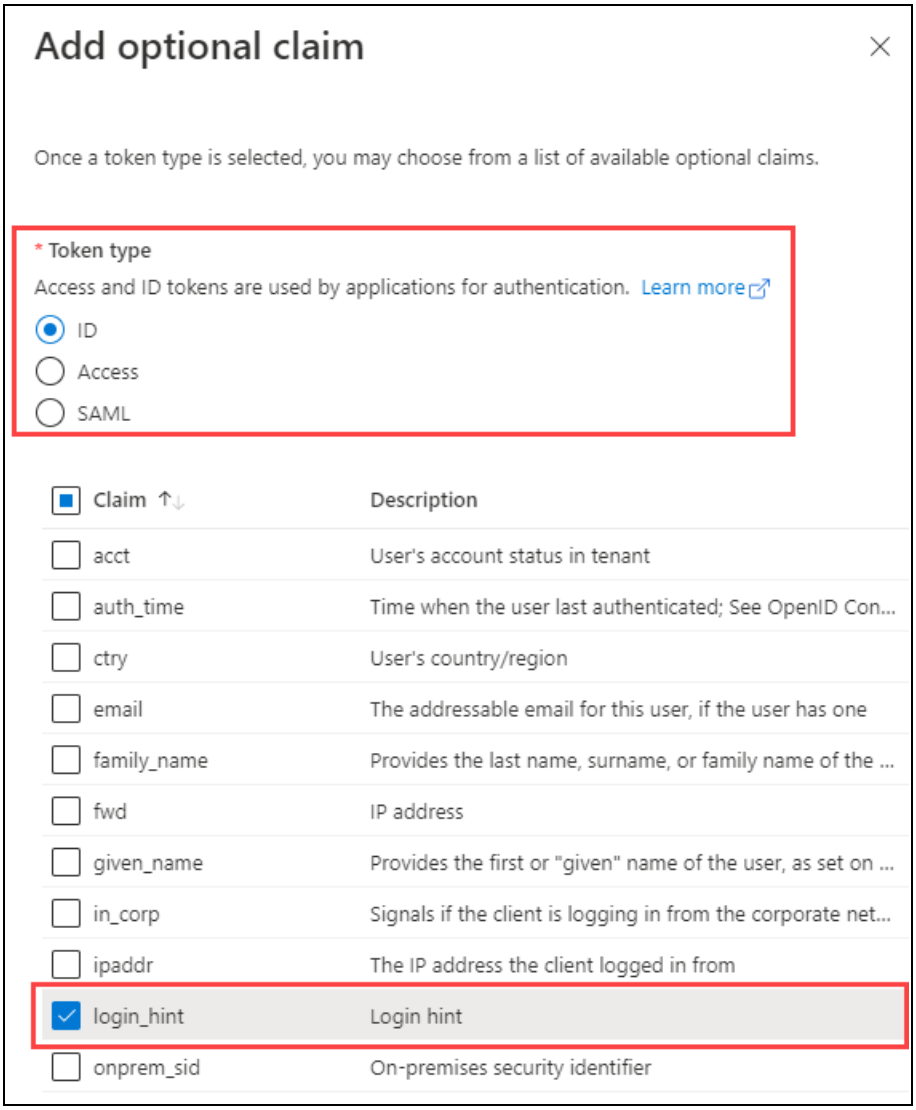
To allow Leap to sign out of specific accounts, you must add an ID token that contains the login_hint claim.

To configure an ID token

1. On the LeapWebApp Overview page, select **Token configuration** from the list of options at the left side of the screen.

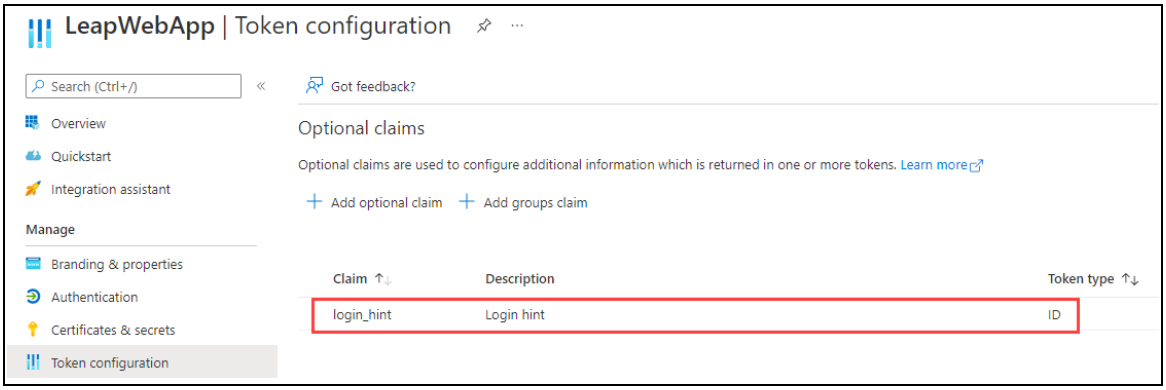
   The Token configuration page appears.
2. Select **Add optional claim**.

   The Add optional claim dialog appears.



3. Set the **Token type** setting to the **ID** option.
4. Select the **login_hint** checkbox.
5. Select **Add**.

   The Azure portal saves the token and closes the Add optional claim dialog.
6. Verify that the new login_hint claim appears on the Token configuration page.



Set Up Users and Groups

To set up users and groups

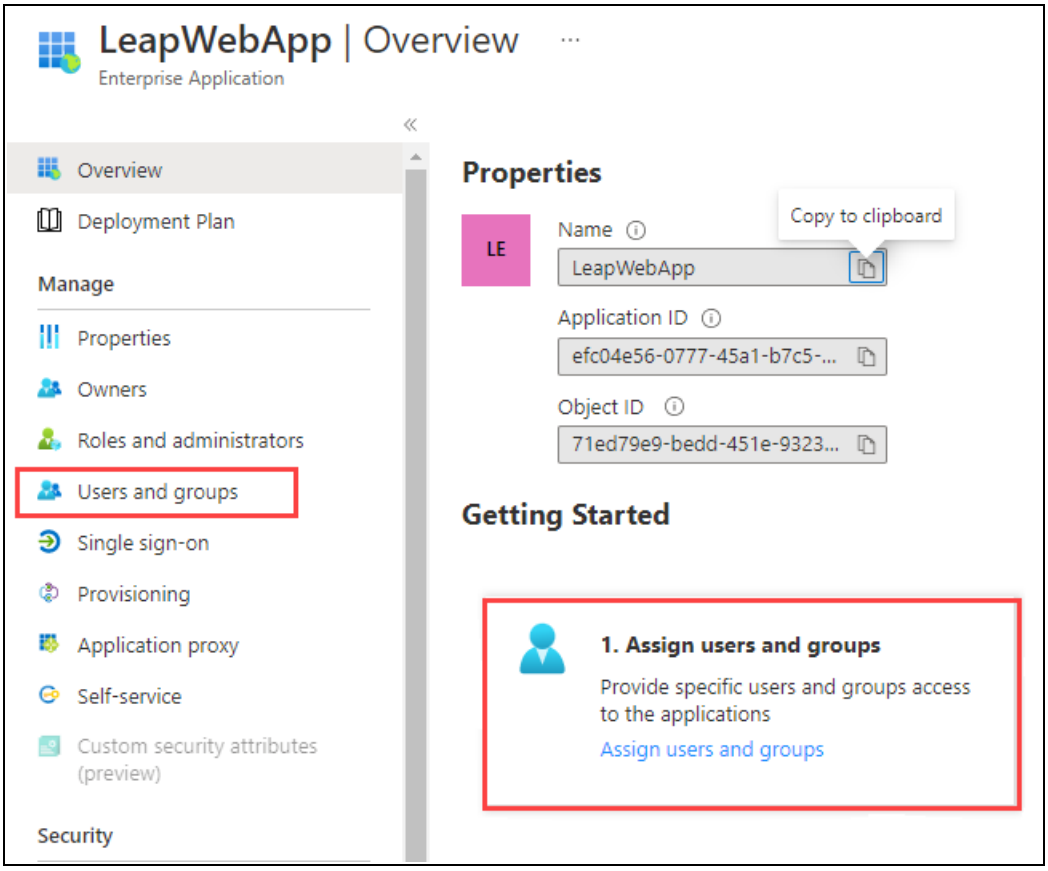
1. On the Azure AD Overview page, select the **Enterprise applications** option from the list at the left side of the screen.

   > **Note:**
   > You can also select **Enterprise applications** from the list of services on the Azure portal home page.

   The All applications page appears.

2. Select the **LeapWebApp** link.

   The LeapWebApp Overview page appears.



3. Select the **Users and groups** option from the list at the left side of the screen. You can also select the **Assign users and groups** tile.

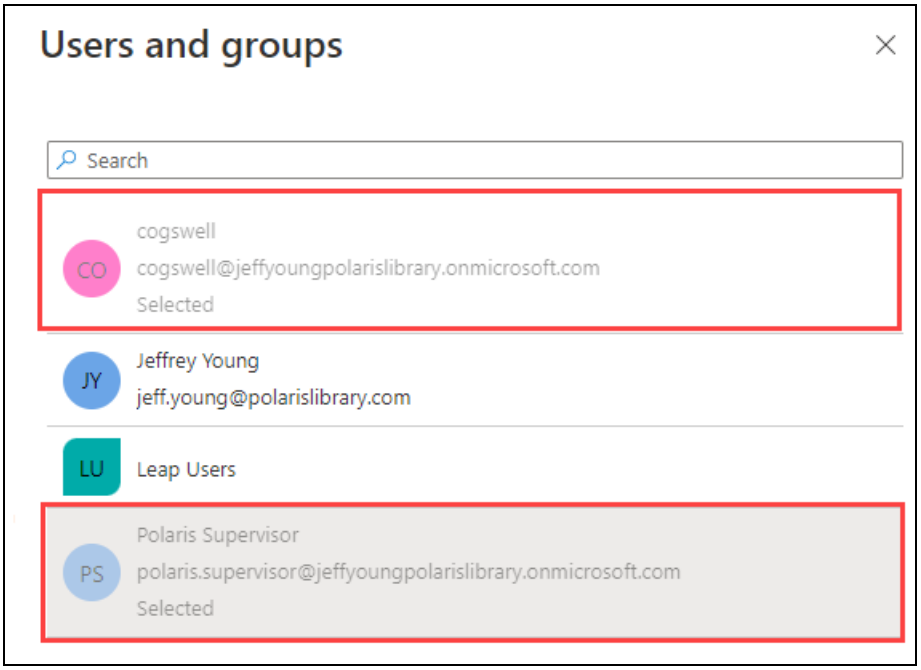
   The Users and groups page appears.

4. Select **Add user/group**.

   The Add Assignment page appears.

5. Select the **None Selected** link.

   The Users and groups dialog appears.



6. Select the users and groups that you want to allow access to LeapWebApp.
7. Click **Select**.

   The Users and groups dialog closes.
8. On the Add Assignment page, select **Assign**.

   The Azure portal saves the user and group assignments.

Control Access to LeapWebApp Using Azure AD

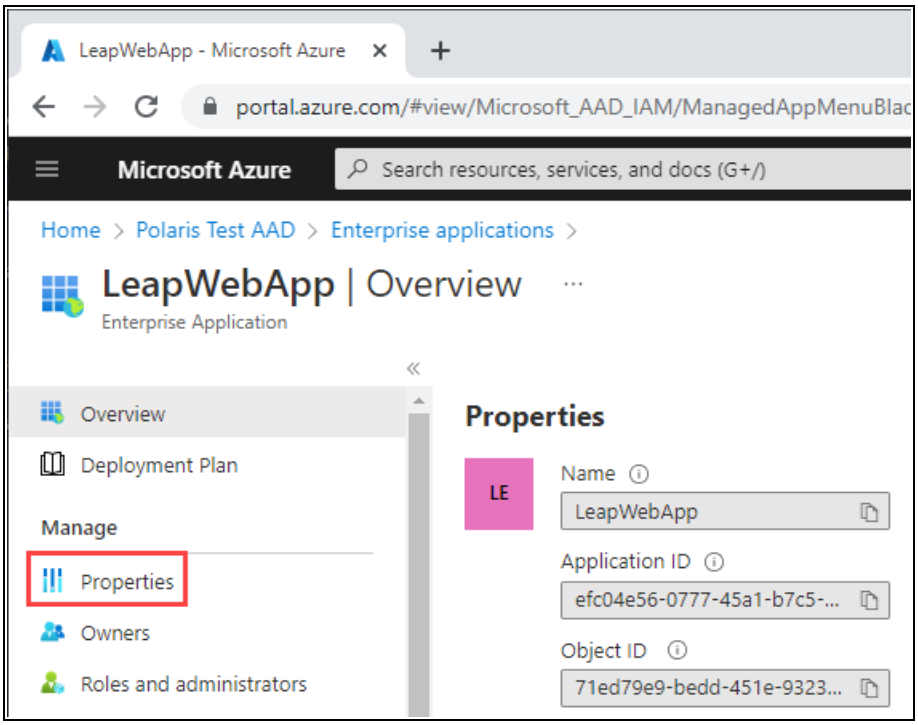
To control access to LeapWebApp using Azure AD

1. On the Azure AD Overview page, select the **Enterprise applications** option from the list at the left side of the screen.

   > **Note:**
   > You can also select **Enterprise applications** from the list of services on the Azure portal home page.

   The All applications page appears.
2. Select the **LeapWebApp** link.

The LeapWebApp Overview page appears.

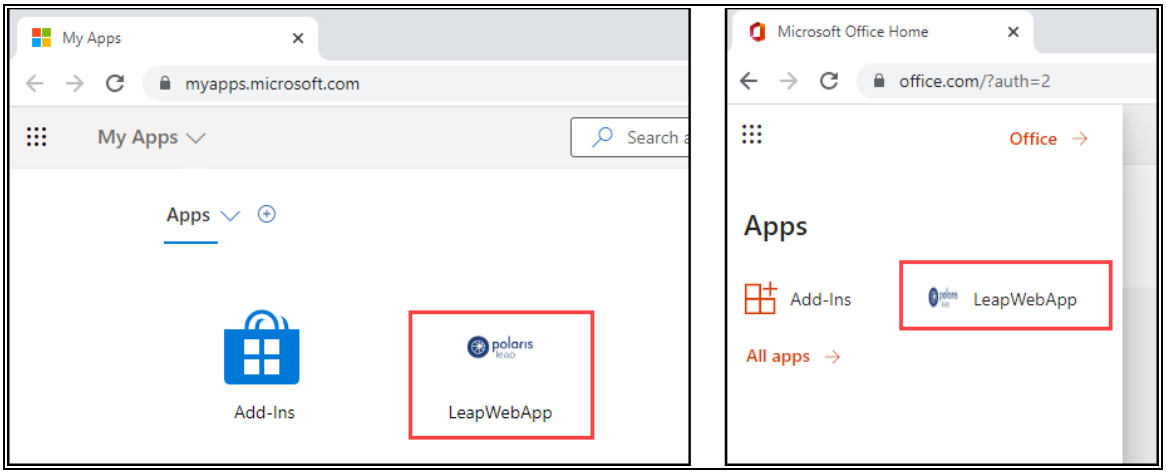


3. Select the **Properties** option from the list at the left side of the screen.

   The Properties page appears.

4. Set the **Assignment requirement?** setting to **Yes**. This allows access to be controlled by the users and groups assigned to the LeapWebApp enterprise application. (When it is set to **No**, all users can sign in.)

5. Set the **Visible to users?** setting to **Yes**. This makes the LeapWebApp application visible to users in their Microsoft My Apps portal and on their Office 365 page.



Important:
When a user accesses LeapWebApp from the Microsoft My Apps portal or their Office 365 page, they might have to click the Polaris Leap Sign In button. This is because cookies are a part of the Leap authentication process.

6. Select **Save**.

   The Azure portal saves your changes.

Set Up Web Services and Applications for OIDC with Azure AD

To set up each of the following web services and applications, you must configure a .json file for each of the following:

- Polaris.Authentication (the application that authenticates Polaris users)
- Polaris.AuthenticationServices (the API service that provides backend support for authentication)
- PolarisAdmin (the web-based Polaris System Administration application)
- LeapWebApp (Leap)
- Polaris.ApplicationServices (Leap's API service)

The five .json files are named appsettings.user.json, but they reside in different directories:

- C:\Program Files\Polaris\8.1\Polaris.Authentication
- C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices
- C:\Program Files\Polaris\8.1\PolarisAdmin\assets
- C:\Program Files\Polaris\8.1\LeapWebApp
- C:\Program Files\Polaris\8.1\Polaris.ApplicationServices

This section contains the following topics:

Configure Polaris.Authentication for Use with Azure AD

To configure Polaris.Authentication, update C:\Program Files\Polaris\8.1\Polaris.Authentication\appsettings.user.json. You will use several values copied from your identity provider.

> **Important:**
> By default, the appsettings.user.json template file contains configuration settings that apply to AD FS. Polaris 8.1 includes a RELEASE-NOTES.md file that contains the template settings that apply to Azure AD. See the Azure AD Example section of the release notes file.

To configure Polaris.Authentication

1. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
    - C:\Program Files\Polaris\8.1\Polaris.Authentication\appsettings.user.json
    - C:\Program Files\Polaris\8.1\Polaris.Authentication\RELEASE-NOTES.md
2. In the RELEASE-NOTES.md file, copy the settings in the Azure AD Example. The image below shows the settings to copy.

```
### AzureAD Example

```
"OAuth": {
  "Enabled": true,
  "SendOAuthAuthorityHeader": false,
  "Authorities": [
    {
      "IsActive": true,
      "Name": "AzureAD",
      "UseOidc": true,
      "UsePkce": true,
      "EndSessionEndpoint": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-
a75887f696cc/oauth2/v2.0/logout",
      "ClientId": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
      "ClientSecret": "YLU8Q~TGdT_KIcDZ5px643zcsLmCzr16S1jiKcZI",
      "CallbackPath": "/signin-oidc",
      "SignedOutCallbackPath": "/signout-callback-oidc",
      "SignedOutRedirectUri": "/login",
      "Authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/",
      "MetaAddress": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/.well-
known/openid-configuration",
      "ResponseMode": "form_post",
      "ResponseType": "code",
      "Scopes": [
        "profile",
        "openid"
      ],
      "SaveTokens": false
    }
  ]
}
```
```

3. In the .json file, replace the entire contents of the file with the settings you copied from the RELEASE-NOTES.md file.
4. In the `EndSessionEndpoint` property, make the following updates:
    a. Replace *`login.microsoftonline.com`* with your Azure AD server address.
    b. Replace *`3e8e440e-d3c1-459d-87a0-a75887f696cc`* with the tenant ID copied from the Azure portal.
5. In the `ClientId` property, replace *`efc04e56-0777-45a1-b7c5-793dbc1dbd68`* with the application (client ID) you copied during the Register LeapWebApp with Azure AD step.
6. In the `ClientSecret` property, replace *`YLU8Q~TGdT_KIcDZ5px643zcsLmCzr16S1jiKcZI`* with the client secret you copied during the Create Client Credentials step.
7. In the `Authority` property, make the following updates:

   a. Replace *login.microsoftonline.com* with your Azure AD server address.

   b. Replace *3e8e440e-d3c1-459d-87a0-a75887f696cc* with the tenant ID copied from the Azure portal.

8. In the `MetaAddress` property, make the following updates:

   a. Replace *login.microsoftonline.com* with your Azure AD server address.

   b. Replace *3e8e440e-d3c1-459d-87a0-a75887f696cc* with the tenant ID copied from the Azure portal.

9. Save the .json file.

Configure Polaris.AuthenticationServices for Use with Azure AD

To configure Polaris.AuthenticationServices, update C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices\appsettings.user.json. You will use several values copied from your identity provider.

> **Important:**
> By default, the appsettings.user.json template file contains configuration settings that apply to AD FS. Polaris 8.1 includes a RELEASE-NOTES.md file that contains the template settings that apply to Azure AD. See the OIDC/OAuth Setup for AzureAD/Entra section of the release notes file.

To configure Polaris.AuthenticationServices

1. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
   - C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices\appsettings.user.json
   - C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices\RELEASE-NOTES.md
2. In the RELEASE-NOTES.md file, copy the settings in the OIDC/OAuth Setup for AzureAD/Entra section. The image below shows the settings to copy.

```
### OIDC/OAuth Setup for AzureAD/Entra

```
"OAuth": {
  "Enabled": true,
  "Authorities": [
        {
          "Name": "AzureAD",
          "UseOidc": true,
          "RequireHttpsMetadata": true,
          "RequireSignedTokens": true,
          "ValidateIssuer": true,
          "ValidateAudience": true,
          "Authority": "https://login.microsoftonline.com/0a586b1e-eeb0-4c8a-8381-50e9cafec240/",
          "Audience": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
          "MetaAddress": "https://login.microsoftonline.com/0a586b1e-eeb0-4c8a-8381-50e9cafec240/v2.0/.well-
known/openid-configuration",
          "ValidIssuers": [
                "https://login.microsoftonline.com/0a586b1e-eeb0-4c8a-8381-50e9cafec240",
                "https://sts.windows.net/0a586b1e-eeb0-4c8a-8381-50e9cafec240/"
          ],
          "ValidAudiences": [
                "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
                "microsoft:identityserver:0a586b1e-eeb0-4c8a-8381-50e9cafec240"
          ],
          "UpnClaimTypes": [ "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn" ],
          "IsUpnExternalId": false
        }
```

3. In the .json file, replace the entire contents of the file with the settings you copied from the RELEASE-NOTES.md file.
4. Verify that `Enabled` is set to `true`.
5. Verify that `UseOidc` is set to `true`.
6. In the `Authority` property, do the following:
    a. Replace *`login.microsoftonline.com`* with the server address for your identity provider.
    b. Replace *`0a586b1e-eeb0-4c8a-8381-50e9cafec240`* with the application (client) ID you copied during the Register LeapWebApp with Azure AD step.
7. In the `Audience` property, replace *`0a586b1e-eeb0-4c8a-8381-50e9cafec240`* with the application (client) ID copied from the Azure AD portal.
8. In the `MetaAddress` property, do the following:
    a. Replace *`login.microsoftonline.com`* with the server address for your identity provider.
```

   b. Replace *0a586b1e-eeb0-4c8a-8381-50e9cafec240* with the application (client) ID copied from the Azure AD portal.

9. In the `ValidIssuers` property, do the following:
   a. Replace *login.microsoftonline.com* with the server address for your identity provider.
   b. Replace *0a586b1e-eeb0-4c8a-8381-50e9cafec240* with the application (client) ID copied from the Azure AD portal.
10. In the `ValidAudiences` property, replace *0a586b1e-eeb0-4c8a-8381-50e9cafec240* with the application (client) ID copied from the Azure AD portal.
11. In the `UpnClaimTypes` property, update the default value if you want to specify a different claim to serve as the user identifier. The default value applies to most configurations, but you can specify a different claim that exists in a JSON Web Token (JWT).
12. Save the .json file.

Configure PolarisAdmin for Use with Azure AD

To configure PolarisAdmin, update C:\Program Files\Polaris\8.1\PolarisAdmin\assets\appsettings.user.json.

To configure PolarisAdmin

1. Open the C:\Program Files\Polaris\8.1\PolarisAdmin\assets\appsettings.user.json file in a text editor. You must run the editing application (for example, Notepad) as administrator.
2. In the .json file, update the server location in the `apiUrlRoot`. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *[my-server-domain-name]* with the FQDN of the server that hosts both the API service for Polaris System Administration (web-based) and the Authentication Application.
3. Update the server location in the `authAppUrl`. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *[my-server-domain-name]* with the FQDN of the server that hosts both the API service for Polaris System Administration (web-based) and the Authentication Application.
4. Save the .json file.

Configure LeapWebApp for Use with Azure AD

Once you have made the changes described in the Upgrading to Polaris 7.7 section, you only need to do additional configuration for LeapWebApp if one or both of the following is true:

- You want to enable permission overrides in Leap.
- You want to set `ReAuthDisabled` to `false` for Leap.

If neither of the above conditions is true, skip to the next section. See Configure Polaris.ApplicationServices for Use with Azure AD.

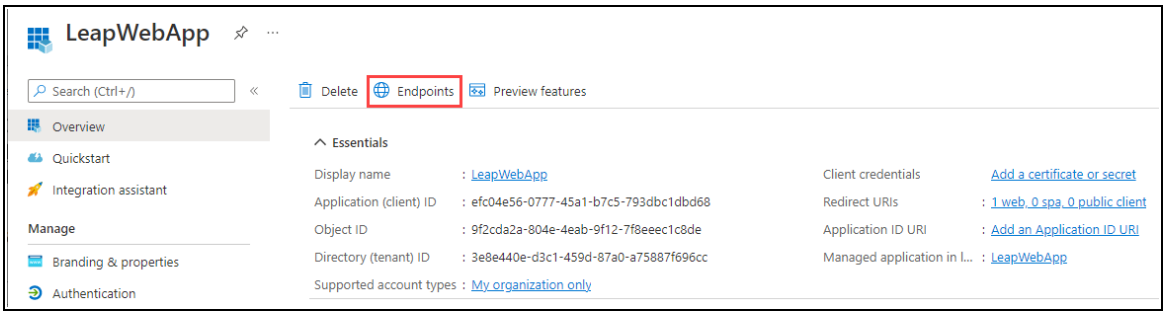
> **Note:**
> Permission overrides and reauthentication are not supported if your system uses multiple identity providers for authentication.

To configure LeapWebApp, update C:\Program Files\Polaris\8.1\LeapWebApp\appsettings.user.json using the following information:

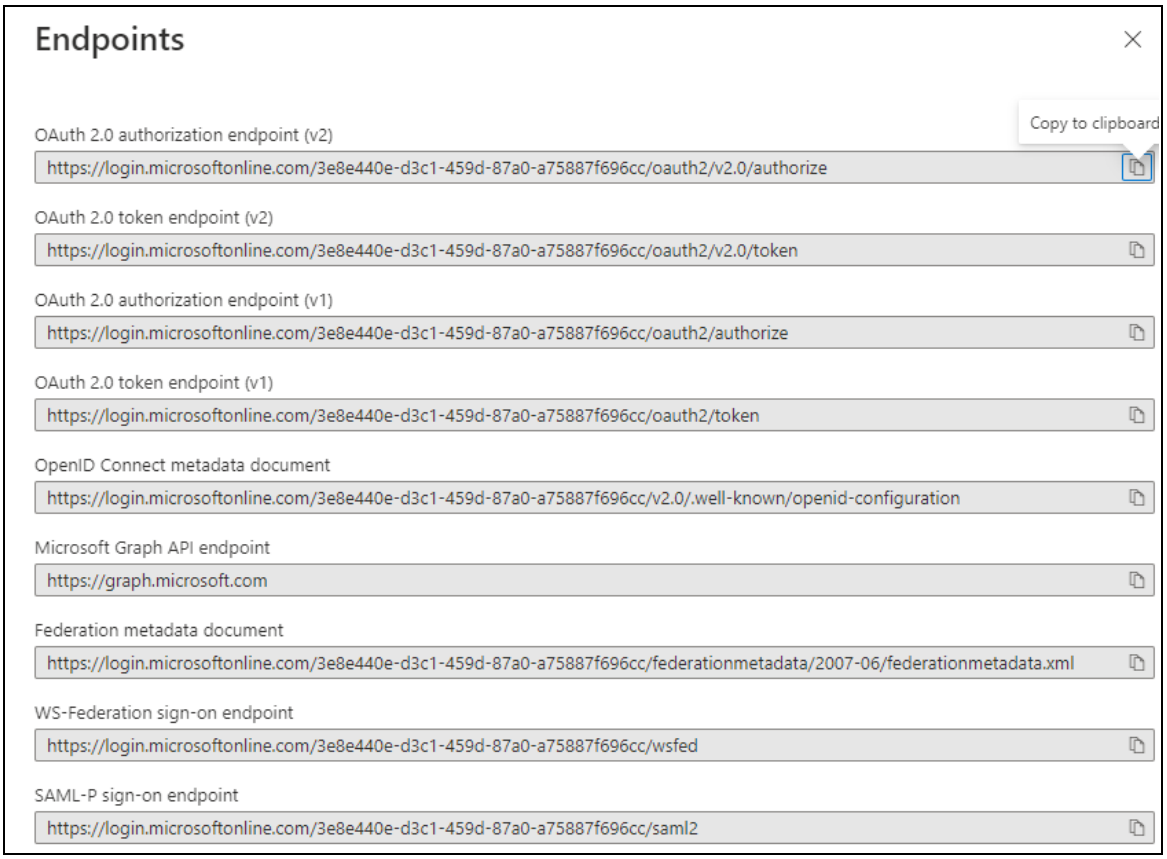
- Endpoint URIs copied from the Azure portal
- Values that you copied and saved during earlier steps in the Azure AD configuration process

To configure LeapWebApp

1. In the Azure portal, select **App registrations** from the list of options at the left side of the screen.
2. Select **LeapWebApp**.
3. On the LeapWebApp page, select **Endpoints**.



The Endpoints dialog appears. Leave this browser tab open so that you can copy endpoint URIs from it and paste them into the LeapWebApp appsettings.user.json file.



4. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
    - C:\Program Files\Polaris\8.1\LeapWebApp\appsettings.user.json
    - C:\Program Files\Polaris\8.1\LeapWebApp\RELEASE-NOTES.md
5. In the RELEASE-NOTES.md file, copy the settings in the OIDC/OAuth Setup for AzureAD section. The image below shows the settings to copy.

```
### OIDC/OAuth Setup for AzureAD

```
  "OAuth": {
    "Authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/",
    "AuthorizationEndpoint": null,
    "TokenEndpoint": null,
    "UserInformationEndpoint": null,
    "ClientId": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
    "ClientSecret": "YLU8Q~TGdT_KIcDZ5px643zcsLmCzr16S1jiKcZI",
    "MetadataAddress": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/.well-
known/openid-configuration",
    "KnownAuthorities": [ "login.microsoftonline.com" ],
    "CallbackPath": "/signin-oidc",
    "SignedOutCallbackPath": "/signout-callback-oidc",
    "SignedOutRedirectUri": "/login",
    "RemoteAuthenticationTimeout": 1,
    "RemoteFailureRedirectUri": "/leapwebapp/logout",
    "ResponseMode": "form_post",
    "ResponseType": "code",
    "SaveTokens": false,
    "Scopes": [ "openid", "profile", "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas" ],
    "UseOIDC": true,
    "UsePkce": true,
    "AlternateUpnClaimType": "preferred_username",
    "AlternateLogoutUri": null,
    "OptionalAuthorizeParameters": null,
    "OptionalEndSessionParameters": null,
    "SendAccessTokenAsHeaderValue": false,
    "AccessTokenHeaderName": "",
    "BasicAuthorizationCredentials": {
      "Username": null,
      "Password": null
    }
  },
```

6. In the .json file, replace the OAuth contents of the file with the settings you copied from the RELEASE-NOTES.md file.
7. On the Endpoints dialog, copy the root value and tenant ID from the **OAuth 2.0 authorization endpoint (v2)** box and paste it into the `Authority` property in the .json file. Your value will be similar to this one:

   `https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/`

   > **Note:**
   > The value must include the trailing slash character.

8. In the .json file, replace the `ClientId` property value with the application (client) ID you copied during the Register LeapWebApp with Azure AD step. Your value will be similar to this one:
```

`efc04e56-0777-45a1-b7c5-793dbc1dbd68`

9. In the .json file, replace the `ClientSecret` property value with the client secret you copied and saved during the Create Client Credentials step. Your value will be similar to this one:

   `4I.8Q~-GpkdoymMQneGIYNg40FRjx2Hr1wWLDcbr`

10. On the Endpoints dialog, copy the value from the **OpenID Connect metadata document** box and paste it into the `MetadataAddress` property in the .json file. Your value will be similar to this one:

    `https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/.well-known/openid-configuration`

11. On the Endpoints dialog, copy the FQDN of the Microsoft server from the **OpenID Connect metadata document** box and paste it into the `KnownAuthorities` property in the .json file. The value will be identical to this one:

    `login.microsoftonline.com`

12. In the .json file, update the `Scopes` property to add the scope you copied and saved during the Expose the Polaris.ApplicationServices API step. Your value will be similar to this one:

    `api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas`

    > **Note:**
    > If you want Azure AD to return the user's email address, add `"email"` to the `Scopes` property. You might choose to do this if the `preferred_username` is a generic name, phone number, or other value.

13. In the .json file, set the `AlternateUpnClaimType` property to `"preferred_username"`.

    > **Note:**
    > You can also set this property to `"email"`, if you choose.

14. Save the .json file.

15. Leave the browser tab displaying the Endpoints dialog open, and continue to the Configure Polaris.ApplicationServices for Use with Azure AD procedure.

Configure Polaris.ApplicationServices for Use with Azure AD

To configure Polaris.ApplicationServices, update C:\Program Files\Polaris\8.1\Polaris.ApplicationServices\appsettings.user.json using the following information:

- Endpoint URIs copied from the Azure portal
- Values that you copied and saved during earlier steps in the Azure AD configuration process

To configure Polaris.ApplicationServices

1. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
    - C:\Program Files\Polaris\8.1\Polaris.ApplicationServices\appsettings.user.json
    - C:\Program Files\Polaris\8.1\Polaris.ApplicationServices\RELEASE-NOTES.md
2. In the RELEASE-NOTES.md file, find the **Azure AD** section and copy the settings in the **Example authority settings** section. The image below shows the settings to copy.

````
## Azure AD

Example authority settings object:

```
{
    "Name": "AzureAD",
    "Authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/",
    "Audience": "api://f9f429e3-8355-4eb3-876b-0e45b6ddd295",
    "MetaAddress": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/.well-known/openid-
configuration",
    "RequireHttpsMetadata": true,
    "RequireSignedTokens": true,
    "ValidateIssuer": true,
    "ValidIssuers": [
        "https://sts.windows.net/3e8e440e-d3c1-459d-87a0-a75887f696cc/",
        "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0"
    ],
    "ValidateAudience": true,
    "ValidAudiences": [
        "f9f429e3-8355-4eb3-876b-0e45b6ddd295",
        "api://f9f429e3-8355-4eb3-876b-0e45b6ddd295"
    ],
    "UPNClaimTypes": [ "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn", "upn", "preferred_username" ],
    "IsUPNExternalID": false,
    "OpaqueToken": false,
    "UserInformationEndpoint": null
},
```
````

3. In the .json file, replace the existing `OAuth.Authorities` property with the settings you copied from the RELEASE-NOTES.md file.
4. In the RELEASE-NOTES.md file, copy the settings in the **Example Swagger settings** section. The image below shows the settings to copy.

Example Swagger settings:

```
"Swagger": {
    "ClientID": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
    "ClientSecret": "",
    "AppName": "Polaris.ApplicationServices",
    "AuthorizationUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-
a75887f696cc/oauth2/v2.0/authorize",
    "TokenUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token",
    "RefreshTokenUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token",
    "LogoutUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/logout",
    "Scopes": [
        { "Name": "openid", "Description": "Use OIDC to verify the user's identity" },
        { "Name": "email", "Description": "Optional to return user's email address" },
        { "Name": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas", "Description": "API Scope defined in AzureAD
for LeapWebApp" }
    ]
}
```

5. In the .json file, replace the existing `OAuth.Swagger` property with the settings you copied from the RELEASE-NOTES.md file.
6. Make the following update to the `Authority` property:
    a. On the Endpoints dialog of the Azure AD portal, copy the value from the **OAuth 2.0 authorization endpoint (v2)** box but omit the trailing *authorize*.
    b. Paste this value into the `Authority` property in the .json file. Your value will be similar to this one:

    `https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/`
7. In the .json file, update the `Audience` property:
    a. Locate the application (client) ID you copied and saved during the Register LeapWebApp with Azure AD step.
    b. Use it to construct a string with the following format:

    api://*[application (client) ID]*

c. Paste this value into the `Audience` property. Your value will be similar to this one:

   `api://efc04e56-0777-45a1-b7c5-793dbc1dbd68`

8. On the Endpoints dialog, copy the value from the **OpenID Connect metadata document** box and paste it into the `MetaAddress` property in the .json file. Your value will be similar to this one:

   `https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/well-known/openid-configuration`

9. In the .json file, add a value to the `ValidIssuers` property:

   a. On the Endpoints dialog, copy the tenant ID from the **OAuth 2.0 authorization endpoint (v2)** box.

   b. Use it to construct a URI with the following format:

      https://sts.windows.net/*[tenant ID]*

   c. Paste this value into the `ValidIssuers` property. Your value will be similar to this one:

      `https://sts.windows.net/3e8e440e-d3c1-459d-87a0-a75887f696cc/`

10. In the .json file, add a second value to the `ValidIssuers` property:

    a. On the Endpoints dialog, copy the value from the **OAuth 2.0 authorization endpoint (v2)** box but omit the trailing *authorize*.

    b. Paste this value into the `ValidIssuers` property in the .json file. Your value will be similar to this one:

       `https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0`

11. In the .json file, add a value to the `ValidAudiences` property:

    a. Locate the application (client) ID you copied and saved during the Register LeapWebApp with Azure AD step.

    b. Paste this value into the `ValidAudiences` property. Your value will be

similar to this one:

`efc04e56-0777-45a1-b7c5-793dbc1dbd68`

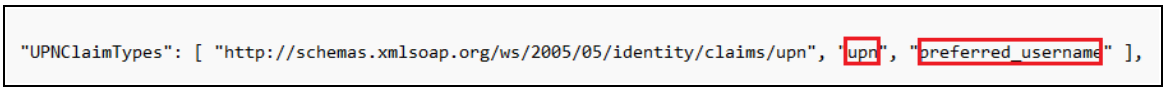
12. In the .json file, add a second value to the `ValidAudiences` property:
    a. Locate the application (client) ID you copied and saved during the Register LeapWebApp with Azure AD step.
    b. Use it to construct a string with the following format:

       api://*[application (client) ID]*

    c. Paste this value into the `ValidAudiences` property. Your value will be similar to this one:

       `api://efc04e56-0777-45a1-b7c5-793dbc1dbd68`

13. In the .json file, update the `UPNClaimTypes` property to add `"upn"` and `"preferred_username"` if those values are not already present. Your `UPNClaimTypes` property will be similar to this one:



14. In the `Swagger` property in the .json file, update the `ClientID` property:
    a. Locate the application (client) ID you copied and saved during the Register LeapWebApp with Azure AD step.
    b. Paste this value into the `ClientID` property. Your value will be similar to this one:

       `efc04e56-0777-45a1-b7c5-793dbc1dbd68`

15. On the Endpoints dialog, copy the value from the **OAuth 2.0 authorization endpoint (v2)** box and paste it into the `AuthorizationUrl` property in the .json file. Your value will be similar to this one:

    `https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/authorize`

16. In the .json file, update the `TokenUrl` property:

a. On the Endpoints dialog, copy the value from the **OAuth 2.0 token endpoint (v2)** box but replace *authorize* with *token*.

b. Paste the value into the `TokenUrl` property in the .json file. Your value will be similar to this one:

```
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token
```

17. In the .json file, update the `RefreshTokenUrl` property:

a. On the Endpoints dialog, copy the value from the **OAuth 2.0 token endpoint (v2)** box but replace *authorize* with *token*.

b. Paste the value into the `RefreshTokenUrl` property in the .json file. Your value will be similar to this one:

```
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token
```

18. In the .json file, update the `LogoutUrl` property:

a. On the Endpoints dialog, copy the value from the **OAuth 2.0 authorization endpoint (v2)** box but replace *authorize* with *logout*.

b. Paste the value into the `LogoutUrl` property in the .json file. Your value will be similar to this one:

```
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/logout
```

19. In the .json file, add a new scope `Name` and `Description`:

a. Locate the scope you copied and saved during the Expose the Polaris.ApplicationServices API step.

b. Copy the scope and use it to construct a new `Name` property. Your value will be similar to this one:

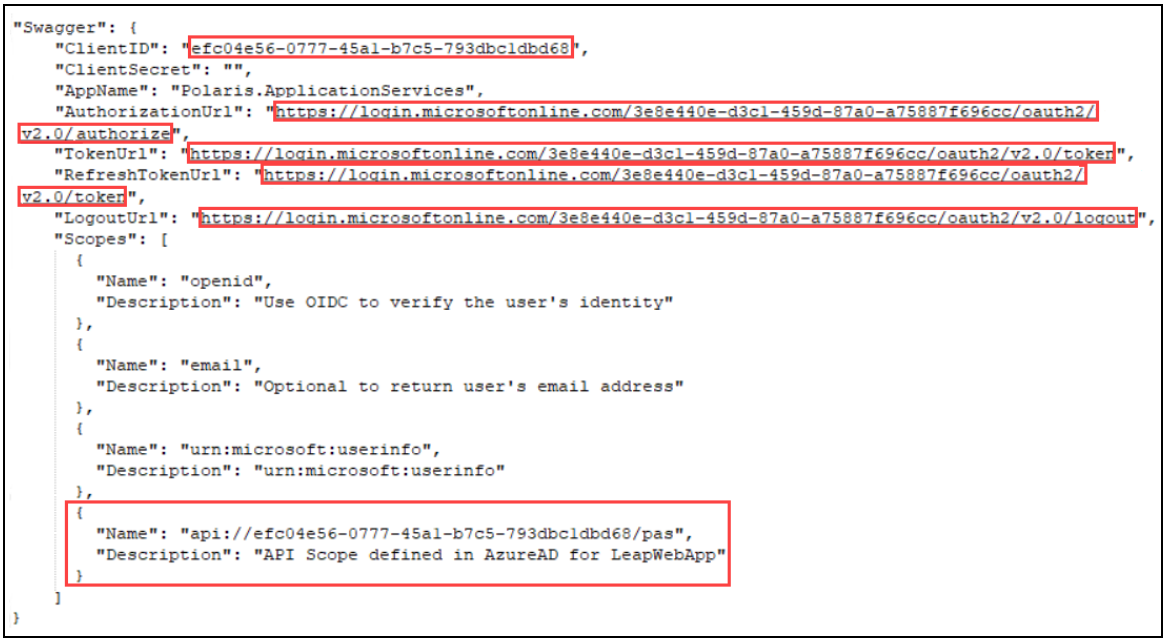
```
"Name": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas"
```

c. Paste the `Name` property into the .json file.

d. Add a `Description` property that matches the example below:

```
"Description": "API Scope defined in AzureAD for
LeapWebApp"
```

20. Save the .json file. Your updated values in the `Swagger` property should look similar to the example below.



> **Note:**
> Changes to the appsettings.user.json files do note take effect until the IIS application pools are restarted or IIS is reset.

Configuring Basic OAuth 2.0

To configure basic OAuth 2.0, perform the following tasks:

1. Set up web services and applications.
2. Add authentication redirect URIs.

After you complete these tasks, Add a URL rewrite rule for LeapWebApp.

Set Up Web Services and Applications for Basic OAuth 2.0

To set up each of the following web services and applications, you must configure a .json file for each of the following:

- Polaris.Authentication (the application that authenticates Polaris users)
- Polaris.AuthenticationServices (the API service that provides backend support for authentication)
- PolarisAdmin (the web-based Polaris System Administration application)
- Polaris.ApplicationServices (Leap's API service)

> **Note:**
> Since basic OAuth 2.0 doesn't support reauthentication or permission overrides, the only configuration needed for the C:\Program Files\Polaris\8.1\LeapWebApp\appsettings.user.json file are the changes described in the Upgrading to Polaris 7.7 section.

The four .json files are all named appsettings.user.json, but they reside in different directories:

- C:\Program Files\Polaris\8.1\Polaris.Authentication
- C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices
- C:\Program Files\Polaris\8.1\PolarisAdmin\assets
- C:\Program Files\Polaris\8.1\Polaris.ApplicationServices

This section contains the following topics:

- Configure Polaris.Authentication for Use with Basic OAuth 2.0
- Configure Polaris.AuthenticationServices for Use with Basic OAuth 2.0
- Configure PolarisAdmin for Use with Basic OAuth 2.0
- Configure Polaris.ApplicationServices for Use with Basic OAuth 2.0

Configure Polaris.Authentication for Use with Basic OAuth 2.0

To configure Polaris.Authentication, update C:\Program Files\Polaris\8.1\Polaris.Authentication\appsettings.user.json. You will use several values copied from your identity provider.

> **Important:**
> By default, the appsettings.user.json template file contains configuration settings that apply to AD FS. Polaris 8.1 includes a RELEASE-NOTES.md file that contains the template settings that apply to basic OAuth 2.0. See the Configuring Basic OAuth 2.0 section of the release notes file.

To configure Polaris.Authentication

1. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
    - C:\Program Files\Polaris\8.1\Polaris.Authentication\appsettings.user.json
    - C:\Program Files\Polaris\8.1\Polaris.Authentication\RELEASE-NOTES.md
2. In the RELEASE-NOTES.md file, copy the settings in the Configuring Basic OAuth 2.0 section. The image below shows the settings to copy.

```
## Configuring Basic OAuth 2.0

### AWS Cognito Template

```
"OAuth": {
  "Enabled": true,
  "SendOAuthAuthorityHeader": false,
  "Authorities": [
    {
      "IsActive": true,
      "Name": "AmazonCognito",
      "UseOidc": false,
      "AuthorizationEndpoint": "[COGNITO-DOMAIN]/oauth2/authorize",
      "TokenEndpoint": "[COGNITO-DOMAIN]/oauth2/token",
      "UserInformationEndpoint": "[COGNITO-DOMAIN]/oauth2/userInfo",
      "EndSessionEndpoint": "[COGNITO-DOMAIN]/logout?logout_uri=https%3A%2F%2F[POLARIS-BASE-URL]%2Fpolarisauth%
2Flogin&client_id=[CLIENT-ID]",
      "ClientId": "[CLIENT-ID]",
      "ClientSecret": "[CLIENT-SECRET]",
      "CallbackPath": "/oauth/callback",
      "UsePkce": false,
      "Scopes": [
        "profile",
        "openid"
      ],
      "SaveTokens": false,
      "SendAccessTokenAsHeaderValue": false,
      "AccessTokenHeaderName": "",
      "BasicAuthorizationCredentials": {
        "Username": "",
        "Password": ""
      },
      "OptionalAuthorizeParameters": null
    }
  ]
}
```

3. In the .json file, replace the entire contents of the file with the settings you copied from the RELEASE-NOTES.md file.
4. Verify that `Enabled` is set to `true`.
5. Verify that `IsActive` is set to `true`.
6. In the `Name` property, replace `AmazonCognito` with a unique name that describes your OAuth identity provider (for example, Azure AD or AD FS).
7. Verify that `UseOidc` is set to `false`.
8. Update the following properties. In each, replace *`[COGNITO-DOMAIN]`* with the server address for your identity provider:
    - `AuthorizationEndpoint`
    - `TokenEndpoint`
```

- `UserInformationEndpoint`
- `EndSessionEndpoint`

9. In the `EndSessionEndpoint` property, replace *`[POLARIS-BASE-URL]`* with the FQDN of the server that hosts the Authentication Application.
10. In the `ClientId` property, replace *`[CLIENT-ID]`* with the client ID from your identity provider.
11. In the `ClientSecret` property, replace *`[CLIENT-SECRET]`* with the client secret from your identity provider.
12. Save the .json file.

Configure Polaris.AuthenticationServices for Use with Basic OAuth 2.0

To configure Polaris.AuthenticationServices, update C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices\appsettings.user.json. You will use several values copied from your identity provider.

> **Important:**
> By default, the appsettings.user.json template file contains configuration settings that apply to AD FS. Polaris 8.1 includes a RELEASE-NOTES.md file that contains the template settings that apply to basic OAuth 2.0. See the OAuth2 Only Setup for AmazonCognito section of the release notes file.

To configure Polaris.AuthenticationServices

1. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
   - C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices\appsettings.user.json
   - C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices\RELEASE-NOTES.md
2. In the RELEASE-NOTES.md file, copy the settings in the OAuth2 Only Setup for AmazonCognito section. The image below shows the settings to copy.

```
### OAuth2 Only Setup for AmazonCognito

```
"OAuth": {
  "Enabled": true,
  "Authorities": [
    {
      "Name": "AmazonCognito",
      "UseOidc": false,
      "RequireHttpsMetadata": false,
      "RequireSignedTokens": false,
      "ValidateIssuer": false,
      "Authority": null,
      "Audience": "4k6l3mtmifukol3bc9a8mjkb8r",
      "MetaAddress": null,
      "UserInformationEndpoint": "https://polaris-leap.auth.us-east-1.amazoncognito.com/oauth2/userInfo",
      "ValidIssuers": [
        "https://polaris-leap.auth.us-east-1.amazoncognito.com/",
        "https://cognito-idp.us-east-1.amazonaws.com/us-east-1_zNpQPq701"
      ],
      "ValidateAudience": false,
      "ValidAudiences": [ "4k6l3mtmifukol3bc9a8mjkb8r" ],
      "UpnClaimTypes": [ "sub" ],
      "IsUpnExternalId": true,
      "SendAccessTokenAsHeaderValue": false,
      "AccessTokenHeaderName": "",
      "BasicAuthorizationCredentials": {
        "Username": "",
        "Password": ""
      }
    }
```

3. In the .json file, replace the entire contents of the file with the settings you copied from the RELEASE-NOTES.md file.
4. Verify that `Enabled` is set to `true`.
5. In the `Name` property, replace *`AmazonCognito`* with a unique name that describes your OAuth identity provider (for example, Azure AD or AD FS).
6. Verify that `UseOidc` is set to `false`.
7. In the `UserInformationEndpoint` property, replace *`polaris-leap.auth.us-east-1.amazoncognito.com`* with the server address for your identity provider.
8. In the `ValidIssuers` property, replace the default URLs with the valid issuer URLs from your identity provider.
9. In the `UpnClaimTypes` property, update the default value if you want to specify a different claim to serve as the user identifier. The default value applies to most configurations, but you can specify a different claim that exists in a JSON Web
```

Token (JWT).

10. Save the .json file.

Configure PolarisAdmin for Use with Azure AD

To configure PolarisAdmin, update C:\Program Files\Polaris\8.1\PolarisAdmin\assets\appsettings.user.json.

To configure PolarisAdmin

1. Open the C:\Program Files\Polaris\8.1\PolarisAdmin\assets\appsettings.user.json file in a text editor. You must run the editing application (for example, Notepad) as administrator.
2. In the .json file, update the server location in the `apiUrlRoot`. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *`[my-server-domain-name]`* with the FQDN of the server that hosts both the API service for Polaris System Administration (web-based) and the Authentication Application.
3. Update the server location in the `authAppUrl`. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *`[my-server-domain-name]`* with the FQDN of the server that hosts both the API service for Polaris System Administration (web-based) and the Authentication Application.
4. Save the .json file.

Configure Polaris.ApplicationServices for Use with Basic OAuth 2.0

To configure Polaris.ApplicationServices, update C:\Program Files\Polaris\8.1\Polaris.ApplicationServices\appsettings.user.json. You will use several values copied from your identity provider.

> **Important:**
> By default, the appsettings.user.json template file contains configuration settings that apply to AD FS. Polaris8.1 includes a RELEASE-NOTES.md file that contains the template settings that apply to basic OAuth 2.0. See the **Example OAuth2 with Opaque Tokens and /userinfo endpoint** section of the release notes file.

To configure Polaris.ApplicationServices

1. Open the following files in a text editor. You must run the editing application (for example, Notepad) as administrator.
    - C:\Program Files\Polaris\8.1\Polaris.ApplicationServices\appsettings.user.json
    - C:\Program Files\Polaris\8.1\Polaris.ApplicationServices\RELEASE-NOTES.md
2. In the RELEASE-NOTES.md file, copy the settings in the **Example OAuth2 with Opaque Tokens and /userinfo endpoint** section. The image below shows the settings to copy.

Example OAuth2 with Opaque Tokens and /userinfo endpoint

```
"OAuth": {
      "Enabled": true,
      "Authorities": [
        {
          "Name": "OAuth2MockService",
          "Authority": null,
          "Audience": null,
          "MetaAddress": null,
          "RequireHttpsMetadata": false,
          "RequireSignedTokens": false,
          "ValidateIssuer": false,
          "ValidIssuers": [
            "http://localhost:8080/"
          ],
          "ValidateAudience": false,
          "ValidAudiences": [],
          "UPNClaimTypes": [ "sub" ],
          "IsUPNExternalID": true,
          "OpaqueToken": true,
          "UserInformationEndpoint": "http://localhost:8080/userinfo",
          "SendAccessTokenAsHeaderValue": false,
          "BasicAuthorizationCredentials": {
            "Username": null,
            "Password": null
          }
        }
      ],
      "Swagger": {
        "ClientID": "7fbe747f-1101-451c-a947-fa6ef898a517",
        "ClientSecret": "",
        "AppName": "Polaris.ApplicationServices",
        "AuthorizationUrl": "http://localhost:8080/authorize",
        "TokenUrl": "http://localhost:8080/token",
        "RefreshTokenUrl": "http://localhost:8080/token",
        "LogoutUrl": "http://localhost:8080/endsession",
        "Scopes": [
          {
            "Name": "profile",
            "Description": "Return basic profile information."
          }
        ]
      }
},
```

3. In the .json file, replace the entire contents of the file with the settings you copied from the RELEASE-NOTES.md file.

4. In the `Authorities` property, update the following properties with information from your identity provider:
    a. In the `Name` property, replace `OAuth2MockService` with a unique name that describes your OAuth identity provider (for example, `Azure AD` or `AD FS`).
    b. Update the `ValidIssuers` property:
        i. Copy the tenant ID from your identity provider.
        ii. Use it to construct a URI with the following format:

        https://sts.windows.net/*[tenant ID]*

        iii. Paste this value into the `ValidIssuers` property. Your value will be similar to this one:

        `https://sts.windows.net/3e8e440e-d3c1-459d-87a0-a75887f696cc/`

    c. Update the `UserInformationEndpoint` property with the user information endpoint from your identity provider.
5. In the `Swagger` property, update the following properties with values copied from your identity provider:
    a. In the `ClientID` property, replace *`7fbe747f-1101-451c-a947-fa6ef898a517`* with the client ID for your configuration.
    b. In the `ClientSecret` property, enter the client secret from your identity provider.
    c. In the following properties, replace *`localhost:8080`* with the server address from your identity provider:
        - `AuthorizationUrl`
        - `TokenUrl`
        - `RefreshTokenUrl`
        - `LogoutUrl`
6. Save the .json file.

Add Authentication Redirect URIs

Add the following URIs to your identity provider's hosted UI configuration:

- Allowed callback URIs:
  - https://*[FQDN]*/leapwebapp/login
  - https://*[FQDN]*/leapwebapp/oauth/callback
  - https://*[FQDN]*/polarisauth/login
  - https://*[FQDN]*/polarisauth/oauth/callback
- Allowed sign-out URIs:
  - https://*[FQDN]*/leapwebapp/login
  - https://*[FQDN]*/leapwebapp/logout
  - https://*[FQDN]*/polarisauth/login
  - https://*[FQDN]*/polarisauth/logout

Notes:

- Replace *[FQDN]* with the fully-qualified domain name of your server.
- Example: https://leap.mylibrary.org/leapwebapp/login

Configuring Basic Authentication

To configure basic authentication, perform the following task:

- Set Up Web Services and Applications for Basic Authentication.

After you complete this task, Add a URL rewrite rule for LeapWebApp.

Set Up Web Services and Applications for Basic Authentication

To set up basic authentication, you must configure a .json file for each of the following:

- Polaris.Authentication (the application that authenticates Polaris users)
- Polaris.AuthenticationServices (the API service that provides backend support for authentication)

> **Note:**
> When setting up basic authentication, the only configuration needed for the C:\Program Files\Polaris\8.1\LeapWebApp\appsettings.user.json file are the changes described in the Upgrading to Polaris 7.7 section.

The two .json files are both named appsettings.user.json, but they reside in different directories:

- C:\Program Files\Polaris\8.1\Polaris.Authentication
- C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices

This section contains the following topics:

- Configure Polaris.Authentication for Use with Basic Authentication
- Configure Polaris.AuthenticationServices for Use with Basic Authentication

Configure Polaris.Authentication for Use with Basic Authentication

To configure Polaris.Authentication, update C:\Program Files\Polaris\8.1\Polaris.Authentication\appsettings.user.json.

To configure Polaris.Authentication

1. Open the C:\Program Files\Polaris\8.1\Polaris.Authentication\appsettings.user.json file in a text editor. You must run the editing application (for example, Notepad) as administrator.
2. In the `BasicAuth` property, verify that the `Enabled` property is set to `true`.
3. In the `OAuth` property, set the `Enabled` property to `false`.
4. Save the .json file.

Configure Polaris.AuthenticationServices for Use with Basic Authentication

To configure Polaris.AuthenticationServices, update C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices appsettings.user.json.

To configure Polaris.AuthenticationServices

1. Open the C:\Program Files\Polaris\8.1\Polaris.AuthenticationServices\appsettings.user.json file in a text editor. You must run the editing application (for example, Notepad) as administrator.
2. In the `BasicAuth` property, set the `Enabled` property to `true`.
3. In the `OAuth` property, set the `Enabled` property to `false`.
4. Save the .json file.

Add a URL Rewrite Rule for LeapWebApp

Adding a URL rewrite rule redirects incoming URLs to the correct address for the LeapWebApp. This must be done manually, since the library may already use other URL rewrite rules.

To add a URL rewrite rule, you must have the Microsoft IIS URL Rewrite 2.1 extension. For more information, see https://www.iis.net/downloads/microsoft/url-rewrite.

To add a URL rewrite rule

1. Open the root IIS web.config file, found in the following location:

   C:\inetpub\wwwroot\web.config

2. Add a rewrite rule to the **system.webServer** node.

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <system.webServer>
        <rewrite>
            <rules>
                <rule name="UrlToLowercase" stopProcessing="true">
                    <match url="(.*)" ignoreCase="true" />
                        <action type="Redirect" url="https://{HTTP_HOST}{ToLower:{PATH_INFO}}" redirectType="Found" appendQueryString="true" />
                        <conditions>
                            <add input="{PATH_INFO}" pattern="^/LeapWebApp(.*)|^/Leapwebapp(.*)|^/LEAPWEBAPP(.*)" ignoreCase="false" />
                        </conditions>
                    </rule>
            </rules>
        </rewrite>
    </system.webServer>
</configuration>
```

> **Note:**
> For sample rewrite rule text that you can copy and paste, see Sample Rewrite Rule Text.

   In the example above, if the incoming URL includes a path that contains any of the following, the rewrite rule redirects to /leapwebapp:

   - /LeapWebApp
   - /Leapwebapp
   - /LEAPWEBAPP

3. Save the web.config file.

> **Note:**
> When registering redirect URIs for LeapWebApp in AD FS, the URIs should

be lowercase. For example:

- https://rd-polaris.polarislibrary.com/leapwebapp/signin-oidc
- https://rd-polaris.polarislibrary.com/leapwebapp/signin-override-oidc
- https://rd-polaris.polarislibrary.com/leapwebapp/signout-callback-oidc

Sample Rewrite Rule Text

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
   <system.webServer>
      <rewrite>
         <rules>
            <rule name="UrlToLowercase" stopProcessing="true">
               <match url="(.*)" ignoreCase="true" />
                  <action type="Redirect" url="https://{HTTP_HOST}
                  {ToLower:{PATH_INFO}}" redirectType="Found"
                  appendQueryString="true" />
                  <conditions>
                     <add input="{PATH_INFO}" pattern="^/LeapWebApp
                     (.*)|^/Leapwebapp(.*)|^/LEAPWEBAPP(.*)"
                     ignoreCase="false" />
                  </conditions>
            </rule>
         </rules>
      </rewrite>
   </system.webServer>
</configuration>
```

Additional URL Rewrite Resources

See Microsoft's URL Rewrite Module Configuration Reference for additional information:

- https://docs.microsoft.com/en-us/iis/extensions/url-rewrite-module/url-rewrite-module-configuration-reference

Glossary

C

Calendar Change

A calendar change indicates the chronological point where the highest level of enumeration changes. For example, if you select Day of Month for the calendar change, and enter 0101, the calendar change will be on January 1.

M

My Term

My definition