LSEG Workspace | Automated Domain Management (ADM)

Installation and configuration guide

LSEG DATA & ANALYTICS

Contents

About this document

The Automated Domain Management (ADM) app helps Microsoft Teams administrators automatically manage external access policies using LSEG's Open Directory (OD) network. This automation minimises manual effort and delivers scalable federation across member organizations.

In this guide

This guide outlines the steps by which Automated Domain Management (ADM) can be installed, configured, and managed.

Intended readership

This guide is intended for LSEG Workspace customers who want to install and use the ADM application within their Azure environment.

Further information

To:

- Request product assistance, contact Support.
- Access other LSEG Workspace technical content, see the Workspace technical documentation site.
- Provide feedback on Workspace technical content, contact DocFeedback@lseg.com.

About Automated Domain Management (ADM)

This section describes the different roles and relationships that are involved in the ADM deployment process.

Roles and relationships

To ensure clarity for all stakeholders, the following roles and relationships should be explicitly defined:

Customer organisation

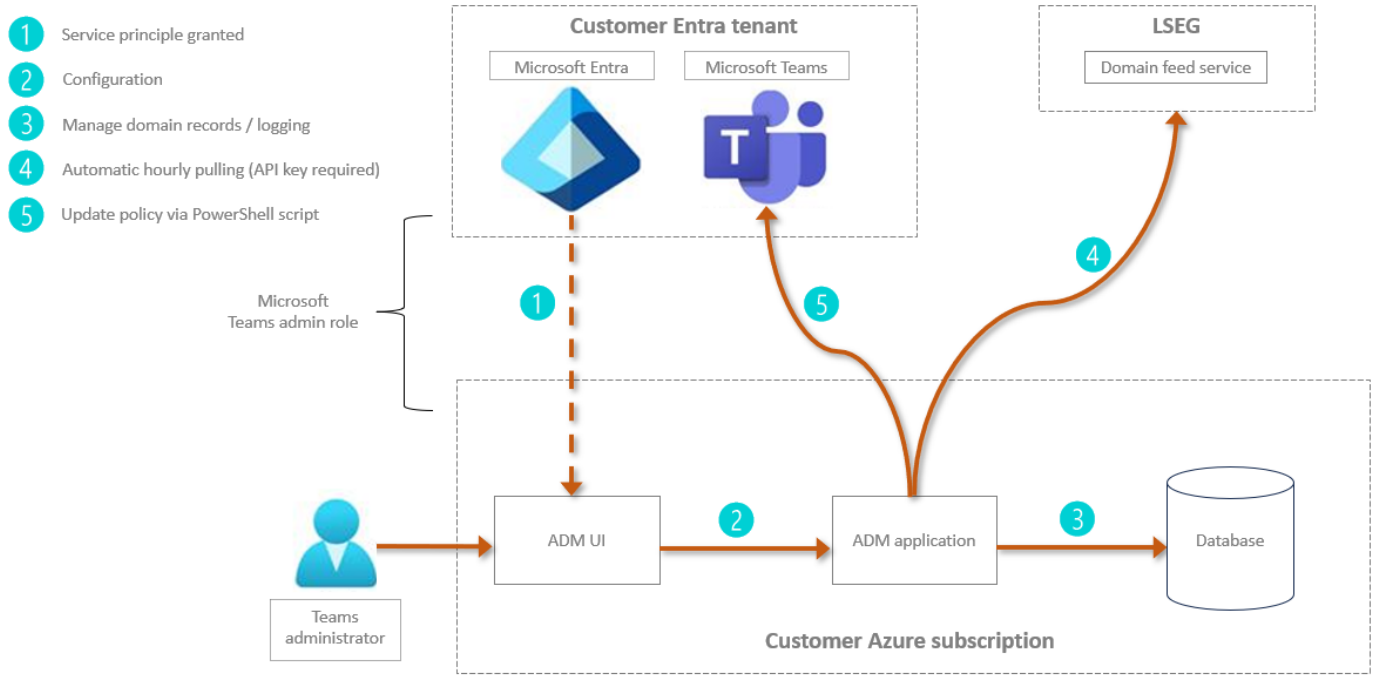
Role	Relationship with LSEG	Relationship with Microsoft
Consumes the Automated Domain Management (ADM) app to manage external access policies for Microsoft Teams in connection with use of Open Directory (OD).	Acts as a participant in LSEG's Open Directory (OD) network, leveraging OD for federation across member organisations.	Uses Microsoft Teams as the collaboration platform.

LSEG (London Stock Exchange Group)

Role	Relationship with customer organisation	Relationship with Microsoft
Provides and maintains the Open Directory (OD) network. Develops and supports the ADM app which supports federation.	Serves as the directory authority, ensuring OD membership integrity and policy enforcement. Provides technical support, documentation, and compliance guidance for ADM deployment.	Collaborates on integration standards to ensure OD and ADM works seamlessly with Microsoft Teams.

ADM workflow

The following diagram presents an architectural overview of ADM.



About deploying the ADM application

The ADM app is a client-side application that must be deployed onto a tenant's Azure cloud environment:

- By an administrator with the appropriate permissions to deploy Azure services onto an Azure cloud environment, and
- Via an Azure Resource Management (ARM) template

📌 More information about the minimum required roles for ADM deployment is described in the Azure resources section. Additionally, you can view Teams administrator details here.

Pre-requisites for deployment

Customers must have the following in place before deploying ADM:

Pre-requisite	Role / permission required	Reason
Azure subscription	For details on the required role / permissions, see Appendix B: Azure resources. Note that a customer's Azure policy must enable public network access for applications.	Required for creating Azure resources.
Azure ID	Teams administrator	Required to use private Azure application
Entra ID	Entra application administrator	Required for creating App Registrations, consent to Graph API permissions and assign Directory roles.
Teams environment	Teams administrator	Required to enable the administrator to perform updates to Teams policies and domains using the ADM app.

📌 Ideally, all the pre-requisites would be part of the same subscription; however, deployment is still possible if this is not the case.

Selecting your deployment method

You can deploy ADM in the following ways (listed from most recommended to least recommended):

- Deploying ADM using the fully-scripted method
- Deploying ADM using the partially-scripted method
- Deploying ADM using the manual method (not recommended)

Customers are strongly advised to deploy ADM using either the fully-scripted or partially-scripted method, as this is the simplest and most straight-forward way of getting ADM up and running.

Deploying ADM using the fully-scripted method

The partially-scripted method is described below, and involves the following steps you should perform (in this order):

1. Registering the deployment application
2. Deploying the standard ARM template
3. Accessing the ADM portal

Registering the deployment application

To register the deployment application:

1. Run the deployment app registration script, as displayed below:

```
# With default app names (adm-deployment)
# If an app with the same name already exists, the script will update any missing configurations on the existing app rather than creating a duplicate.
./adm-deployment.ps1 -RegisterDeploymentApp

# With custom app name
./adm-deployment.ps1 -RegisterDeploymentApp -DeploymentAppName "my-deployment-app"
```

Registering the deployment application:

- Creates deployment app registration with elevated permissions
- Assigns the Privileged Role Administrator directory role
- Configures the following Microsoft Graph API permissions:
  - Application.ReadWrite.All (Application)
  - AppRoleAssignment.ReadWrite.All (Application)
  - RoleManagement.ReadWrite.Directory (Application)
- Grants admin consent automatically
- Generates a deployment app client secret (which expires in one year)
- Saves credentials to app-credentials.json

2. Check that the output of this script is displayed as below:

```
...
================================
AZURE AD APPLICATION CREDENTIALS
================================

adm-deployment-vasanchai:
  Deployment App Client ID: 97318ddd-3b53-49b3-827f-6510b69091eb
  Deployment App Client Secret: <BACKEND_CLIENT_SECRET>

================================

Next Steps:
  Click to open deployment template:
  https://portal.azure.com/#create/Microsoft.Template/uri/https%3A%2F%2Fcdn.refinitiv.com%2Fpublic%2Fadm-arm%2Fassets%2FmainTemplate-low-touch.json
```

You are now ready to deploy the ARM template.

Deploying the standard ARM template

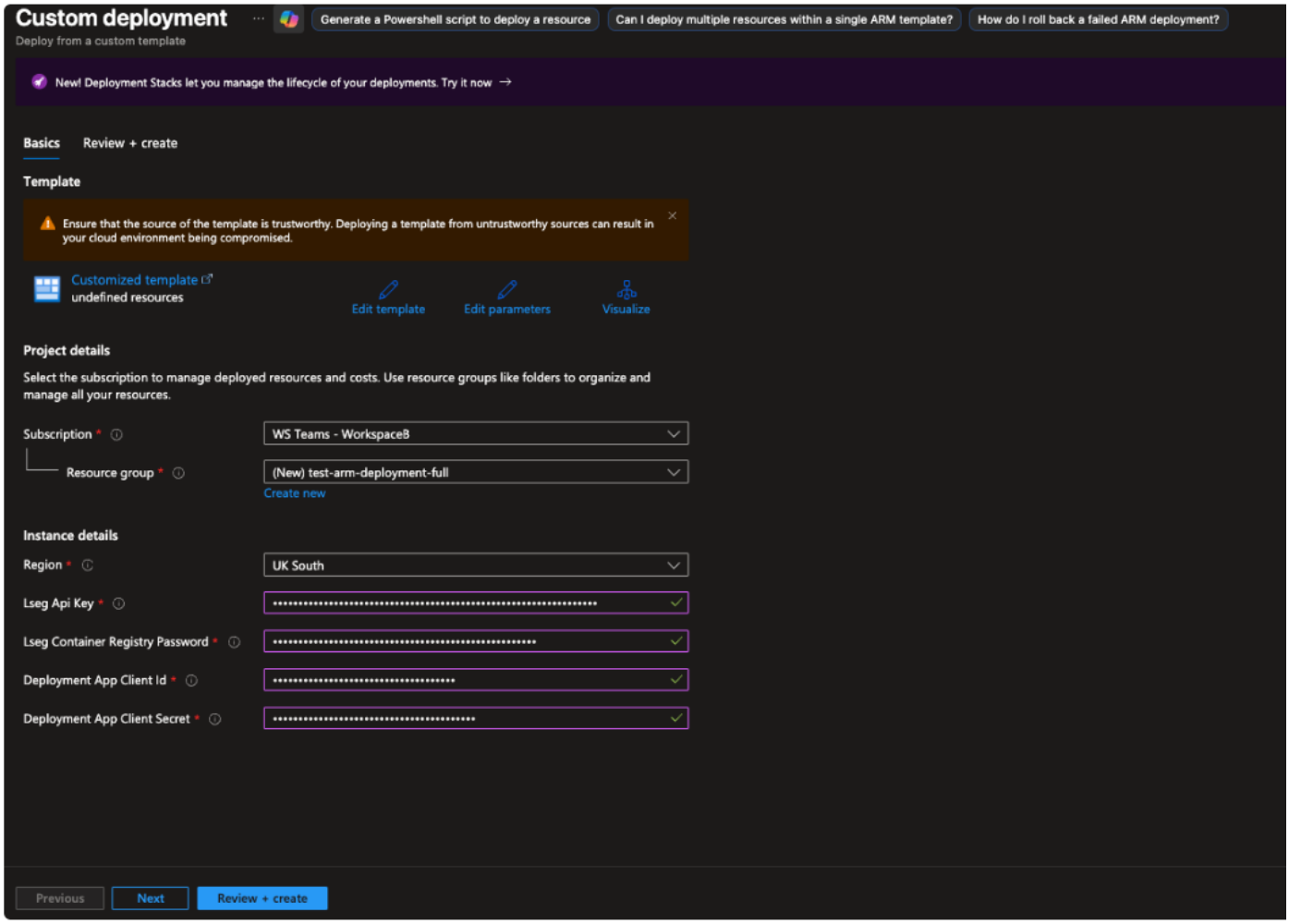
To deploy the standard ARM template:

1. Open the following Azure Portal deployment link shown in the script output.

   https://portal.azure.com/#create/Microsoft.Template/uri/https%3A%2F%2Fcdn.refinitiv.com%2Fpublic%2Fadm-arm%2Fassets%2FmainTemplate-low-touch.json

2. In the resulting screen, complete the required parameters:

   - **Subscription** – Select the Azure subscription from the dropdown list.
   - **Resource group** – Select an existing resource group from the dropdown list or click **Create new** to make a new one.
   - **Region** – Select the deployment region from the dropdown list.
   - **LSEG Api Key** – Enter the LSEG domain feed service API key (free text).
   - **LSEG Container Registry Password** – Enter the LSEG Container registry password (free text).
   - **Deployment App Client ID** – Enter the app client ID (free text).
   - **Deployment App Client Secret** – Enter the app client secret (free text).



3. Click **Review + create**.

   The deployment will take around 10-15 minutes to complete.

Accessing the ADM portal

You can access the ADM portal by using the URL displayed in the **adm portal url** field.

You have now completed the ADM deployment process using the fully-scripted method, and can proceed to configure ADM in the following ways:

- Configuring domain feeds
- Subscribing Admin users to notifications
- Notifying Admin users of domain changes
- Managing domains
- Managing policies

Deploying ADM using the partially-scripted method

The partially-scripted method is described below, and involves the following steps you should perform (in this order):

1. Registering applications
2. Deploying the standard ARM template
3. Configuring ADM after deployment
4. Accessing the ADM portal

> If you want to deploy ADM using the fully-scripted method, see Deploying ADM using the fully-scripted method; or, if you would like to deploy it manually (not recommended for most customers), see Deploying ADM using the manual method.

Registering applications

To create the required frontend and backend app registrations, download and run the script available here:

```
# With default app names (adm-frontend, adm-backend)
# If an app with the same name already exists, the script will update any missing configurations on the existing app rather than creating a duplicate.
./adm-deployment.ps1 -RegisterApp

# With custom app names
./adm-deployment.ps1 -RegisterApp -FrontendAppName "my-frontend-app" -BackendAppName "my-backend-app"
```

You may now run this script using either PowerShell or Microsoft Azure.

To run this script from a PowerShell terminal, either:

1. Open a PowerShell environment and run the script from one of the following:
   - Local PowerShell (PowerShell 7.x or later, on Windows), or
   - Azure Cloud Shell in a browser at https://shell.azure.com, using the PowerShell environment.
2. Sign in under an administrator's account.

> You must be able to sign in with an account that has <required role> (for example, Global Administrator, Teams Administrator) in the <tenant name> tenant.

3. Navigate to the folder where you saved the script and run it. Running this script will:
   - Create an adm-frontend app registration
   - Create an adm-backend app registration
   - Configure API permissions and scopes
   - Grant administrator consent for required permissions
   - Assign the Teams Administrator role to the backend app
   - Generate a backend client secret (which expires one year after generation)
   - Save your credentials to the app-credentials.json file

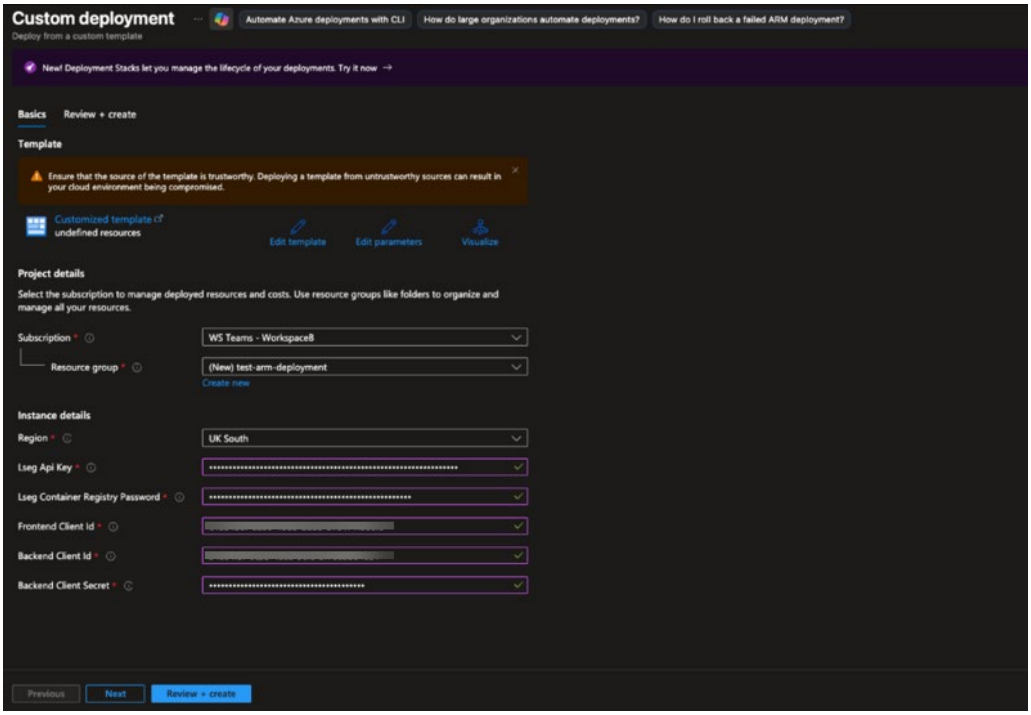
This output of running this script is as follows:

```
...
================================
AZURE AD APPLICATION CREDENTIALS
================================

adm-frontend-vasanchai:
  Frontend Client ID: e133456f-6200-40d6-b2b3-57a471f986fc

my-backend-vasanchai:
  Backend Client ID: 54a04f9f-6c35-4a6a-9efa-3f7eeadd4eb1
  Backend Client Secret: <BACKEND_CLIENT_SECRET>

================================

Next Steps:
  Click to open deployment template:
  https://portal.azure.com/#create/Microsoft.Template/uri/https%3A%2F%2Fcdn.refinitiv.com%2Fpublic%2Fadm-arm%2Fassets%2FmainTemplate-standard.json
```

Deploying the standard ARM template

To deploy the standard ARM template:

1. Open the Azure Portal deployment link provided in the script output (see previous section).
2. In the resulting screen, complete the required Custom deployment parameters.



The fields are as follows:

- Subscription: Select Azure subscription
- Resource Group: Create a new group or select an existing group
- Region: Select the deployment region
- Frontend Client ID
- Backend Client ID
- Backend Client Secret
- Lseg API Key: LSEG domain feed service API key (provided by LSEG)
- Lseg Container Registry Password (provided by LSEG)

3. Click the **Review + create** button.

📌 The deployment will take approximately 10 to 15 minutes to complete.

Configuring ADM after deployment

To configure ADM after deployment:

1. Navigate to the Outputs section in the Azure portal.

2. Copy the content of the **script for post-deployment** field.
3. Go back to the Azure Portal Cloud Shell.
4. In the Azure Portal Cloud Shell, paste the copied value to run the following command, which will update the frontend redirect URI:

```
# Example
./adm-deployment.ps1 -AddRedirectUri -AppId e133456f-6200-40d6-b2b3-57a471f986fc -Url https://adm-fe-app-ri4zqg.kindwave-73af7dc7.uksouth.azurecontainerapps.io/
```

The output of running this script is as follows:

```
...
================================
UPDATE COMPLETE
================================

Total redirect URIs configured: 1
New URIs added: 1

Updated redirect URIs:
  - https://adm-fe-app-ri4zqg.kindwave-73af7dc7.uksouth.azurecontainerapps.io
```

Accessing the ADM Portal

To access the ADM Portal that has been created, obtain the relevant URI from the deployment outputs (as shown above).

You have now completed the ADM deployment process using the standard template, and can proceed to configure ADM in the following ways:

- Configuring domain feeds
- Subscribing Admin users to notifications
- Notifying Admin users of domain changes
- Managing domains
- Managing policies

Deploying ADM using the manual method

📌 If you want to deploy ADM using the fully-scripted or partially scripted methods, which are the recommended methods, see Deploying ADM using the full-scripted method, or Deploying ADM using the partially-scripted method.

Creating an app registration

This is a required step so ADM can manage domains and policies on the client tenant, including allowing specific domains.

📌 Note that this step is currently manual, but will be automated later.

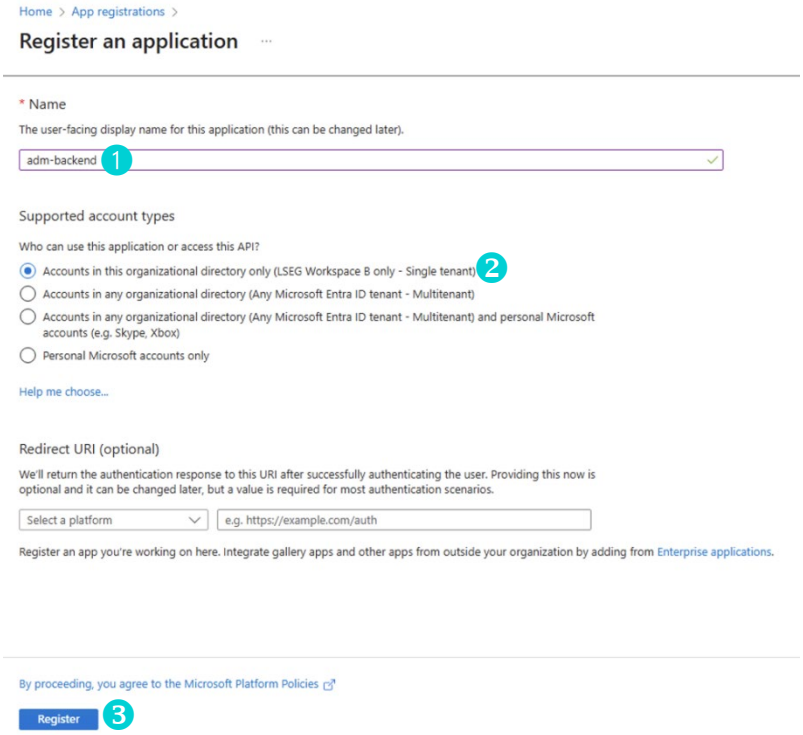
To create an app registration, you must:

- Register the backend
- Register the frontend, and
- Add the Teams administrator role

Registering the backend

To register the backend:

1. Go to the Azure Portal and login with your account.
2. Go to App Registrations.
3. Click **New registration**.
4. In the **Name** field, type **adm-backend** ❶.
5. Select the **Accounts in this organisational directory only (<tenant name> only - Single tenant)** radio button ❷.
6. Click the **Register** button ❸.



Configuring the backend

📌 IMPORTANT: If you do not make a note of the values which are required in this step, you will not be able to deploy an ARM template.

To configure adm-backend:

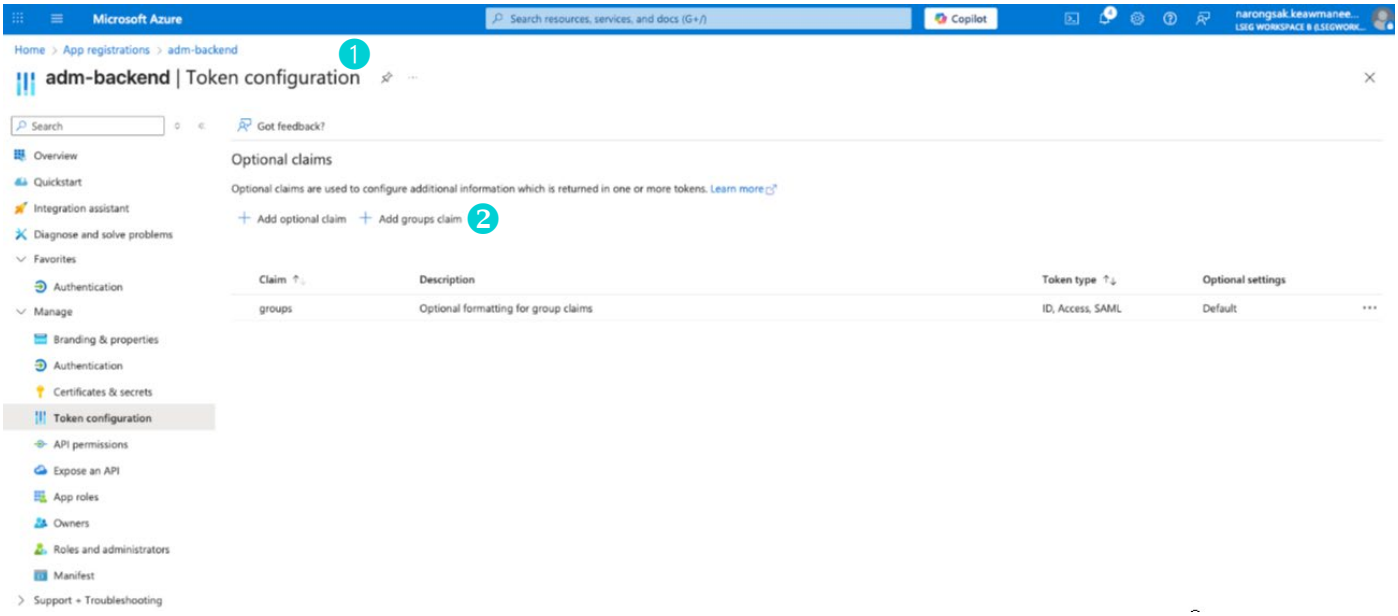
1. Select App Registration > adm-backend.
2. In the **Certificates & secrets** tab, click **New client secret**.
   a) Enter the **Description**: adm-backend-secret
   b) In the **Expires** dropdown, select 730 days (24 months)
   c) Click the **Add** button
3. Copy the **Value** that is generated. This is an important step, as you cannot go back to view these values.

Configuring a token

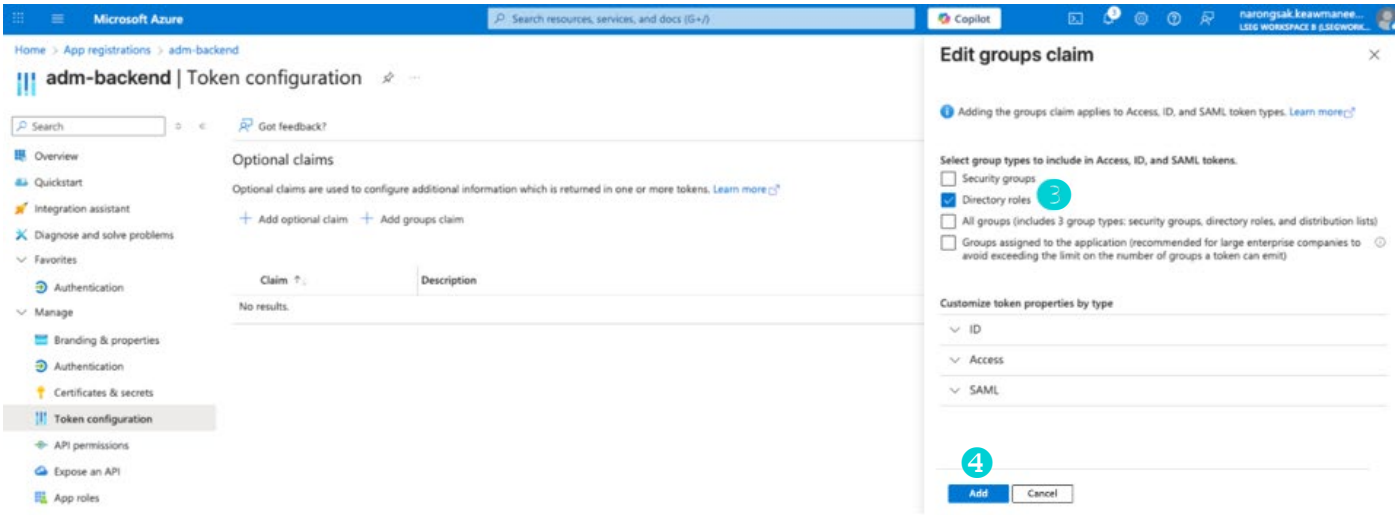
You must add a groups claim to enable ADM to verify that the login user is a Teams Administrator.

To do this:

1. Select **Token configuration** ❶.
2. Click **Add groups claim** ❷.



3. Check the **Directory roles** box ❸.



4. Click the **Add** button ❹.

Adding API permissions

To add an API permission:

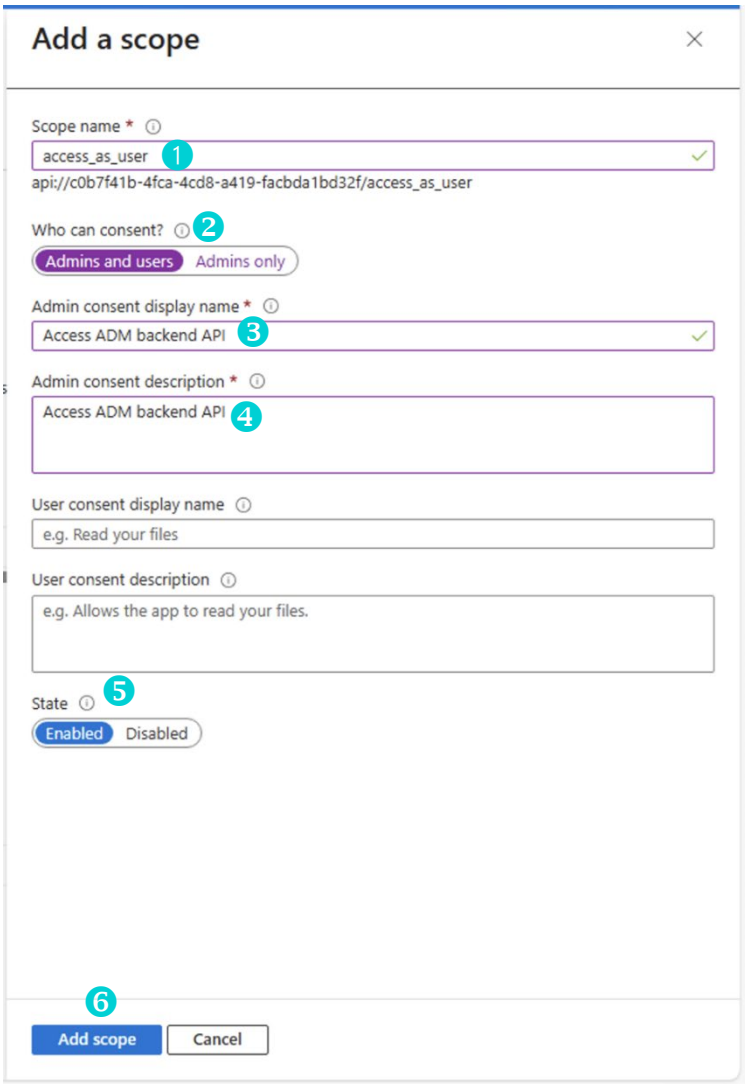
1. Click **Add a permission**.
2. Select **Microsoft Graph** and then choose **Application Permissions**.
3. The permissions are as follows:
   - Application.Read.All – read all applications
   - Group.ReadWrite.All – read and write all groups
   - GroupMember.Read.All – read all group memberships
   - Mail.Send – send mail as any user
   - Organization.Read.All – read organisation information
   - User.ReadBasic.All – read all users' basic profiles
4. If it was created automatically, you should remove the User.Read permission. This permission is not required for the adm-backend.
5. Click **Grant admin consent** for all permissions.

📌 This is a required step for the app to work.

Exposing an API

To expose an API, you need to:

1. Create an application ID URI and click on it.
2. Add a scope as follows:
   a) In the **Scope name** field, type 'access_as_user' ❶.
   b) Select 'Admins and users' in the **Who can consent?** field ❷.
   c) In the **Admin consent display name** field, type 'Access ADM backend API' ❸.
   d) In the **Admin consent description**, type 'Access ADM backend API' ❹.
   e) Ensure the **State** is 'Enabled' ❺.
   f) Click the **Add scope** button ❻.



Obtaining the adm-backend client ID

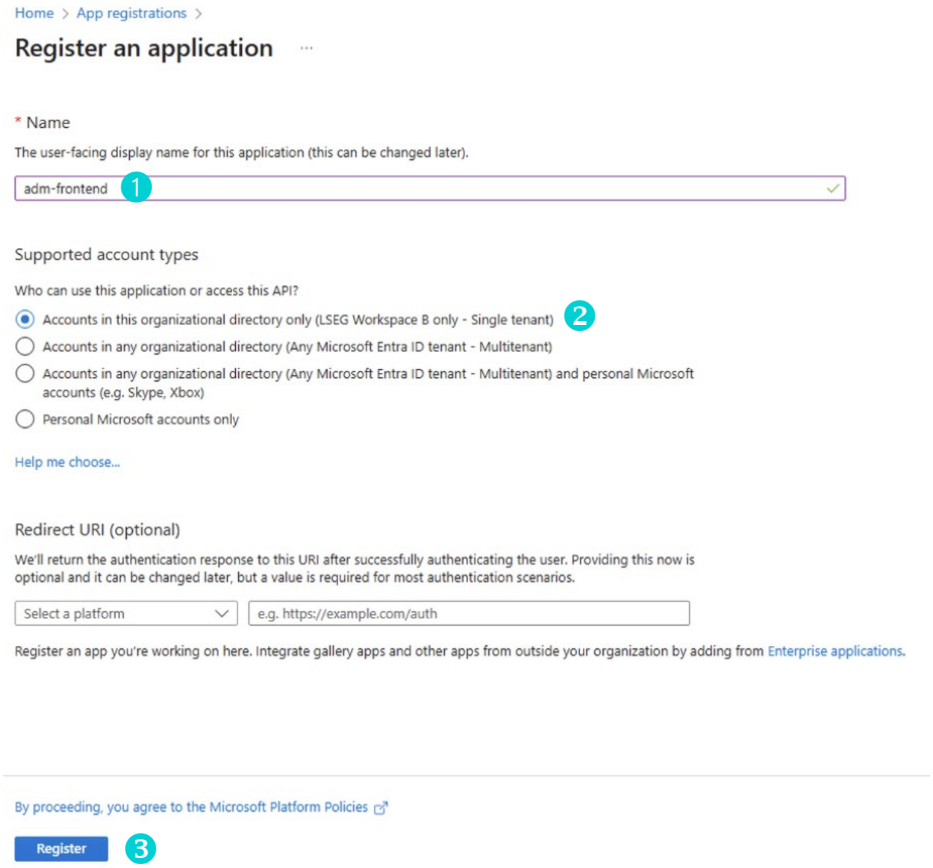
To obtain the adm-backend client ID:

1. Return to **Overview** menu in App Registration > adm-backend
2. Copy the Application (client) ID. This will be used for the 'Backend Azure Client ID' in the ARM Template.

Registering the frontend

To register the frontend:

1. Click **New registration** to create a new App Registration for the frontend app.
2. In the **Name** field, type **adm-frontend** 1.
3. Select the **Accounts in this organisational directory only (<tenant name> only - Single tenant)** radio button 2.
4. Click the **Register** button 3.



Configuring the frontend

To configure adm-frontend:

1. Select **App Registration** > adm-frontend.
2. Select **API permission** and click **Add a permission**.
3. If 'User.Read' was not automatically created, select **Microsoft Graph** and add it as a permission, ensuring the type of permission is 'Delegated'.

4. Click **Add a permission** > select **APIs my organization uses** > adm-backend.
5. Select **Delegated permissions**. The resulting screen displays as follows:

6. Select the permission **access_as_user** and click the **Add permissions** button.
7. Grant admin consent to all permissions. This is a required step for the app to work.

Adding the Teams Administrator role to ADM

To add the Teams Administrator role to ADM:

1. Go to **Microsoft Entra roles and administrators** in Azure.
2. Search for ‘Teams Administrator’ and click it.
3. To add the required assignments to the Teams administrator role:
   a) Click on **Add assignments**.
   b) Select the member(s) for whom you want to add assignments.
4. Search for **adm-backend** and select it.

📌 If you do not see 'adm-backend' in the results, go to the adm-backend resource and copy the ‘client ID’ attribute. Then return to this screen and search for the client ID value.

5. Click the **Next** button.
6. Select **Active**.
7. Select **Permanently assigned**.
8. Click the **Assign** button.

   The Teams Administrator role is now created and assigned.

Name	Principal name	Type	Scope	Membership	State
Teams Administrator					
adm-backend	e4cd6ce0-2a55-4883-93a	Service principal	Directory	Direct	Assigned

Obtaining the adm-frontend client ID

To obtain the adm-frontend client ID:

1. Return to the **Overview** menu in App Registration > adm-frontend
2. Copy the **Application (client) ID**. This will be used as the ‘Frontend Client ID’ in the ARM template.
3. Ensure the following are all saved for use in ADM template deployment:
   - Backend Azure Client Id
   - Backend Azure Client Secret
   - Frontend Client Id

Obtaining an API key and a Container Registry password

The LSEG API Key and LSEG Container Registry Password will be provided to customers by LSEG as part of the onboarding process.

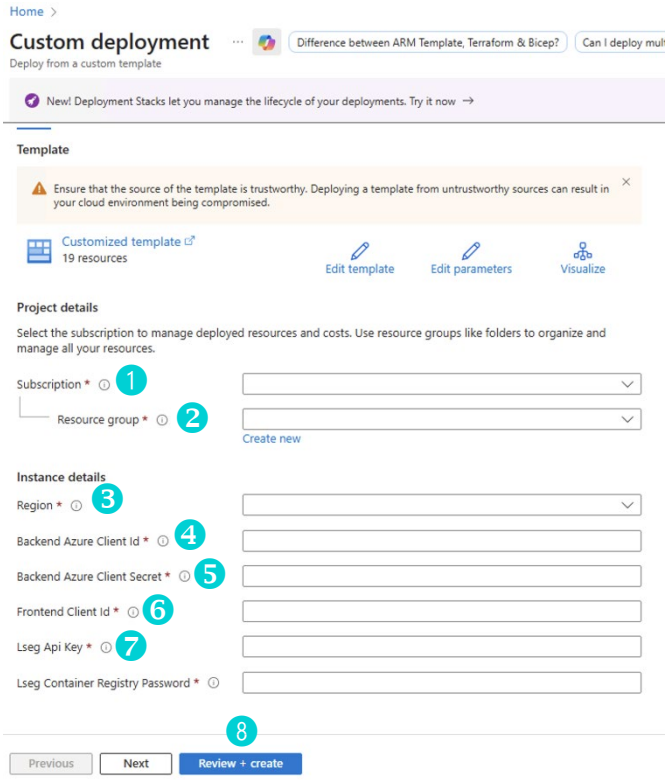
- The API Key is unique for each client.
- The LSEG Container Registry Password is required for accessing container resources needed for ADM backend deployment.

📌 Contact WSTEAMSonboarding@lseg.com if you experience any issues with your API key or password.

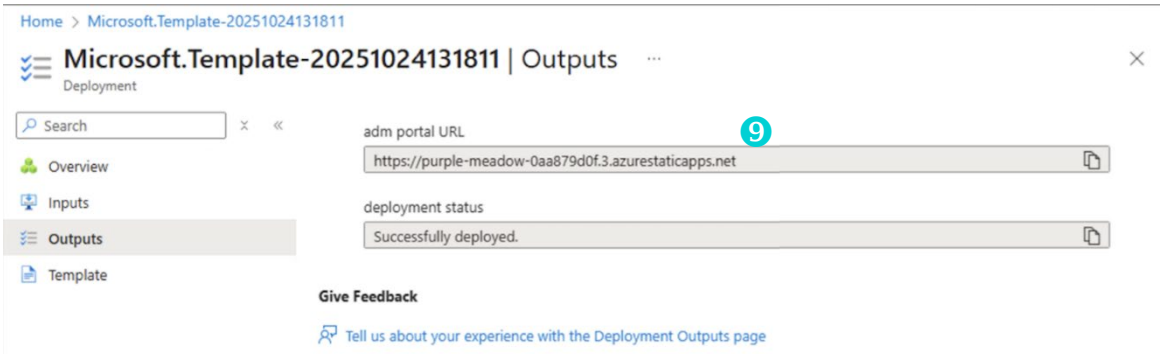
Deploying the ARM template

To deploy the ARM template:

1. Ensure you have the following required information:
   - Backend Azure Client Id
   - Backend Azure Client Secret
   - Frontend Client Id
   - LSEG API Key
   - LSEG Container Registry Password
2. Open the Azure Portal and load the LSEG ARM template for ADM deployment.
3. In the **Project details** section of the screen:
   a) Select your **Subscription** 1.
   b) Select existing **Resource group** 2 or create a new one (recommended).
4. In the **Instance details** section of the screen:
   a) Select the **Region** 3 where the ADM should be deployed.
5. Enter the **Backend Azure Client Id** 4 (Application Client ID).
6. Enter **Backend Azure Client Secret** 5.
7. Enter **Frontend Client Id** 6 (Application Client ID).
8. Enter the **LSEG API Key** 7.
9. Enter the LSEG Container Registry Password, and then click **Review + create** 8.
10. Review the terms and click **Create** to start the deployment.



After the deployment has been completed, the ADM application URL will be displayed in **Outputs** 9.



Setting up a redirect URI for authentication

This step is required to bind the Entra login and make it redirect to the ADM app after a successful login.

1. Copy the URL created when Deploying the ARM template (see previous page). This is required for adding the URI in the adm-frontend.
2. Go to **App Registration**.
3. Search for, and select, **adm-frontend**.
4. Go to Manage > Authentication.
5. Click **Add a platform**.
6. In the displayed panel, select **Single-page application**.
7. Enter **Redirect URIs** using the adm-frontend URL you have copied (see Step 1).
8. Click **Configure**.
9. Select the **Accounts in this organizational directory only (<tenant name> only - Single tenant)** radio button.
10. Click the **Save** button.

The redirect URI has now been set up for authentication.

Post-deployment administration

After ADM has been installed / deployed, customers can manage a range of administrative tasks, including:

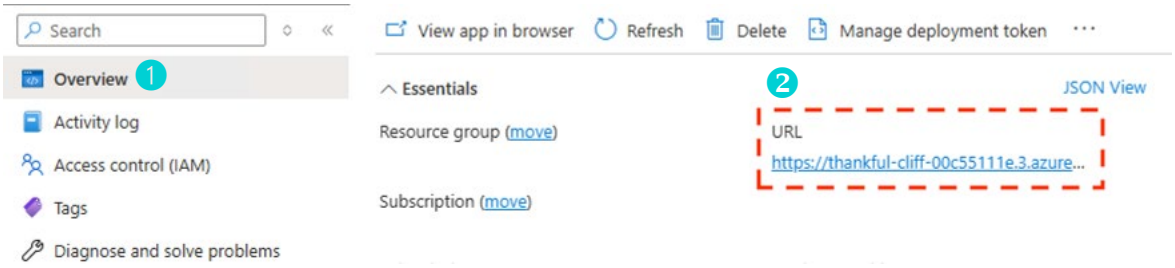
- Managing configuration
- Creating a base policy
- Managing domains
- Managing policies

Managing configuration

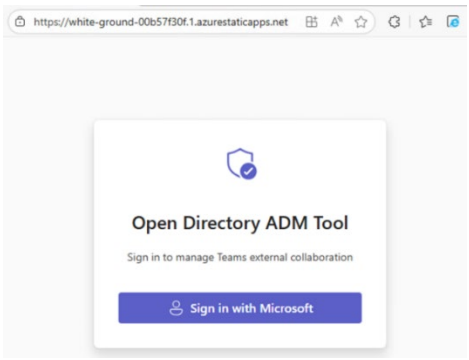
Customers are required to configure ADM before using the unique API Key provided to them by LSEG.

To configure ADM:

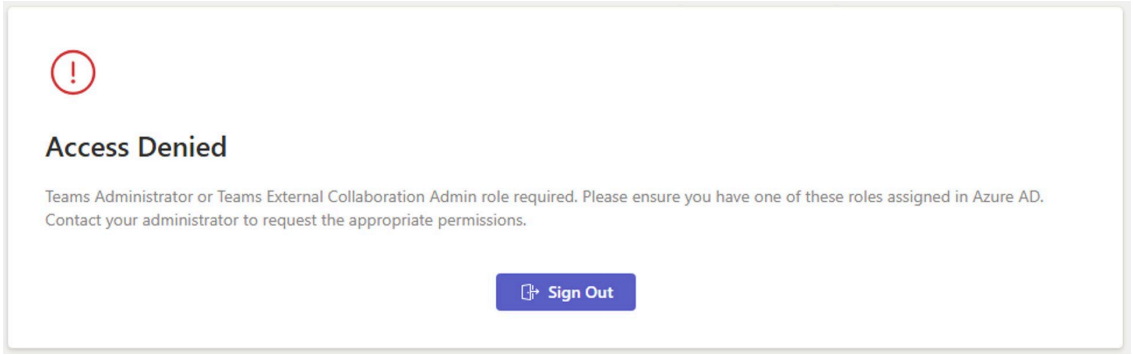
1. Open the ADM tool by using adm-frontend URL from Deploying the ARM template or by finding it in Container Apps > adm-frontend > Overview 1 > URL 2.



2. In the resulting popup window, click **Sign in with Microsoft**.



If you have not been assigned the Microsoft Teams Administrator role, you will be blocked from accessing the app and the following window will be displayed.



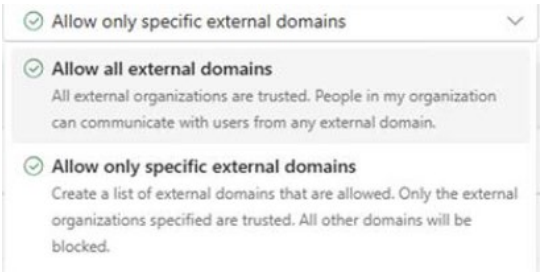
Creating a base policy

The base policy is managed in the Teams Admin Center, outside the ADM app. ADM will not interfere with the existing policy because it will create an inherited version of the base policy instead.

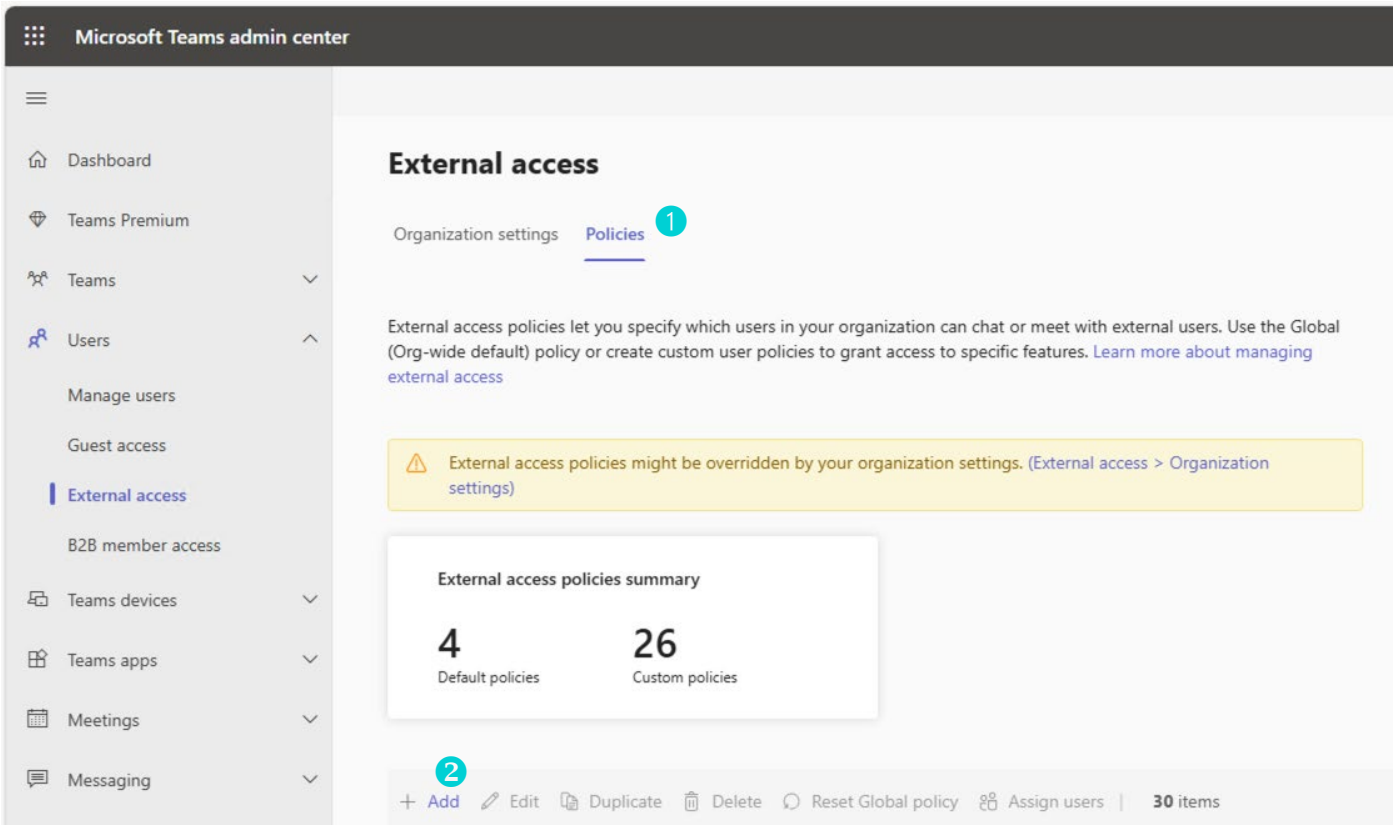
Creating a policy in the Teams Admin Center

To create a base policy in the Teams Admin Center:

1. In the **Organization Settings** tab, select either **Allow all external domains** or **Allow only specific external domains** from the dropdown list.

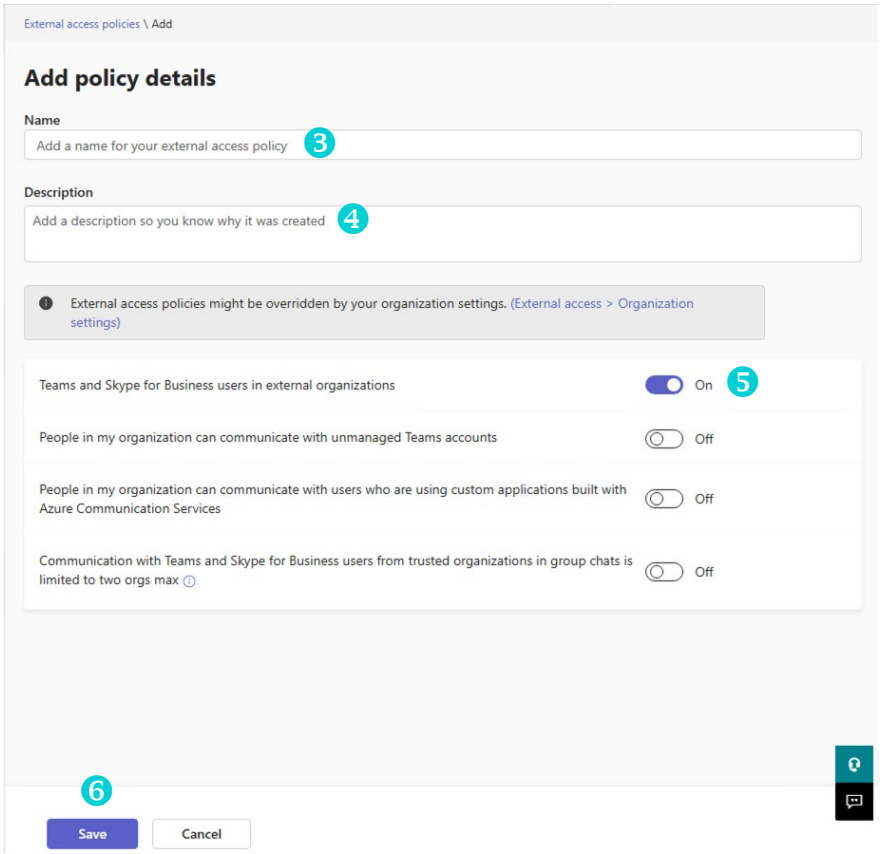


2. Go to the **Policies** tab ❶.



3. Click **Add** ❷.

4. Enter the **Name** of the new policy ❸.
5. Enter a **Description** of the policy (Optional) ❹.
6. Turn the **Teams and Skype for Business users in external organizations** switch to **On.** ❺ This option is a minimum requirement for chat with external organizations.



7. Click the **Save** ❻ button.

You have now completed the ADM deployment process using the custom template, and can proceed to configure ADM in the following ways:

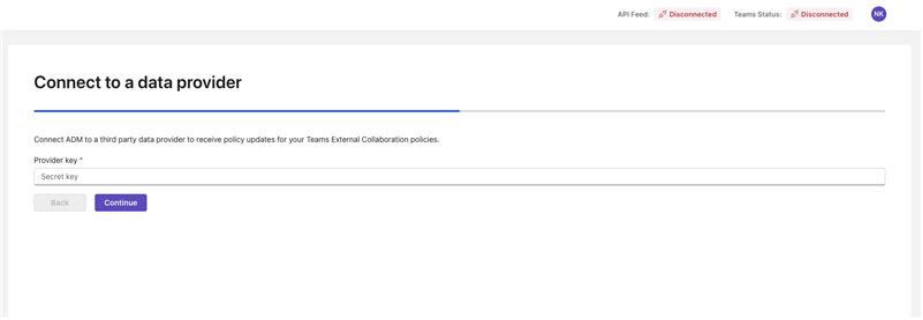
- Configuring domain feeds
- Subscribing Admin users to notifications
- Notifying Admin users of domain changes
- Managing domains
- Managing policies

Configuring domain feeds

The first time you connect to a domain feed, you will use an API key provided to you by LSEG.

To do this:

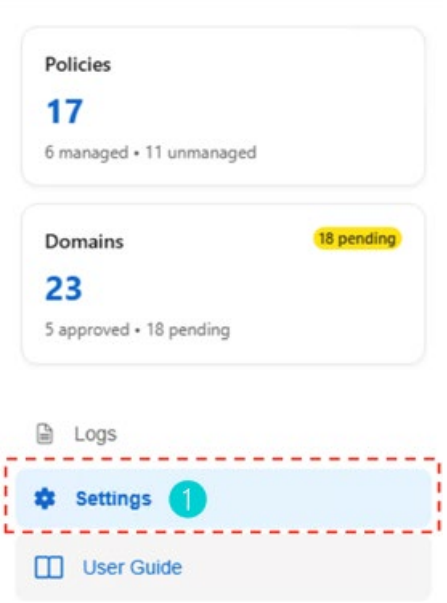
1. In the **Provider key** field, enter the API key.
2. Click the **Continue** button.



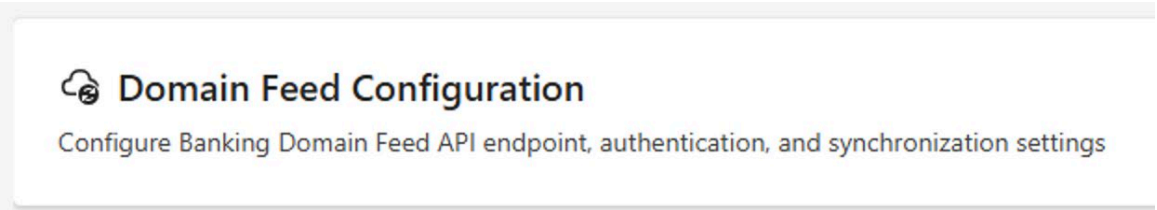
3. Click **Go to Policies** to be directed to the Policies screen.

Thereafter, you may need to configure these domain feeds:

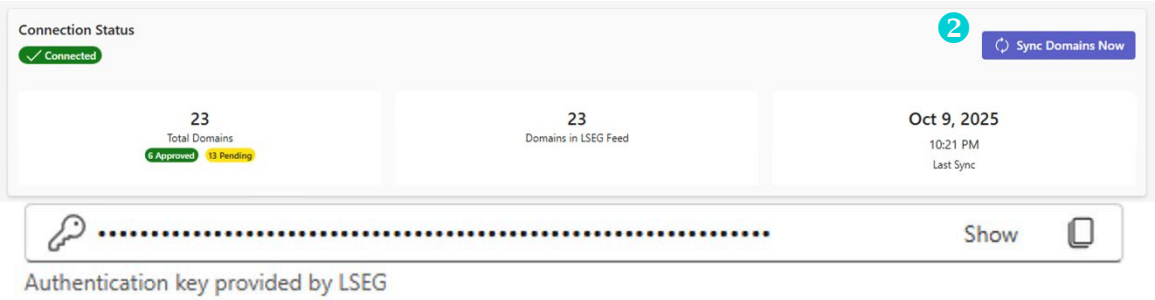
1. In ADM, select **Settings** 1.



2. In the resulting window, select **Domain Feed Configuration**.



3. In the **API Configuration** section, add the API key provided by LSEG. If you have any issues, contact LSEG Support.
4. Click **Save Configuration**.
5. Validate connectivity by clicking the **Sync Domains Now** ❷ button.

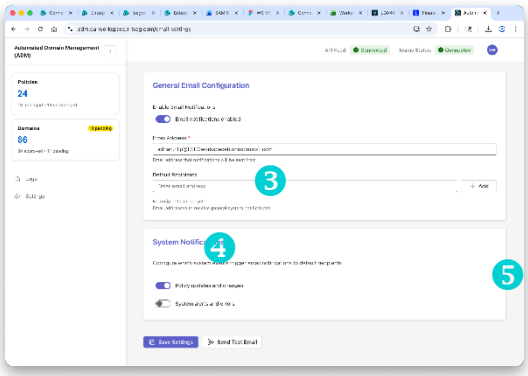


Subscribing Admin users to notifications

To subscribe Admin users to general system notifications:

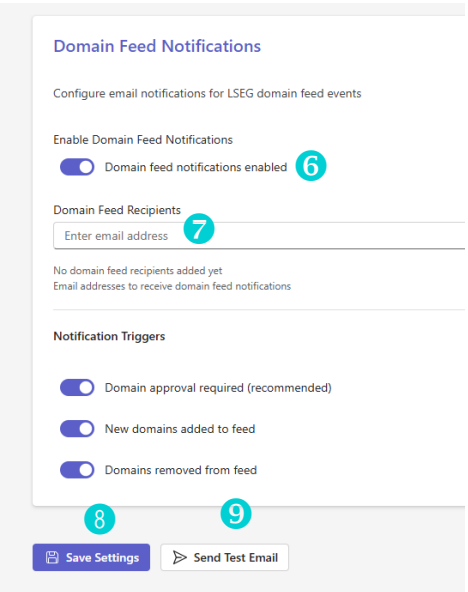
1. In ADM, select **Settings**.
2. Select **Email Notification Settings** and ensure **Email notifications enabled** ❸ is switched on.
3. Add the relevant email addresses in the **Default Recipients** ❹ field and click the **Add** button ❺.

Notifying Admin users of domain changes



To notify Admin users of changes to the domains list:

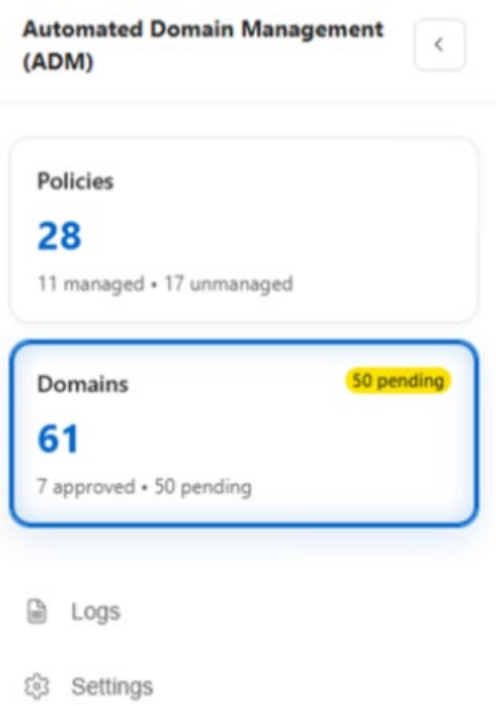
1. In ADM, select **Settings**.
2. Select **Domain Feed Notifications** and ensure **Domain feed notifications enabled** ❻ is switched on.
3. Add the relevant email addresses in the **Default Feed Recipients** ❼ field.
4. Click the **Save Settings** ❽ button and then click **Send Test Email** ❾ to verify the setup.



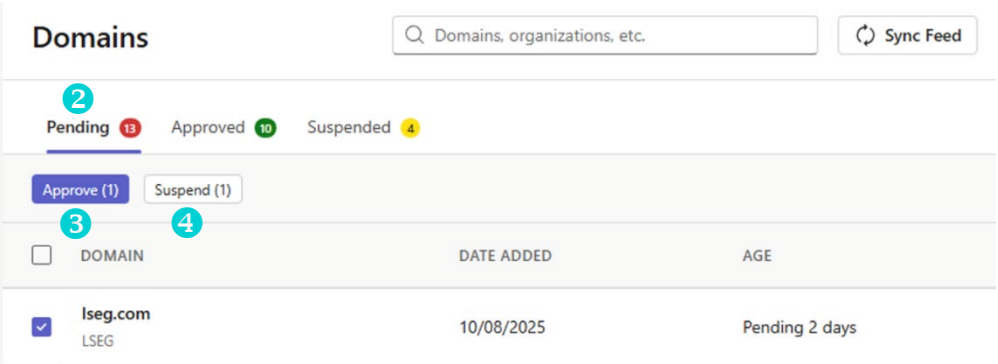
Managing domains

To either approve or suspend domains:

1. Select **Domains** 1.



2. Select domains in the **Pending** 2 list.
3. Approve 3 or 4 suspend the selected domains.



As a result:

- The approved domain will be added to the Approved list, and these domains will be available in the domain selection in Policies Management.
- The suspended domain will be added to the Suspended list, and these domains will be removed from all ADM managed policies.

Managing policies

There are two types of policies that are relevant to ADM:

Policies managed by Teams Admin Center

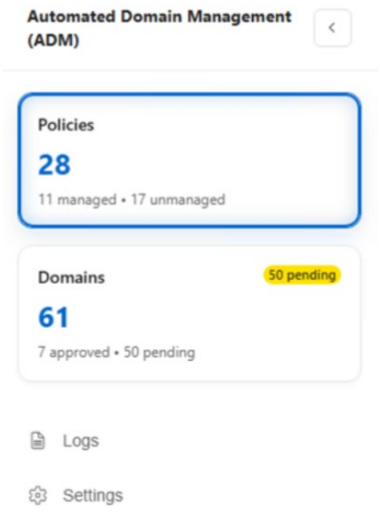
- These policies are not managed in ADM. They are created by the administrator in the Teams Admin Center, and can be used as base policies for ADM managed policies. See Creating a base policy for more information.
- Each ADM managed policy needs to be mapped 1:1 to each org policy (base policy for ADM policy).
- ADM does not change the base org policy.

Policies managed by ADM

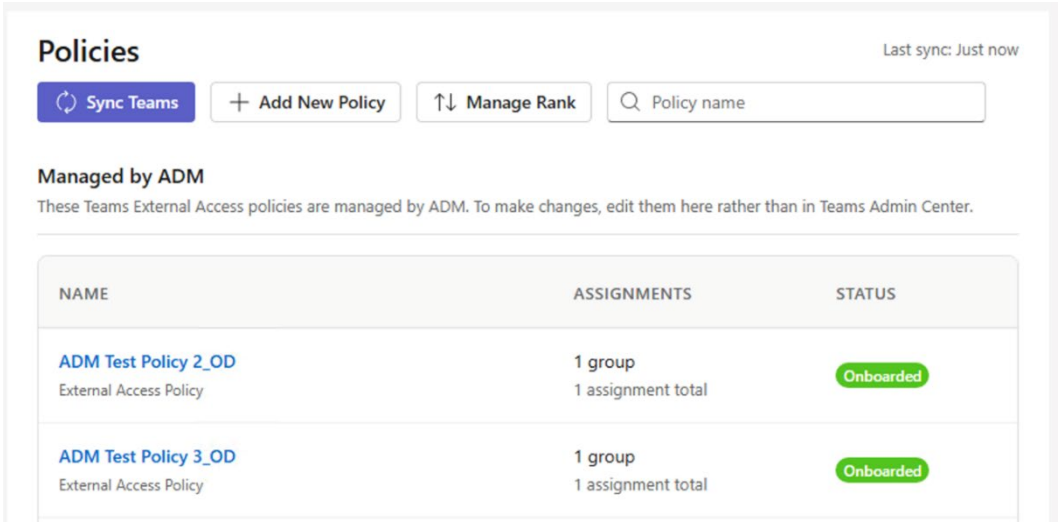
- These policies are created in ADM and can be modified within ADM.
- ADM is assigning domains, users and groups to these policies without touching anything on the org policy.

To navigate to the screen where you can manage policies for your ADM:

1. Select **Policies**.



2. Click **Sync Teams** to sync the policies available for Microsoft Teams.



Creating a new policy

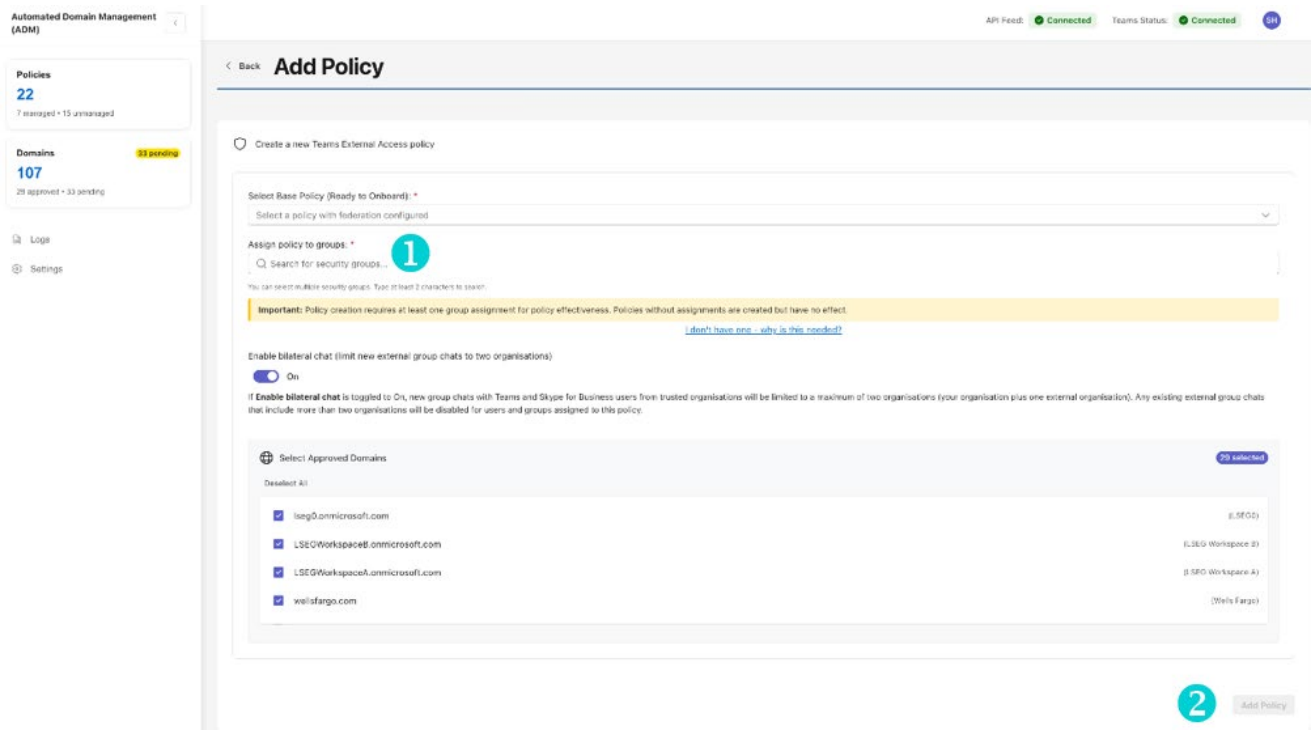
To create a new policy:

1. Click **Add New Policy** (next to the Sync Teams button).

   The Create Policy screen appears.

2. From the **Select Base Policy (Ready to Onboard)** field, select the required base policy.

3. In the **Assign policy to groups** field, select the relevant **Security Group** 1.



4. Click the **Add policy** button 2.

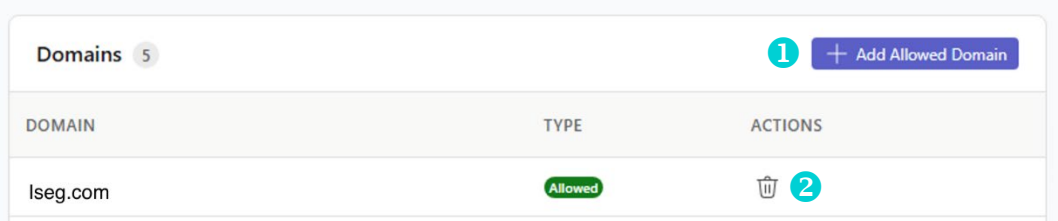
Adding or deleting a domain

To add a domain to a policy:

- Select an existing ADM managed policy and click the **Add Allowed Domain** button 1.

To delete a domain:

- Select an existing domain and click the button 2.

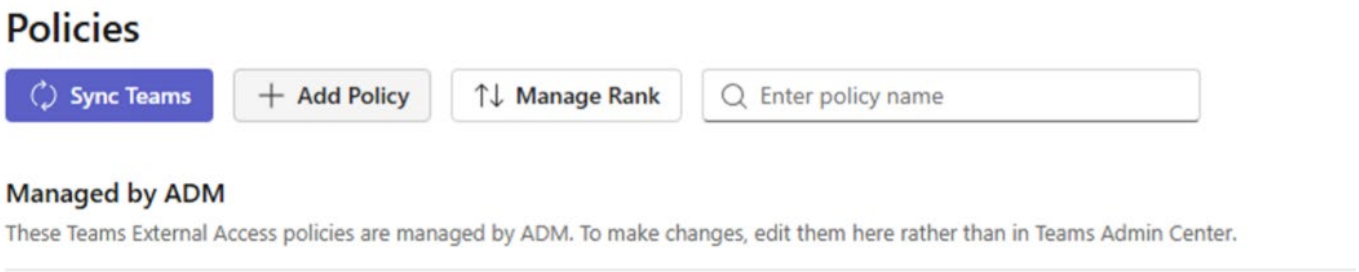


Managing a ranking

A rank refers to the priority of Teams policies applied individually to security groups or users. The lower the number of the policy ranking, the higher the priority.

To manage a ranking:

1. Click the **Manage Rank** button.



2. Adjust the rank as required.
3. Click the **Save Changes** button.

Support

If you need support during any stage of the installation and deployment process, during the preview phase you can contact us here: WSTEAMSonboarding@lseg.com.

Appendix A: Required permissions

The following table describes the permissions that should be granted to enable seamless collaboration and personalized user experiences within the LSEG Workspace app:

Resource Type	What it allows	Why it is needed	Purpose
Chat Create	Create new 1:1 or group chats.	This is the base permission to start a new chat thread.	Enables proactive communication—users do not need to manually start a chat before sharing content.
Chat.Read.Basic	View basic information about chats (such as chat IDs and participants).	Helps the app identify existing chats or confirm chat creation.	Retrieves a list of recent chats the user has participated in to generate type-ahead suggestions for recipients.
ChatMessage.Send	Send messages into a chat.	Needed to post content into the chat after it has been created.	Deliver the message with the shared content.
openid	Enables silent single sign on (SSO). Silent SSO allows users to access Microsoft Teams without re-entering credentials by using a session cookie and Microsoft Entra ID.	Required to verify the end user's Workspace license.	Allows Workspace end users to seamlessly access Workspace Teams.
TeamsAppInstallation.ReadWriteAndConsentForChat	Allows the app and app bot to install itself into a chat before sending a message.	Ensures the app is properly set up to deliver the shared content.	Ensures the app is present in the chat, to support features such as adaptive cards or bots.
User.ReadBasic.All	Accesses basic profile information such as name and photo.	Useful for showing user details in the chat UI or suggesting contacts.	Displays user details such as name and profile photo to enhance the experience and give users confidence that they are messaging the correct person.

Example of permissions being used

📌 The scenario below illustrates how some of these permissions are utilised in a typical workflow.

Scenario: A financial analyst is researching a company in LSEG Workspace and wants to quickly share insights with a colleague via Microsoft Teams. The following workflow is initiated:

1. User clicks **'**Send via Teams' in LSEG Workspace.
2. The app uses the **chat.create** permission to check if a 1:1 or group chat already exists.
3. If not, the app will create a new chat thread between the analyst and the recipient.
4. The app uses **chatmessage.send** to post a message with a link to the company insights.
5. If the app is not already installed in the chat, it uses **teamsappinstallation.readwriteandconsentforchat** to install itself.
6. The app may also use **user.readbasic.all** to display the recipient's name and profile picture in the UI.

Appendix B: Azure resources

The following resources will be deployed on the customer's Azure subscription during deployment of ADM:

Resource Type	Default Specification	Minimum Role Required
Action Group	Default value	Contributor or Monitoring Contributor
Application Insights	Default value	Contributor or custom role with Microsoft.Insights/components/write
Azure Database for PostgreSQL flexible server	Name:Standard_B1ms Tier:Burstable PostgreSQL version: 14.19	Contributor or DBAas Contributor
Container App	Default value	Contributor or Azure Kubernetes Service RBAC Writer
Container Apps Environment	workloadProfileType: Consumption	Contributor or custom role with Microsoft.App/managedEnvironments/write
Key vault	Family:A Name:Standard	Contributor or Key Vault Contributor or custom role with Microsoft.KeyVault/vaults/* permissions
Log Analytics workspace	Name:PerGB2018	Contributor or Log Analytics Contributor
Static Web App	Name:Free Tier:Free	Contributor or Website Contributor
Storage account	Name:Standard_LRS Tier:Standard	Contributor or Storage Account Contributor
App Registration - frontend	Graph API delegated permissions (admin consent required): • User.Read – Required for reading user info of the current admin user who is using the app	Contributor
App Registration - backend	Graph API Application permissions (admin consent required): • Application.Read.All - Required for checking app consents are configured correctly • Group.ReadWrite.All - Required for creating security groups to assign newly created policies • GroupMember.Read.All - Required for read security group members to analyze assignments • Mail.Send - required for sending email • Organization.Read.All - Required for teams powershell authentication • User.ReadBasic.All - Required for reading user info for individual user search	Contributor – for

Appendix C: Frequently asked questions

What is the Automated Domain Management (ADM) app?

The ADM app is a management tool for Microsoft Teams administrators designed to keep external collaboration policies aligned with LSEG's Open Directory network. It automates the process of:

- Subscribing to a domain feed
- Updating federation policies
- Managing collaboration rules at scale

What problem does the ADM app solve?

To communicate externally in Microsoft Teams, organisations must federate with numerous entities in a point-to-point way. Existing workflows require manual processing, which is time-consuming and prone to error. The ADM app automates this process, reduces administrative overhead, and allows policies to remain up to date. ADM features a user-friendly front-end, robust backend services, and is deployed inside the customer's own environment to ensure that sensitive data does not leave the customer data boundary.

How does the app work?

The ADM app creates external access policies within Microsoft Teams, facilitating communication between users and other members of the Open Directory network. These policies, also referred to as external collaboration or federation policies, ensure that only specific individuals in the organisation (in other words, Open Directory users) can communicate only with other Open Directory customers, and to those within your existing federation policies. This capability is enabled by the new Granular Federation Controls feature in Microsoft Teams.

The ADM app will:

- Create new external access policies in Teams
- Synchronise created policies with:
  - Approved domains received from LSEG
  - Other, organisation-managed, policies in Teams
- Assign policies to appropriate users / groups
- Provide workflow for admins to approve / reject domains received from LSEG

The ADM app does not:

- Send tenant configuration or messaging data to LSEG
- Store tenant configuration other than for policies it manages (which it does locally in your environment)
- Edit existing organisation configuration or policies, except in very limited circumstances and with admin consent (see What is Granular Federation Control?, below).

What is Granular Federation Control?

Granular Federation Control is a new feature in Microsoft Teams which enables administrators to configure different federation policies for different groups of users in their organisation.

For granular federation controls to work, the property AllFederatedUsers must be set to true. This is a tenant-wide setting. The ADM app will check this and inform the administrator that it must be set correctly before continuing. The administrator can do this themselves, or the ADM app can do it on their behalf. Changing this value from false to true will enable federation at the tenant-wide level. If this was set to prevent any federation within the tenant, the admin should set the AllowedDomains property to null.

For more information, see Set Tenant Federation Configuration and Set External Access Policy on Microsoft Learn.

How does the ADM app know which settings and domains to configure?

New policies generated by the ADM app are based upon a pre-existing policy that is managed by the organisation's administrator within the Teams Admin Centre (TAC). The ADM app continuously synchronises these new policies with the corresponding base policy. Administrators continue to update their org policies as usual, and the ADM app keeps the policies it manages aligned with any updates made by administrators to the base policy in the TAC.

Federated domains are configured on the ADM-managed policy by referencing both the original baseline policy and the list of approved domains provided by LSEG. This approach allows approved domains from LSEG to be configured, while retaining the organisation’s ability to customise policies enabling communication beyond Open Directory.

ADM does not edit any organisation-managed policies, meaning that administrators can continue to manage their existing policies as usual and the ADM app will resolve any policy conflicts as per the settings chosen by the administrator.

How does the ADM app know which users should be assigned the policy?

The app knows which users should be assigned the policy once administrators have created the relevant security groups and specified users to them.

Do I have control over these policies?

Yes. Administrators can specify whether new domains added to the network by LSEG should be automatically approved and applied to their organisation’s policies, or if approval is required first.

Additionally, adminitrators can specify ’always-block‘ lists that take precedence over the domain feed, ensuring critical domains remain blocked regardless of feed updates.

As policies are synced with base organisation policies, administrators continue to manage their organisation policies as usual, and changes will be replicated to the corresponding ADM-managed policy. This allows administrators to add additional domains which are not members of Open Directory, ensuring users are still able to communicate with these organisations.

How does the ADM app handle security and compliance?

The ADM app leverages Microsoft Azure’s platform-managed services to allow high availability, security, and compliance. It supports secure authentication with Entra ID and maintains comprehensive audit logs. The ADM app requires a service principal in your Entra tenant so that it can connect to the Teams PowerShell service. This service principal requires the Teams Administrator role.

The ADM app also requires administrators to create an app registration in Entra to enable SSO to the management portal.

What technology does this use and how is it deployed?

The solution uses Azure platform-managed services, which provide native high availability without requiring custom application-level high availability logic.

The ADM app will have dependencies on the following Azure platform services:

- Azure Container Apps
- Azure Static Web Apps
- Key Vault
- Application Insights
- PostgreSQL

Services such as Azure App Service, Function Apps, Key Vault, and Entra ID are inherently resilient and distributed, ensuring uptime through zone and regional redundancy.

For disaster recovery, the solution is designed to be re-deployable by administrators in alignment with their specific disaster recovery requirements. Infrastructure-as-Code templates support rapid provisioning in alternate regions, while data services use

geo-redundant configurations to protect against regional failures. Key Vault secrets and configuration settings can be replicated across vaults, and monitoring via Application Insights allows visibility and supports proactive recovery actions.

Deploying the ADM app requires provisioning infrastructure in Azure, which can be automated using Azure Resource Manager (ARM) and PowerShell deployment scripts.

Customers are free to tailor the solution per their requirements, for example selecting alternative high availability / disaster recovery options, scaling, network connectivity, load balancing, and so on.

lseg.com

LSEG DATA & ANALYTICS

Document version: 100.05