

Automate uploading logs to Defender for Cloud Apps Cloud Discovery

This sample guide/scripts will help you automate uploading firewall logs to Defender for Cloud Apps using API.

This helps eliminate the need to deploy a log collector, and makes it more flexible to deploy and maintain.

In addition, this guide is using the Microsoft Graph API, instead of the legacy Defender for Cloud Apps APIs, which is in a path to being deprecated in the future.

Note from [Managing API tokens - Microsoft Defender for Cloud Apps | Microsoft Learn](#)

ⓘ Note

The legacy method of accessing the Defender for Cloud Apps API is still supported. However, it is on a deprecation path, so we recommend using the methods described on this page.

This guide is based on the following documentation, but some customization is needed as described below to use it for Shadow IT Discovery purposes .

Intro:	REST API - Microsoft Defender for Cloud Apps Microsoft Learn
Authentication	Managing API tokens - Microsoft Defender for Cloud Apps Microsoft Learn
Discovery API	Defender for Cloud Apps Cloud Discovery API - Microsoft Defender for Cloud Apps Microsoft Learn

Here are the main tasks:

- 1: Create an Azure AD App
- 2: Give the App permissions over Defender for Cloud Apps APIs
- 3: Document key elements from the App, such as App ID, Tenant ID and App Secret
- 4: Document the Defender for Cloud Apps API URL
- 5: Create a new Data Source in Defender for Cloud Apps to receive the logs
- 6: Configure the PowerShell scripts with the data collected above and test it.
- 7: Schedule the script to run according to your preferences.

Here are detailed steps:

Step 1-Create an App using either [Application Context](#) or [User Context](#) and give it consent.
In general, you'll need to take the following steps to use the APIs:

- Create an Azure Active Directory (Azure AD) application
- Get an access token using this application
- Use the token to access the Defender for Cloud Apps API

To create an App in Azure AD, follow the steps 1, 2 and 3 on this documentation
<https://learn.microsoft.com/en-us/defender-cloud-apps/api-authentication-application#create-an-app>

When you get to step 4 on, use the instructions below instead of the steps in the public doc.
You will need to customize the permissions needed for managing Cloud Discovery

Step 4: Give Permissions to the "Microsoft Cloud App Security" (Application ID 05a65629-4c1b-48c1-a78b-804c4abdd4af)

Request API permissions

Select an API


Microsoft APIs APIs my organization uses My APIs

Apps in your directory that expose APIs are shown below

microsoft cloud	
Name	Application (client) ID
Microsoft Cloud App Security	05a65629-4c1b-48c1-a78b-804c4abdd4af
Microsoft Cloud App Security	25a6a87d-1e19-4c71-9cb0-16e88ff608f1

Step 5: Give it permissions over Discovery.manage and discovery.read

[← All APIs](#)

 Microsoft Cloud App Security
<https://microsoft.onmicrosoft.com/873153a1-b75b-46d9-8a18-ccaaa0785781>

What type of permissions does your application require?

Delegated permissions

Your application needs to access the API as the signed-in user.

Application permissions

Your application runs as a background service or daemon without a signed-in user.

Select permissions

[expand all](#)

Start typing a permission to filter these results	
Permission	Admin consent required
✓ discovery (2)	
<input checked="" type="checkbox"/> discovery.manage ⓘ discovery.manage	Yes
<input checked="" type="checkbox"/> discovery.read ⓘ discovery.read	Yes

Step 6: Grant Admin Consent

Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as all the permissions the application needs. [Learn more about permissions and consent](#)

+ Add a permission ✓ Grant admin consent for Contoso M365SECURITY

Step 7: Get an app secret under "Certificates & Secrets" and copy the value

MDA Discovery App | Certificates & secrets

< Got feedback?

Overview

Quickstart

Integration assistant

Manage

Branding & properties

Authentication

Certificates & secrets

Token configuration

API permissions

Expose an API

App roles

Owners

Credentials enable confidential applications to identify themselves to the authentication service when receiving tokens at a web address at scheme). For a higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

Application registration certificates, secrets and federated credentials can be found in the tabs below.

Certificates (0) Client secrets (1) Federated credentials (0)

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

+ New client secret

Description	Expires	Value	Secret ID
MDADiscover	8/9/2023	[REDACTED]	c162af92-3f2e-

Step 8: Go to "Overview and copy the Application (Client ID) / Directory (Tenant ID)

MDA Discovery App

< Delete Endpoints Preview features

Overview

Quickstart

Integration assistant

Manage

Branding & properties

Authentication

Certificates & secrets

...

Got a second? We would love your feedback on Microsoft identity platform (previ

Essentials

Display name : [MDA Discovery App](#)

Application (client) ID : dea6c215-4e6e-442f-8aef-95818a54128c

Object ID : 1d60ba08-da6b-4c41-b039-7c1f3629fcee

Directory (tenant) ID : [REDACTED]

Supported account types : [My organization only](#)

Step 9: Copy your Tenant API URL from the Defender for Cloud Apps settings/Settings > System > About page

<https://security.microsoft.com/cloudapps/settings?tabid=about>

System

About

Organization details

Mail settings

Scoped deployment and
privacy

Preview Features

IP address ranges

About

Version

246.189

Region

West US 2

Data center

US3

Defender for Cloud Apps Tenant ID

101097833

Azure Active Directory Tenant ID

fba8356f-c86d-4408-9bd0-776f655478a1

API URL

<https://msdx132695.us3.portal.cloudappsecurity.com>

Step 10: Create a Data Source in the Defender for Cloud Apps / Settings > Cloud apps > Cloud Discovery > Automatic Log upload

Copy the data source name

<https://security.microsoft.com/cloudapps/settings?tabid=discovery-autoUpload&innertab=dataSources>

Add data source

Name *

PaloAlto

Comment

Source *

PA Series Firewall

[View sample of expected log file](#), and compare it with yours

Receiver type *

Syslog - TLS

☐

Anonymize private information

Store and display only encrypted usernames.

Add

Cancel

Make sure to copy these values to add to the script later:

Data	From	Value
Application (Client ID)	Azure AD App / Overview	
Directory (Tenant ID)	Azure AD App / Overview	
Application Secret (value)	Azure AD App / Certificates & Secrets	
Tenant API URL	Defender for Cloud Apps	
Data Source Name	Defender for Cloud Apps	

Step 11: Download the Discovery-sample.zip folder and extract to a place of your preference and modify these two files:

Get-AppToken-MDA.ps1

```
$tenantId = '<Your Tenant ID>' ### Paste your tenant ID here  
$appId = '<Your app ID>' ### Paste your Application ID here  
$appSecret = '<Your App secret key value>' ### Paste your Application key here
```

Upload-DiscoveryLogs-MDA-sample.ps1

```
$TenantURL = "<Your Tenant URL>" # you can find this at (link)  
$LogSource = "PALO_ALTO" #Change to your log source (link)  
$DataSourceName = "PaloAltoDemo" #Create a Data Source in "Automatic log upload" in (link)  
$sourceScriptFolder = "<Ex:C:\Scripts\Discover>" #This is the path to the Discover folder where the  
PowerShell scripts are
```

Step 12: Once the variables are configured, it's time to test the scripts.

- Copy some sample logs to the "New Logs" folder
- Run the Upload-DiscoveryLogs-MDA.ps1 script.
- Verify if the logs were uploaded successfully by running the List-ContinuousReports-MDA.ps1 and comparing the numbers for the data source you are using.
- Verify in the Defender for Cloud Apps portal if you can see a new report in Cloud Discovery for the data source you created.

Step 13: Configure your firewall/proxy logs to save new logs to the "New logs" folder of the machine running the script, or to a location the machine can read from.

Step 14: If everything works well, you can schedule the Upload-DiscoveryLogs-MDA.ps1 script to run a couple of times a day, depending how often you generate and save your firewall logs to the "New Logs" folder.