
# Microsoft 365 Registration Apps and Security

SEPTEMBER 2025

LIONEL.TRAVERSE@ADMIN365.FR

# 1. Table of contents

# 2. Introduction

This document provides a comprehensive overview of Microsoft 365 applications utilized for Single Sign-On (SSO), Application Proxy and direct access protocols such as EWS, POP, IMAP, SMTP, and GRAPH.

Here, you will find detailed guidance on how to configure these applications within your environment, ensuring seamless integration and reliable connectivity for diverse services.

Special attention will be given to critical aspects of security, including scoping of permissions and protection of application usage, so that your organization’s data remains secure and compliant.

You will also find examples of token requests that can help you reproduce the behavior of certain applications and better assess your security.

# 3. Overview and concepts

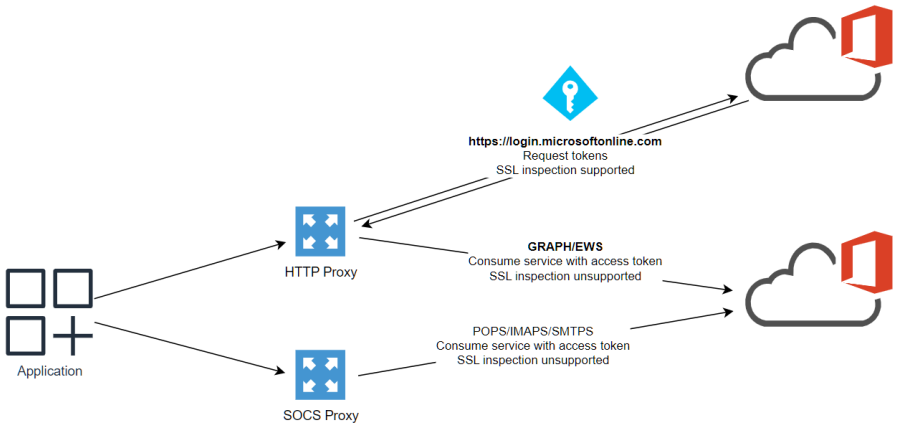
## 3.1. Apps and authentication

For a program to connect to Microsoft 365, it is imperative to use an APP object that has been created within the M365 tenant. This APP object can be either an application created by the tenant administrator or a trusted application that already exists in another tenant, as is the case for multi-tenant apps. Without utilizing an APP, it is impossible to establish a connection to Microsoft 365; simply having an account and password is insufficient. The APP object is essential as it provides the necessary authorization and permissions required for the program to access user resources securely.

When requesting an access token with Microsoft 365, it is mandatory to provide the application ID along with additional authentication components such as a secret, certificate, account with a password, access code, or a previously obtained refresh token.

The access token provided upon successful authentication is valid for 1 hour, while a refresh token is valid for 90 days by default. Both access tokens and refresh tokens are formatted as JSON Web Tokens (JWT), ensuring secure and standardized token representation.

It is important to note that the access token is always issued by the tenant that hosts the resource. The site responsible for providing these tokens is https://login.microsoftonline.com.

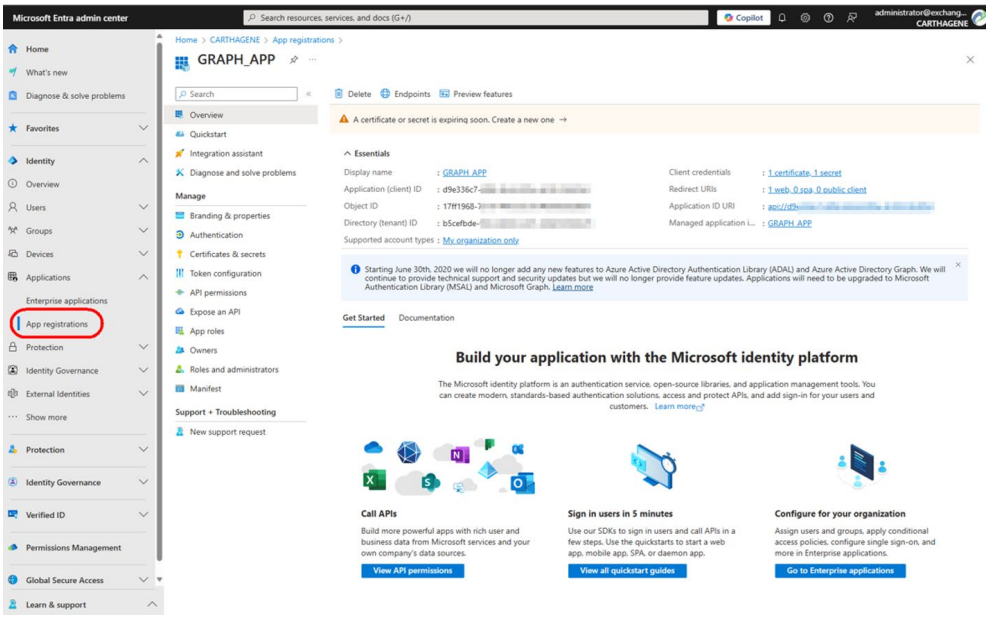



## 3.2. Creation of registered apps

To create a registered application (APP) in Microsoft 365, follow these steps:

- Access the Azure Portal: Log in to the Azure portal using your Microsoft 365 administrator account at https://admin.microsoft.com.
- Navigate to Entra: From the left-hand navigation pane, select "Identity."

- App Registrations: In the Identity menu, select "App registrations."
- New Registration: Click on the "New registration" button at the top of the screen.
- Configure Application Details:
- **Name**: Enter a name for your application. This can be anything that helps you identify the application.
- **Supported account types**: Choose who can use the application. Options include accounts in your organization only or multi-tenant options.
- Redirect URI (optional): Configure the location (redirect URI) where the tokens will be sent by the authorization server after authentication.
- Register the Application: Complete the registration by clicking the "Register" button at the bottom of the screen.
- Note the Application (client) ID: Once registered, you will be taken to the app's overview page. Note down the "Application (client) ID," as this will be required for authentication.
- Configure API Permissions:
- Select "API permissions" from the navigation menu.
- Click "Add a permission" and choose the APIs your app needs to access.
- Add the necessary **delegated** or **application** permissions.
- Don't forget to click "Grant admin consent" for the required permissions.
- Generate Client Secret or Certificate or select “Allow public flows”.
- Navigate to "Certificates & secrets" in the app's menu.
- Choose to add a new client secret or upload a certificate. Note down the secret value immediately, as it will be hidden later.
- You can also configure the app without a certificate or secret and use the “Allow public client flows” option..
- Additional Settings: Configure any additional settings such as branding, user roles, and token configuration as needed.

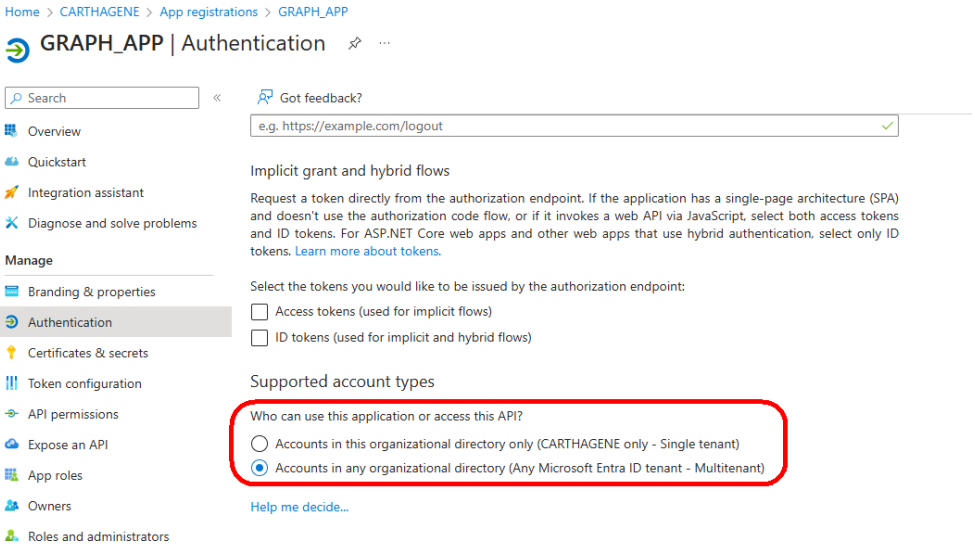



## 3.3. Multi-tenant apps

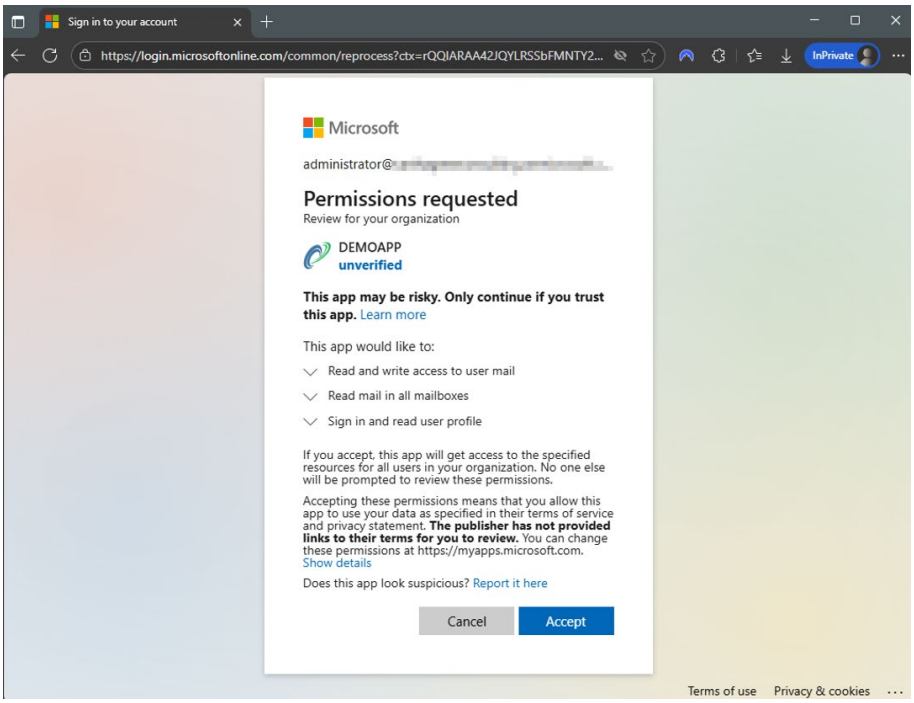
A significant feature of registered apps within M365 is the ability to configure them as multi-tenant applications. This means the app can be accessed by users across multiple organizations rather than being restricted to a single tenant (organization). Benefits of Multi-Tenant Apps:

- Wider Reach: Multi-tenant apps can serve users from multiple organizations, expanding the app's potential user base.
- Central Management: Developers can manage a single instance of the app that serves multiple tenants, simplifying updates and maintenance.
- Resource Sharing: Users from different organizations can collaborate and share resources through the app.

To configure an app as a multi-tenant app, developers need to set the app's "Supported account types" to include " **Accounts in any organizational directory (Any Microsoft Entra ID tenant - Multitenant)**" during the app registration process. This allows the app to be available across multiple organizations.



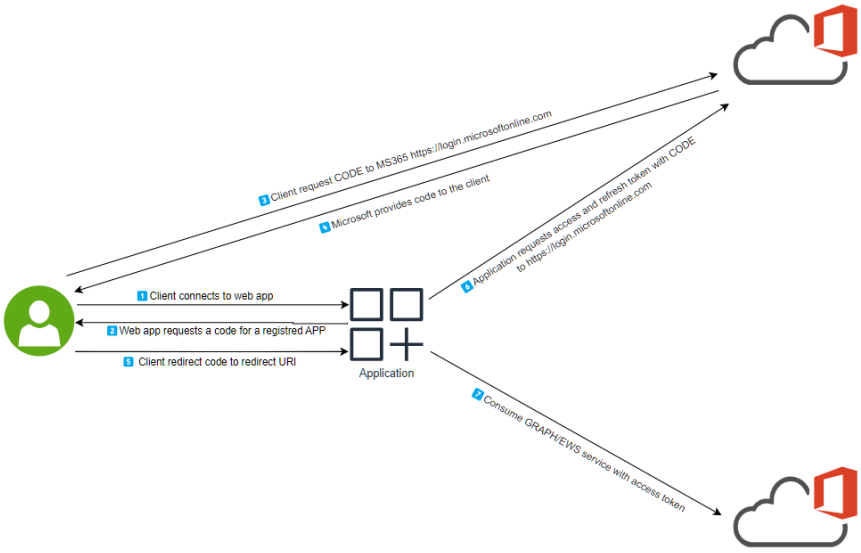

To deploy the application to another tenant, simply use the link **https://login.microsoftonline.com/common/adminconsent?client_id=<CLIENT_ID>** with an admin account from the target tenant. A message will then prompt you to consent permissions. Next the application will appear in the list of enterprise apps.




When a multitenant APP is integrated into another tenant, this APP is going to be able to request TOKEN from this tenant. The certificates/secrets are located inside the owner's tenant and the consumer's tenant can't access these parameters.

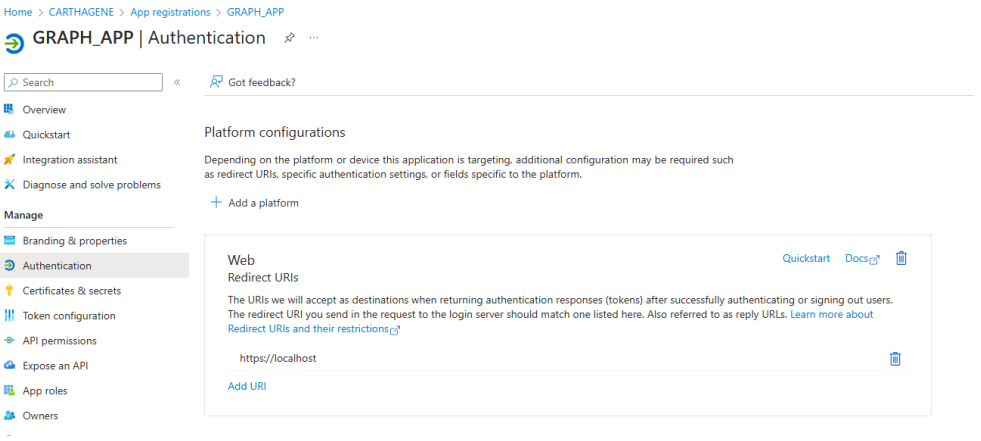
## 3.4. Apps with redirect URL

This application leverages code-based authentication to ensure secure access and operations. It utilizes delegate permissions to allow the application to act on behalf of the user, providing a seamless and secure user experience.




The authentication Process is next:

- User connects to a Web Application. The Web Application requests the user to provide an Access Code
- User Initiates Login: The user starts the login process by entering their M365 credentials.
- Code Generation: A unique authentication code is generated and sent to the user. The value of the code is redirected to the URL defined in the registration APP or the user enters the received code into the application.
- The application can request refresh and access tokens on behalf of the user.




## 3.5. Apps' credentials

A Registration App can be secured using different methods to ensure the safety and integrity of the application. The available protection options include:

**Secret**: A string of characters used for authentication.

**Certificate**: A digital document for higher security.

**Allow Public Flows**: No secret or certificate required. Only delegate permissions can be used.

- ### Secret

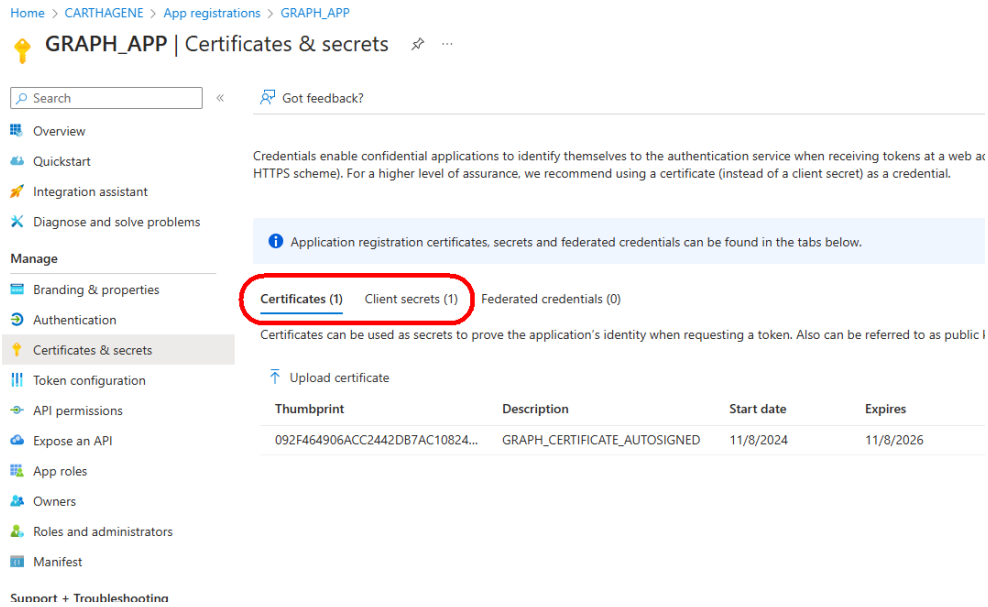
The application can be protected by a client secret, which is a string of characters known only to the application and the authentication server. This client secret is used to verify the identity of the application during the authentication process.

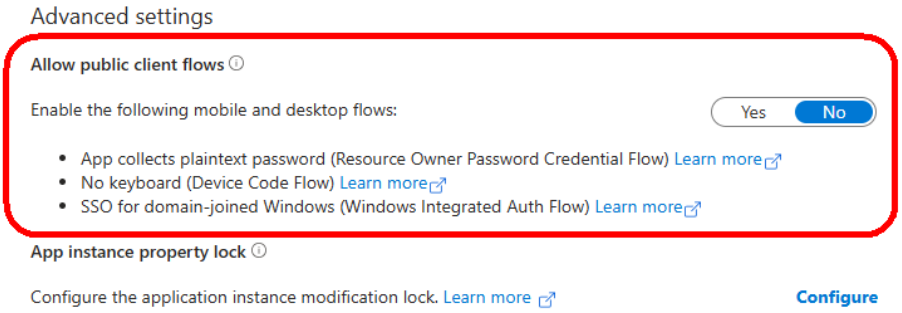
- ### Certificate

Alternatively, the application can use a certificate for authentication. A certificate is a digital document that uses a public key infrastructure (PKI) to verify the identity of the application. This method provides a higher level of security compared to a secret key.

- ### Allow Public Flows

If the "allow public flows" option is enabled, the application does not require a secret or certificate for authentication. This option is designed for scenarios where the application needs to be accessible to the public. However, it is important to note that when "allow public flows" is enabled, the application can only consume delegate permissions. Delegate permissions allow the application to act on behalf of the user.




## 3.6. Apps' permissions

When configuring a Registration Application, it's important to understand the distinction between application permissions and delegate permissions. Both types of permissions define how the application interacts with resources, but they serve different purposes and have different scopes.

- ### Application Permissions

Scope: Application permissions are granted directly to the application itself, independent of any user context.

Usage: These permissions are typically used for background services or daemon applications that need to access resources without user interaction.

Authority: The application acts as its own entity, with permission granted to it by an administrator.

Consent: Application's permissions must be consented by **administrators** to become usable.

Example: An application with application permissions might access a database to perform scheduled data processing tasks or access a mailbox without user interaction.

- **Delegate Permissions**

Scope: Delegate permissions are granted to the application **on behalf of a signed-in user**.

Usage: These permissions are used when the application needs to perform actions on behalf of the user, requiring user interaction and consent.

Authority: The application acts with the same permissions as the user, limited to what the user can do.

Consent: Delegate permissions can be consented by **administrators or users to become usable**.

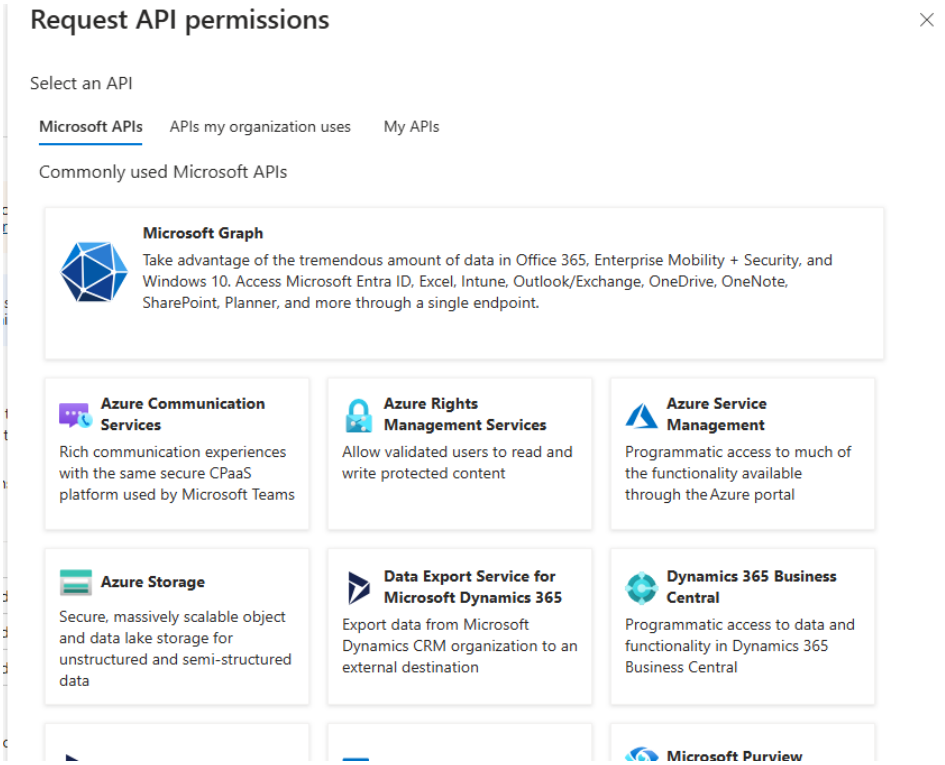
Example: An application with delegate permissions might access a user's email to send messages on their behalf (Thunderbird, Samsung Email).

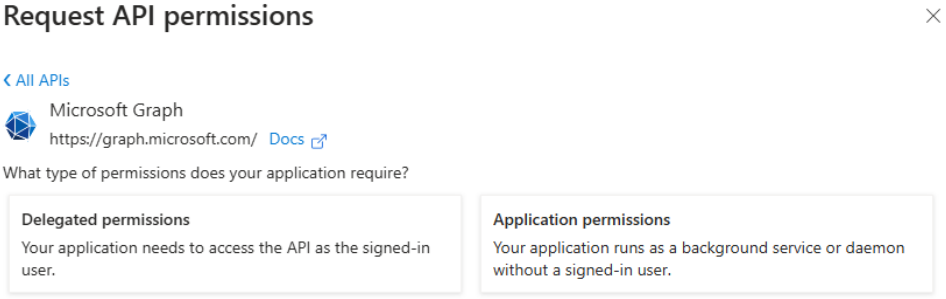
- **Key Differences**

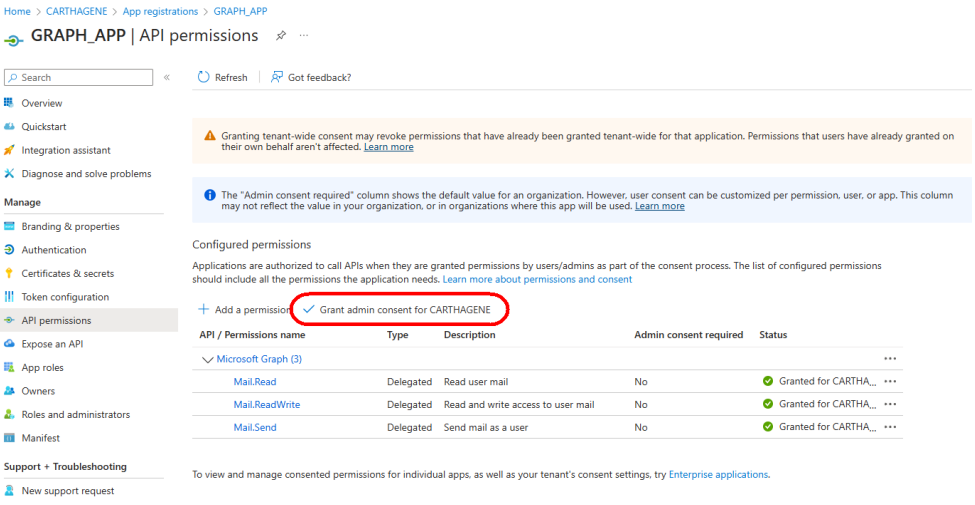
Context: Application permissions operate without user context, while delegate permissions require a user context.

Authority: Application permissions are granted by an administrator, whereas delegate permissions are granted by the user.

Use Cases: Application permissions are suited for background tasks, while delegate permissions are ideal for user-interactive scenarios or use case where a refresh token is sent to the application server (with authentication by code).


Request API permissions
Select an API
Microsoft APIs
APIs my organization uses
My APIs
Commonly used Microsoft APIs
Microsoft Graph
Take advantage of the tremendous amount of data in Office 365, Enterprise Mobility + Security, and Windows 10. Access Microsoft Entra ID, Excel, Intune, Outlook/Exchange, OneDrive, OneNote, SharePoint, Planner, and more through a single endpoint.
Azure Communication Services
Rich communication experiences with the same secure CPaaS platform used by Microsoft Teams
Azure Rights Management Services
Allow validated users to read and write protected content
Azure Service Management
Programmatic access to much of the functionality available through the Azure portal
Azure Storage
Secure, massively scalable object and data lake storage for unstructured and semi-structured data
Data Export Service for Microsoft Dynamics 365
Export data from Microsoft Dynamics CRM organization to an external destination
Dynamics 365 Business Central
Programmatic access to data and functionality in Dynamics 365 Business Central
Microsoft Purview


Request API permissions
All APIs
Microsoft Graph
https://graph.microsoft.com/ Docs
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.


Home > CARTHAGENE > App registrations > GRAPH_APP
GRAPH_APP | API permissions
Search
Refresh
Got feedback?
Overview
Quickstart
Integration assistant
Diagnose and solve problems
Manage
Branding & properties
Authentication
Certificates & secrets
Token configuration
API permissions
Expose an API
App roles
Owners
Roles and administrators
Manifest
Support + Troubleshooting
New support request
Granting tenant-wide consent may revoke permissions that have already been granted tenant-wide for that application. Permissions that users have already granted on their own behalf aren't affected. Learn more
The "Admin consent required" column shows the default value for an organization. However, user consent can be customized per permission, user, or app. This column may not reflect the value in your organization, or in organizations where this app will be used. Learn more
Configured permissions
Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. Learn more about permissions and consent
Add a permission
Grant admin consent for CARTHAGENE
API / Permissions name
Type
Description
Admin consent required
Status
Microsoft Graph (3)
Mail.Read
Delegated
Read user mail
No
Granted for CARTHA...
Mail.ReadWrite
Delegated
Read and write access to user mail
No
Granted for CARTHA...
Mail.Send
Delegated
Send mail as a user
No
Granted for CARTHA...
To view and manage consented permissions for individual apps, as well as your tenant's consent settings, try Enterprise applications.

## 3.7. Tokens

When working with authentication and authorization in M365 applications, it's important to understand the roles of **Access Tokens and Refresh Tokens**. Both tokens play crucial roles in securing access to services, but they serve different purposes and have different lifespans.

- **Access Token**

Purpose: An Access Token is used to access various services such as Microsoft Graph, POP, IMAP, SMTP.

Format: Access Tokens are in JWT (JSON Web Token) format, which is readable and contains encoded information.

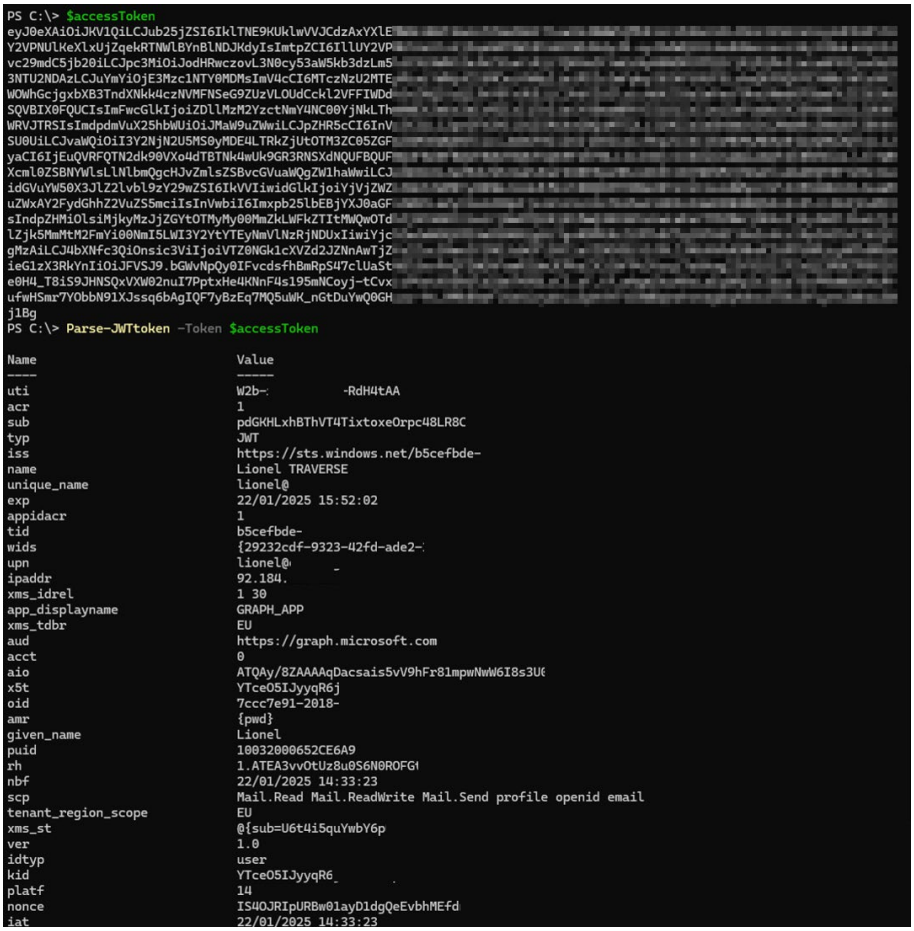
Permissions: The token includes application permissions in the scp (scope) property.

Validity: An Access Token is typically **valid for 1 hour**. But if the application and the endpoint are compatible with CAE, the Access Token **can be valid around 24 hours.**

Usage: It is presented to the service to authenticate and authorize the application's requests.

Web Services Access: For web services like Microsoft Graph or EWS (Exchange Web Services), the Access Token is included in the HTTP request's Authorization header as a Bearer token.

Example: When an application needs to read emails from a user's mailbox, it uses an Access Token to authenticate the request to the Microsoft Graph API.




- **Refresh Token**

Purpose: A Refresh Token is used to obtain new Access Tokens and Refresh Tokens.

Validity: A Refresh Token is generally valid for 3 months. Refresh token rotation is not set by default on M365.

Usage: When an Access Token expires, the application can use the Refresh Token to request a new Access Token without requiring the user to re-authenticate.

Example: After the initial Access Token expires, the application uses the Refresh Token to get a new Access Token to continue accessing the user's mailbox.


- **Key Differences**

Access Scope: Access Tokens are used directly to access services, while Refresh Tokens are used to obtain new Access Tokens.

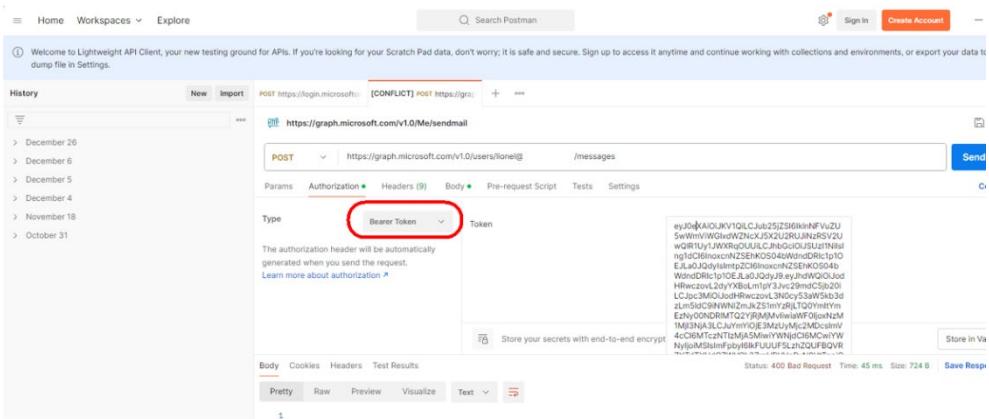
Lifespan: Access Tokens are short-lived (1 hour or 24 hours), whereas Refresh Tokens are long-lived (3 months).

Renewal: To obtain a new Access Token using a Refresh Token, the application must have the secret or certificate of the app, unless the "Allow public flows" option is enabled.

Revocation: It is possible to revoke only refresh token with the Graph PowerShell command “Revoke-MgUserSignInSession”.

- **Using access tokens**

When accessing Microsoft Web resources (like GRAPH endpoint), an Access Token is required to authenticate and authorize the application's requests. The Access Token is included in the HTTP request's Authorization header as a **Bearer token**.



# 4. Security aspects

## 4.1. Scoping

When an application is granted permissions for Exchange Online, it inherently gains access **to all mailboxes within the tenant**. This broad access can pose significant security and compliance risks if not properly managed. Therefore, implementing scoping is essential to limit the application's reach and ensure that it only accesses the necessary mailboxes.

When configuring M365 applications for Exchange Online, 3 types of scoping can be implemented to ensure secure and appropriate access: **Application Access Policies,** AppRBAC (**Application Role-Based Access Control**) rules (these scoping modes can be defined only by PowerShell) and **strict Assignment**.

- **Application Access Policies**

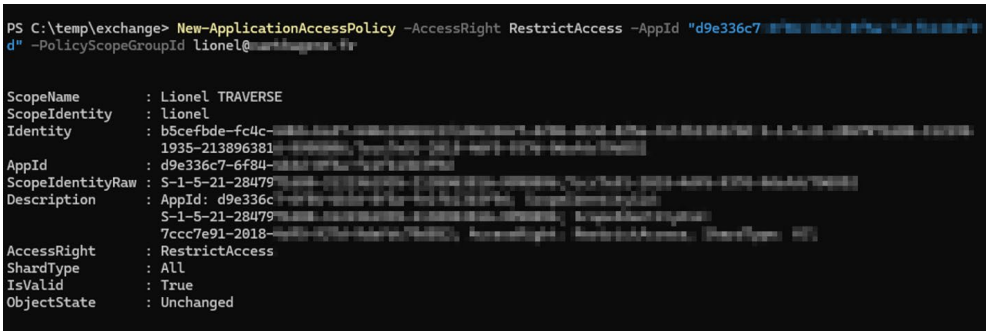
Application Access Policies are used to control which applications can access specific mailboxes within Exchange Online. These policies help ensure that only authorized applications, with application type permissions can interact with user data.

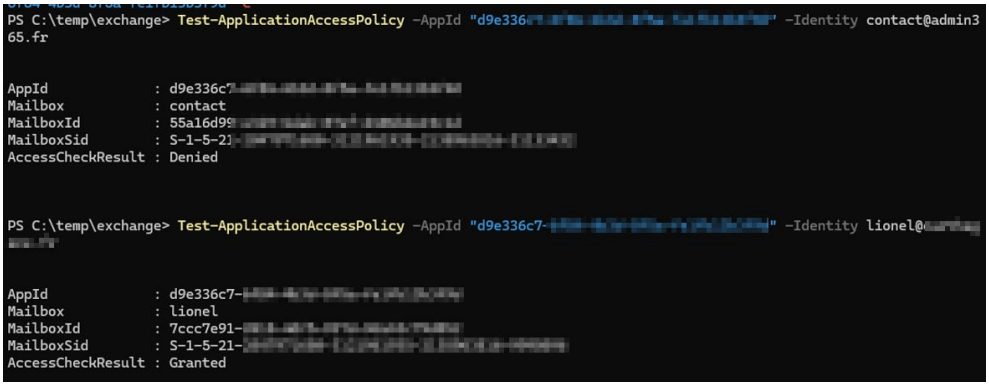
Mailbox-Specific Access: Policies can be configured to allow or deny access to specific mailboxes based on the application's identity.

Granular Control: Administrators can define precise rules to control which applications have access to which mailboxes, enhancing security and compliance.

Access Restrictions: Policies can restrict access to sensitive mailboxes, ensuring that only trusted applications can access critical data.

Example: An Application Access Policy might allow a CRM application to access sales team mailboxes but deny access to executive mailboxes.






- **AppRBAC (Application Role-Based Access Control) Rules:**

AppRBAC will replace Application Access Policies and permits to bypass the limit of 300 application access policies.

AppRBAC rules define the roles and permissions that an application has within Exchange Online. These rules ensure that applications only have the necessary permissions to perform their intended functions, minimizing security risks.

Role Assignment: Applications are assigned specific roles that determine what actions they can perform within Exchange Online.

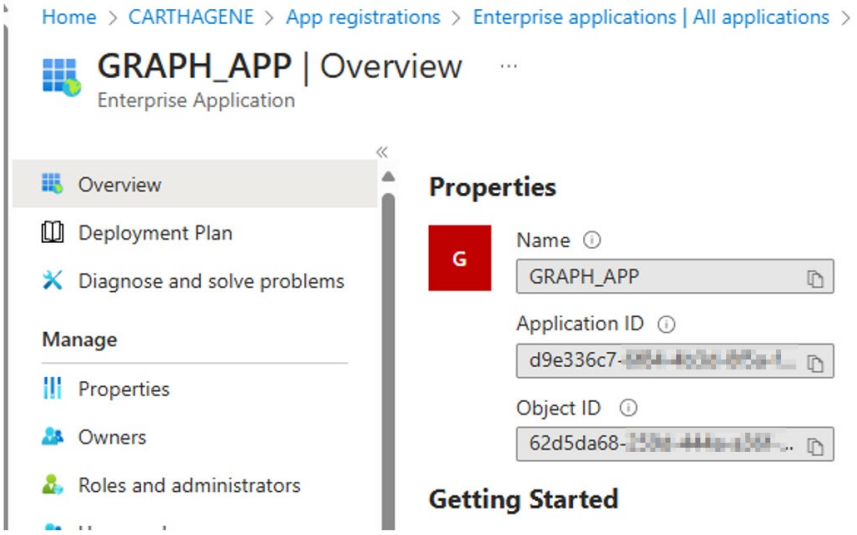
Granular Permissions: Permissions can be finely tuned to grant only the necessary access, such as read-only access to mailboxes or the ability to send emails on behalf of a user.

Administrator Control: Administrators can manage and update AppRBAC rules to adapt to changing security requirements.

Example: An application might be assigned a role that allows it to read user calendars but not modify them, ensuring that it only has the permissions needed for its functionality.

APPRBAC is based on several concepts:

Creation of a “serviceprincipal” associated with the APPID of the APP (present in Enterprise APP portal)




```
PS C:\> Get-ServicePrincipal

DisplayName                           ObjectId                              AppId
-----------                           --------                              -----
GRAPH_APP                             62d5da68-                             d9e336c7-
```

Creation of a management scope with a lot of criteria available.

```
PS C:\> New-ManagementScope -name "MAILBOX_LIONEL" -RecipientRestrictionFilter {RecipientType -eq "UserMailbox" -and Pri
marySmtpAddress -eq "                    "}

Name           ScopeRestrictionType Exclusive RecipientRoot RecipientFilter
----           -------------------- --------- ------------- ---------------
MAILBOX_LIONEL RecipientScope       False                   ((RecipientType -eq 'UserMailbox') -and (PrimarySmtpAddr...
```

Check permissions available for scoping (only application type permissions are available).

```
PS C:\> Get-ManagementRole | Sort-Object -Property name

Name                                RoleType
----                                --------
Address Lists                       AddressLists
Application Calendars.Read          ApplicationCalendarsRead
Application Calendars.ReadWrite     ApplicationCalendarsReadWrite
Application Contacts.Read           ApplicationContactsRead
Application Contacts.ReadWrite      ApplicationContactsReadWrite
Application EWS.AccessAsApp         ApplicationEWSAccessAsApp
Application Exchange Full Access    ExchangeFullAccessApp
Application Mail Full Access        MailFullAccessApp
Application Mail.Read               ApplicationMailRead
Application Mail.ReadBasic          ApplicationMailReadBasic
Application Mail.ReadWrite          ApplicationMailReadWrite
Application Mail.Send               ApplicationMailSend
Application MailboxSettings.Read    ApplicationMailboxSettingsRead
Application MailboxSettings.ReadWrite ApplicationMailboxSettingsReadWrite
Application SMTP.SendAsApp          ApplicationSMTPSendAsApp
ApplicationImpersonation            ApplicationImpersonation
ArchiveApplication                  ArchiveApplication
Audit Logs                          AuditLogs
Communication Compliance Admin      CommunicationComplianceAdmin
Communication Compliance Investigation CommunicationComplianceInvestigation
```

Application permissions are assigned one by one with Management Role Assignments to the service principal.

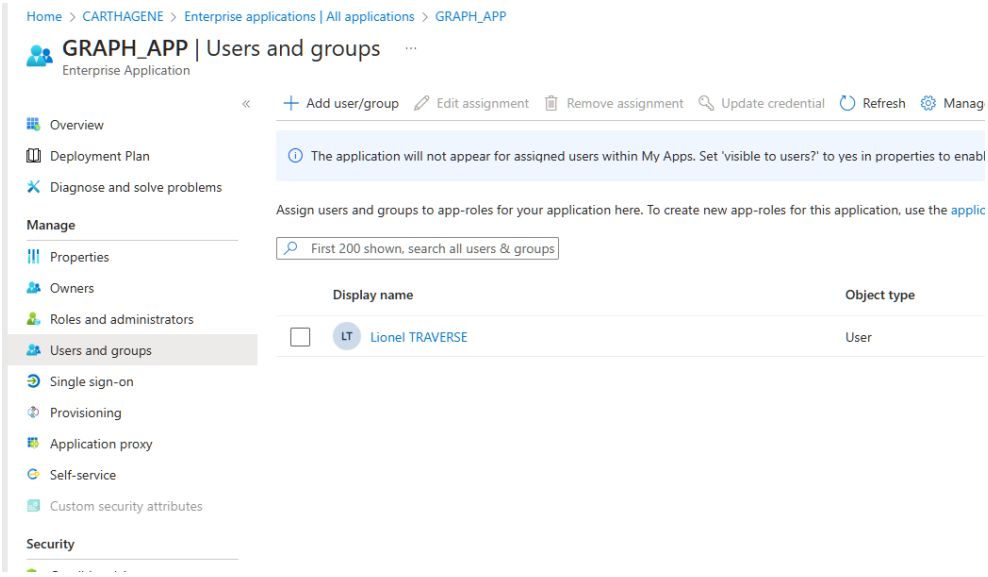
```
PS C:\> New-ManagementRoleAssignment -Name "GRAPH_APP Application.Mail.ReadWrite" -Role "Application Mail.ReadWrite" -Ap
p d9e336c7-[redacted] -CustomResourceScope "MAILBOX_LIONEL"

Name                          Role             RoleAssigneeName  RoleAssigneeType  AssignmentMethod  EffectiveUserNam
                                                                                                     e
----                          ----             ----------------  ----------------  ----------------  ----------------
GRAPH_APP Application.Mail.... Application Ma... 62d5da68-259d-... ServicePrincipal  Direct
```

Permissions can be tested with a PowerShell command.

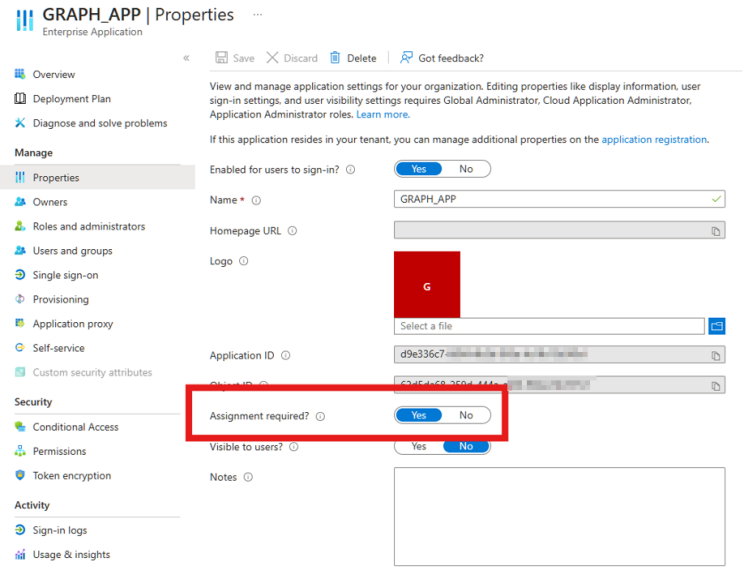
```
PS C:\> Get-ServicePrincipal

DisplayName                   ObjectId                             AppId
-----------                   --------                             -----
GRAPH_APP                     62d5da68-[redacted]                  d9e336c7-[redacted]

PS C:\> Test-ServicePrincipalAuthorization -Identity 62d5da68-[redacted] -Resource contact@admin365.fr

RoleName                  GrantedPermissions  AllowedResourceScope ScopeType              InScope
--------                  ------------------  -------------------- ---------              -------
Application Mail.ReadWrite Mail.ReadWrite      MAILBOX_LIONEL       CustomRecipientScope   False

PS C:\> Test-ServicePrincipalAuthorization -Identity 62d5da68-[redacted] -Resource traverse@[redacted]

RoleName                  GrantedPermissions  AllowedResourceScope ScopeType              InScope
--------                  ------------------  -------------------- ---------              -------
Application Mail.ReadWrite Mail.ReadWrite      MAILBOX_LIONEL       CustomRecipientScope   True
```

- ### Strict users and groups assignment

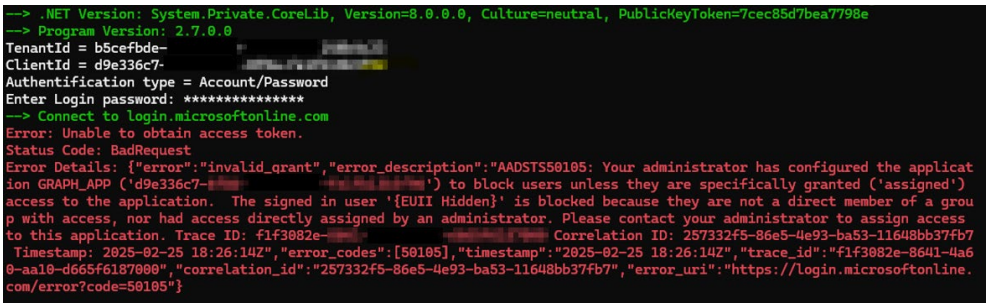
It is also possible to consider the 'scoping' of an application at another level. Indeed, an application that only consumes 'delegate' type rights can be configured to work only for the users and groups defined in the **'assignment'** part of the application viewed as an ENTERPRISE app and not as a REGISTRED app. Thus, this third type of scoping will cover the case of applications that consume delegate type rights or no rights (just SSO transfers).



The “Assignment required” option will limit the use of the application to certain users or groups.




If a user does not have the right to use the application, they will be denied when requesting a token.




- **Summary**

| Scoping / APP type | Application access policy | App RBAC | User assignment |
|---|---|---|---|
| APP for auth and SSO | | | ✓ |
| APP with delegate permissions | | | ✓ |
| APP with application permissions | ✓ | ✓ | |

## 4.2. Conditional access

Conditional Access rules for Microsoft Graph applications are specifically **applied during the token requests made by the application entity or the user entity**. This means that when an application or a user requests an access token to interact with Microsoft Graph APIs, the Conditional Access policies are enforced to ensure that the request meets the organization's security requirements.

However, it is important to note that these Conditional Access rules do not apply to token requests made using authorization codes or refresh tokens. In other words, when an application or user requests a new access token using an authorization code or refresh token, the Conditional Access policies are not evaluated. This distinction is crucial for understanding how and when Conditional Access rules are enforced. By focusing on the initial token requests from the application or user entity, organizations can ensure that the primary access points are secured, while subsequent token exchanges using authorization codes or refresh tokens are not subjected to the same Conditional Access policies.

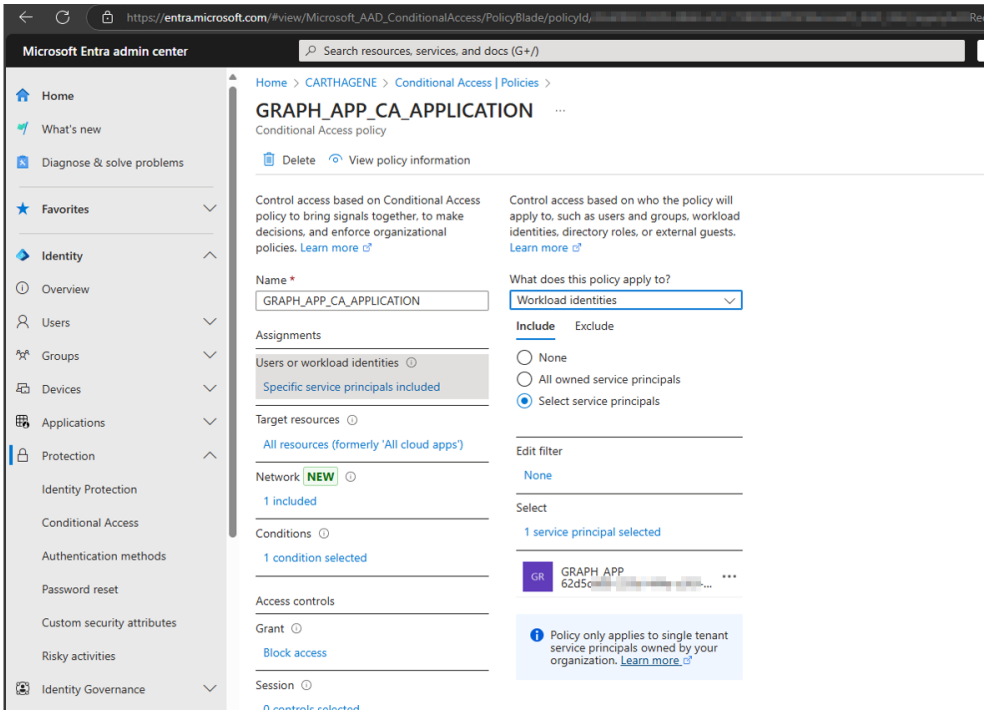
When it comes to Microsoft Graph applications, Conditional Access rules can be applied based on two different scenarios: the user consuming delegated permissions or the application itself consuming application permissions.

- **User-Based Conditional Access (Delegated Permissions)**

In this scenario, the Conditional Access policy is applied to the user who is accessing the Microsoft Graph API.

- **Application-Based Conditional Access (Application Permissions)**

In this scenario, the Conditional Access policy is applied to the application itself, which is accessing the Microsoft Graph API with application permissions.




## 4.3. Tenant restriction

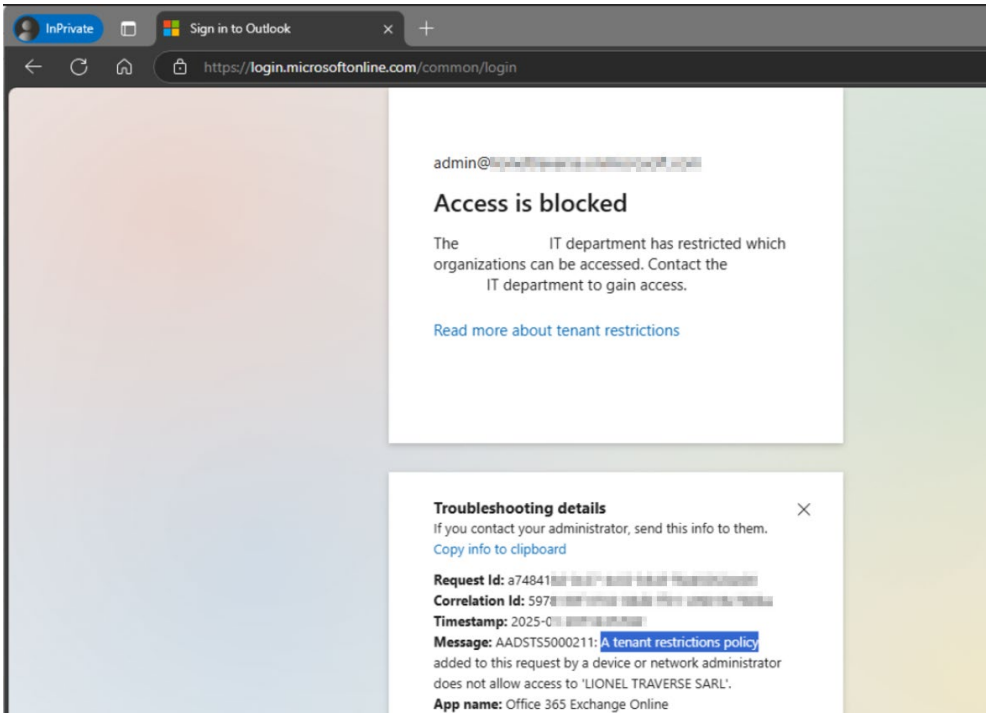
Tenant Restriction is a security feature in Microsoft 365 that allows organizations to control and restrict access to their resources based on the tenant from which the request originates. This feature is particularly useful for enterprises that use proxies to manage and secure their internet connectivity.

When an organization implements Tenant Restriction, it configures its enterprise proxies to include specific headers in the HTTP requests that are sent to Microsoft 365 services. These headers contain information about the allowed tenants, which are the tenants that the organization trusts and permits access to its resources.

Enterprise **Proxies**: These proxies act as intermediaries between the users and the internet. They ensure that all internet traffic passes through them, allowing the organization to monitor and control access to external resources. By configuring the proxies to include Tenant Restriction headers, the organization can enforce access policies based on the tenant information.

Tenant Restriction **Headers**: The header used for Tenant Restriction is called **Restrict-Access-To-Tenants**. This header is added to the HTTP requests by the enterprise proxies and specifies the allowed tenants (tenantid format). When a request reaches Microsoft 365 services, the service checks the header to determine if the request is coming from an allowed tenant.

**Access Control**: If the request originates from an allowed tenant, access to the Microsoft 365 resources is granted. If the request comes from a tenant that is not allowed, access is denied. This ensures that only users from trusted tenants can access the organization's resources, enhancing security and preventing unauthorized access.

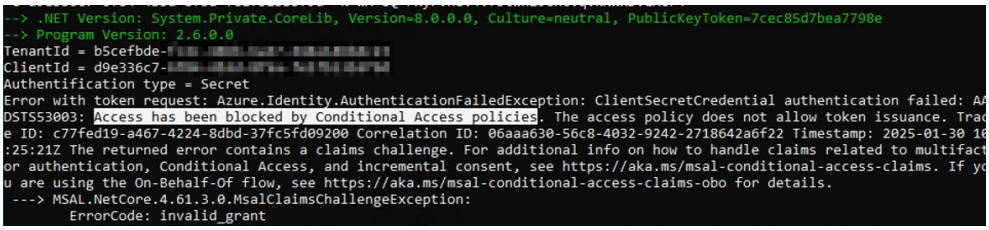



## 4.4. Workload ID

Microsoft Entra Workload ID licenses are essential for organizations that want to implement Conditional Access policies on their applications.


These licenses enable the application of security controls and access policies to ensure that only authorized users and applications can access sensitive resources.

Enabling Conditional Access: Without Microsoft Entra Workload ID licenses, it is not possible to apply Conditional Access rules to applications. These licenses are required to enforce security policies that control access based on various conditions such as user location, device compliance, and risk level.

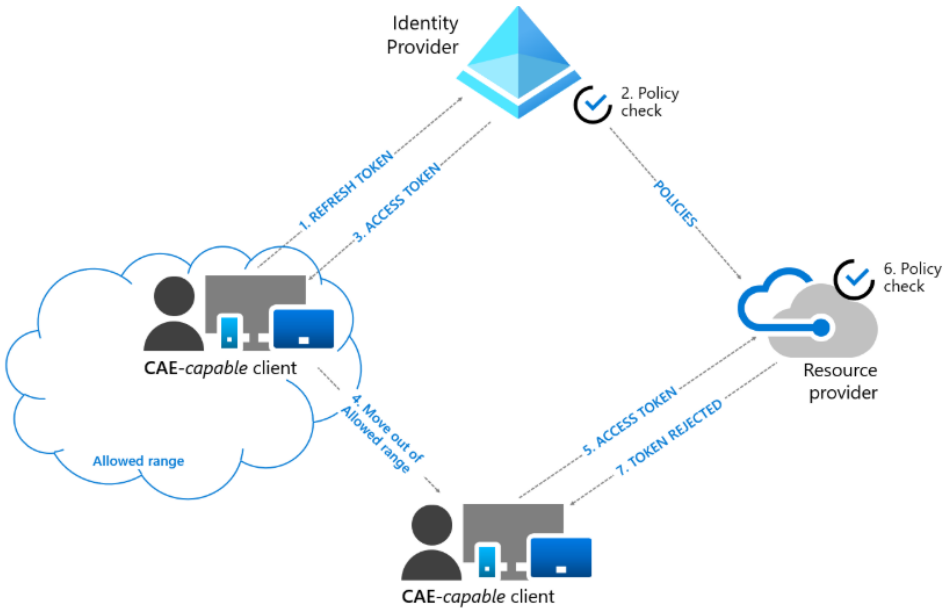



Note: Even with Microsoft 365 Workload Entities licenses, it is not possible to apply Conditional Access rules to multi-tenant applications from the source tenant. Multi-tenant applications are designed to be used by multiple organizations; therefore, Conditional Access policies can only be enforced by the target tenant, not by the source tenant.

## 4.5. Continuous access evaluation

Continuous Access Evaluation (CAE) is a **security feature** in Microsoft 365 applications that ensures real-time evaluation of access policies.

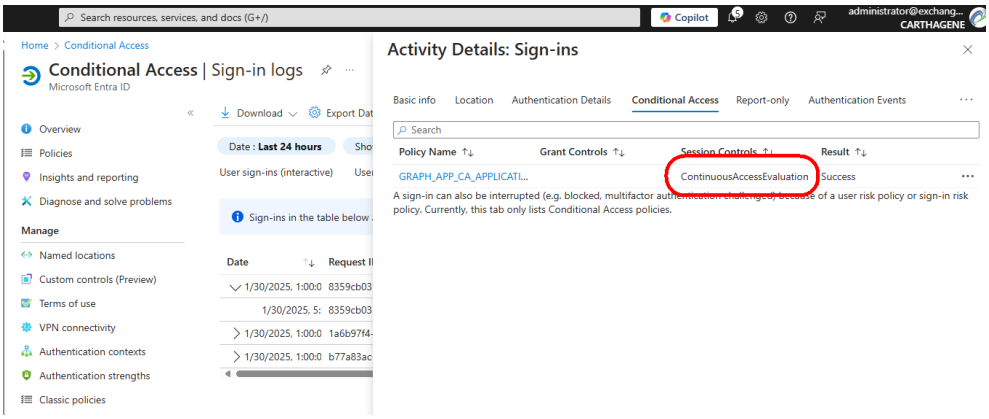
This feature helps protect against the misuse of refresh tokens outside the desired network.




When CAE is enabled, the system continuously monitors and evaluates the user's access based on predefined policies.

If any suspicious activity is detected, such as an attempt to use an ACCESS TOKEN from an unauthorized network, the system will deny access.

This ensures that only authorized users within the specified network can access the applications, providing an additional layer of security.



When a CAE ACCESS TOKEN is requested by an application, Microsoft provides an ACCESS TOKEN valid 24 hours with the claims “**xms_cc=cp1**”.


## 4.6. Compromise of tokens

Security Risks of Stolen or Improperly Secured Tokens in Microsoft 365

In Microsoft 365, tokens play a crucial role in authenticating and authorizing access to resources. However, if a refresh token or access token is stolen or improperly secured, it can pose significant security risks.

- **Refresh Tokens**

A refresh token is used to obtain a new access token without requiring the user to re-authenticate. If a refresh token associated with an application that allows public flows (i.e., without authentication) is stolen, it can be used from anywhere and will not be protected by Conditional Access rules or the security measures provided by Workload Entities licenses. This means that an attacker can use the stolen refresh token to continuously obtain new access tokens and access the organization's resources without any restrictions.

- **Access Tokens**

An access token is used to access specific resources in Microsoft 365. If an access token is stolen, it can be used from any location and will not be subject to Conditional Access policies or the security controls offered by Workload Entities licenses. This allows an attacker to gain unauthorized access to sensitive data and applications, posing a significant security threat.

- **Developer Access**

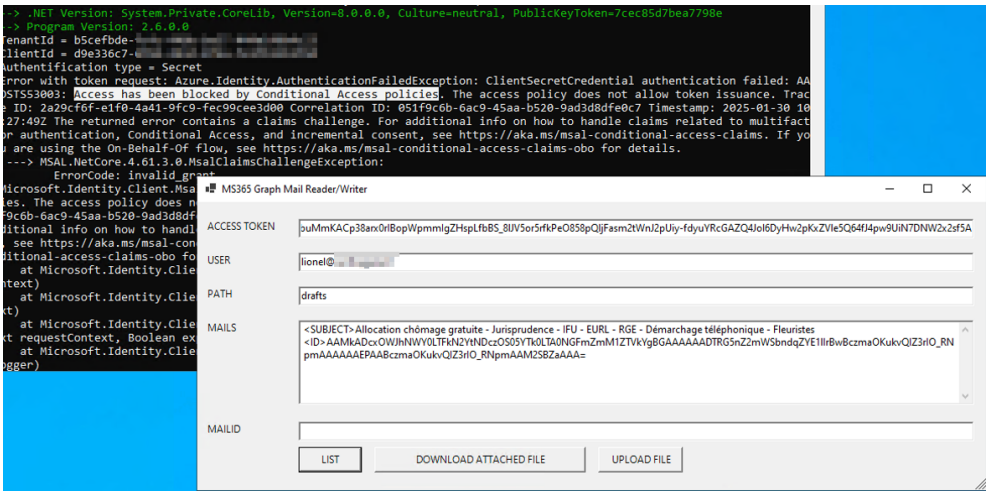
'If developers have access to these tokens, there is a significant risk of misuse or accidental exposure.The developer could potentially misuse the tokens to access resources they are not authorized to, leading to data breaches and unauthorized access to sensitive information.

- **Storage in Insecure Applications**

If refresh tokens are stored in a Software as a Service (SaaS) application with a lower security level than that provided by Microsoft 365, it further exacerbates the security risk. An attacker could exploit the weaker security measures of the SaaS application to steal the tokens and use them to access the organization's resources.

## Summary

T**he security of refresh tokens and access tokens is paramount in Microsoft 365**. If these tokens are stolen or stored in insecure environments, they can be used from anywhere and will not be protected by Conditional Access rules or Workload Entities licenses. This highlights the importance of securing tokens and ensuring they are stored in environments with robust security measures to prevent unauthorized access and protect sensitive data.

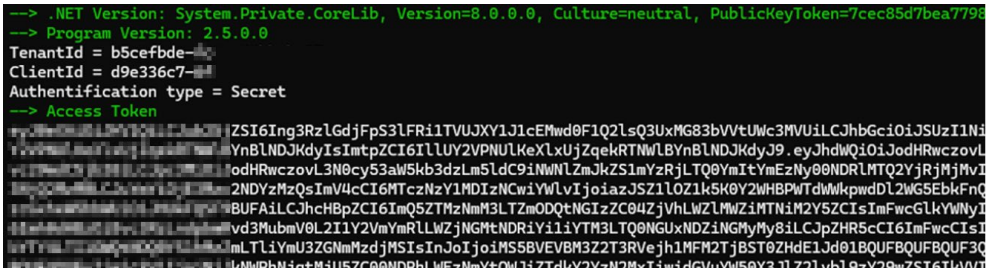


# 5. Requesting tokens for testing security

## 5.1. Requesting with client secret

Demonstration using TOKEN365.EXE (https://admin365.fr/token365/)

- TOKEN365.exe -a secret -t tenantid -c clientid -k secret

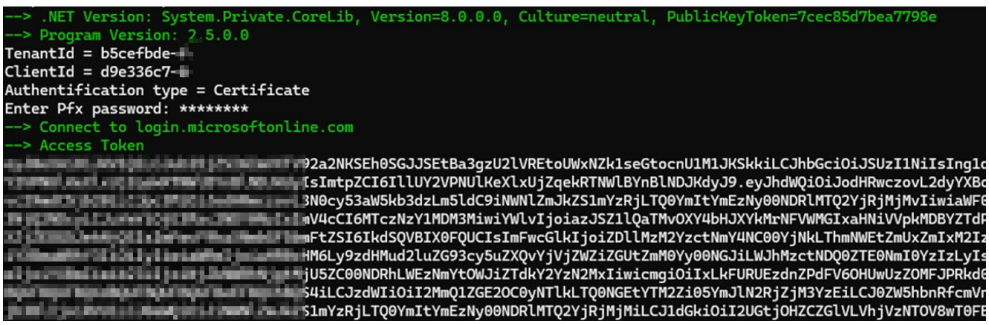



## 5.2. Requesting with certificate

Demonstration using TOKEN365.EXE (https://admin365.fr/token365/)

Connection by certificate requests to have the PRIVATE key of the certificate.

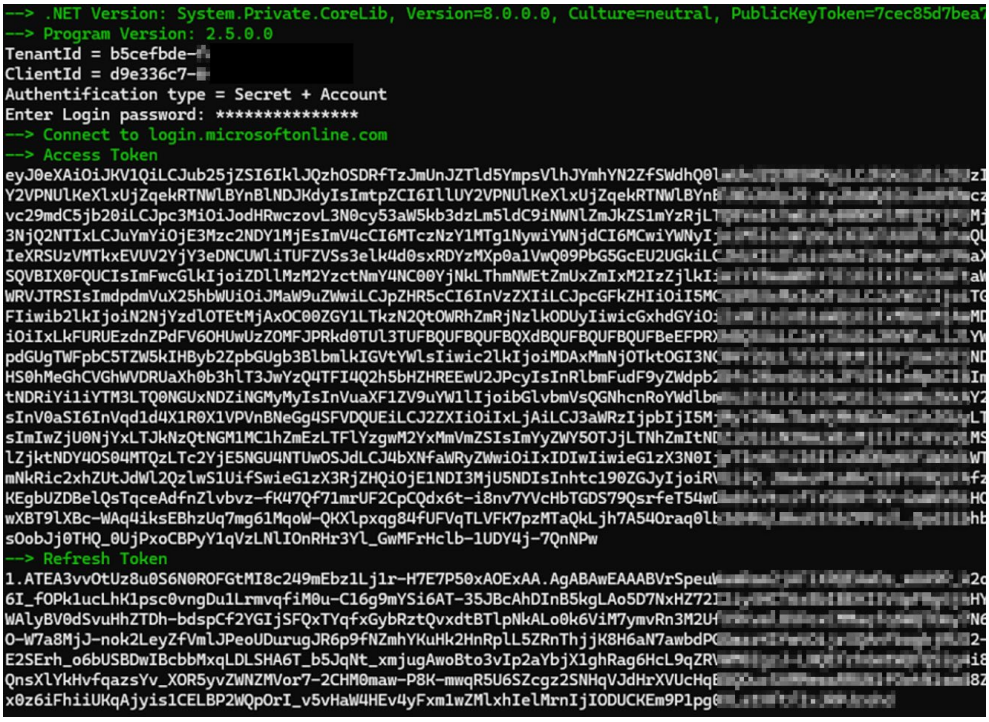
- TOKEN365.exe -a cert -t tenantid -c clientid -x pfx_file




## 5.3. Requesting with account/password

Demonstration using TOKEN365.EXE (https://admin365.fr/token365/)

- TOKEN365.exe -a account -t tenantid -c clientid -l login_name
- TOKEN365.exe -a secret-account -t tenantid -c clientid -l login_name -k secret
- TOKEN365.exe -a cert-account -t tenantid -c clientid -l login_name -x pfx_file

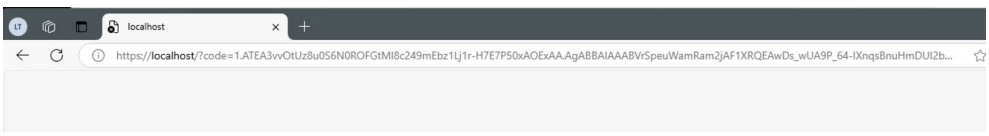



## 5.4. Requesting with code and/or user's MFA

Demonstration using TOKEN365.EXE and CODE365.EXE

CODE365 allows you to generate a code authentication request (with the code identifier). CODE365 uses EDGE navigator (standard or private mode).

- CODE365.exe -t tenantid -c clientid -r url_redirect -s "scope1 scop2…"
- CODE365.exe -t tenantid -c clientid -r url_redirect -s "scope1 scop2…" -p

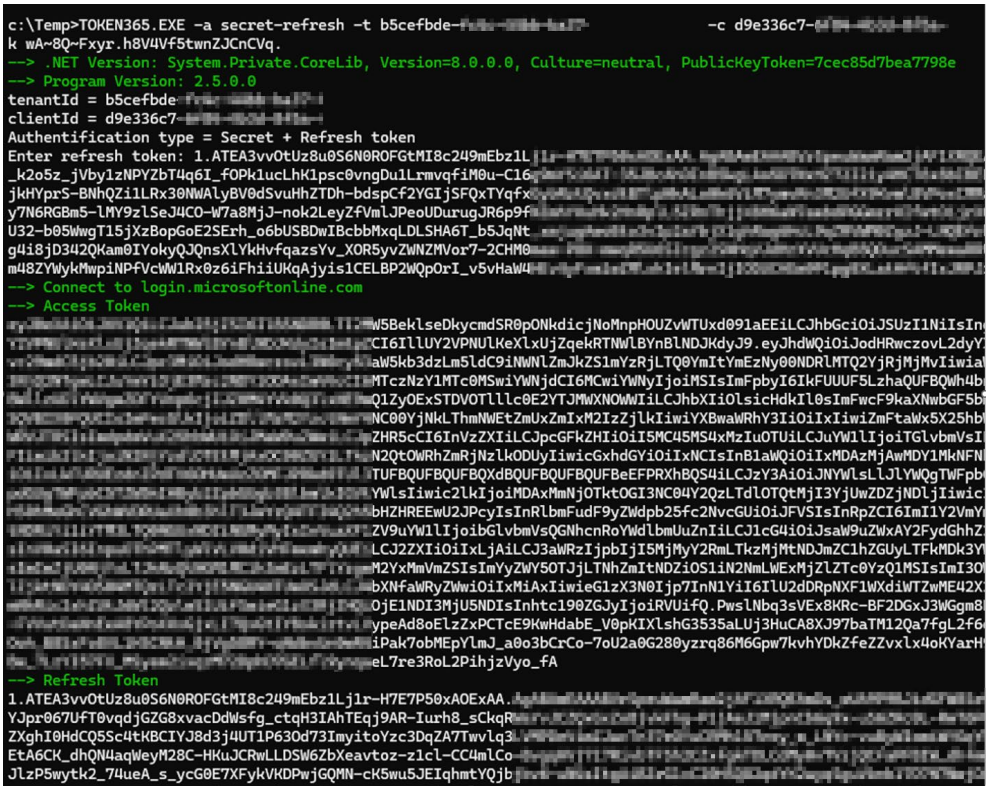

With the code (**valid 10 minutes**) and the code identifier, it is possible to obtain a couple of ACCESS TOKEN and REFRESH TOKEN.

- TOKEN365.exe -a code -t tenantid -c clientid -r redirecturl
- TOKEN365.exe -a secret-code -t tenantid -c clientid -r redirecturl -k secret
- TOKEN365.exe -a cert-code -t tenantid -c clientid -r redirecturl -x pfx_file

## 5.5. Requesting with refresh token

Demonstration using TOKEN365.EXE (https://admin365.fr/token365/)

- TOKEN365.exe -a refresh -t tenantid -c clientid
- TOKEN365.exe -a secret-refresh -t tenantid -c clientid -k secret
- TOKEN365.exe -a cert-refresh -t tenantid -c clientid -x pfx_file




## 5.6. Requesting with CAE option

To request an access token with the Continuous Access Evaluation (CAE) option in Microsoft 365, you need to include a specific claim in your token request. This claim is xms_cc=cp1, which indicates that your application can handle claim challenges.

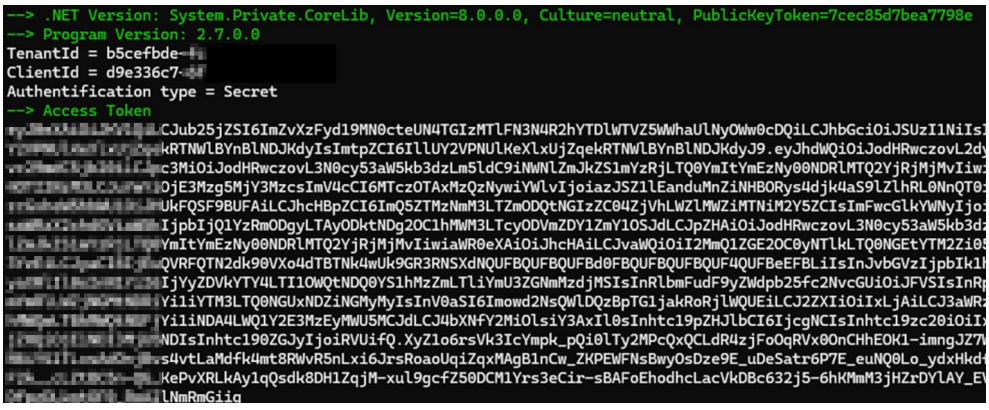
This option can be used for every type of credential used:

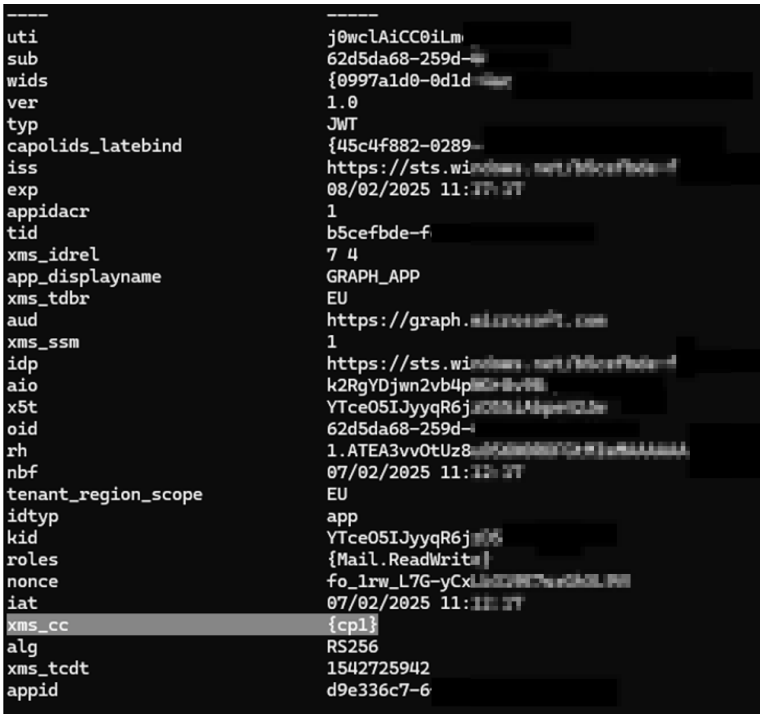
- Secret
- Certificate
- Account
- Account with Secret

- Account with Certificate
- Code
- Code with Secret
- Code with Certificate
- Refresh token
- Refresh token with Secret
- Refresh token with Certificate

TOKEN365 provides “-e” option to activate evaluation and request tokens with CAE support.

- TOKEN365.exe -a secret -t tenantid -c clientid -k secret -e
- TOKEN365.exe -a cert -t tenantid -c clientid -x pfx_file -e
- TOKEN365.exe -a account -t tenantid -c clientid -l login_name -e
- TOKEN365.exe -a secret-account -t tenantid -c clientid -l login_name -k secret -e
- TOKEN365.exe -a cert-account -t tenantid -c clientid -l login_name -x pfx_file -e
- TOKEN365.exe -a code -t tenantid -c clientid -r redirecturl -e
- TOKEN365.exe -a secret-code -t tenantid -c clientid -r redirecturl -k secret -e
- TOKEN365.exe -a cert-code -t tenantid -c clientid -r redirecturl -x pfx_file -e
- TOKEN365.exe -a refresh -t tenantid -c clientid -e
- TOKEN365.exe -a secret-refresh -t tenantid -c clientid -k secret -e
- TOKEN365.exe -a cert-refresh -t tenantid -c clientid -x pfx_file -e





| ---- | ----- |
| uti | j0wclAiCC0iLm |
| sub | 62d5da68-259d- |
| wids | {0997a1d0-0d1d |
| ver | 1.0 |
| typ | JWT |
| capolids_latebind | {45c4f882-0289- |
| iss | https://sts.wi |
| exp | 08/02/2025 11: |
| appidacr | 1 |
| tid | b5cefbde-f |
| xms_idrel | 7 4 |
| app_displayname | GRAPH_APP |
| xms_tdbr | EU |
| aud | https://graph. |
| xms_ssm | 1 |
| idp | https://sts.wi |
| aio | k2RgYDjwn2vb4p |
| x5t | YTceO5IJyyqR6j |
| oid | 62d5da68-259d- |
| rh | 1.ATEA3vvOtUz8 |
| nbf | 07/02/2025 11: |
| tenant_region_scope | EU |
| idtyp | app |
| kid | YTceO5IJyyqR6j |
| roles | {Mail.ReadWrit |
| nonce | fo_1rw_L7G-yCx |
| iat | 07/02/2025 11: |
| xms_cc | {cp1} |
| alg | RS256 |
| xms_tcdt | 1542725942 |
| appid | d9e336c7-6 |