25. – 27.
SEPTEMBER
2023
PORTOROŽ

NT KONF
NT KONFERENCA

Managing Entra ID with Microsoft Graph PowerShell

Aleksandar Nikolić | Microsoft MVP

Day 1

Agenda

1. Introduction to Entra ID and Microsoft Graph PowerShell
2. Authentication and Authorization
3. Query Parameters and Advanced Queries
4. Working with Entra ID Users and Groups
5. Using Microsoft Graph API with PowerShell
6. Q & A

MODULE 01

Introduction to Entra ID and Microsoft Graph PowerShell

Microsoft Graph is a unified endpoint for accessing data, intelligence, and insights in Microsoft 365.

Why Microsoft Graph PowerShell Module?

Microsoft Graph PowerShell module acts as an API wrapper for Microsoft Graph APIs, exposing the entire API set for use in PowerShell.

It contains a set of commands that help you manage identities at scale; from automating tasks to managing users in bulk using Entra ID (formerly known as Azure Active Directory (Azure AD)).

It will help administer every Entra ID feature that has an API in Microsoft Graph.

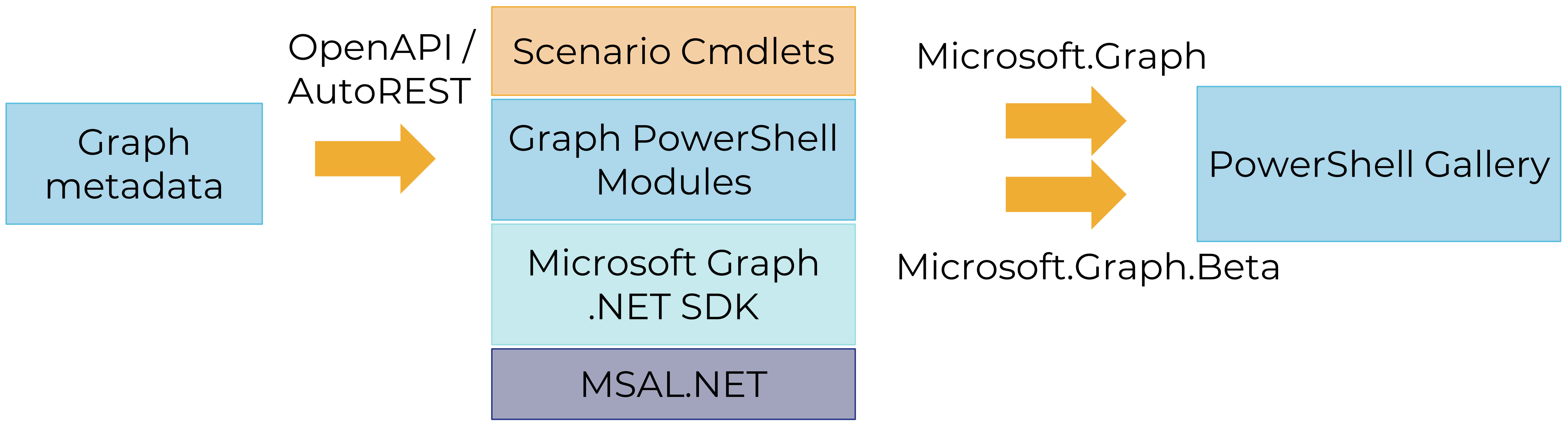
Microsoft Graph PowerShell

The commands in Microsoft Graph PowerShell are **autogenerated** from the Microsoft Graph API schema making it easier to get faster updates and functionality.

The cmdlet reference content is **also autogenerated** from the API reference.

Microsoft Graph PowerShell is the replacement for the Azure AD PowerShell and MSOnline modules and is recommended for interacting with Entra ID.

Microsoft Graph PowerShell



Microsoft Graph PowerShell features & benefits

Access to all Microsoft Graph APIs: Based on Microsoft Graph API. In addition to Entra ID, the Microsoft Graph API includes APIs from other Microsoft services

Supports PowerShell 7: Works with PowerShell 7 and later. It's also compatible with Windows PowerShell 5.1.

Cross-platform support: Works on all platforms including Windows, macOS, and Linux.

Supports modern authentication: Supports the Microsoft Authentication Library (MSAL) which offers more security. For example, you can use passwordless sign-in experiences.

Microsoft Graph PowerShell features & benefits

Supports external identities: Users from other Entra ID tenants can authenticate to services in your tenant with Microsoft Graph PowerShell.

Uses least privilege: Microsoft Graph PowerShell permissions are not pre-authorized and users must perform one-time request for app permissions depending on their needs.

Advanced queries: Microsoft Graph PowerShell supports rich, advanced queries via *eventual consistency*. For example, you can get a near-instant count of all users using advanced queries.

Open source: Feature teams and the community can create great PowerShell experiences and share them with everyone.

Receives regular updates: Microsoft Graph PowerShell commands are updated regularly to support the latest Graph API updates.

Supported PowerShell versions

PowerShell 7 and later is the recommended PowerShell version for use with the Microsoft Graph PowerShell SDK on all platforms.

There are no additional prerequisites to use the SDK with PowerShell 7 or later.

The following prerequisites must be met to use the Microsoft Graph PowerShell SDK with Windows PowerShell.

- Upgrade to PowerShell 5.1 or later
- Install .NET Framework 4.7.2 or later
- Update PowerShellGet to the latest version using Install-Module PowerShellGet -Force

Installation

GitHub: https://github.com/microsoftgraph/msgraph-sdk-powershell

# 1.28.0 and 2.6.1
https://www.powershellgallery.com/packages/Microsoft.Graph

# 2.6.1
https://www.powershellgallery.com/packages/Microsoft.Graph.Beta

```
Install-Module Microsoft.Graph -Scope CurrentUser
Install-Module Microsoft.Graph.Beta -Scope CurrentUser
```

Microsoft Graph PowerShell v2: Breaking changes

The new version drops profile support by having separate v1.0 and beta modules. (~~Select-MgProfile~~)

Microsoft.Graph.* and Microsoft.Graph.Beta.* modules

v1.0 API: Get-MgUser / Beta API: Get-MgBetaUser

In v2, Connect-MgGraph's -AccessToken changed from String to SecureString type.

Dropped support for -ForceRefresh on Connect-MgGraph command.

Microsoft Graph PowerShell v2: New features

Authentication

- Managed Identity
- Client Secret Credentials
- Environment Variable Based Auth

v2 adds HTTP/2 support for supported API endpoints. HTTP/2 improves performance by adding support for multiplexing, header compression, and server push.

v2 commands will correctly handle and return the right output when an HTTP request responds back with a successful code (2xx).

Installation

DEMO

Finding Microsoft Graph PowerShell commands

The `Find-MgGraphCommand` allows to:

Pass a Microsoft Graph URL (relative and absolute) and get an equivalent Microsoft Graph PowerShell command.

Pass a command and get the URL it calls.

Pass a command or URI wildcard (.*) to find all commands that match it.

The output of this cmdlet also includes **the permissions** required to authenticate the specified cmdlet.

Least Permission Model

Permission handling differs significantly between the Azure AD PowerShell module and the Microsoft Graph PowerShell SDK.

Unlike permissions inherited from signed-in account, the permissions used by the SDK are granted to the service principal used to run SDK cmdlets.

For interactive sessions, the service principal is the Microsoft Graph

You must request permissions to perform actions, even when connecting with a highly-permissioned account

How to Figure Out What Microsoft Graph Permissions You Need

Guess and hope it will work ☺

- The danger here is that you end up with a heavily-permissioned service principal

Use the Graph Explorer (https://aka.ms/ge)

- The permissions required to run the query in the *Modify permissions* tab

Read the Microsoft Graph documentation

- Microsoft Graph REST API references (v1.0 and beta)
- Microsoft Graph permissions reference

Use `Find-MgGraphPermission` and `Find-MgGraphCommand` to identify Graph permissions

Finding permissions

`Find-MgGraphPermission` helps to answer the following questions:

How do I find the values to supply to the permission-related parameters of commands like `New-MgApplication` and other application and consent related commands?

What permissions are applicable to a certain domain, for example, application, directory?

DEMO

Finding commands and permissions

MODULE 02

Authentication and Authorization

2

Authentication

The PowerShell SDK supports two types of authentication: delegated access and app-only access.

Delegated access: Log in as a user, grant consent to the SDK to act on our behalf, and call the Microsoft Graph.

App-only access: Grants permissions directly to an application, and requires an administrator to consent to the required permission scopes.

Delegated Access

Delegated access uses a public client to get an access token and consume Microsoft Graph resources on behalf of the signed-in user.

Microsoft Graph PowerShell module supports the following delegated access scenarios:

- **Interactive Browser**

```
Connect-MgGraph -Scopes "User.ReadBasic.All", "Calendars.Read.Shared“
```

- **Device Code**

```
Connect-MgGraph -Scopes "User.ReadBasic.All", "Calendars.Read.Shared" -UseDeviceCode
```

App-only Access

App-only access uses a confidential client to get an access token and consume Microsoft Graph resources without a user context (uses an app's context).

Microsoft Graph PowerShell module supports the following app-only access scenarios:

- Client credential via a certificate
- Client credential via client secret (v2)
  - Using PSCredential object
  - Using environment variables
- Managed Identity (v2)

Bring Your Own Token

Customers can acquire an access token using their preferred auth library and pass the access token using -AccessToken parameter on Connect-MgGraph.

The module will then use the provided access token to consume Microsoft Graph resources.

The following considerations should be made before using -AccessToken:

- Access Token Expiry
- Access Token Scopes (scp) Claims

Configuration of app-only access with a certificate

Before you can use app-only access with the SDK, you need the following:

A certificate to use as a credential for the application.

This can be a self-signed certificate or a certificate from an authority.

Register an application in Entra ID, configure it with the permission scopes your scenario requires, and share the public key for your certificate.

DEMO

App-only access with a certificate

MODULE 03

Query Parameters and Advanced Queries

3

Use query parameters to customize PowerShell query outputs

Microsoft Graph PowerShell supports optional query parameters that you can use to control the amount of data returned in an output.

Support for the exact query parameters varies from one cmdlet to another, and depending on the API, can differ between the v1.0 and beta endpoints.

OData system query options

Microsoft PowerShell cmdlets may support one or more of the following OData system query options, which are compatible with OData v4.0 query language and are only supported in the **GET operations**:

-Count

-Expand

-Filter

-OrderBy

-Search

-Select

-Top

OData query options in the Microsoft Graph API use lowercase names and specify the dollar ($) prefix.

In Microsoft Graph PowerShell, their names are Pascal-cased and prefixed with a hyphen (-).

For example, $count and $orderBy are to Microsoft Graph API while -Count and -OrderBy, respectively, are to Microsoft Graph PowerShell.

Query parameters

Name	Description	Example
-Count	Retrieves the total count of matching resources	`Get-MgUser -ConsistencyLevel eventual -Count count` `$count`
-Expand	Retrieves related resources	`Get-MgGroup -GroupId '0e06b38f-931a-47db-9a9a-60ab5f492005' -Expand members | Select -ExpandProperty members`
-Filter	Filters results (rows)	`Get-MgUser -Filter "startsWith(DisplayName, 'Conf')"`
-OrderBy	Orders results	`Get-MgUser -OrderBy DisplayName`
-Search	Returns results based on search criteria	`Get-MgUser -ConsistencyLevel eventual -Search '"DisplayName:Conf"'`
-Select	Filters properties (columns)	`Get-MgUser -Select DisplayName, Id` (`-Property` is a parameter alias)
-Top	Sets the page size of results.	`Get-MgUser -Top 10`

Count parameter

Use the `-Count` query parameter to store the count of the total number of items in a collection in the specified count variable.

On resources that derive from DirectoryObjects (Administrative Unit, Applications, OrgContact, Devices, Groups, Service Principal, and Users), `-Count` is only supported in advanced queries.

```
PS> Get-MgUser -ConsistencyLevel eventual -Count userCount
PS> $userCount
```

Expand parameter

Many Microsoft Graph resources expose both declared properties of the resource and their relationships with other resources.

These relationships are also called **reference properties** or **navigation properties**.

- For example, the mail folders, manager, and direct reports of a user are all exposed as relationships.

Use the `–Expand` parameter to include the expanded resource or collection referenced by a single relationship (navigation property) in your results.

Filter parameter

Use the `-Filter` query parameter to retrieve a subset of a collection.

The `-Filter` query parameter can also be used to retrieve relationships like *Members*, *MemberOf*, *TransitiveMember*, and *TransitiveMemberOf*.

```
Get-MgUser -Filter "startsWith(DisplayName, 'J')"
```

Support for `-Filter` operators varies across Microsoft Graph PowerShell cmdlets.

OrderBy parameter

Use the `-OrderBy` query parameter to specify the sort order of the items returned by a cmdlet.

```
PS> Get-MgUser -OrderBy DisplayName
```

Search parameter

Use the `-Search` query parameter to restrict the results of a request to match a search criterion.

```
Get-MgUser -Search '"DisplayName:De"' -ConsistencyLevel
eventual -Count UserCount
```

Pay attention to quotes.

Support for `-Search` varies by module and some properties support `-Search` only in advanced queries.

Select parameter

Use the `–Select` query parameter to return a set of properties that are different from *the default set* for an individual resource or a collection of resources.

Use `–Select` to limit the properties returned by a query to those properties needed by your script. This is especially true of queries that might potentially return a large result set.

Limiting the properties returned in each row will reduce network load and help improve your script's performance.

Select parameter

Some Entra ID resources that derive from *DirectoryObject*, like *User* and *Group*, return a limited, default subset of properties on reads.

For these resources, you must use **`-Select`** to return properties outside of the default set.

`-Property` is an alias parameter.

```
PS> Get-MgUser -Select Id, DisplayName
```

Only properties passed to –Select will be returned.
Selected properties won’t be appended to the default properties.

Top parameter

Use the **`-Top`** query parameter to specify the page size of the result set.

The minimum value of **`-Top`** is 1 and the maximum depends on the corresponding API.

```
PS> Get-MgUser -Top 5
```

Attention!

Query parameters specified in a request might **fail silently**.

This can be true for unsupported query parameters and for unsupported combinations of query parameters.

In these cases, you should examine the data returned by the request to determine whether the query parameters you specified had the desired effect.

Advanced query capabilities on Entra ID objects

Microsoft Graph supports advanced query capabilities on various Entra ID objects, also called directory objects, and their properties.

The Microsoft Graph query engine uses an index store to fulfill query requests.

To add support for additional query capabilities on some properties, these properties are now *indexed in a separate store*.

Query request needs to be redirected to the index store.

Advanced query parameters

Advanced query capabilities are not available by default but, the requestor must also set the **ConsistencyLevel** header to eventual and, with the exception of **$search**, use the **$count** query parameter.

The **ConsistencyLevel** header and **$count** are referred to as advanced query parameters.

```
-CountVariable countVar -ConsistencyLevel eventual
```

Using -Filter and -OrderBy together is supported only with advanced queries.

-Expand is not currently supported with advanced queries.

The advanced query capabilities are currently not available for Entra ID B2C tenants.

https://learn.microsoft.com/en-us/graph/aad-advanced-queries?tabs=powershell

Query Parameters and Advanced Queries

DEMO

MODULE 04

Working with Entra ID Users and Groups

4

Working with users in Microsoft Graph

You can use Microsoft Graph to build compelling app experiences based on users, their relationships with other users and groups, and the resources they access for example their mails, calendars, files, administrative roles, group memberships.

You can access users through Microsoft Graph in two ways:

- By their ID, /users/{id}
- By using the /me alias for the signed-in user, which is the same as /users/{signed-in user's id} (PowerShell doesn't support directly the /me endpoint.)

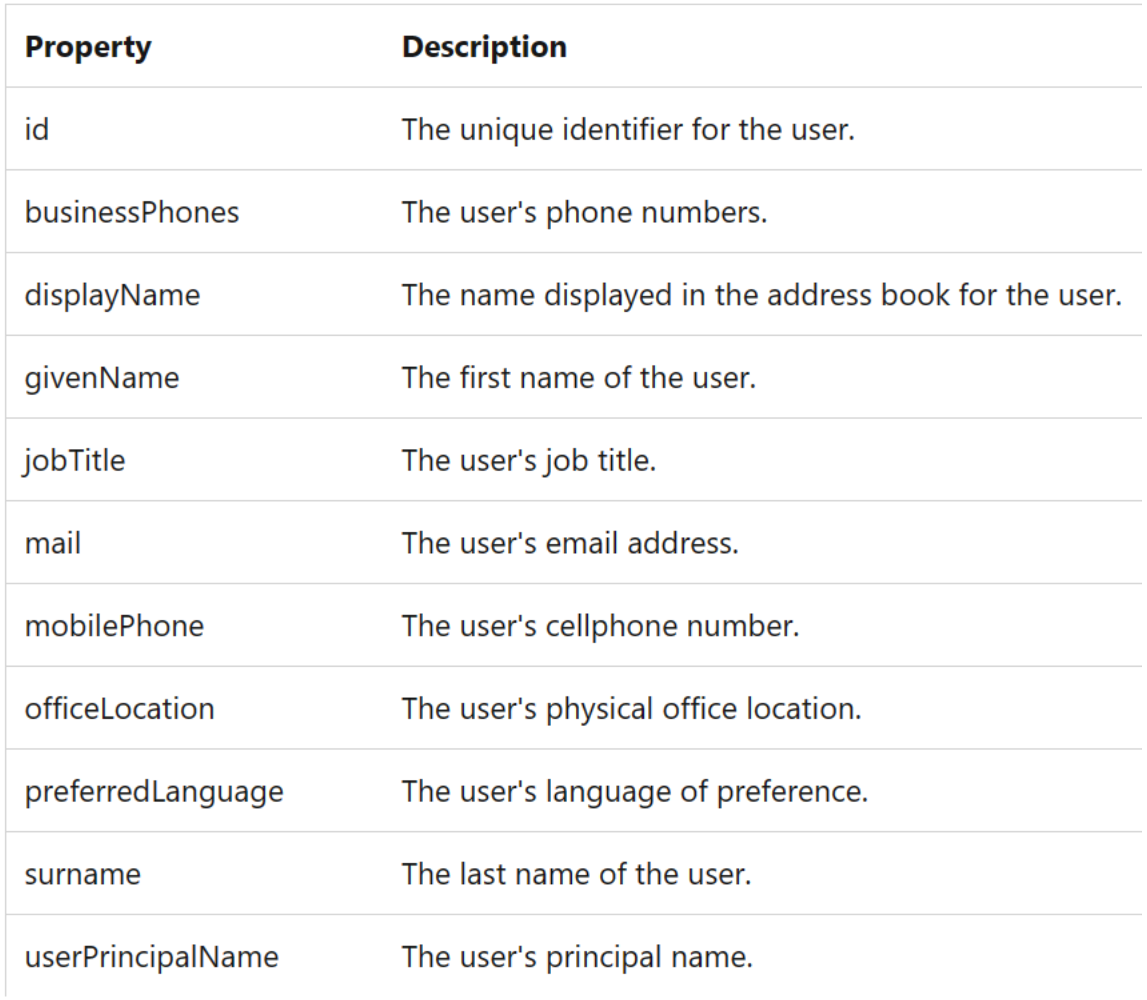
There are two types of users in Entra ID - members and guest users.

Authorization and privileges

One of the following permissions is required to access user operations.
The first three permissions can be granted to an app by a user.
The rest can only be granted to an app by the administrator.

- User.ReadBasic.All
- User.Read
- User.ReadWrite
- User.Read.All
- User.ReadWrite.All
- Directory.Read.All
- Directory.ReadWrite.All
- Directory.AccessAsUser.All

Default set of properties

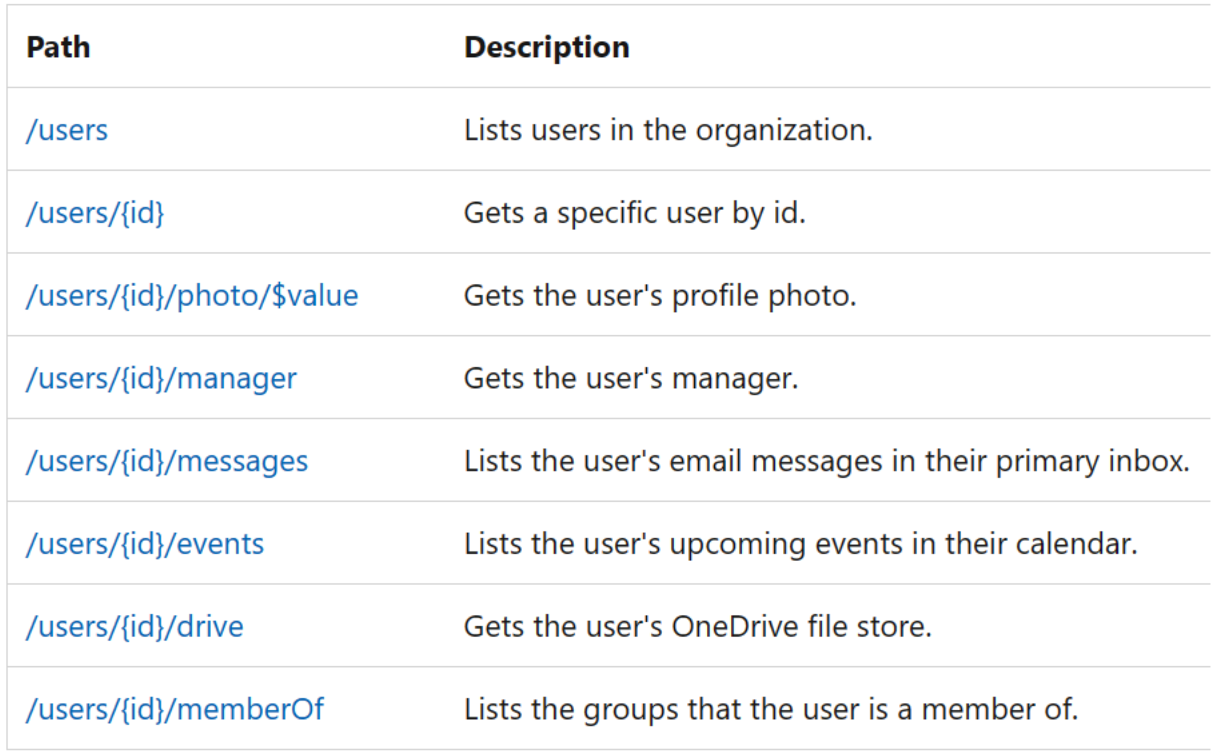


Property	Description
id	The unique identifier for the user.
businessPhones	The user's phone numbers.
displayName	The name displayed in the address book for the user.
givenName	The first name of the user.
jobTitle	The user's job title.
mail	The user's email address.
mobilePhone	The user's cellphone number.
officeLocation	The user's physical office location.
preferredLanguage	The user's language of preference.
surname	The last name of the user.
userPrincipalName	The user's principal name.

These are a subset of all available properties.

To get more user properties, use the $select query parameter.

Common operations



Path	Description
/users	Lists users in the organization.
/users/{id}	Gets a specific user by id.
/users/{id}/photo/$value	Gets the user's profile photo.
/users/{id}/manager	Gets the user's manager.
/users/{id}/messages	Lists the user's email messages in their primary inbox.
/users/{id}/events	Lists the user's upcoming events in their calendar.
/users/{id}/drive	Gets the user's OneDrive file store.
/users/{id}/memberOf	Lists the groups that the user is a member of.

DEMO

Working with users

Working with groups in Microsoft Graph

Groups are collections of principals with shared access to resources in Microsoft services or in your app.

Different principals such as users, other groups, devices, and applications can be part of groups.

Using groups helps you avoid working with individual principals and simplifies management of access to your resources.

“

All group-related operations in Microsoft Graph require administrator consent.

Group types in Entra ID and Microsoft Graph

Entra ID supports the following types of groups.

Microsoft 365 groups

Security groups

Mail-enabled security groups

Distribution groups

“

Microsoft also supports dynamic distribution groups which cannot be managed or retrieved through Microsoft Graph.

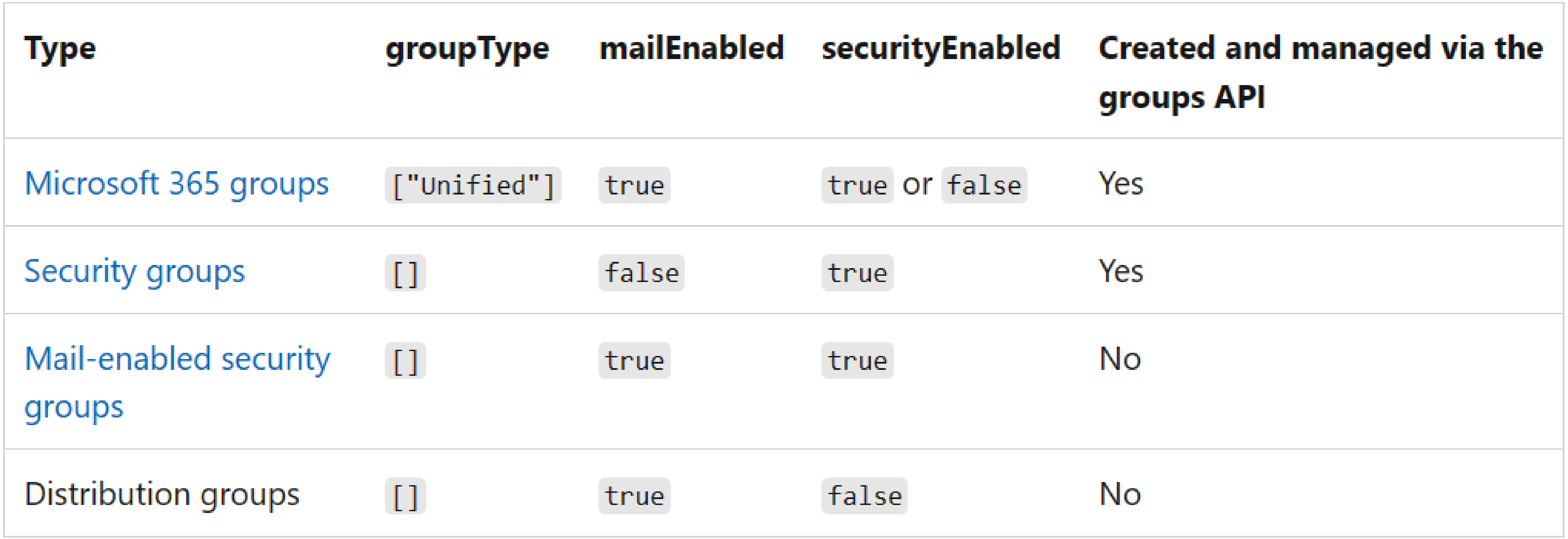
Group types in Entra ID and Microsoft Graph

Only Microsoft 365 and security groups can be managed through the Microsoft Graph groups API.

Mail-enabled and distribution groups are read-only through Microsoft Graph.

In Microsoft Graph, the type of group can be identified by the settings of its **groupType**, **mailEnabled**, and **securityEnabled** properties.

Group types in Entra ID and Microsoft Graph



Type	groupType	mailEnabled	securityEnabled	Created and managed via the groups API
Microsoft 365 groups	`["Unified"]`	`true`	`true` or `false`	Yes
Security groups	`[]`	`false`	`true`	Yes
Mail-enabled security groups	`[]`	`true`	`true`	No
Distribution groups	`[]`	`true`	`false`	No

Microsoft 365 groups

The power of Microsoft 365 groups is in its collaborative nature, perfect for people who work together on a project or a team.

They're created with resources that members of the group share, including:

Outlook conversations

Outlook calendar

SharePoint files

OneNote notebook

SharePoint team site

Planner plans

Intune device management

Security groups

Security groups are for controlling user access to resources.

By checking whether a user is a member of a security group, your app can make authorization decisions when that user is trying to access some secure resources in your app.

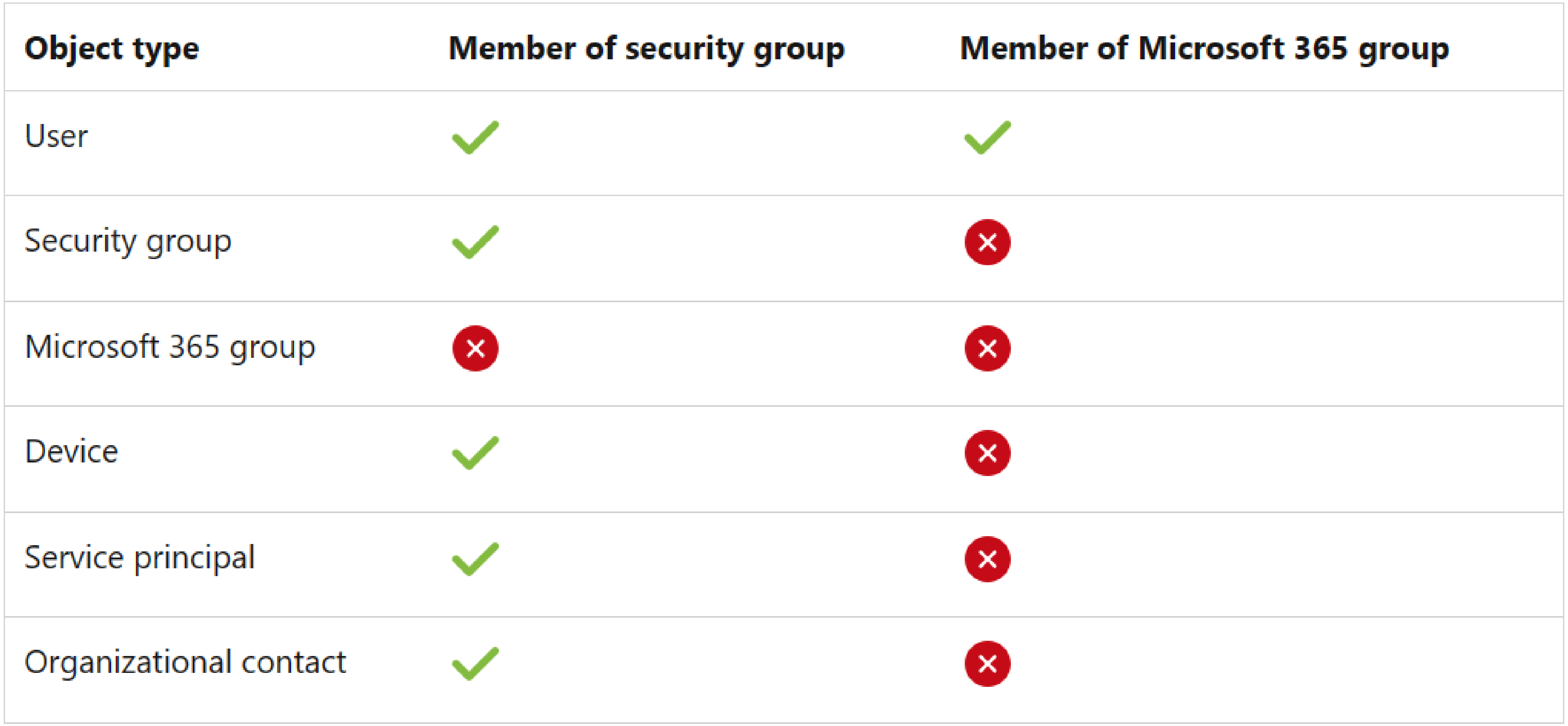
Security groups can have users, other security groups, devices, and service principals as members.

Mail-enabled security groups

Mail-enabled security groups are used in the same way as security groups, but can be used to send emails to group members.

Mail-enabled security groups can't be created or updated through the API; instead, they're read-only.

Group membership



Object type	Member of security group	Member of Microsoft 365 group
User	✓	✓
Security group	✓	✗
Microsoft 365 group	✗	✗
Device	✓	✗
Service principal	✓	✗
Organizational contact	✓	✗

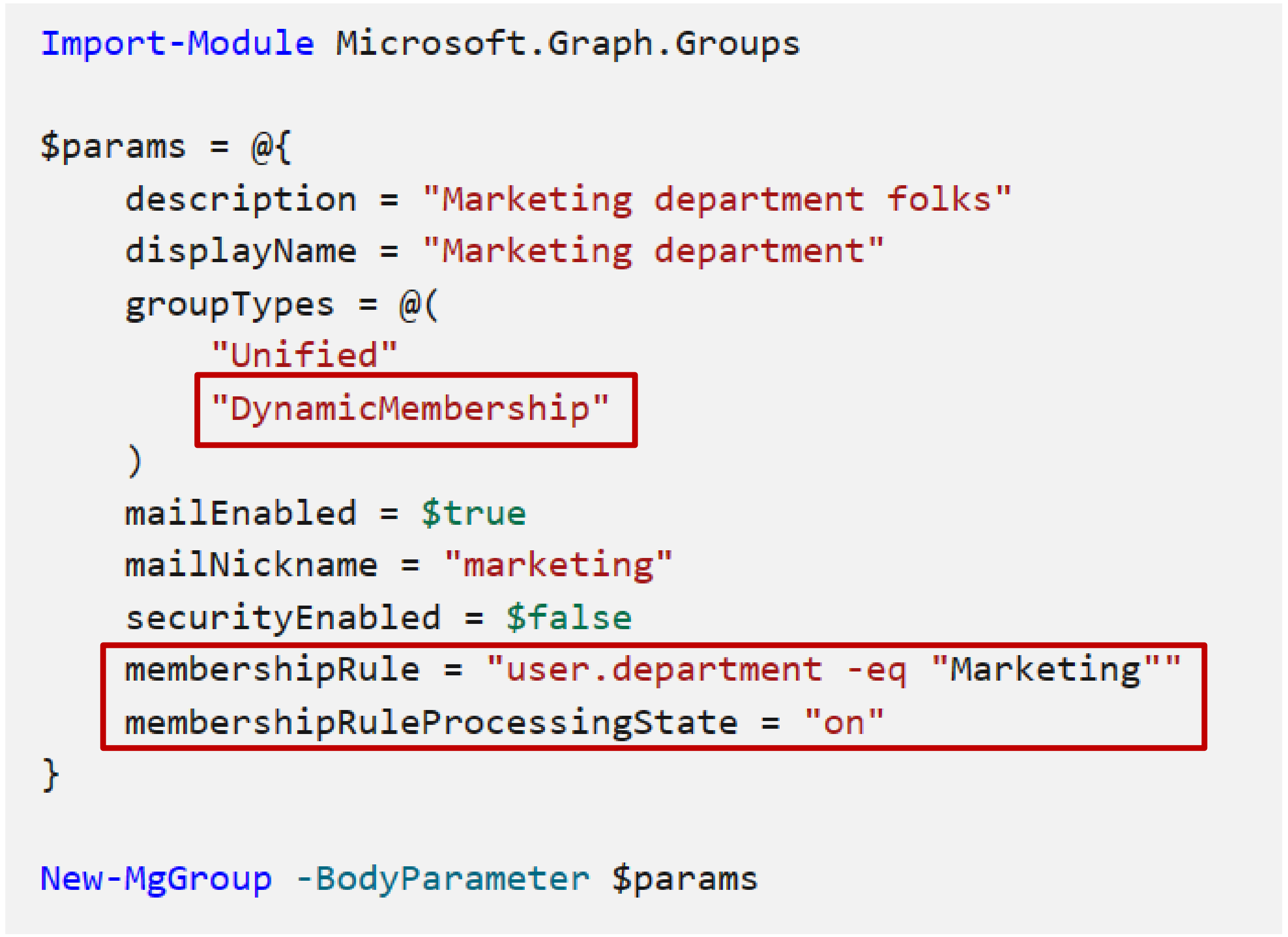
Dynamic membership

Microsoft 365 and security groups can have dynamic membership rules that automatically add or remove members from the group based on the principal's properties.

Only users and devices are supported as members in dynamic membership groups.

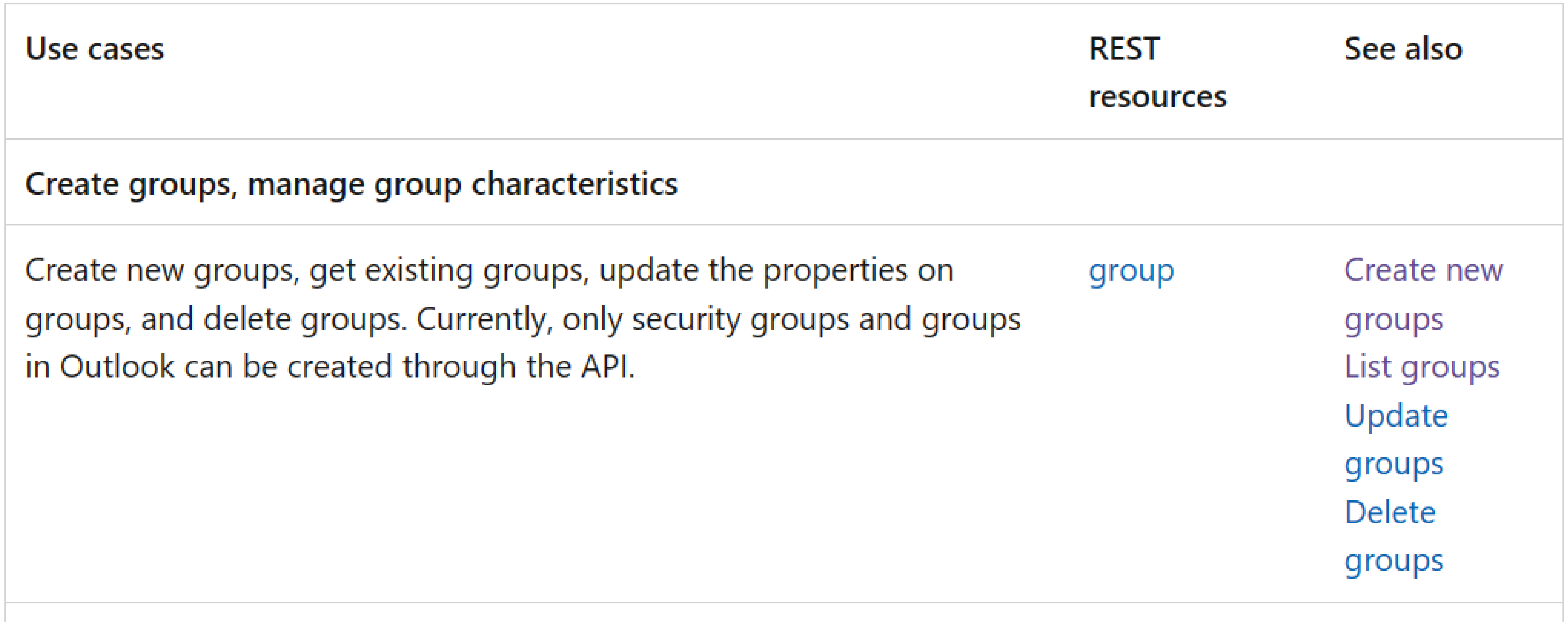
You can create a dynamic membership group for devices or users, but not both.

Properties for dynamic groups



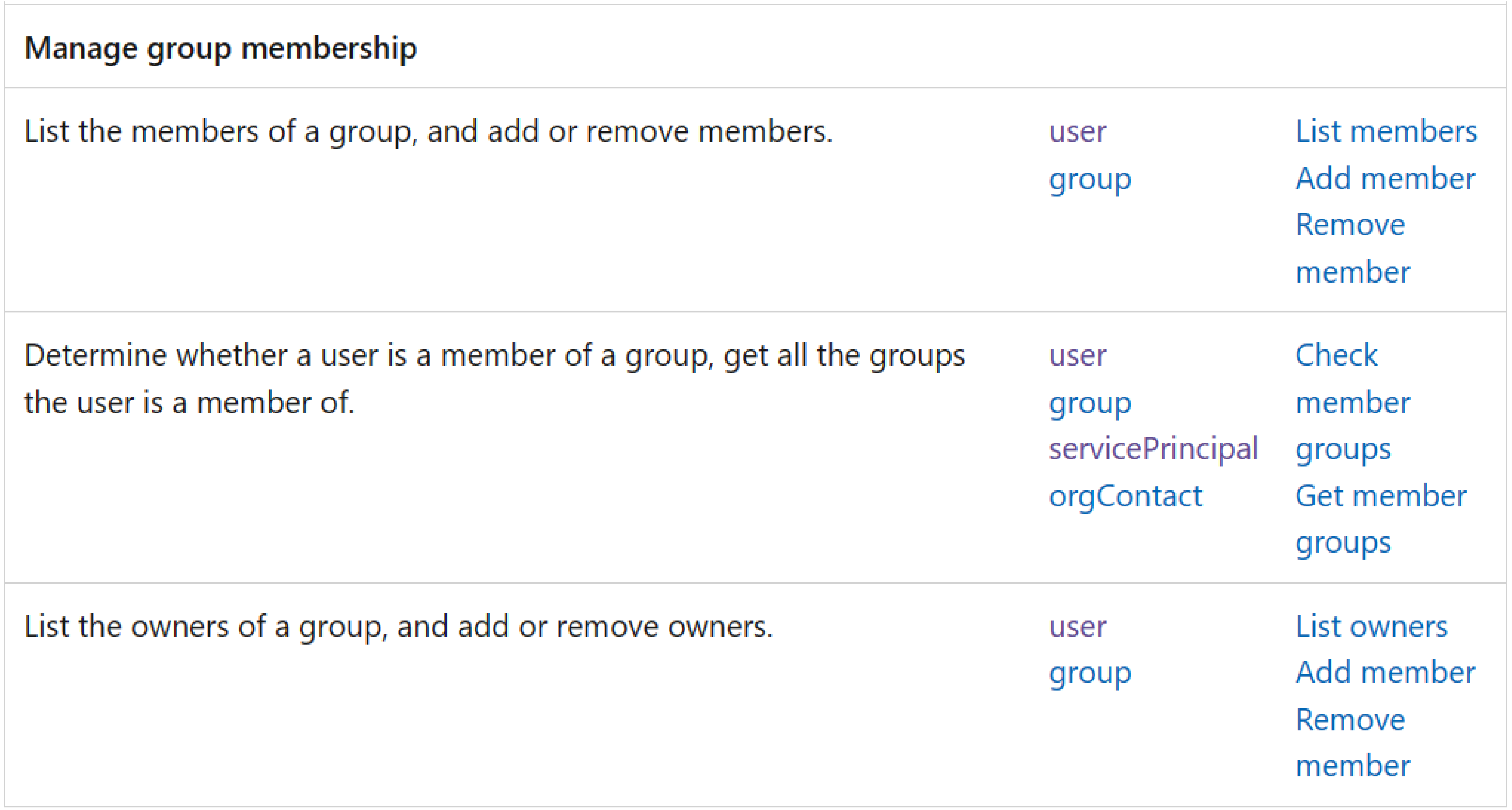
```powershell
Import-Module Microsoft.Graph.Groups

$params = @{
    description = "Marketing department folks"
    displayName = "Marketing department"
    groupTypes = @(
        "Unified"
        "DynamicMembership"
    )
    mailEnabled = $true
    mailNickname = "marketing"
    securityEnabled = $false
    membershipRule = "user.department -eq "Marketing""
    membershipRuleProcessingState = "on"
}

New-MgGroup -BodyParameter $params
```

Common operations



Use cases	REST resources	See also
Create groups, manage group characteristics		
Create new groups, get existing groups, update the properties on groups, and delete groups. Currently, only security groups and groups in Outlook can be created through the API.	group	Create new groups List groups Update groups Delete groups

Common operations



Manage group membership		
List the members of a group, and add or remove members.	user group	List members Add member Remove member
Determine whether a user is a member of a group, get all the groups the user is a member of.	user group servicePrincipal orgContact	Check member groups Get member groups
List the owners of a group, and add or remove owners.	user group	List owners Add member Remove member

Working with groups

DEMO

MODULE 05

Using Microsoft Graph API with PowerShell

5

When do we need direct Microsoft Graph API calls?

Microsoft Graph PowerShell doesn’t support access to all Microsoft Graph API endpoints

`Invoke-MgGraphRequest` issues REST API requests to the Graph API.

It works for any Graph API if you know the REST URI, method and optional body parameter.

This command is especially useful for accessing APIs for which there isn't an equivalent cmdlet yet.

You can also use `Invoke-RestMethod`.

What is a REST URI?

The URI (or Universal Resource Identifier) identifies where a specific resource can be found in Microsoft Graph.

The rest URI path for a specific user would be:

`https://graph.microsoft.com/v1.0/users/{id}`

`https://graph.microsoft.com` – The endpoint of the Microsoft Graph service.

`v1.0/` – The Microsoft Graph profile version.

`users/` – The path to the resource.

`{id}` – The unique resource ID.

Read information from Microsoft Graph

To read information from Microsoft Graph, you first need to create a request object and then run the GET method on the request.

# GET https://graph.microsoft.com/v1.0/users/{user-id}

```
Invoke-MgGraphRequest –Uri
‘https://graph.microsoft.com/v1.0/users/{user-id}’ –Method GET
```

Use $select to control the properties returned

When retrieving an entity, not all properties are automatically retrieved; sometimes they need to be explicitly selected.

In some scenarios it isn't necessary to return the default set of properties.

Selecting just the required properties can improve the performance of the request.

You can customize the request to include the $select query parameter with a list of properties.

```
# GET https://graph.microsoft.com/v1.0/users/{user-
id}?$select=displayName,jobTitle
```

Retrieve a subset of entities

The `$filter` query parameter can be used to reduce the result set to only those rows that match the provided condition.

Some requests for Entra ID resources require the use of advanced query capabilities.

If you get a response indicating a bad request, unsupported query, or a response that includes unexpected results, including the **`$count`** query parameter and **`ConsistencyLevel`** header may allow the request to succeed.

```
GET ~/users?$count=true&$filter=endsWith(mail,'@live.com')
```

Use $expand to access related entities

You can use the `$expand` filter to request a related entity, or collection of entities, at the same time that you request the main entity.

```
# GET https://graph.microsoft.com/v1.0/users/{user-
id}/messages?$expand=attachments
```

Delete an entity

Delete requests are constructed in the same way as requests to retrieve an entity, but use a `DELETE` request instead of a `GET`.

```
# DELETE https://graph.microsoft.com/v1.0/users/{user-
id}/messages/{message-id}
```

Creating a new entity

`POST` requests are used to create new resources

```
POST https://graph.microsoft.com/v1.0/applications
Content-type: application/json

{
  "displayName": "Display name"
}
```

Updating an existing entity with PATCH

Most updates in Microsoft Graph are performed using a `PATCH` method and therefore it is only necessary to include the properties that you want to change in the object you pass.

```
PATCH https://graph.microsoft.com/v1.0/servicePrincipals/{id}

Content-type: application/json


{
  "appRoleAssignmentRequired": true
}
```

DEMO

Access Microsoft Graph API directly with PowerShell

https://linkedin.com/in/alexandair

@alexandair

Thank You!

NT KONF

NT KONFERENCA

25. – 27.
SEPTEMBER
2023
PORTOROŽ