Configurer une application enregistrée dans Entra ID pour gérer une équipe Sharepoint

Inscription d'une application

Une inscription d'application dans Entra se fait :

- depuis le **portail Azure**,
- en **CLI/script** avec l'**API Microsoft Graph** par exemple en utilisant **Powershell**.

</WRAP>

Installer le module Microsoft.Graph de Powershell

Installer Powershell 7

Installer Microsoft.Graph de Powershell

```
* prérequis :
    * permissions API (Microsoft Graph) : Sites.Selected (Application)

* ouvrir une session Powershell en tant, qu'administrateur
```

```
Install-Module Microsoft.Graph
Import-Module Microsoft.Graph
```

Préparation Azure AD (OAuth2)

Depuis l'interface d'administration du portail Azure

- création d'une **Inscription d'applications** Entra ID :
    - Portail Azure ⇒ Entra ID
    - Inscription d'applications ⇒ Nouvelle inscription
    - Nom : app
    - Locataire unique seulement
    - S'incrire
- Récupérer :
    - Tenant ID
    - Client ID
    - l'ajout du certificat se fera ultérieurement (dans Certificates & Secrets)
- Ajouter la permission Microsoft Graph :
    - Autorisations d'application (et non Autorisations déléguées) :
        - Sites.Selected

      La permission **Sites.Selected** restreint l'accès au site explicitement autorisé via Graph.

> Une autorisation d'application ne nécessite pas de compte utilisateur :
>
> - L'application agit en autonomie (service, script, automatisation).
> - Nécessite un consentement administrateur.

- Demander le consentement de l'administrateur.

En CLI

- Connexion avec les droits nécessaires sur le tenant Entra ID

```
Connect-MgGraph -Scopes "Application.ReadWrite.All","AppRoleAssignment.ReadWrite.All"
Welcome to Microsoft Graph!

Connected via delegated access using 14d82eec-204b-4c2f-b7e8-296a70dab67e
Readme: https://aka.ms/graph/sdk/powershell
SDK Docs: https://aka.ms/graph/sdk/powershell/docs
API Docs: https://aka.ms/graph/docs

NOTE: You can use the -NoWelcome parameter to suppress this message.
```

- Création d'une nouvelle application

```
$app = New-MgApplication -DisplayName "Application"
```

- Création du Principal de service associé

```
$sp = New-MgServicePrincipal -AppId $app.AppId
```

Autoriser l'app enregistrée à mettre à jour les dossiers de l'équipe Teams

Cela ne peut pas se faire depuis l'interface d'administration Entra ID mais en utilisant Powershell.

Utilisation d'un certificat

- Création d'un certificat

```
$cert = New-SelfSignedCertificate `
  -Subject "CN=applicaton" `
  -CertStoreLocation "Cert:\CurrentUSer\My" `
  -KeySpec Signature `
  -KeyUsage DigitalSignature `
  -KeyExportPolicy Exportable `
  -HashAlgorithm SHA256 `
  -KeyLength 2048 `
  -NotAfter (Get-Date).AddYears(4)

$cert.Thumbprint
```

- Notez l'empreinte du certficat

```
$cert.Thumbprint
```

- Vérifier que le certificat est BIEN présent au BON endroit dans Cert:\CurrentUser\My

```
Get-Item "Cert:\CurrentUSer\My\<TON_THUMBPRINT>"
```

- Vérifier que la CLÉ PRIVÉE est présente dans le certificat

```
$cert = Get-Item "Cert:\CurrentUSer\My\<THUMBPRINT>"
$cert.HasPrivateKey
```

$cert.HasPrivateKey doit renvoyer True

- Export du certificat PUBLIC à importer dans Entra ID

```
Export-Certificate `
    -Cert "Cert:\CurrentUSer\My\$($cert.Thumbprint)" `
    -FilePath "application.cer"
```

```
* importer le certificat dans l'application
```

```
Add-MgApplicationKey  `
     -ApplicationId $app.Id  `
     -KeyCredentialFile "application.cer"
```

- script Powershell pour se conncter t etest la création d'un dossier

> Il est important de se connecter en tant qu'application et non en tant qu'utilisateur.

```
#############################################################
# CONFIGURATION pour l'accès avec l'application enregistrée
#############################################################

$TenantId     = "Tenant_educ-valadon#
$AppId        = "APP_ID_DE_TON_APPLICATION"

$Thumbprint = "EMPREINTE_DU_CERTIFICAT"

$SitePath      = "/sites/Signaturesnumriques"
$NewFolderName = "Test-SitesSelected-Cert-OK"



$SitePath = "/sites/Signaturesnumriques"
$NewFolderName = "Test-SitesSelected-OK"

#############################################################
# ÉTAPE 1 – OBTENIR UN TOKEN D'APPLICATION (CLIENT CREDENTIALS)
#############################################################

Write-Host "Authentification par certificat…"

$cert = Get-Item "Cert:\LocalMachine\My\$Thumbprint"

$assertion = [Microsoft.Identity.Client.ConfidentialClientApplicationBuilder]::Create($AppId).
    WithTenantId($TenantId).
    WithCertificate($cert).
    Build()

$tokenResult = $assertion.AcquireTokenForClient(@("https://graph.microsoft.com/.default")).
    ExecuteAsync().GetAwaiter().GetResult()

$Headers = @{
    Authorization = "Bearer $($tokenResult.AccessToken)"
}

Write-Host "Token d'application obtenu via certificat"


#############################################################
# ÉTAPE 2 – RÉCUPÉRER LE SITE SHAREPOINT PAR SON URL
#############################################################

Write-Host "Récupération du site SharePoint..."

$site = Invoke-RestMethod `
    -Method GET `
    -Uri "https://graph.microsoft.com/v1.0/sites/educvaladonlimogesfr.sharepoint.com:$SitePath" `
    -Headers $Headers

$SiteId = $site.id

Write-Host "Site trouvé : $($site.displayName)"
Write-Host "SiteId : $SiteId"

#############################################################
# ÉTAPE 3 – ATTRIBUER LA PERMISSION WRITE
#############################################################

Write-Host "Attribution de la permission WRITE à l'application..."

$permissionBody = @{
```

```
    roles = @("write")
    grantedToIdentities = @(
        @{
            application = @{
                id = $AppId
            }
        }
    )
} | ConvertTo-Json -Depth 5

Invoke-RestMethod `
    -Method POST `
    -Uri "https://graph.microsoft.com/v1.0/sites/$SiteId/permissions" `
    -Headers $Headers `
    -Body $permissionBody `
    -ContentType "application/json"

Write-Host "Permission Sites.Selected (write) appliquée"

#############################################################
# ÉTAPE 4 — TEST : CRÉATION D'UN DOSSIER DANS SHAREPOINT
#############################################################

Write-Host "Test : création d'un dossier SharePoint..."

$folderBody = @{
    name = $NewFolderName
    folder = @{}
} | ConvertTo-Json

Invoke-RestMethod `
    -Method POST `
    -Uri "https://graph.microsoft.com/v1.0/sites/$SiteId/drive/root/children" `
    -Headers $Headers `
    -Body $folderBody `
    -ContentType "application/json"

Write-Host "SUCCÈS : le dossier '$NewFolderName' a été créé dans SharePoint"
```

- Donner accès à l'application sur le site

```
Grant-MgSitePermission -SiteId $site.Id -Roles "write" -GrantedToIdentities @{ application = @{ id =
"APP_ID" } }

Grant-MgSitePermission `
    -SiteId $site.Id `
    -Roles "write" `
    -GrantedToIdentities @{ application = @{ id = "APP_ID" } }
```

- Tester l'accès avec Microsoft Graph (avec client secret)

Variables requises

```
$tenantId     = "<TON_TENANT_ID>"
$clientId     = "<APP_ID>"
$clientSecret = "<CLIENT_SECRET>"
$siteId       = $site.Id
```

- Obtenir un token avec l'application

```
$tokenBody = @{
    client_id     = $clientId
    scope         = "https://graph.microsoft.com/.default"
    client_secret = $clientSecret
    grant_type    = "client_credentials"
}

  *  Créer un dossier dans le site SharePoint
```

```
<code>
$headers = @{ Authorization = "Bearer $token" }
$body = '{ "name": "DossierTestCopilot", "folder": {} }'

Invoke-RestMethod `
    -Uri "https://graph.microsoft.com/v1.0/sites/$siteId/drive/root/children" `
    -Headers $headers `
    -Method POST `
    -Body $body `
    -ContentType "application/json"
```

Installer le module module PnP.PowerShell

PnP.PowerShell supporte :

- L'authentification Graph App-only
- La commande Grant-PnPAzureADAppSitePermission
- Les opérations moderne de gestion des Sites.Selected

```
Install-Module PnP.PowerShell -Scope CurrentUser
```

Se connecter

- génération d'un certificat autosigné

```
$cert = New-SelfSignedCertificate `
    -Subject "CN=MinioS3Sync" `
    -KeyAlgorithm RSA `
    -KeyLength 2048 `
    -CertStoreLocation "Cert:\CurrentUser\My" `
    -NotAfter (Get-Date).AddYears(3)
```

- Exporter le certificat .CER (public)

```
Export-Certificate `
    -Cert $cert `
    -FilePath "./minio-s3.cer"
```

- Exporter la clé privée .PFX

```
$password = Read-Host "Mot de passe du certificat" -AsSecureString

Export-PfxCertificate `
    -Cert $cert `
    -FilePath "./minio-s3.pfx" `
    -Password $password
``
```

- Uploader le fichier minio-s3.cer dans Entra ID
    - Entra ID → App Registration → Ton App → Certificates & secrets → Upload certificate → sélectionne minio-s3.cer
- se connecter

```
Connect-PnPOnline `
  -Url "https://educvaladonlimogesfr-admin.sharepoint.com" `
  -ClientId "ad39ff78-4116-43d6-86cf-3dfa8f120aa3" `
  -Tenant "educvaladonlimogesfr.onmicrosoft.com" `
  -CertificatePath "./minio-s3.pfx" `
  -CertificatePassword $password
```

Donner l'autorisation

```
Grant-PnPAzureADAppSitePermission `
  -AppId "ad39ff78-4116-43d6-86cf-3dfa8f120aa3" `
  -DisplayName "Minio-S3c" `
  -Site "https://educvaladonlimogesfr.sharepoint.com/sites/Signaturenumrique" `
  -Permissions Write
```

Tester la commande

```
Connect-PnPOnline -Url "https://<tenant>-admin.sharepoint.com" -ClientId "<appID>" -ClientSecret
"<secret>"
```

Script bash pour configurer l'accès à l'équipe Sharepoint associée à l'équipe Teams

Le script :

- génère un token OAuth2 (client_credentials)
- Récupère automatiquement le {site-id} via Graph API
- Récupère automatiquement le {drive-id} (bibliothèque SharePoint)
- Assigne les permissions Sites.Selected (write)` à l'application inscrite
- Teste l'accès via rclone (liste des fichiers)

- Prérequis :

```
apt install jq
```

setup_sharepoint_access.sh

```
#!/usr/bin/env bash

### ------------------------------------------------------------------------
###  CONFIGURATION À RENSEIGNER
### ------------------------------------------------------------------------

TENANT_ID="XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
CLIENT_ID="XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
CLIENT_SECRET="YOUR_APP_SECRET"

HOSTNAME="contoso.sharepoint.com"          # domaine SharePoint
SITE_PATH="DocumensoSite"                  # nom du site SharePoint
DRIVE_NAME="Documents"                     # nom de la bibliothèque (ex: "Documents")

RCLONE_REMOTE="sharepoint"                 # nom du remote rclone
RCLONE_MINIO="s3minio:documenso"           # bucket minio
RCLONE_SHAREPOINT="${RCLONE_REMOTE}:${DRIVE_NAME}"

### ------------------------------------------------------------------------
echo "1. Obtention du token OAuth2..."

ACCESS_TOKEN=$(
curl -s -X POST \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "client_id=$CLIENT_ID" \
  -d "scope=https://graph.microsoft.com/.default" \
  -d "client_secret=$CLIENT_SECRET" \
  -d "grant_type=client_credentials" \
  "https://login.microsoftonline.com/$TENANT_ID/oauth2/v2.0/token" \
  | jq -r .access_token
)

if [[ "$ACCESS_TOKEN" == "null" || -z "$ACCESS_TOKEN" ]]; then
  echo "❌ Impossible de générer un token ! Vérifie ton CLIENT_ID / SECRET."
  exit 1
fi

echo " Token obtenu."

### ------------------------------------------------------------------------
echo "2. Récupération automatique du site-id..."

SITE_INFO=$(curl -s -X GET \
  -H "Authorization: Bearer $ACCESS_TOKEN" \
  "https://graph.microsoft.com/v1.0/sites/$HOSTNAME:/sites/$SITE_PATH?$select=id,webUrl")
```

```
SITE_ID=$(echo "$SITE_INFO" | jq -r '.id')

if [[ "$SITE_ID" == "null" || -z "$SITE_ID" ]]; then
  echo "❌ Impossible de récupérer le site-id !"
  echo "Réponse brute:"
  echo "$SITE_INFO"
  exit 1
fi

echo "✅ SITE_ID trouvé : $SITE_ID"

### ------------------------------------------------------------------------
echo "3. Récupération automatique du drive-id de la bibliothèque $DRIVE_NAME..."

DRIVES_LIST=$(curl -s -X GET \
  -H "Authorization: Bearer $ACCESS_TOKEN" \
  "https://graph.microsoft.com/v1.0/sites/$SITE_ID/drives")

DRIVE_ID=$(echo "$DRIVES_LIST" | jq -r ".value[] | select(.name==\"$DRIVE_NAME\") | .id")

if [[ -z "$DRIVE_ID" ]]; then
  echo "❌ Impossible de trouver la bibliothèque '$DRIVE_NAME'"
  echo "Réponse brute:"
  echo "$DRIVES_LIST"
  exit 1
fi

echo "DRIVE_ID trouvé : $DRIVE_ID"

### ------------------------------------------------------------------------
echo "4. Attribution de la permission Sites.Selected (write) à l'application..."

ASSIGN_RESPONSE=$(curl -s -X POST \
  -H "Authorization: Bearer $ACCESS_TOKEN" \
  -H "Content-Type: application/json" \
  -d "{
    \"roles\": [\"write\"],
    \"grantedToIdentities\": [
      {
        \"application\": {
          \"id\": \"$CLIENT_ID\"
        }
      }
    ]
  }" \
  "https://graph.microsoft.com/v1.0/sites/$SITE_ID/permissions")

echo "✅ Permission appliquée."
echo "Réponse API :"
echo "$ASSIGN_RESPONSE"

### ------------------------------------------------------------------------
echo "5. Test d'accès SharePoint avec rclone..."

docker exec -it rclone-sync rclone lsd "$RCLONE_SHAREPOINT"

echo "✅ Configuration terminée avec succès !"
echo "SITE_ID  = $SITE_ID"
echo "DRIVE_ID = $DRIVE_ID"
```

- exécuter le script

```
bash setup_sharepoint_access.sh
```