Mise en place d’un script Powershell

Ce script PowerShell est conçu pour automatiser la création d'utilisateurs, de groupes, d'unités organisationnelles (OU) et de dossiers dans un environnement Active Directory (AD), ainsi que la gestion des autorisations NTFS sur ces dossiers.

Voici une explication détaillée de chaque partie du script :

Importer les données du fichier CSV :

Le script commence par importer les données d'un fichier CSV qui contient les informations sur les utilisateurs, telles que leur **nom, prénom, identifiant, mot de passe, filière, classe, etc**.

Création de l'unité organisationnelle principale et du groupe d'élèves :

Il vérifie d'abord si l'unité organisationnelle principale **"Eleves"** existe déjà dans l'AD. Si elle n'existe pas, elle est créée, ainsi qu'un groupe nommé **"eleves"** pour les élèves.

Création des dossiers pour les filières et les classes :

Le script crée deux dossiers principaux : **"Filieres"** et **"Classes"** dans le répertoire spécifié. Ces dossiers seront utilisés pour organiser les dossiers des filières et des classes.

Parcours des données du CSV pour créer les filières :

Pour chaque entrée du CSV, le script vérifie si une OU correspondante à la filière existe déjà dans l'AD. Si elle n'existe pas, elle est créée. Il crée également un groupe de sécurité correspondant à chaque filière.

Parcours des données du CSV pour créer les classes :

Pour chaque entrée du CSV, le script vérifie si une OU correspondante à la classe existe déjà dans l'AD, sous l'OU de sa filière respective. Si elle n'existe pas, elle est créée. Il crée également un groupe de sécurité correspondant à chaque classe.

Création des utilisateurs et attribution des groupes :

Le script parcourt à nouveau les données du CSV pour créer chaque utilisateur dans l'AD. Il attribue chaque utilisateur à son OU de classe et de filière, ainsi qu'au groupe d'élèves global. Il crée également un dossier pour chaque élève dans le répertoire **"Eleves"**, s'il n'existe pas déjà, et lui accorde les autorisations NTFS appropriées sur son dossier personnel.

Ajout des autorisations NTFS sur les dossiers des classes :

Pour chaque classe, le script ajoute des autorisations NTFS pour permettre aux membres de la classe de lire et exécuter les fichiers dans leur dossier de classe respectif.

Le script est commenté à chaque étape pour expliquer ce qu'il fait. Il automatise efficacement la création et la gestion des comptes d'utilisateurs dans un environnement AD, tout en organisant les utilisateurs dans des unités organisationnelles appropriées et en attribuant les autorisations nécessaires sur les dossiers.

```
# Importer les données
$CSVFile = "E:\Powershell\comptes_eleves.csv"
$CSVData = Import-Csv -Path $CSVFile -Delimiter ";" -Encoding Default

$ouPrincipale = "OU=Eleves,DC=lprs,DC=org"
$groupEleveName = "eleves"
$filieres = @()        # Créer un tableau pour stocker les filières
$increment = 0         # Déclarer une variable pour l'incrémentation

# Chemin de l'UO principale
if (-not (Get-ADOrganizationalUnit -Filter {Name -eq "Eleves"} -
SearchBase "DC=lprs,DC=org")) {
    New-ADOrganizationalUnit -Name "Eleves" -Path "DC=lprs,DC=org" -
ProtectedFromAccidentalDeletion $false
    New-ADGroup -Name $groupEleveName -Path $ouPrincipale -GroupCategory
Security -GroupScope Global
    Write-Output "Création de l'UO et du Groupe : Eleves"
}

# Créer le dossier Filieres dans $rootDirectory
$filiereDirectory = Join-Path -Path "E:\Peda\" -ChildPath "Filieres"
if (-not (Test-Path $filiereDirectory)) {
    New-Item -Path $filiereDirectory -ItemType Directory -Force
    Write-Output "Création du dossier Filieres"
}

# Créer le dossier Classes dans $rootDirectory
$classesDirectory = Join-Path -Path "E:\Peda\" -ChildPath "Classes"
if (-not (Test-Path $classesDirectory)) {
    New-Item -Path $classesDirectory -ItemType Directory -Force
    Write-Output "Création du dossier Classes"
}

# Parcourir chaque ligne du CSV pour collecter les filières
foreach ($entry in $CSVData) {
    $filiere = $entry.FILIERE

    # Vérifier si l'UO pour cette filière existe déjà
    $filiereOU = "OU=$filiere,$ouPrincipale"
    if (-not (Get-ADOrganizationalUnit -Filter {Name -eq $filiere} -
SearchBase $ouPrincipale)) {
        New-ADOrganizationalUnit -Name $filiere -Path $ouPrincipale -
ProtectedFromAccidentalDeletion $false
        New-ADGroup -Name $filiere -Path $filiereOU -GroupCategory
Security -GroupScope Global
        Write-Output "Création de l'UO et du Groupe : $filiere"
    }

    # Ajouter la filière au tableau
    $filieres += $filiere

    # Créer le dossier de la filière
    $filiereFolderPath = Join-Path -Path $filiereDirectory -ChildPath
$filiere
```

```
    if (-not (Test-Path $filiereFolderPath)) {
        New-Item -Path $filiereFolderPath -ItemType Directory -Force
        Write-Output "Création du dossier pour la filière : $filiere"
    }
}

# Parcourir à nouveau le CSV pour créer les classes
foreach ($entry in $CSVData) {
    $filiere = $entry.FILIERE
    $classe = $entry.CLASSE

    # Créer une UO pour la classe sous l'UO de la filière
    $filiereOU = "OU=$filiere,$ouPrincipale"
    $classeOU = "OU=$classe,$filiereOU"
    if (-not (Get-ADOrganizationalUnit -Filter {Name -eq $classe} -
SearchBase $filiereOU)) {
        New-ADOrganizationalUnit -Name $classe -Path $filiereOU -
ProtectedFromAccidentalDeletion $false
        New-ADGroup -Name $classe -Path $classeOU -GroupCategory Security
-GroupScope Global
        Write-Output "Création de l'UO et du Groupe : $classe ($filiere)"
    }

    # Créer le chemin pour le dossier de la classe
    $classFolderPath = Join-Path -Path $classesDirectory -ChildPath
$classe

    # Créer le dossier de la classe s'il n'existe pas encore
    if (-not (Test-Path $classFolderPath)) {
        New-Item -Path $classFolderPath -ItemType Directory -Force
    }

    # Ajouter les droits NTFS pour la classe sur son dossier
    $classACL = Get-Acl -Path $classFolderPath
    $classACL.AddAccessRule((New-Object
System.Security.AccessControl.FileSystemAccessRule($classe,
"ReadAndExecute", "ContainerInherit,ObjectInherit", "None", "Allow")))
    Set-Acl -Path $classFolderPath -AclObject $classACL
}

# Emplacement racine pour les dossiers classe et eleves
$rootDirectory = "E:\Peda\"

# Boucle Foreach qui crée les utilisateurs et les dossiers
foreach ($User in $CSVData) {
    $UserNom = $User.NOM
    $UserPrenom = $User.PRENOM
    $UserLogin = $User.IDENTIFIANT
    $UserPassword = $User.PASSWORD
    $UserFiliere = $User.FILIERE
    $UserClasse = $User.CLASSE
    $UserEmail = $User.MESSAGERIE
    $UserExpirationDate = $User.ExpirationDate
```

```
    $UserExpirationDateTime = [datetime]::ParseExact($UserExpirationDate,
"dd/MM/yyyy", $null)
    $UserChangePass = $User.ChangePassword
    $UserScript = $User.Script

    # Vérifie si l'utilisateur existe déjà dans l'AD
    if (Get-ADUser -Filter {SamAccountName -eq $UserLogin}) {
        Write-Warning "L'identifiant $UserLogin existe déjà dans l'AD"
    } else {
        $UserOU = "OU=$UserClasse,OU=$UserFiliere,$ouPrincipale"
        New-ADUser -Name "$UserNom $UserPrenom" `
            -DisplayName "$UserNom $UserPrenom" `
            -GivenName $UserPrenom `
            -Surname $UserNom `
            -SamAccountName $UserLogin `
            -UserPrincipalName "$UserLogin@lprs.org" `
            -EmailAddress $UserEmail `
            -Path $UserOU `
            -AccountPassword (ConvertTo-SecureString $UserPassword -
AsPlainText -Force) `
            -AccountExpirationDate $UserExpirationDateTime `
            -ChangePasswordAtLogon $false `
            -CannotChangePassword $true `
            -PasswordNeverExpires $true `
            -Enabled $true `
            -ScriptPath $UserScript

        # Ajouter l'utilisateur aux groupes de filière et de classe
        Add-ADGroupMember -Identity $UserFiliere -Members $UserLogin
        Add-ADGroupMember -Identity $UserClasse -Members $UserLogin
        Add-ADGroupMember -Identity $groupEleveName -Members $UserLogin

        # Créer le chemin pour le dossier de l'élève
        $UserDirectory = Join-Path -Path "E:\Peda\Eleves\$UserClasse" -
ChildPath $UserLogin

        # Créer le dossier de l'élève s'il n'existe pas encore
        if (-not (Test-Path -Path $UserDirectory)) {
            New-Item -Path $UserDirectory -ItemType Directory -Force
            Write-Output "Création du dossier pour l'élève : $UserLogin
dans la classe : $UserClasse"
        }

        # Ajouter les droits NTFS pour l'élève sur son dossier
        $UserDirectory = Join-Path -Path "E:\Peda\Eleves\$UserClasse" -
ChildPath $UserLogin
        $userACL = Get-Acl -Path $UserDirectory
        $userACL.AddAccessRule((New-Object
System.Security.AccessControl.FileSystemAccessRule($UserLogin,
"ReadAndExecute", "ContainerInherit,ObjectInherit", "None", "Allow")))
        Set-Acl -Path $UserDirectory -AclObject $userACL

        $increment++
```

```
        Write-Output "$increment. Création de l'utilisateur : $UserLogin
($UserNom $UserPrenom)"
    }
}
```