

La Neuville du Bosc, 27890

14/02/25

Script AD

Maël Poisson

Création d'un script pour l'AD

Consigne : Créer un script qui permet de désactiver les comptes inactifs depuis 90 jours.

Dans notre cas, nous allons créer 2 scripts.

1^{er} script

Il permet de rechercher les utilisateurs qui ont plus de 90 jours d'inactivité et de les enregistrer dans un fichier **.csv**.

```
1 # Importer le module Active Directory
2 Import-Module ActiveDirectory
3
4 # Définir la période d'inactivité (90 jours)
5 $InactiveDays = 90
6 $DateThreshold = (Get-Date).AddDays(-$InactiveDays)
7
8 # Rechercher les comptes d'utilisateur inactifs
9 $InactiveAccounts = Get-ADUser -Filter {
10     LastLogonDate -lt $DateThreshold -and Enabled -eq $true
11 } -Properties LastLogonDate, Enabled |
12 Where-Object { $_.LastLogonDate -ne $null }
13
14 # Vérifier si des comptes inactifs ont été trouvés
15 if ($InactiveAccounts) {
16     # Préparer les données avec une colonne de jours d'inactivité
17     $ExportData = $InactiveAccounts | Select-Object SamAccountName, Name, LastLogonDate, @{Name="DaysInactive"; Expression={(New-TimeSpan -Start $_.LastLogonDate -End (Get-Date)).Days}}
18
19     # Exporter les résultats dans un fichier CSV avec encodage UTF-8
20     $ExportData | Export-Csv -Path "chemin-réseaux" -NoTypeInformation -Encoding UTF8
21
22     Write-Host "Les comptes inactifs ont été exportés vers le fichier CSV." -ForegroundColor Green
23 } else {
24     Write-Host "Aucun compte inactif trouvé." -ForegroundColor Red
25 }
26
```

Explication du script

Ligne 2

On importe le module **ActiveDirectory** pour qu'il puisse interagir avec l'AD.

Lignes 5 et 6

On fixe la période d'inactivité et il la calcule en fonction de la date actuelle.

Lignes 9 à 11

Il récupère les comptes utilisateurs dont la date de dernière connexion (**LastLogonDate**) est antérieure à la date seuil et qui sont toujours activés.

Ligne 12

Il s'assure que seuls les comptes ayant une date de dernière connexion non nulle sont pris en compte.

Lignes 15 à 20

- Il crée une liste contenant le **SamAccountName**, le **Name**, la **LastLogonDate** et le nombre de jours d'inactivité.
- Il exporte ces données dans un fichier **.csv** en UTF-8.

Lignes 22 à 24

- Ensuite, dans ce fichier, on effectue le tri des comptes à ne pas désactiver, comme les comptes administrateurs ou autres, puis on enregistre et on ferme.

[illegible]

2ème script

Il permet d'utiliser le fichier **.csv** créé par le premier script pour désactiver les comptes qui y figurent. De plus, il crée un autre fichier **.csv** qui répertorie les comptes qui viennent d'être désactivés ainsi que ceux qui étaient déjà désactivés. Enfin, ce fichier rapporte le nom de l'utilisateur, s'il vient d'être désactivé ou s'il était déjà désactivé, la date et le nombre de jours d'inactivité du compte.

```

1  # Définir le chemin du fichier CSV d'entrée et de sortie
2  $cheminCsvEntree = "chemin-reseau"
3  $cheminCsvSortie = "chemin-reseau"
4
5  # Vérifier si le fichier de sortie existe déjà
6  if (Test-Path $cheminCsvSortie) {
7      # Charger les données existantes du fichier CSV de sortie
8      $donneesExistantes = Import-Csv -Path $cheminCsvSortie
9  } else {
10     # Si le fichier de sortie n'existe pas, créer un tableau vide pour ajouter les données
11     $donneesExistantes = @()
12 }
13
14 # Importer les données du fichier CSV d'entrée
15 $donneesCsv = Import-Csv -Path $cheminCsvEntree
16
17 # Parcourir chaque ligne du CSV
18 foreach ($ligne in $donneesCsv) {
19     # Récupérer les informations pour chaque utilisateur
20     $nomUtilisateur = $ligne."SamAccountName" # Nom d'utilisateur (SamAccountName)
21     $nomComplet = $ligne."Name" # Nom complet
22     $dateDerniereConnexion = $ligne."LastLogonDate" # Date de dernière connexion
23     $joursInactivite = [int]$ligne."DaysInactive" # Nombre de jours d'inactivité (assurer qu'il soit bien un entier)
24
25     # Vérifier que la donnée DaysInactive est valide
26     if ($null -eq $joursInactivite) {
27         Write-Host "Erreur : Nombre de jours d'inactivité non valide pour l'utilisateur $nomUtilisateur"
28         continue
29     }
30
31     # Vérifier si le nombre de jours d'inactivité est supérieur à 90
32     if ($joursInactivite -gt 90) {
33         try {
34             # Vérifier si le compte est déjà désactivé
35             $utilisateurAD = Get-ADUser -Identity $nomUtilisateur -Properties Enabled -ErrorAction SilentlyContinue
36
37             if ($utilisateurAD -eq $null) {
38                 Write-Host "Erreur : Le compte $nomUtilisateur n'existe pas."
39                 continue
40             }
41
42             if ($utilisateurAD.Enabled -eq $false) {
43                 # Si le compte est déjà désactivé, ajouter au fichier avec une indication spécifique
44                 $donneesExistantes += New-Object PSObject -property @{
45                     "Nom complet" = $nomComplet
46                     "Nom d'utilisateur" = $nomUtilisateur
47                     "Date de désactivation" = (Get-Date).ToString("yyyy-MM-dd HH:mm:ss")
48                     "Statut" = "Déjà désactivé"
49                 }
50                 Write-Host "Le compte $nomUtilisateur est déjà désactivé."
51             } else {
52                 # Si le compte est actif, le désactiver
53                 Disable-ADAccount -Identity $nomUtilisateur -ErrorAction SilentlyContinue
54                 Write-Host "Compte désactivé : $nomUtilisateur"
55
56                 # Ajouter l'utilisateur qui a été désactivé au fichier CSV
57                 $donneesExistantes += New-Object PSObject -property @{
58                     "Nom complet" = $nomComplet
59                     "Nom d'utilisateur" = $nomUtilisateur
60                     "Date de désactivation" = (Get-Date).ToString("yyyy-MM-dd HH:mm:ss")
61                     "Statut" = "Désactivé"
62                 }
63             }
64         } catch {
65             Write-Host "Erreur lors de la gestion du compte $nomUtilisateur : $_"
66         }
67     }
68 }
69
70 # Exporter les données mises à jour dans le fichier CSV de sortie
71 $donneesExistantes | Export-Csv -Path $cheminCsvSortie -NoTypeInformation -Force
72
73 Write-Host "Le fichier CSV a été mis à jour avec succès."
74

```

Explication du script

Lignes 2 et 3

Il définit le chemin des fichiers **.csv** d'entrée (généré par le premier script) et de sortie.

Lignes 5 à 12

Il teste si le fichier de sortie **.csv** existe.

- Si oui : il charge les données existantes du fichier.
- Si non : il crée un tableau vide pour y ajouter les données.

Lignes 14 et 15

Il importe les données du fichier d'entrée.

Lignes 19 à 23

Il récupère les informations du fichier d'entrée :

- **SamAccountName** (identifiant AD de l'utilisateur)
- **Name** (nom complet)
- **LastLogonDate** (date de dernière connexion)
- **DaysInactive** (nombre de jours d'inactivité)

Lignes 25 à 29

Il vérifie si **DaysInactive** est bien un nombre.

Lignes 32 à 68

Il vérifie que le compte est inactif depuis plus de 90 jours.

Si le compte est inactif depuis plus de 90 jours :

- Il vérifie si le compte existe et est actif.
- Si le compte est déjà désactivé : il l'ajoute au fichier avec le statut « **Déjà désactivé** ».
- Si le compte est actif : il le désactive avec **Disable-ADAccount** et l'ajoute au fichier avec le statut « **Désactivé** ».

Mise à jour du fichier .csv avec les données

Ligne 71

Il exporte le fichier de sortie avec les données mises à jour.

Ligne 73

Il renvoie le message : « **Le fichier CSV a été mis à jour avec succès.** »

Maël Poisson

Exemple de Fichier .cvs de sortie