Scripts du Stage

Pour les utilisateurs inactifs depuis X jours :

```
# Importer le module Active Directory

Import-Module ActiveDirectory

# Définir la période d'inactivité (180 jours)

$inactivityThreshold = (Get-Date).AddDays(-180)

# Rechercher les utilisateurs dont la dernière connexion est antérieure à la date seuil

Get-ADUser -Filter {Enabled -eq $true -and LastLogonTimeStamp -lt $inactivityThreshold} `

  -Properties DisplayName, LastLogonTimeStamp |

  Select-Object DisplayName, SamAccountName,
@{Name="LastLogonDate";Expression={[DateTime]::FromFileTime($_.LastLogonTimeStamp)}} |

  Sort-Object LastLogonDate
```

Explication :

- `Import-Module ActiveDirectory` : charge les commandes AD.
  `LastLogonTimeStamp` : propriété répliquée (approximative, mais suffisante pour ce type de recherche).
- `.AddDays(-180)` : calcule la date 180 jours en arrière.
- `[DateTime]::FromFileTime(...)` : convertit le timestamp brut en date lisible.

Pour les ordinateurs inactifs depuis X jours :

```
# Importer le module Active Directory

Import-Module ActiveDirectory

# Définir la date de seuil (180 jours d'inactivité)

$inactivityThreshold = (Get-Date).AddDays(-180)

# Rechercher les ordinateurs dont le dernier logon est antérieur à cette date

Get-ADComputer -Filter {Enabled -eq $true -and LastLogonTimeStamp -lt $inactivityThreshold} `

  -Properties Name, LastLogonTimeStamp |
```

Select-Object Name,
@{Name="LastLogonDate";Expression={[DateTime]::FromFileTime($_.LastLogonTimeStamp)}} |

Sort-Object LastLogonDate

Explication :

- `Get-ADComputer` : commande pour récupérer les objets ordinateurs.
- `LastLogonTimeStamp` : date approximative du dernier logon (répliquée).
  `AddDays(-180)` : calcule la date d'il y a 180 jours.

Script PowerShell pour lister les utilisateurs inactifs depuis 180 jours et afficher ses appartenances (groupes AD, OU) et les droits NTFS (lecture, écriture…) sur les dossiers/fichiers d'un chemin donné

1. Détection des utilisateurs inactifs

```
# Charger le module Active Directory

Import-Module ActiveDirectory


# Seuil d'inactivité : 180 jours

$inactivityThreshold = (Get-Date).AddDays(-180)


# Récupérer les utilisateurs inactifs

$inactiveUsers = Get-ADUser -Filter {

    Enabled -eq $true -and LastLogonTimeStamp -lt
$inactivityThreshold

} -Properties SamAccountName, LastLogonTimeStamp, MemberOf,
DistinguishedName


# Afficher un résumé

$inactiveUsers | Select-Object SamAccountName,

    @{Name="LastLogonDate";
Expression={[DateTime]::FromFileTime($_.LastLogonTimeStamp)}},
```

```
    DistinguishedName,
    @{Name="GroupMemberships"; Expression={
        ($_ | Get-ADUser -Properties MemberOf).MemberOf |
ForEach-Object {
            (Get-ADGroup $_).Name
        } -join ", "
    }}
```

2. Vérification des droits NTFS sur un répertoire cible

```
# Répertoire à scanner
$folderPath = "C:\DossiersPartages"


# Pour chaque utilisateur inactif, vérifier les permissions sur les
fichiers/dossiers
foreach ($user in $inactiveUsers) {
    Write-Host "Permissions NTFS pour $($user.SamAccountName):"
-ForegroundColor Cyan


# Parcourir tous les fichiers et dossiers dans le chemin donné
    Get-ChildItem -Path $folderPath -Recurse -Force | ForEach-Object
{
        try {
            $acl = Get-Acl $_.FullName
            foreach ($access in $acl.Access) {
                if ($access.IdentityReference -like
"*$($user.SamAccountName)") {
                    [PSCustomObject]@{
                        Utilisateur = $user.SamAccountName
                        FichierOuDossier = $_.FullName
                        Accès = $access.FileSystemRights
```

```
                        Hérité = $access.IsInherited
                    }
                }
            }
        } catch {
            Write-Warning "Impossible d'accéder à $_"
        }
    }
}
```