Fiche 07 : Premiers scripts d'administration d'Active Directory via ADSI

1. Script de création de comptes utilisateurs à partir d'un fichier texte au format CSV

```powershell
# Premier script ADSI, script03.ps1
# Création de comptes utilisateur à partir d'un fichier texte
# au format CSV : (prenom,nom,description,tel)

$domaine=",dc=labo,dc=ig"
$unite="test"
$ActiverCompte=512                      # Pour activer le compte
$pwd="toto"                             # Mot de passe première connexion
$fichierUser="c:\temp\users.txt"
$cible="LDAP://ou="+$unite+$domaine

$ldapUnit=[ADSI] $cible
Import-Csv $fichierUser | ForEach-Object {
     $user=$ldapUnit.create("user","cn="+$_.prenom+" "+$_.nom)
     $user.put("sAMAccountName",$_.nom)
     $user.put("sn",$_.nom)
     $user.put("givenName",$_.prenom)
     $user.put("DisplayName",$_.nom+" "+$_.prenom)
     $user.put("userPrincipalName",$_.nom)
     $user.put("description", $_.description)
     $user.put("telephoneNumber",$_.tel)
     $user.SetInfo()
     $user.SetPassword($pwd)
     $user.put("userAccountControl", $ActiverCompte)      # Active le compte
     $user.put("pwdLastSet", 0)  # Changement du mot de passe 1ère connexion
     $user.SetInfo()
}
```

Valeurs de userAccountControl, voir site : msdn.microsoft.com/fr-fr/ms677286(en-

Cette commande permet de lire un fichier CSV, chaque objet (ligne) est transmis au pipeline.

Chaque élément du compte utilisateur est accessible par la désignation définie dans le fichier (1ère ligne) et la variable $_ qui représente la ligne en cours.

2. Idem mais avec les tests d'existence de l'unité d'organisation et des comptes utilisateurs

Extrait du script :

```powershell
# avec tests d'existence de l'organisation et du compte utilisateur
$ldapUnit=[ADSI] $cible
if (!$ldapUnit.DistinguishedName){
     Write-Host "$unite : cette unité d'organisation n'existe pas"
}
else{
     $recherche=New-Object DirectoryServices.DirectorySearcher
     Import-Csv $fichierUser | ForEach-Object {
          $cn=$_.prenom+" "+$_.nom
          $recherche.Filter="(cn=$cn)"
          $user=$recherche.FindOne()
          if($user -eq $null){
               $user=$ldapUnit.create("user","cn="+$cn)
```

En fait, on teste l'existence de l'unité ou du

Avec la version 2.0 : ldapDomaine=[ADSI]"LDAP://labo.ig"
$recherche=[ADSISearcher]
$ldapDomaine

Cet objet permet de faire une recherche dans tout le domaine et pas seulement dans l'unité

Remarque : Ligne 23, on peut aussi écrire : "cn=$cn", comme à la ligne 20, ce qui facilite le traitement ses parenthèses () dans la chaîne de caractères.

3. Script qui liste tous les noms de poste qui ne sont pas des serveurs.

```
1  #script03_03, Liste des noms des postes de travail sans les serveurs
2  $searcher=New-Object System.DirectoryServices.DirectorySearcher
3  $searcher.filter="(&(Objectcategory=computer)(!operatingsystem=*server*))"
4  $searcher.findall() | foreach-object {$_.properties.name}
```

Le résultat de la méthode est transmis au pipeline qui lui ne liste que le nom du poste

Le filtre sélectionne les objets de la catégorie 'computer' avec un 'operatingsystem' différent (!) de '*server*'. L'opérateur (&) est placé devant les

Le résultat de cette recherche peut être placé dans un fichier texte avec Add-content, mais aussi avec Export-Csv :

```
4  $searcher.findall() | select @{name="Nom";Expression={$_.properties.name}},
5                        @{name="Date creation";Expression={$_.properties.whencreated}}| Export-Csv "c:\temp\poste.txt"
```

Remarques : Ne sont sélectionnées que les propriétés noms et date de création du poste. La transmission est réalisée avec 2 pipelines.