

POWERSHELL

Module Active Directory

1) INTRODUCTION AU MODULE ACTIVE DIRECTORY	3
A. EXERCICE 1	3
B. LES NOMS LDAP	4
C. RECHERCHE LDAP	5
<i>Get-ADUser -Filter *</i>	5
<i>Get-ADUser -LDAPFilter '(name=*elo*)'</i>	5
<i>Get-ADUser -Filter * -SearchBase 'OU=nomOU,DC=nom_domaine,DC=tld_domaine'</i>	5
D. EXERCICE 2	5
2) UTILISATEURS	6
A. LES PARAMETRES « NAME »	6
<i>Name</i>	6
<i>Surname</i>	6
<i>GivenName</i>	6
<i>SAMAccountName</i>	7
<i>UserPrincipalName</i>	7
<i>Exercice 3</i>	7
B. LES PARAMETRES « PASSWORD »	7
<i>AccountPassword</i>	7
<i>CannotChangePassword</i>	7
<i>ChangePasswordAtLogon</i>	8
<i>PasswordNeverExpires</i>	8
<i>Exercice 4</i>	8
C. LE PARAMETRE ENABLE	8
<i>Exercice 5</i>	8
D. READ-HOST	9
<i>Exercice 6</i>	10
<i>Exercice 7</i>	10
<i>Boucle Do-While</i>	10
<i>Exercice 8</i>	10
E. IMPORT DE FICHIER CSV	11
<i>Import-csv</i>	12
<i>Exercice 9</i>	12
<i>Import-csv commande</i>	12
<i>Exercice 10</i>	13
<i>Switch</i>	13
<i>Exercice 11</i>	13
3) GROUPES	14
A. EXERCICE 12	14
B. EXERCICE 13	14
C. IMPORT-CSV ET BOUCLE FOREACH	14
<i>Boucle ForEach</i>	14
<i>Exercice 14</i>	17
<i>Exercice 15</i>	17
D. EXERCICE 16 (SYNTHESE UTILISATEURS – GROUPES GLOBAUX)	17

1) Introduction au module Active Directory

On a vu précédemment que PowerShell 3.0 disposait d'un jeu de commandes par défaut, enrichi avec des modules complémentaires, dont le module Active Directory. Le module Active Directory permet d'automatiser un grand nombre de tâches courantes ; ce support n'est pas exhaustif et ne traitera que des tâches les plus communes, à savoir la gestion des utilisateurs et des groupes (globaux et de domaine locaux).

a. Exercice 1

Quelle commande permet d'obtenir la liste des commandes du module Active Directory ?

Le module est installé par défaut dès qu'un serveur devient contrôleur de domaine. Il importe avec lui un fournisseur (aussi appelé Provider), qui permet de naviguer et gérer l'Active Directory comme un «simple lecteur ». Pour connaître le point d'entrée du module (le lecteur associé), entrez la commande Get-PSDrive :

```
PS C:\Windows> get-PSDrive
```

Name	Used (GB)	Free (GB)	Provider	Root	CurrentLocation
AD			ActiveDire...	//RootDSE/	cn=users,dc=ps,dc=test
Alias			Alias		
C	10,64	14,26	FileSystem	C:\	Windows
Cert			Certificate	\	
D	2,98		FileSystem	D:\	
Env			Environment		
Function			Function		
HKCU			Registry	HKEY_CURRENT_USER	
HKLM			Registry	HKEY_LOCAL_MACHINE	
Variable			Variable		
WSMan			WSMan		

Le lecteur associé à l'Active Directory est donc AD.

NB : Notez au passage que d'autres Providers existent ; on peut notamment naviguer dans la base de registre sur les clés HKCU et HKLM via une console Powershell.

On navigue donc dans l'Active Directory comme dans une arborescence de fichiers :

```
PS C:\Windows> cd AD:
PS AD:\cn=users,dc=ps,dc=test> get-ChildItem
```

Name	ObjectClass	DistinguishedName
Administrateur	user	CN=Administrateur,CN=Users,DC=ps,DC=test
Administrateurs d...	group	CN=Administrateurs de l'entreprise,CN=Users,DC=ps,DC=test
Administrateurs d...	group	CN=Administrateurs du schéma,CN=Users,DC=ps,DC=test
Admins du domaine	group	CN=Admins du domaine,CN=Users,DC=ps,DC=test
Contrôleurs de do...	group	CN=Contrôleurs de domaine,CN=Users,DC=ps,DC=test
Contrôleurs de do...	group	CN=Contrôleurs de domaine d'entreprise en lecture seule,CN=Users,DC=ps,DC=test
Contrôleurs de do...	group	CN=Contrôleurs de domaine en lecture seule,CN=Users,DC=ps,DC=test
DnsAdmins	group	CN=DnsAdmins,CN=Users,DC=ps,DC=test
DnsUpdateProxy	group	CN=DnsUpdateProxy,CN=Users,DC=ps,DC=test
Editeurs de certi...	group	CN=Editeurs de certificats,CN=Users,DC=ps,DC=test
Groupe de répliqua...	group	CN=Groupe de répliqua... dont le mot de passe RODC est autorisé,CN=Users,DC=ps,DC=test
Groupe de répliqua...	group	CN=Groupe de répliqua... dont le mot de passe RODC est refusé,CN=Users,DC=ps,DC=test
Invité	user	CN=Invité,CN=Users,DC=ps,DC=test
Invités du domaine	group	CN=Invités du domaine,CN=Users,DC=ps,DC=test
krbtgt	user	CN=krbtgt,CN=Users,DC=ps,DC=test
Ordinateurs du do...	group	CN=Ordinateurs du domaine,CN=Users,DC=ps,DC=test
Propriétaires cré...	group	CN=Propriétaires créateurs de la stratégie de groupe,CN=Users,DC=ps,DC=test
Serveurs RAS et IAS	group	CN=Serveurs RAS et IAS,CN=Users,DC=ps,DC=test
Utilisateurs du d...	group	CN=Utilisateurs du domaine,CN=Users,DC=ps,DC=test
WinRMRemoteWMIUse...	group	CN=WinRMRemoteWMIUse...,CN=Users,DC=ps,DC=test

Une fois positionné sur le « lecteur » AD, on utilise les noms LDAP (DistinguishedName) pour se déplacer, en prenant bien soin de les mettre entre cotes `` :

Exemple :

```
PS AD:\> cd 'DC=ps,DC=test'
PS AD:\DC=ps,DC=test> dir
```

Name	ObjectClass	DistinguishedName
Builtin	builtinDomain	CN=Builtin,DC=ps,DC=test
Computers	container	CN=Computers,DC=ps,DC=test
DIRECTION	organizationalUnit	OU=DIRECTION,DC=ps,DC=test
Domain Controllers	organizationalUnit	OU=Domain Controllers,DC=ps,DC=test
EXPLOITATION	organizationalUnit	OU=EXPLOITATION,DC=ps,DC=test
ForeignSecurityPr...	container	CN=ForeignSecurityPrincipals,DC=ps,DC=test
Infrastructure	infrastructureUpdate	CN=Infrastructure,DC=ps,DC=test
LostAndFound	lostAndFound	CN=LostAndFound,DC=ps,DC=test
Managed Service A...	container	CN=Managed Service Accounts,DC=ps,DC=test
NTDS Quotas	msDS-QuotaContainer	CN=NTDS Quotas,DC=ps,DC=test
OU_COMPTA	organizationalUnit	OU=OU_COMPTA,DC=ps,DC=test
Program Data	container	CN=Program Data,DC=ps,DC=test
System	container	CN=System,DC=ps,DC=test
test	organizationalUnit	OU=test,DC=ps,DC=test
Users	container	CN=Users,DC=ps,DC=test

On retrouve les containers et les unités d'organisations présentes dans notre structure, comme dans la console Utilisateurs et Ordinateurs Active Directory.

b. Les Noms LDAP

Les noms LDAP des objets Active Directory ne sont pas compliqués, mais long.

Un objet container est identifié par CN=nom_objet,DC,nom_netbios_domaine,DC=tld_domaine :

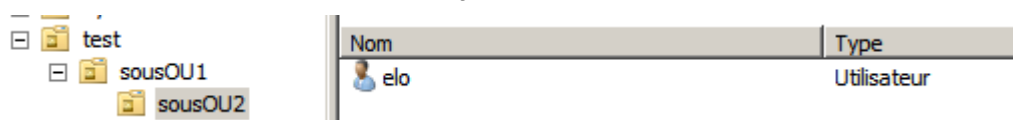
CN=Users,DC=ps,DC=test est le nom LDAP du contenair Users du domaine ps.test.

Un objet unité d'organisation est identifié par

OU=nom_objet,DC=nom_netbios_domaine,DC=tld_domaine :

OU=OU_COMPTA,DC=ps,DC=test est le nom LDAP de l'unité d'organisation OU_COMPTA du domaine ps.test.

L'identification d'un nom LDAP s'écrit toujours « à l'envers » :



CN=elo,OU=sousOU2,OU=sousOU1,OU=test,DC=ps,DC=test

On part de l'objet et on remonte.

c. Recherche LDAP

Les recherches dans l'annuaire LDAP s'organisent autour de filtre LDAP. Les plus courantes étant les recherches sur les utilisateurs, voici quelques filtres qui peuvent s'avérer pratiques.

Get-ADUser -Filter *

Recherche tous les objets utilisateurs d'un annuaire complet

Get-ADUser -LDAPFilter '(name=*elo*)'

Recherche tous les objets utilisateurs dont le nom contient elo.

Si on avait voulu chercher tous les utilisateurs dont le nom commence par elo, on aurait utilisé '(name=elo*)'.

Si on avait voulu chercher tous les utilisateurs dont le nom commence par une lettre suivi de elo, on aurait utilisé '(name= ?elo*)'.

Les caractères génériques sont en fait les mêmes que sous une basique console DOS.

Get-ADUser -Filter * -SearchBase 'OU=nomOU,DC=nom_domaine,DC=tld_domaine'

Le paramètre SearchBase permet de spécifier l'emplacement de recherche ; on peut le combiner avec les paramètres -Filter ou -LDAPFilter.

d. Exercice 2

- Créez au préalable une unité d'organisation TEST, dans laquelle vous créez 3 utilisateurs : toto, titi, tata.

Indiquez les commandes effectuées pour :

- Dans le Provider, positionnez-vous sur l'unité d'organisation TEST :
- Afficher la liste des utilisateurs :
- Remonter à la racine du Provider :
- Afficher la liste des utilisateurs de l'unité d'organisation TEST :
- Afficher la liste des utilisateurs du domaine dont le nom contient 'a' :
- Afficher la liste des utilisateurs de l'unité d'organisation dont le nom comment par 'to' :

2) Utilisateurs

La commande New-ADUser permet la création de comptes utilisateurs.

Elle peut s'utiliser quasiment sans argument :

```
New-ADUser -Name ElodieLECLERC
```

Création basique, sans aucun paramètre, sauf le paramètre Name qui est obligatoire. L'utilisateur est créé dans le container Users (emplacement par défaut).

Les champs Prénom, Nom de famille, Nom d'ouverture de session – appelé nom UPN -, et Nom d'ouverture de session (antérieur Windows 2000) – appelé nom SAM -, ne sont pas renseignés ; le compte n'ayant pas de mot de passe n'est pas activé. Dans une entreprise, l'utilisation basique de la commande New-ADUser n'est donc pas exploitable.

On va donc devoir utiliser des paramètres supplémentaires ; le nombre de paramètres de cette commande est impressionnant. Même si, au premier abord, ils peuvent paraître incompréhensibles, ils sont en fait assez parlant.

Pour obtenir la liste des paramètres de cette commande : Get-Help New-ADUser.

a. Les paramètres « name »

New-ADUser propose une série de paramètres comportant le mot « name ». Certains sont obligatoires, d'autres non. Pour donner le maximum d'information à la commande et avoir une organisation d'utilisateurs « propre », il est fortement conseillé de renseigner au moins les paramètres suivants :

Name

C'est le nom de l'objet Active Directory ; c'est celui qu'on retrouve lorsqu'on navigue dans le Provider :
CN='ElodieLECLERC',DC=ps,DC=test.

Ce paramètre est obligatoire ; à utiliser entre cotes `` dans la commande New-ADUser si le paramètre comporte un espace : New-ADUser -Name 'Elodie LECLERC'.

Surname

C'est le nom de famille de l'utilisateur.

GivenName

C'est le prénom de l'utilisateur.

SAMAccountName

C'est le nom d'ouverture de session antérieur à Windows 2000. C'est celui qui vous permet d'ouvrir une session au format DOMAINE\nom_user).

UserPrincipalName

C'est le nom d'ouverture de session principale. C'est celui qui vous permet d'ouvrir une session au format [nom_user@domaine.tld](#)).

Exercice 3

- Créez le compte utilisateur Naima MANSOURI, en respectant le cahier des charges suivants :
 - Les champs Nom et Prénom doivent être renseignés ;
 - Les noms SAM et UPN doivent être au format p.nom (n.mansouri).
- Notez la commande utilisée :
- Vérifiez en mode graphique que tous les champs sont correctement renseignés.

b. Les paramètres « password »

Pour pouvoir activer un compte utilisateur, il est obligatoire de lui attribuer un mot de passe. Pour des raisons de sécurité, le système n'accepte que les chaînes de caractères SecureString. Pour obtenir une chaîne de caractères SecureString, il faut convertir une chaîne de caractères existante. Il va donc falloir repasser par une variable ...

```
$pwd = 'P@ssword69'
```

```
$pwd = ConvertTo-SecureString $pwd -AsPlainText -Force
```

Une fois la variable créée, il suffit de s'en servir avec le paramètre -AccountPassword

AccountPassword

Ce paramètre indique donc un mot de passe (sécurisé).

A renseigner dans la commande New-ADUser, à la suite des autres paramètres, en se servant de la variable créée précédemment :

```
-AccountPassword $pwd
```

CannotChangePassword

Nom parlant : ne peut pas changer le mot de passe. C'est un paramètre qui attend un argument de type <System.Nullable[bool]>, qui est en fait un booléen (vrai ou faux). L'argument se passe en utilisant \$True ou \$False.

ChangePasswordAtLogon

Parlant aussi : doit changer le mot de passe à l'ouverture de session. Même type d'argument que le paramètre précédent : un booléen \$True ou \$False.

PasswordNeverExpires

Parlant aussi : le mot de passe n'expire jamais. Même type d'argument que le paramètre précédent : un booléen \$True ou \$False.

Exercice 4

- Supprimez le compte utilisateur que vous venez de créer (en mode graphique pour l'instant).
- Enrichissez la commande New-ADUser précédente pour créer le compte utilisateur Naima MANSOURI (qui doit toujours respecter le cahier des charges précédent), qui doit avoir un mot de passe sécurisé, et qui doit en plus avoir les paramètres suivants :
 - L'utilisateur peut changer de mot de passe
 - L'utilisateur doit changer le mot de passe à l'ouverture de session
 - Le mot de passe peut expirer
- Notez la commande utilisée :

Attention aux négations dans les valeurs booléennes !

c. Le paramètre Enable

Comme son nom l'indique, le paramètre enable active ou pas un compte utilisateur. Il attend une valeur booléenne pour pouvoir s'exécuter \$True ou \$False.

Exercice 5

- Supprimez le compte utilisateur que vous venez de créer (en mode graphique pour l'instant).
- Enrichissez la commande New-ADUser précédent pour créer le compte utilisateur Naima MANSOURI activé à la création.
- Notez la commande utilisée :

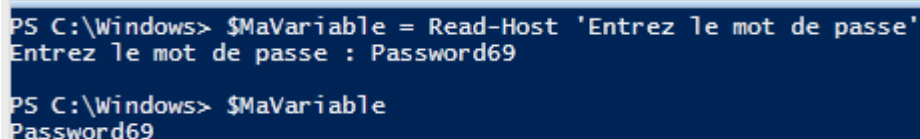
d. Read-Host

La commande Read-Host n'est pas spécifique au module Active-Directory, mais elle va nous être utile ici.

Cette commande permet de récupérer les informations que l'on entre au clavier dans une variable. Typiquement, pour le mot de passe, il est déconseillé de le noter « en dur et en clair » ; on préfère demander le mot de passe, le récupérer de la saisie clavier et le convertir en SecureString dans la foulée.

La commande Read-Host s'utilise de cette façon :

```
$MaVariable = Read-Host 'Entrez le mot de passe'
```

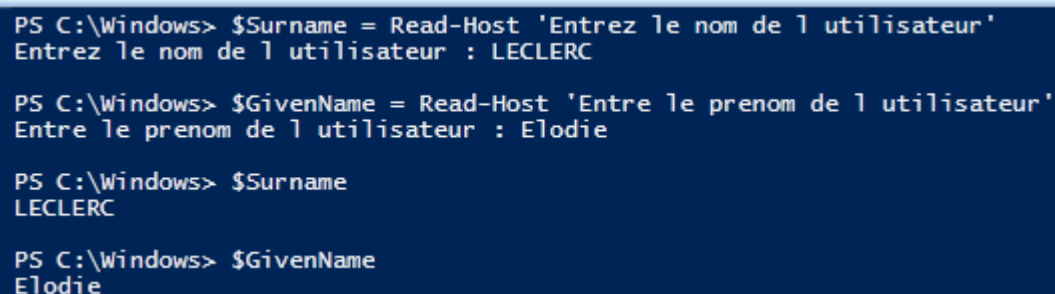


```
PS C:\Windows> $MaVariable = Read-Host 'Entrez le mot de passe'
Entrez le mot de passe : Password69

PS C:\Windows> $MaVariable
Password69
```

Ici, c'est pour l'exemple ; on s'en sert dans un script de telle sorte que le mot de passe ne soit pas renseigné directement dans le fichier de script.

Il en est de même pour toutes les variables ; on peut donc utiliser le read-host pour récupérer les noms et prénoms d'un utilisateur par exemple :



```
PS C:\Windows> $Surname = Read-Host 'Entrez le nom de l'utilisateur'
Entrez le nom de l'utilisateur : LECLERC

PS C:\Windows> $GivenName = Read-Host 'Entrez le prénom de l'utilisateur'
Entrez le prénom de l'utilisateur : Elodie

PS C:\Windows> $Surname
LECLERC

PS C:\Windows> $GivenName
Elodie
```

Ce qui va nous servir pour la suite ; en effet, pour le moment, nous n'avons pas créé de script de création d'utilisateurs, nous n'avons fait qu'utiliser la commande New-ADUser et ses paramètres, avec un nom d'utilisateur fixe. En utilisant le Read-Host et en insérant la commande New-ADUser dans un script, on devrait pouvoir récupérer le contenu des variables et les passer en arguments aux paramètres de la commande New-ADUser.

Sauf que ... (encore) ... sauf que, autant pour les paramètres Surname (nom de famille) et GivenName (prénom), ça ne posera pas de problèmes, autant pour les paramètres Name (Prénom Nom) et SAMAccountName et UserPrincipalName (p.nom), ça va se compliquer un peu. On pourrait demander à entrer tous ces paramètres via un Read-Host, mais ce serait un peu idiot (autant créer l'utilisateur en mode graphique). Il va donc falloir trouver un moyen pour créer de nouvelles variables qui reprennent une partie des informations des variables existantes et les combine avec des espaces et des points.

Exercice 6

Le contenu d'une partie d'une variable qui contient des caractères peut être obtenu de la même façon que lorsqu'on cherche à obtenir le contenu d'une case d'un tableau.

- Créer un script qui demande à l'utilisateur d'entrer un mot et qui n'affiche que la 1^{ère} lettre de ce mot.
- Créer un script qui demande à l'utilisateur d'entrer 2 mots et qui affiche la concaténation de ces 2 mots séparés par un espace.
- Créer enfin un script qui demande à l'utilisateur d'entrer 2 mots et qui affiche la concaténation de la 1^{ère} lettre du 1^{er} mot, suivi d'un point, suivi du 2^{ème} mot.

Exercice 7

En reprenant la commande New-ADUser et tous ses paramètres, créez un script de création d'utilisateur qui demande :

- D'entrez un mot de passe
- D'entrez le nom de l'utilisateur
- D'entrez le prénom de l'utilisateur

Et qui crée l'utilisateur en respectant le cahier des charges initial.

Boucle Do-While

La boucle Do-While est quasi la même que la boucle While, sauf que la condition n'est pas testée au début, mais à la fin. La syntaxe est la suivante :

Do

{

#instructions

}

While (condition)

Fréquemment utilisée, en combinant avec le read-host pour poser la question : voulez-vous continuer O/N ?

Exercice 8

- Modifiez votre script de création d'utilisateur en incluant vos instructions dans une boucle Do-While.
- Tant que vous répondez O (oui) à la question voulez-vous continuer ?, le script vous redemande d'entrer un mot de passe, un nom et prénom d'utilisateur ; si vous répondez N (non), le script s'arrête.

e. Import de fichier csv

On a vu précédemment comment créer automatiquement des utilisateurs avec la commande New-ADUser et l'entrée de paramètres.

Il existe une autre méthode pour créer automatiquement les comptes utilisateurs : l'import de fichier csv. Cette méthode est utilisée pour importer massivement des utilisateurs ; si vous n'avez que 2 ou 3, préférez la méthode précédente.

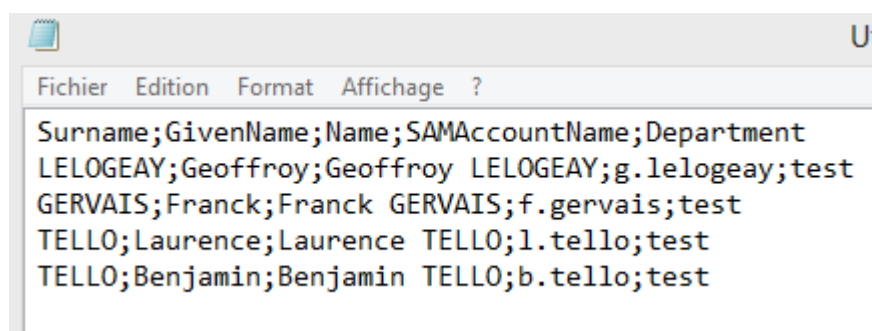
Un fichier csv est un simple fichier texte, dont les valeurs sont séparées par des points-virgules (ou des virgules en environnement de langue anglaise) – sans espace. La 1^{ère} ligne correspond à l'en-tête (le nom des colonnes) et les lignes suivantes au contenu :

```
Surname ;GivenName ;Name
LECLERC ;Elodie ;Elodie LECLERC
PESTRE ;Hugues ;Hugues PESTRE
GOURRU ;Benjamin ;Benjamin GOURRU
```

Un fichier csv peut être créé directement via Notepad, ou, plus simple, avec Excel ; il suffit de créer un tableau et de l'enregistrer au format CSV (DOS) :

A	B	C	D	E
Surname	GivenName	Name	SAMAccountName	Department
LELOGEAY	Geoffroy	Geoffroy LELOGEAY	g.lelogeay	test
GERVAIS	Franck	Franck GERVAIS	f.gervais	test
TELLO	Laurence	Laurence TELLO	l.tello	test
TELLO	Benjamin	Benjamin TELLO	b.tello	test

Qui se traduit par :



Import-csv

La commande import-csv « lit » le fichier csv qu'on lui indique et le traite dans la commande qu'on lui indique via un pipe.

Pour que l'import fonctionne, il faut impérativement que les noms de l'en-tête soit identiques aux paramètres de la commande qu'on souhaite faire passer dans le pipe.

Par exemple, si on reprend les copies d'écran ci-dessus ; je souhaite utiliser la commande New-ADUser ; les paramètres de cette commande sont, entre autres, Surname, GivenName, etc ... Mon fichier csv doit impérativement avoir un en-tête qui correspond aux paramètres de la commande New-ADUser.

La commande s'utilise de cette façon :

Import-csv chemin_nom_csv

Pour être sûr d'interpréter correctement le fichier :

Import-csv chemin_nom_csv -delimiter ';' ;'

Exercice 9

- Créer un fichier Excel qui comporte en en-tête Surname, Name, GivenName, SAMAccountName et UserPrincipalName ; puis renseignez 4 ou 5 utilisateurs dans ce fichier.

Optionnel : Excel offre une panoplie de commandes qui permettent, entre autres, de concaténer des cellules, de ne récupérer qu'une partie d'une cellule (la 1^{ère} lettre par exemple), de passer une cellule tout en minuscules ou tout en majuscules. Tant qu'à faire, utilisez ces formules pour générer un fichier « propre ». Vous aurez besoin de stxt, &, minuscule() et majuscule().

- Une fois le fichier créé, passez le en csv et vérifiez votre csv via la commande import-csv et le paramètre delimiter.
- Notez la commande utilisée :

Import-csv | commande

Une fois le fichier csv vérifié, il ne reste plus qu'à combiner la commande import-csv à la commande new-ADUser pour créer nos utilisateurs à la volée.

Comme tous les paramètres concernant « name » sont renseignés dans le fichier, il ne reste plus qu'à entrer dans le script les paramètres concernant le « password » et le paramètre Enable.

La syntaxe est simple : import-csv fichier.csv -delimiter ';' | New-ADUser -Password \$password etc ...

Exercice 10

- Créer un script qui importe des utilisateurs via un fichier csv, en respectant le cahier des charges initial.

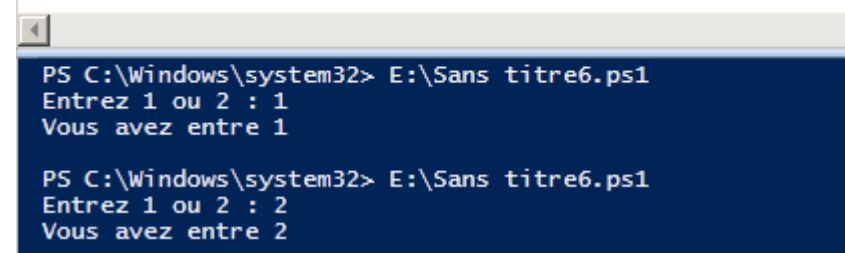
Switch

L'instruction Switch permet d'orienter l'exécution d'une partie d'un script vers tel ou tel bloc d'instructions. L'exécution d'une partie du script est conditionné par le choix de l'utilisateur. La syntaxe est la suivante :

Switch (expression)

```
{
    Valeur_1 {bloc instruction 1}
    Valeur_2 {bloc instruction 2}
}
```

```
1 $valeur = Read-Host 'Entrez 1 ou 2'
2
3 switch ($valeur)
4 {
5     1 {Write-Host 'Vous avez entre 1'}
6     2 {Write-Host 'Vous avez entre 2'}
7 }
```



```
PS C:\Windows\system32> E:\Sans titre6.ps1
Entrez 1 ou 2 : 1
Vous avez entre 1

PS C:\Windows\system32> E:\Sans titre6.ps1
Entrez 1 ou 2 : 2
Vous avez entre 2
```

Exercice 11

- Reprenez les 2 scripts précédents (création d'utilisateur avec new-ADUser et import d'utilisateur via csv) pour n'en former qu'un.
- Utilisez le switch pour demande au début du script ce que vous souhaitez faire :
 - 1 : création d'utilisateur 1 à 1
 - 2 : création d'utilisateur via csv

3) Groupes

La création de groupes globaux, universels ou de domaine locaux passe par la commande New-ADGroup.

Selon le même principe que la commande new-ADUser, la commande New-ADGroup possède plusieurs paramètres :

- **Name** : c'est le nom de l'objet
- **DisplayName** : c'est le nom d'affichage de l'objet
- **GroupCategory** : Security ou Distribution (rappel : pour pouvoir poser des permissions NTFS, il faut créer des groupes de sécurité)
- **GroupScope** : Global, DomainLocal ou Universal

a. Exercice 12

- Créez un groupe global, appelé GG_TEST01 (nom et nom d'affichage).
- Notez la commande utilisée :

b. Exercice 13

- Créez un script qui :
 - Demande d'entrer le nom d'un service (exemple : compta)
 - Crée le groupe global associé au format GG_Compta
 - Demande si on veut continuer (O/N)

c. Import-csv et boucle ForEach

L'import massif de groupes via un fichier csv ne fonctionne pas comme l'import massif d'utilisateurs. On ne peut pas importer un fichier et passer la commande New-ADGroup via le pipe ; il faut utiliser une boucle ForEach.

Boucle ForEach

Cette boucle sert à parcourir ce qu'on appelle une collection de données ; un fichier csv par exemple, peut-être considéré comme une collection de données (ça tombe bien).

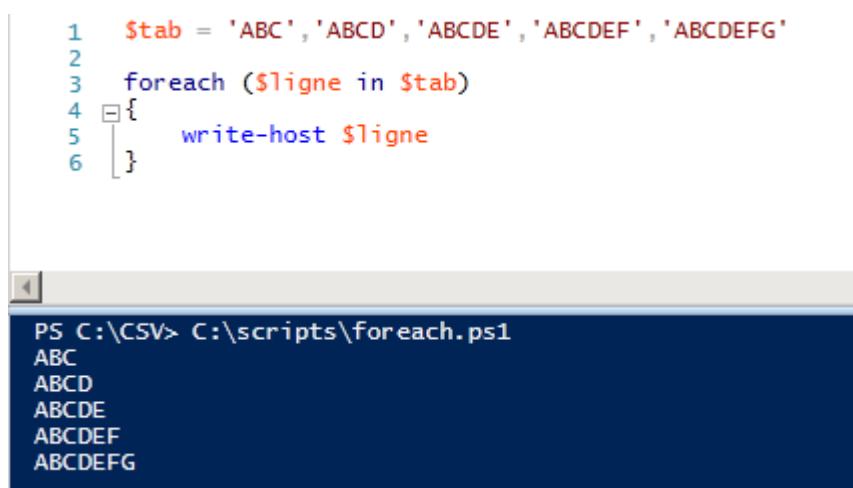
La syntaxe est la suivante :

Foreach (element in collection)

```
{  
    #instructions  
}
```

Le principe, c'est que « element » est une variable, qu'on n'a pas besoin de déclarer au préalable, et qui n'existe que dans la boucle foreach ; l'élément « collection » est aussi une variable, cette fois-ci déclarée au préalable, et qui correspond donc à une « collection », c'est-à-dire un tableau ou un fichier csv.

Un exemple étant toujours plus parlant :



```
1 $tab = 'ABC', 'ABCD', 'ABCDE', 'ABCDEF', 'ABCDEFG'  
2  
3 foreach ($ligne in $tab)  
4 {  
5     write-host $ligne  
6 }
```

PS C:\CSV> C:\scripts\foreach.ps1
ABC
ABCD
ABCDE
ABCDEF
ABCDEFG

Ma collection est la variable \$tab, un tableau qui comporte 5 cases (ou lignes)

J'utilise ensuite Foreach, qui, par principe, sait qu'elle va lire chaque élément de ma collection. La variable \$ligne est donc mon élément. Cette variable \$ligne n'a pas été déclarée au préalable, elle n'existe que pour la boucle Foreach.

Je demande donc au script à ce que pour chaque élément (\$ligne) de mon tableau (\$tab), il affiche le contenu de cet élément.

Mais je pourrais tout aussi bien utilisé les propriétés de ma variable \$ligne et demander à afficher la longueur de mon élément (en utilisant la propriété .length) :

```

1  $stab = 'ABC', 'ABCD', 'ABCDE', 'ABCDEF', 'ABCDEFG'
2
3  foreach ($ligne in $stab)
4  {
5      write-host $ligne.length
6  }

```

```

PS C:\CSV> C:\scripts\foreach.ps1
3
4
5
6
7
PS C:\CSV> |

```

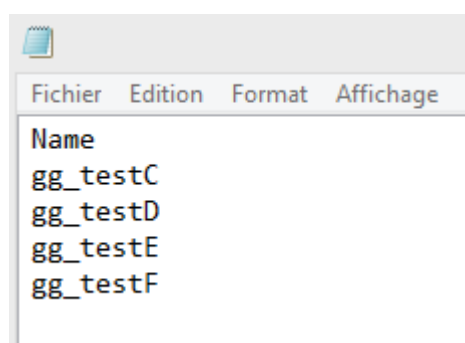
Pour traiter un fichier csv (une collection donc), il suffit de l'importer avant dans une variable :

```

1  set-location C:\CSV
2  $MesGroupes = import-csv 'globaux1.csv'
3
4  foreach ($ligne in $MesGroupes)
5  {
6      write-host $ligne.Name
7  }
8
PS C:\CSV> C:\scripts\exempleforeach.ps1
gg_testC
gg_testD
gg_testE
gg_testF

```

Le fichier csv est le suivant :



Je l'importe dans la variable \$MesGroupes

Et ensuite, pour chaque élément de ce fichier (\$ligne), je demande à afficher le contenu (\$ligne.Name).

On aurait pu obtenir le même résultat en utilisant les structures conditionnelles If et If-Else ; le gros avantage de la boucle ForEach, c'est qu'on n'a pas besoin de connaître ou de calculer la longueur de nos tableaux (ou fichiers) pour balayer toutes les lignes.

Exercice 14

- Créez un fichier csv qui ne contient qu'un seul champ : « Name » et une dizaine de lignes (GG_TEST01 à GG_TEST10 par exemple).
- Créez un script qui importe le fichier csv et qui, pour chaque ligne, crée un groupe global, en utilisant Foreach.

Exercice 15

- Reprenez les 2 scripts précédents (création de groupes globaux avec new-ADUser et import de groupes via csv) pour n'en former qu'un.
 - Utilisez le switch pour demander au début du script ce que vous souhaitez faire :
 - 1 : création de groupe 1 à 1
 - 2 : création de groupe via csv
-

La création de groupes de domaine locaux et de groupes universels suit exactement le même principe. Il suffit de remplacer la paramètre -GlobalScope par DomainLocal ou Universal.

d. Exercice 16 (synthèse utilisateurs – groupes globaux)

En combinant vos scripts de création d'utilisateurs et de création de groupes globaux, créez un script qui suit l'organigramme ci-après.

En plus des commandes, structures conditionnelles et boucles que vous avez déjà utilisé, vous allez avoir besoin :

- De la commande Add-ADGroupMember : pour ajouter des utilisateurs à un groupe.
- D'utiliser les filtres de recherche LDAP (cf. 1^{ère} partie de ce document) : pour chercher tous les groupes contenant le nom du service et les afficher.
- D'appeler le script de création de groupe : pour créer un groupe s'il n'en n'existe aucun. Pour appeler un script à l'intérieur d'un script, il suffit d'indiquer son emplacement et son nom.

Bon courage !

