

Module 5: Gestion des usagers locaux avec powershell

Table des matières

Module 5: Gestion des usagers locaux avec powershell.....	1
Faire la gestion des usagers locaux avec Powershell.....	2
Établir la liste des utilisateurs et de leurs propriétés avec PowerShell.....	3
Créer un utilisateur local avec PowerShell.....	5
Modifier le mot de passe d'un utilisateur local ou les propriétés du mot de passe avec PowerShell.....	6
Supprimer un compte d'utilisateur local avec PowerShell.....	6
Gérer les groupes locaux avec PowerShell.....	7
Consulter les groupes locaux avec PowerShell.....	7
Ajouter un groupe local avec PowerShell.....	7
Ajouter des utilisateurs à un groupe local avec PowerShell.....	7
Afficher les membres d'un groupe particulier avec PowerShell.....	8
Afficher tous les groupes dont un utilisateur est membre avec PowerShell.....	8
Enlever un usager d'un groupe local avec PowerShell.....	9
Gérer les utilisateurs et groupes locaux à distance avec PowerShell.....	9
Gestion des usagers Active Directory avec PowerShell.....	11
DN ("Distinguished Name" ou nom unique).....	11
Exercice DN.....	12
Obtenir un usager Active Directory.....	13
Get-ADUser.....	13
Exemple.....	13
Ajouter un usager Active Directory.....	13
New-ADUser.....	13
Exemple.....	13
Modifier un usager Active Directory.....	14
Set-ADUser.....	14
Exemple.....	14
Supprimer un usager Active Directory.....	15
Remove-ADUser.....	15
Exemple.....	15
Ajouter une UO (unité organisationnelle) (OU).....	15
New-ADOrganizationalUnit.....	15
Exemple.....	15
Supprimer une UO (unité organisationnelle).....	16
Remove-ADOrganizationalUnit.....	16
Exemple.....	16

Faire la gestion des usagers locaux avec Powershell

Pour que les administrateurs puissent gérer plus facilement les utilisateurs et groupes locaux avec PowerShell, Microsoft propose une collection de cmdlets appelée **Microsoft.PowerShell.LocalAccounts**. Auparavant, il fallait la télécharger et l'importer dans PowerShell, et aussi installer Windows Management Framework 5.1 ; dans les systèmes d'exploitation Windows Server 2016 et Windows 10, la collection de cmdlets est incluse en tant que module standard.

Le module **LocalAccounts** contient 15 cmdlets. Pour voir la liste complète, exécutez la commande suivante :

```
Get-Command -Module Microsoft.PowerShell.LocalAccounts
```

```
PS C:\WINDOWS\system32> Get-Command -Module Microsoft.PowerShell.LocalAccounts
```

CommandType	Name	Version	Source
Cmdlet	Add-LocalGroupMember	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Disable-LocalUser	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Enable-LocalUser	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Get-LocalGroup	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Get-LocalGroupMember	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Get-LocalUser	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	New-LocalGroup	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	New-LocalUser	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Remove-LocalGroup	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Remove-LocalGroupMember	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Remove-LocalUser	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Rename-LocalGroup	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Rename-LocalUser	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Set-LocalGroup	1.0.0.0	Microsoft.PowerShell.Loc
Cmdlet	Set-LocalUser	1.0.0.0	Microsoft.PowerShell.Loc

```
PS C:\WINDOWS\system32> Get-Command -Module Microsoft.PowerShell.LocalAccounts | Measure-Object
```

```
Count      : 15
```

- **Add-LocalGroupMember** — Ajouter un utilisateur au groupe local
- **Disable-LocalUser** — Désactiver un compte d'utilisateur local
- **Enable-LocalUser** — Activer un compte d'utilisateur local
- **Get-LocalGroup** — Afficher les préférences du groupe local

- **Get-LocalGroupMember** — Afficher la liste de tous les membres du groupe local
- **Get-LocalUser** — Afficher les préférences d'un compte d'utilisateur local
- **New-LocalGroup** — Créer un nouveau groupe local
- **New-LocalUser** — Créer un nouveau compte d'utilisateur local
- **Remove-LocalGroup** — Supprimer un groupe local
- **Remove-LocalGroupMember** — Supprimer un membre d'un groupe local
- **Remove-LocalUser** — Supprimer un compte d'utilisateur local
- **Rename-LocalGroup** — Renommer un groupe local
- **Rename-LocalUser** — Renommer un compte d'utilisateur local
- **Set-LocalGroup** — Modifier les paramètres d'un groupe local
- **Set-LocalUser** — Modifier les paramètres de compte d'un utilisateur local

Voyons comment utiliser ces commandes pour effectuer des tâches courantes de gestion des utilisateurs locaux sur un ordinateur sous Windows 10.

Établir la liste des utilisateurs et de leurs propriétés avec PowerShell

Commençons par obtenir la liste de tous les comptes d'utilisateur locaux de la machine. Pour cela, nous utilisons le cmdlet **Get-LocalUser** :

```
Get-LocalUser
```

```
PS C:\Windows\system32> Get-LocalUser

Name           Enabled Description
-----
Administrator True    Built-in account for administering the computer/domain
Guest          False   Built-in account for guest access to the computer/domain
```

Comme vous le voyez, nous avons deux comptes d'utilisateur locaux, dont l'un est désactivé (celui pour lequel est indiqué « False » (Faux) dans la colonne « Enabled » (Activé)).

Si vous voulez obtenir toutes les propriétés et leurs valeurs pour un compte d'utilisateur local, utilisez le cmdlet **Get-LocalUser** avec les paramètres suivants :

```
Get-LocalUser -Name 'guest' | Select-Object *
```

```
PS C:\Windows\system32> Get-LocalUser -Name 'guest' | Select-Object *

AccountExpires      :
Description         : Built-in account for guest access to the computer/domain
Enabled            : False
FullName           :
PasswordChangeableDate :
PasswordExpires    :
UserMayChangePassword : False
PasswordRequired   : False
PasswordLastSet    :
LastLogon          :
Name               : Guest
SID                : S-1-5-21-3404964596-197790689-1311383245-501
PrincipalSource    :
ObjectClass        : User
```

Pour obtenir la valeur d'un attribut de compte d'utilisateur local particulier, saisissez son nom après le paramètre `Select-Object`. Dans cet exemple, nous souhaitons connaître la valeur de l'attribut `PasswordLastSet` du compte dont le nom d'utilisateur est « administrator » :

```
Get-LocalUser -Name 'administrator' | Select-Object PasswordLastSet
```

```
PS C:\Windows\system32> Get-LocalUser -Name 'administrator' | Select-Object PasswordLastSet

PasswordLastSet
-----
1/3/2018 1:51:09 PM
```

Créer un utilisateur local avec PowerShell

Créons maintenant un nouvel utilisateur à l'aide du cmdlet **New-LocalUser**. Ce cmdlet peut créer les types de comptes d'utilisateur suivants :

- Comptes d'utilisateur locaux Windows
- Comptes Microsoft
- Comptes Azure Active Directory

Lorsque vous créez un compte d'utilisateur local, ne tapez jamais le mot de passe en texte simple ; convertissez-le toujours en une chaîne sécurisée à l'aide du paramètre "**AsSecureString**" ou "**ConvertTo-SecureString**". La commande suivante permet de créer un nouveau compte d'utilisateur local :

```
$UserPassword = Read-Host -AsSecureString
```

```
New-LocalUser "nouvelusager" -Password $UserPassword -FullName "Le nouvel usager"  
-Description "CompleteVisibility"
```

Dans un environnement Windows 10, les utilisateurs peuvent accorder des autorisations depuis leurs comptes Microsoft, ce qui permet de créer un nouveau compte d'utilisateur local lié aux informations d'identification d'un compte Microsoft. À cette fin, utilisez le script suivant (notez que vous n'avez pas besoin de saisir le mot de passe car il est stocké dans le Cloud Microsoft) :

```
New-LocalUser -Name "MicrosoftAccount\SomeAccount@outlook.com" -Description  
"Microsoft Account"
```

Pour créer un compte local lié à votre Azure AD, utilisez la commande suivante :

```
New-LocalUser -Name "AzureAD\usager@enterprise.com" -Description "Azure AD  
Account"
```

Modifier le mot de passe d'un utilisateur local ou les propriétés du mot de passe avec PowerShell

Pour changer le mot de passe d'un compte d'utilisateur local, utilisez le cmdlet **Set-LocalUser**. Pour changer le mot de passe de l'administrateur local :

```
$UserPassword = Read-Host -AsSecureString
```

```
Set-LocalUser -Name Administrator -Password $UserPassword -Verbose
```

Pour définir avec PowerShell que le **mot de passe d'un utilisateur local ne doit jamais expirer**, exécutez le script suivant :

```
Set-LocalUser -Name nom_usager -PasswordNeverExpires $False
```

Supprimer un compte d'utilisateur local avec PowerShell

Pour supprimer un compte d'utilisateur local, utilisez le cmdlet **Remove-LocalUser** :

```
Remove-LocalUser -Name nom_compte -Verbose
```

Gérer les groupes locaux avec PowerShell

Après nous être intéressés aux utilisateurs locaux, penchons-nous maintenant sur les groupes locaux.

Consulter les groupes locaux avec PowerShell

Commençons par établir une liste de tous les groupes sur notre Windows Server :

```
Get-LocalGroup
```

```
PS C:\Windows\system32> Get-LocalGroup

Name                                     Description
----                                     -
Access-Denied Assistance Users          Members of this group are provided access-denied assistance whe
HelpLibraryUpdaters
Message Capture Users                   Users belonging to this group can capture messages using Micros
NetWrix Account Help Desk               Users allowed to access Account Lockout Examiner
Netwrix Auditor Administrators          Members of this group are allowed to set up and modify all Netw
Netwrix Auditor Client Users           Members of this group are allowed to access the audit data coll
WinRMRemoteWMIUsers__                  Members of this group can access WMI resources over management
Access Control Assistance Operators     Members of this group can remotely query authorization attribut
Administrators                          Administrators have complete and unrestricted access to the com
Backup Operators                        Backup Operators can override security restrictions for the sol
Certificate Service DCOM Access         Members of this group are allowed to connect to Certification A
Cryptographic Operators                 Members are authorized to perform cryptographic operations.
Distributed COM Users                   Members are allowed to launch, activate and use Distributed COM
Event Log Readers                       Members of this group can read event logs from local machine
Guests                                  Guests have the same access as members of the Users group by de
```

Ajouter un groupe local avec PowerShell

Pour créer un nouveau groupe :

```
New-LocalGroup -Name 'Nom_du_groupe' -Description 'Le groupe super'
```

Ajouter des utilisateurs à un groupe local avec PowerShell

Pour ajouter un utilisateur (ou un groupe) à un groupe local, utilisez le cmdlet **Add-LocalGroupMember**. Supposez par exemple que vous vouliez ajouter des utilisateurs au groupe local Administrators, mais sans avoir à les ajouter un par un. Pour ajouter le groupe « users » au groupe local Administrators :

```
Add-LocalGroupMember -Group 'Administrators' -Member ('usager', 'users') -Verbose
```

Si votre ordinateur ou votre serveur fait partie du domaine, vous pouvez également ajouter un compte et des groupes de domaine aux groupes locaux pour accorder à ces utilisateurs des droits locaux spéciaux sur le serveur. Ajoutez-les au format « DomainName\User » (pour un utilisateur) ou « DomainName\Domain Group » (pour un groupe).

Afficher les membres d'un groupe particulier avec PowerShell

Pour établir la liste de tous les membres d'un groupe local particulier :

```
Get-LocalGroupMember -Group 'nom_du_groupe'
```

```
PS C:\windows\system32> Get-LocalGroupMember administrateurs
ObjectClass Name PrincipalSource
-----
Utilisateur DESKTOP-0JV00DL\Administrateur Local
Utilisateur DESKTOP-0JV00DL\Stephane Chasse Local
```

Comme vous le voyez, cette commande affiche tous les comptes et groupes locaux qui sont membres du groupe « administrateurs ». Bien que seuls les comptes et groupes locaux soient mentionnés dans cette liste, cette commande affiche également tous les utilisateurs et groupes de domaine, ainsi que tous les comptes Microsoft et Azure AD.

Afficher tous les groupes dont un utilisateur est membre avec PowerShell

Pour établir la liste de tous les groupes dont un utilisateur particulier est membre, exécutez le script suivant : (Exemple avec l'utilisateur "guest")

```
foreach ($LocalGroup in Get-LocalGroup)
{
    if (Get-LocalGroupMember $LocalGroup -Member 'Guest' -ErrorAction SilentlyContinue)
```

```
{  
  
$LocalGroup.Name  
  
}  
  
}
```

```
PS C:\Windows\system32> foreach ($LocalGroup in Get-LocalGroup)  
if (Get-LocalGroupMember $LocalGroup -Member 'Guest' -ErrorAction SilentlyContinue)  
$LocalGroup.Name  
Netwrix Users  
Guests
```

Enlever un usager d'un groupe local avec PowerShell

Pour supprimer un compte d'utilisateur local d'un groupe, utilisez le cmdlet **Remove-LocalGroupMember** :

```
Remove-LocalGroupMember -Group 'nom_du_groupe -Member 'nom_compte'
```

Gérer les utilisateurs et groupes locaux à distance avec PowerShell

Pour gérer à distance les comptes d'utilisateur et groupes locaux, connectez-vous aux postes de travail distants via WinRM en utilisant les cmdlets **Invoke-Command** et **Enter-PSSession**. Par exemple, pour obtenir à distance les membres du groupe d'administrateurs local sur plusieurs ordinateurs, exécutez le script suivant :

```
$search = new-pssession -computer pcname1,pcname2,pcname3
```

```
invoke-command -scriptblock {Get-LocalGroupMember -Group 'Administrators'} -session  
$search -hidecomputername | select * -exclude RunspaceID | out-gridview -title "LocalAdmins"
```

Gestion des usagers Active Directory avec PowerShell¹

DN (“Distinguished Name” ou nom unique)

Un chemin hiérarchique, dans Active Directory, est appelé **DN** (“Distinguished Name”) et peut être utilisé pour référencer de manière unique un objet. Le **DN** d’un usager, d’un groupe d’usager ou d’un ordinateur est constitué de son **CN** (Common Name, nom commun) suivi de l’**OU** (Organizational Unit, unité d’organisation) suivi d’un ou plusieurs **DC** (Domain Components, composantes du domaine). Les **DC** contiennent les différentes parties du nom de domaine (une pour chaque étiquette), dans l’ordre.

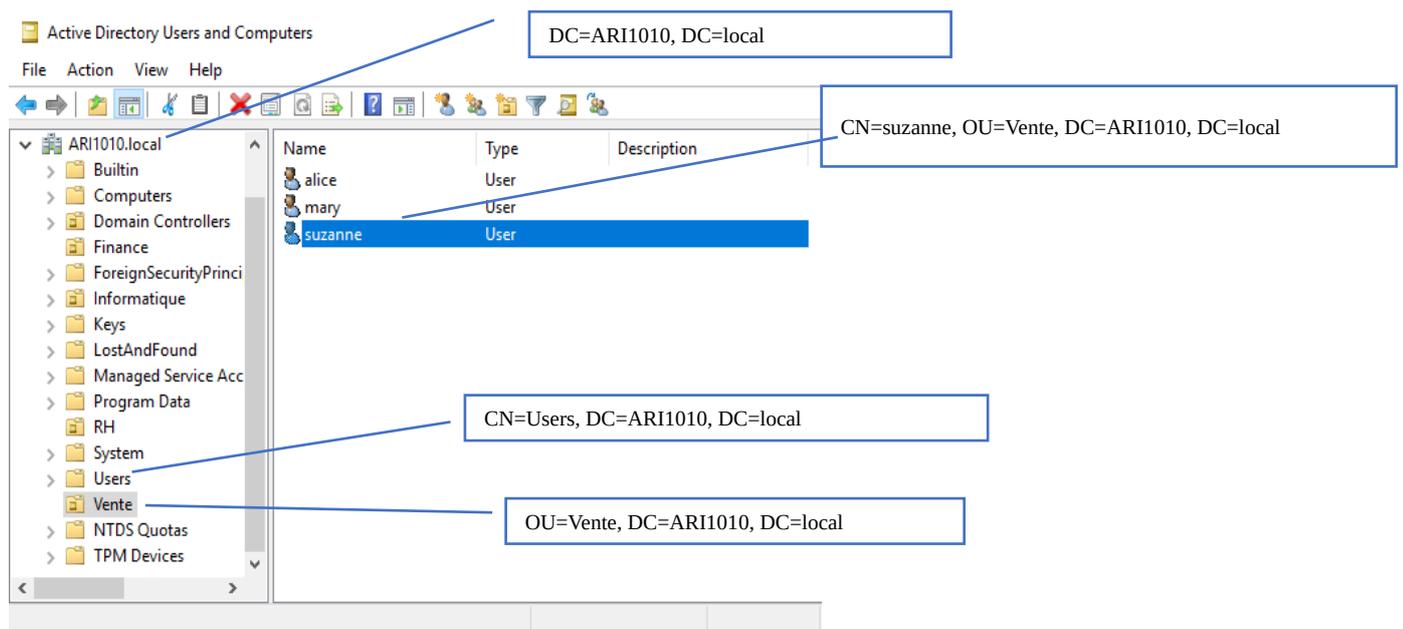
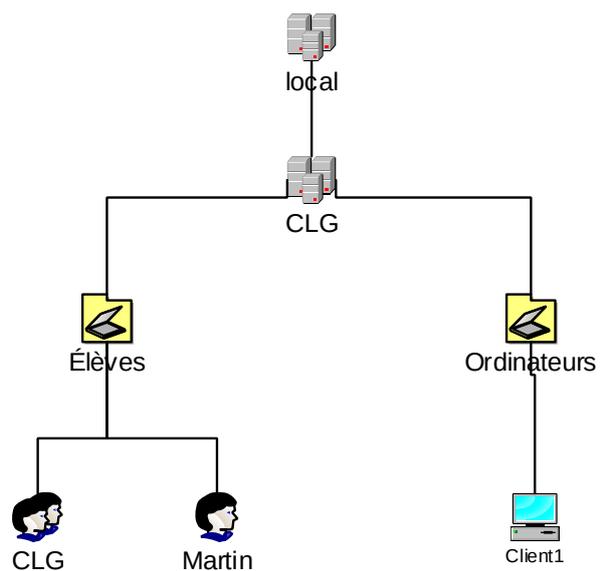


Figure 1 : Différents objets Active Directory avec leur DN (Distinguished Name)

1 Portion de ces notes développées par M. Baulne, Collaboration: J. Beaudet et R. Kadouche, Modification: Stéphane Chassé

Exercice DN



Quel est le DN de l'utilisateur Martin ?

Quel est le DN du groupe CLG ?

Obtenir un usager Active Directory

Get-ADUser

Exemple

```
Get-ADUser -filter *
```

On peut lister tous les comptes dans un OU

```
Get-ADUser -Filter * -SearchBase "OU=Elevés, DC=CLG, DC=local"
```

```
Get-ADUser [-Identity] <ADUser>
```

-Identity peut prendre les valeurs suivantes:

- DN distinguished name
- GUID (objectGUID)
- A security identifier (objectSid)
- A SAM account name (sAMAccountName)

Autre exemple: On veut afficher les informations du compte AD de "albert"

```
get-aduser albert
```

Ajouter un usager Active Directory

New-ADUser

Par défaut les usagers sont ajoutés dans le conteneur système **Users**. Vous pouvez utiliser l'option **-Path** pour spécifier une UO (unité organisationnelle) particulière.

Exemple

```
$mot_de_passe = 'pass1234'
$mot_de_passeSS = ConvertTo-SecureString -String $mot_de_passe `
    -AsPlainText -Force

$attr_communs = @{}
$attr_communs.AccountPassword = $mot_de_passeSS
$attr_communs.Enabled = $true
$attr_communs.ChangePasswordAtLogon = $true

$attr_usager_base = @{}
$attr_usager_base.Name = 'tomtom'

New-ADUser @attr_communs @attr_usager_base `
    -Path 'OU=Finance, DC=ARI1010, DC=local' `
    -DisplayName 'The king'
```

On peut également réaliser l'exemple précédent avec le path intégré au dictionnaire comme ceci:

```
$mot_de_passe = 'pass1234'  
$mot_de_passeSS = ConvertTo-SecureString -String $mot_de_passe `   
-AsPlainText -Force
```

```
$attr_communs = @{}  
$attr_communs.AccountPassword = $mot_de_passeSS  
$attr_communs.Enabled = $true  
$attr_communs.ChangePasswordAtLogon = $true  
$attr_communs.Path = "OU=Finance, DC=CLG, DC=local"
```

```
New-ADUser @attr_communs @attr_usager_base
```

Note:

On pourrait également bâtir la variable `$attr_communs.Path` avec des variables qui seraient lues d'un fichier CSV comme ci-dessous:

```
$attr_communs.Path = "OU=" + $ligneFichiercsv.nom_ou + "  
" + $ligneFichiercsv.domaine
```

Modifier un usager Active Directory

Set-ADUser

Exemple

```
$attr_usager_supp = @{}  
  
$attr_usager_supp.GivenName = "Thomas"  
  
#La ligne précédente peut également se faire en utilisant une variable  
#qui aurait été lue depuis un fichier csv comme ci-dessous:  
  
$attr_usager_supp.GivenName = $ligneFichiercsv.nom  
  
$attr_usager_supp.Surname = "Garneau"  
$attr_usager_supp.OfficePhone = "1234567890"  
  
$nom_unique = "CN=tomtom" + ", " + "OU=Finance, DC=CLG, DC=local"  
  
Set-ADUser $nom_unique @attr_usager_supp
```

Note:

Il est important ici de bâtir la variable qui spécifie le nom unique de l'usager en question avec son DN complet. (Voir variable `$nom_unique` ci-dessus)

Supprimer un usager Active Directory

Remove-ADUser

Exemple

```
Remove-ADUser -Identity $nom_unique -Confirm $false  
OU  
Remove-ADUser $nom_unique -Confirm $false
```

Note:

Il est important ici de bâtir la variable qui spécifie le nom unique de l'utilisateur en question avec son DN complet. (Voir variable \$nom_unique de l'exemple de Set-ADUser)

Ajouter une UO (unité organisationnelle) (OU)

New-ADOrganizationalUnit

Exemple

```
$attr_ou = @{}  
$attr_ou.Name = "Finance"  
$attr_ou.DisplayName = "Finance"  
$attr_ou.Path = "DC=CLG, DC=local"
```

```
New-ADOrganizationalUnit @attr_ou `
    -ProtectedFromAccidentalDeletion $false
```

Note:

On peut évidemment utiliser une variable qui serait lue d'un fichier CSV pour assigner les éléments du dictionnaire.
Exemple:
\$attr_ou.Name = \$ligneFichiercsv.nom_ou

Supprimer une UO (unité organisationnelle)

Remove-ADOrganizationalUnit

Exemple

```
$nom_unique = "OU=" + $attr_uo.Name + ", " + $attr_uo.Path
```

```
Remove-ADOrganizationalUnit -Identity $nom_unique `
                             -Confirm $false
```