**PowerShell - WMI - CIM**

Ce laboratoire doit être fait individuellement sur votre serveur réel

**Objectifs**

* Maîtriser l’accès aux objets CIM en utilisant PowerShell
* Utiliser les cmdlets Format-Table et Format-List

note: -AutoSize et -Wrap sont deux paramètres du cmdlet Format-Table

**Documentation:**

* Le fichier "C:\\_Outils\documentation - PowerShell 4.0\Quick\_Reference\_WMI\_ITPro\_WS12R2.pdf"
* Sur internet le site https://technet.microsoft.com

Voici trois raisons de ne plus utiliser les cmdlets WMI

Raison 1: Les cmdlets CIM utilisent WS-MAN pour les accès distants.

Les cmdlets WMI utilisent DCOM qui est bloqué par défaut dans le "Pare-feu Windows".

Raison 2: Les cmdlets CIM effectuent du travail à notre place, voir l'exemple 1

**Exemple 1 - Avantage de CimInstance versus WmiObject**

**# Utilisation de Win32\_OperatingSystem pour afficher la date d'installation du système d'exploitation**

**# Retourne la date d'installation du système d'exploitation**

**$rep1 = (Get-WmiObject Win32\_OperatingSystem).InstallDate**

$rep1 contient "20160910092428.000000-240"

**# Utilisation de Win32\_OperatingSystem pour afficher la date d'installation du système d'exploitation**

**# Retourne la date d'installation du système d'exploitation**

**$rep2 = (Get-CimInstance Win32\_OperatingSystem).InstallDate**

$rep2 contient "10 septembre 2016 09:24:28"

**Conclusion: La date est plus facile à lire avec Get-CimInstance.**

**Exemple 2 - Utilisation de Get-CimInstance**

**Exemple de l'utilisation de Get-CimInstance avec la classe Win32\_OperatingSystem pour afficher la description d'un ordinateur.**

**Get-CimInstance -ClassName Win32\_OperatingSystem | Select-Object Description**

**# Voici trois façons de vérifier la description d'un ordinateur**

**Get-CimInstance -Query "Select \* from Win32\_OperatingSystem where Description ='ORDI1'"**

**Get-CimInstance -ClassName Win32\_OperatingSystem -Filter "Description ='ORDI1'"**

**Get-CimInstance -ClassName Win32\_OperatingSystem | Where-Object Description -eq "ORDI1"**

**Exemple 3 - Utilisation de Set-CimInstance**

**Exemple de l'utilisation de Set-CimInstance avec la classe Win32\_OperatingSystem pour modifier la description d'un ordinateur.**

**# EXEMPLE 1**

**$nom = Get-CimInstance -ClassName Win32\_OperatingSystem**

**$nom.Description = "ORDI1"**

**Set-CimInstance -CimInstance $nom**

**# EXEMPLE 2**

**$nom = Get-CimInstance -ClassName Win32\_OperatingSystem**

**Set-CimInstance -CimInstance $nom -Property @{Description="ORDI2"}**

**# EXEMPLE 3**

**Set-CimInstance -Query 'Select \* from Win32\_OperatingSystem' `**

**-Property @{Description="ORDI3"}**

**# EXEMPLE 4**

**Get-CimInstance -ClassName Win32\_OperatingSystem | `**

**Set-CimInstance -Property @{Description = "ORDI4"}**

**Exemple 4 - Utilisation de Get-CimInstance et Set-CimInstance**

**# VERSION 1**

**# IMPORTANT: ce code est valide si l'ordinateur est dans un "Groupe de travail"**

NOTE: il n'y a pas de cmdlet pour **déverrouiller** le compte d'un utilisateur local

**Voici le code pour déverrouiller un compte d'un utilisateur local**

**$liste = "ETU","PROF","TECH"**

**$users = Get-CimInstance -ClassName Win32\_UserAccount `**

**-Filter "LocalAccount=True and Lockout=True" | `**

**Where-Object Name -in $liste**

**foreach ($user in $users)**

**{**

**# Affiche le nom de l'utilisateur**

**$nom = $User.name**

**Write-Warning "Nom de l'utilisateur: $nom"**

**# Affiche le contenu de la propriété Lockout avant la modification**

**$user.Lockout**

**# Modification de la propriété Lockout**

**$user.Lockout = "False"**

**# Affiche le contenu de la propriété Lockout après la modification**

**$user.Lockout**

**# Mise à jour de l'utilisateur**

**Set-CimInstance -CimInstance $user**

**}**

**# VERSION 2**

**# IMPORTANT: ce code est valide si l'ordinateur est dans un "Groupe de travail"**

NOTE: il n'y a pas de cmdlet pour **déverrouiller** le compte d'un utilisateur local

**Voici le code pour déverrouiller un compte d'un utilisateur local**

**$liste = "PROF","TECH"**

**$users = Get-CimInstance -ClassName Win32\_UserAccount `**

**-Filter "LocalAccount=True and Lockout=True" | `**

**Where-Object Name -in $liste**

**foreach ($user in $users)**

**{**

**# Mise à jour de l'utilisateur pour déverrouiller son compte**

**Set-CimInstance -CimInstance $user `**

**-Property @{LockOut=$false}**

**$nom = $user.name**

**Write-Warning "Le compte de l'utilisateur est déverrouillé: $nom"**

**}**

**Étape 1 - Utilisation de plusieurs fonctions du namespace ROOT\CIMV2**

Dans PowerShell ISE, écrire les commandes permettant d’obtenir les informations demandées.

**Vous devez utiliser le cmdlet Get-CimInstance du module CimCmdlets.**

**Win32\_ComputerSystem**

Écrire la commande pour afficher toutes les propriétés de win32\_ComputerSystem

get-ciminstance Win32\_ComputerSystem | select-object \*

Écrire la commande qui affiche

* Le nom de votre ordinateur
* Le nom du domaine ou du groupe de travail
* Le nom de l’utilisateur qui est connecté pour exécuter le test
* Pour l'affichage utiliser Format-Table avec le paramètre -AutoSize

get-ciminstance Win32\_ComputerSystem | format-table -autosize Caption,Domain,UserName

**Win32\_Group**

Écrire la commande pour afficher le nom et le SID de tous les groupes de votre ordinateur

* Pour l'affichage utiliser Format-Table avec le paramètre -AutoSize

get-ciminstance Win32\_Group | format-table -autosize Name,SID

**Win32\_UserAccount**

Écrire la commande pour afficher le nom et le SID de tous les usagers de votre ordinateur

* Pour l'affichage utiliser Format-Table avec le paramètre -AutoSize

get-ciminstance Win32\_UserAccount | format-table -autosize Name,SID

**Win32\_Environment**

Écrire la commande pour afficher la valeur de toutes les variables d’environnement de votre ordinateur

* Pour l'affichage utiliser Format-Table avec les paramètres -AutoSize et -Wrap

get-ciminstance Win32\_Environment | Select-Object \* | format-table -autoSize -wrap

**Win32\_VideoController**

Écrire la commande pour afficher des informations sur la carte vidéo de votre ordinateur

* Le nom de votre ordinateur
* Le nom du modèle de la carte vidéo
* Le nom du processeur vidéo
* La résolution de l'écran utilisée et le nombre de couleurs
* La quantié de mémoire sur la carte vidéo
* La version du pilote de la carte vidéo

**Get-CimInstance -ClassName Win32\_VideoController | `**

**Select-Object -Property SystemName,**

**Name,**

**VideoProcessor,**

**AdapterRAM,**

**VideoModeDescription,**

**CurrentRefreshRate,**

**MaxRefreshRate,**

**DriverDate,**

**DriverVersion | Format-List**

**SystemName : VM70035316**

**Name : NVIDIA Quadro M2000**

**VideoProcessor : Quadro M2000**

**AdapterRAM : 4293918720**

**VideoModeDescription : 1680 x 1050 x 4294967296 couleurs**

**CurrentRefreshRate : 59**

**MaxRefreshRate : 75**

**DriverDate : 2021-02-22 19:00:00**

**DriverVersion : 27.21.14.6172**

**Win32\_LogicalDisk**

Afficher la liste des disques locaux présents avec les informations suivantes:

* Leur nom (lettre)
* Leur type
* Leur système de fichier
* Les lecteurs de disque qui ne contiennent pas de média n’ont pas de système de fichier.
* Consulter le fichier "C:\\_Outils\documentation - WMI\wmisdk\_book.chm" pour connaître les valeurs de l’attribut DriveType.
* La taille
* Espace libre
* Pour l'affichage utiliser Format-Table avec les paramètres -AutoSize et -Wrap

Écrire trois réponses en utilisant trois façons différentes de faire cette requête (-filter, -query, where-object)

get-ciminstance Win32\_LogicalDisk -Filter "DriveType = 3" | `

format-table -autosize –wrap DeviceID,Description,FileSystem,Size,FreeSpace

get-ciminstance -Query "Select \* from Win32\_LogicalDisk Where DriveType = 3" | `

format-table -autosize –wrap DeviceID,Description,FileSystem,Size,FreeSpace

get-ciminstance Win32\_LogicalDisk | where-object drivetype -eq 3 | `

format-table -autosize –wrap DeviceID,Description,FileSystem,Size,FreeSpace

**Win32\_Service**

Faire afficher la liste des services dont l’état est à "démarrer" avec les informations suivantes:

* le nom du service
* le nom complet du service
* le chemin
* le mode de démarrage
* Pour l'affichage utiliser Format-List

Écrire trois réponses en utilisant trois façons différentes de faire cette requête (-filter, -query, where-object)

get-ciminstance Win32\_Service -Filter 'state = "running"' | `

format-list name,displayname,pathname,startmode

get-ciminstance -Query 'select \* from Win32\_Service where state = "running"' | `

format-list name,displayname,pathname,startmode

get-ciminstance Win32\_Service | where-object state -eq "Running" | `

format-list name,displayname,pathname,startmode

**Win32\_NetWorkConnection**

Faire afficher le nom et le chemin des disques réseaux présentement connectés

* Pour l'affichage utiliser Format-Table avec le paramètre -AutoSize

get-ciminstance win32\_NetworkConnection | format-table -autosize LocalName,RemoteName

get-ciminstance win32\_NetworkConnection | select-object LocalName,RemoteName | Out-GridView

**Win32\_NetworkAdapter**

Afficher l’index, l’adresse MAC et le nom des cartes réseaux (NetConnectionID).

note 1: le paramètre MACAddress ne doit pas être nul

note 2: le paramètre NetConnectionID ne doit pas être nul

note 3: Pour l'affichage utiliser Format-Table avec les paramètres -AutoSize et -Wrap

Écrire trois réponses en utilisant trois façons différentes de faire cette requête (-filter, -query, where-object)

get-ciminstance Win32\_NetworkAdapter `

-Filter "MACAddress is not null and NetConnectionID is not null" | `

format-table -autosize –wrap Index,MACAddress,NetConnectionID

get-ciminstance -Query "select \* from Win32\_NetworkAdapter where `

MACAddress is not null and NetConnectionID is not null" | `

format-table -autosize –wrap Index,MACAddress,NetConnectionID

get-ciminstance Win32\_NetworkAdapter | `

where-object { $PSItem.MACAddress -ne $null -and $PSItem.NetConnectionID -ne $null } | `

format-table -autosize –wrap Index,MACAddress,NetConnectionID

**Win32\_NetworkAdapterConfiguration**

Afficher l’index, la description des cartes réseaux, l'adresse MAC, l’adresse IP, le masque de sous-réseau, la passerelle, les adresses des serveurs DNS

note 1: le paramètre MACAddress ne doit pas être nul

note 2: le paramètre Description net doit pas être nul

note 3: Pour l'affichage utiliser Format-List

Écrire trois réponses en utilisant trois façons différentes de faire cette requête (-filter, -query, where-object)

get-ciminstance Win32\_NetworkAdapterConfiguration `

-Filter "MACAddress is not null and Description is not null" | `

FL Index,Description,MACAddress,IPAddress,IPSubnet,DefaultIPGateway,DNSServerSearchOrder

get-ciminstance -Query "select \* from Win32\_NetworkAdapterConfiguration where `

MACAddress is not null and Description is not null" | `

FL Index,Description,MACAddress,IPAddress,IPSubnet,DefaultIPGateway,DNSServerSearchOrder

get-ciminstance Win32\_NetworkAdapterConfiguration | `

where-object { $PSItem.MACAddress -ne $null -and $PSItem.Description -ne $null } | `

FL Index,Description,MACAddress,IPAddress,IPSubnet,DefaultIPGateway,DNSServerSearchOrder

**Win32\_BIOS**

Les propriétés de Win32\_BIOS sont en lecture seulement.

**Clear-Host**

**$computer = $env:COMPUTERNAME**

**$namespace = "ROOT\CIMV2"**

**$classname = "Win32\_BIOS"**

**Write-Output "====================================="**

**Write-Output "Computer : $computer"**

**Write-Outpui "NameSpace : $namespace"**

**Write-Output "ClassName : $classname"**

**Write-Output "====================================="**

**Get-CimInstance -Namespace $namespace -ClassName $classname**

**La compagnie "Hewlett Packard Enterprise" offre un module PowerShell pour administrer le BIOS des serveurs. Le module "HPEBIOSCmdlets" permet de modifier les propriétés du BIOS.**

Le module "**HPEBIOSCmdlets**" est offert avec des scripts pour faciliter la gestion du BIOS.

* ConfigureAdminInfo.ps1
* ConfigureAMDCorePerformanceBoosting.ps1
* ConfigureBIOSAdminPassword.ps1
* ConfigureBootMode.ps1
* ConfigureBootOrder.ps1
* ConfigureEMSConsoleAndSerialPort.ps1
* ConfigureIntelCoreBoosting.ps1
* ConfigureIntelTurboBoost.ps1
* ConfigureNetworkBootsettings.ps1
* ConfigureNVDIMMConfiguration.ps1
* ConfigurePCIDeviceWithPCIeLinkSpeedForGen10.ps1
* ConfigurePowerOnPassword.ps1
* ConfigureProcessorJitterControl.ps1
* ConfigureProcessorPower.ps1
* ConfigureServerAvailability.ps1
* ConfigureServerSecurity.ps1
* ConfigureThermalAndFanOption.ps1
* ConfigureTPM.ps1
* ConfigureUEFIOptimizedBoot.ps1
* ConfigureVirtualInstallDisk.ps1
* ConfigureWorkloadProfileForGen10servers.ps1
* ResetBIOSAdminPassword.ps1
* ResetBIOSDefaultManufacturingSettings.ps1
* ResetPowerOnPassword.ps1

**Étape 2 - Exemple avec une fonction dans le namespace ROOT\virtualization\V2**

Programme pour afficher le GUID du BIOS de chaque ordinateur virtuel.

**Clear-Host**

**$VMNames = (Get-VM).Name**

**ForEach ($VMName in $VMNames)**

**{**

**Get-CimInstance -Namespace Root\Virtualization\V2 `**

**-ClassName Msvm\_VirtualSystemSettingData `**

**-Filter "ElementName = '$VMName'" `**

**| Select-Object ElementName,BiosGUID**

**}**