Fiche 05 : Premiers scripts d'administration via WMI

1. Script WMI à partir de noms de postes contenus dans un fichier texte

Une première version sans pipeline :

```powershell
# Premier script avec WMI script02.ps1
# Recherche des adresses IP et des adresses MAC des ordinateurs dont les noms
# sont contenus dans un fichier texte listeposte.txt (1 nom par ligne)
# Le résultat est placé dans un fichier texte nommé adresseposte.txt
# les postes non trouvés sont placés dans le fichier erreur.txt

$fichierPoste="c:\temp\listeposte.txt"
$fichierAdresse="c:\temp\adresseposte.txt"
$fichierErreur="c:\temp\erreur.txt"
$listePoste=Get-Content $fichierPoste
foreach($nom in $listeposte) {
    $filtre="address='"+$nom+"'"
    $test=Get-WmiObject Win32_PingStatus -Filter $filtre
        if ($test.StatusCode -eq 0){
            $colCarte=Get-WmiObject Win32_NetworkAdapterConfiguration -Filter IPEnabled=True -ComputerName $nom
            foreach($carte in $colCarte) {
                $ligne=$nom+"/"+$carte.MACAddress+"/"+$carte.IPAddress
                Add-Content $fichierAdresse $ligne
            }
        }
        else{
            Add-Content $fichierErreur "pas trouvé le poste : $nom"
        }
}
```

Cette classe WMI permet de faire un test de connexion vers le poste avec un filtre construit à partir du nom. La connexion est OK si StatusCode est égal à 0.

Cette classe WMI retourne une collection d'objet (configuration adaptateur réseau) du poste spécifié par –computeurName.

Pour chaque configuration retournée, écriture d'une ligne dans le fichier

Remarque : Ne sont pas traités les différents tests d'existence des fichiers.

Une autre version avec un pipeline pour récupérer la configuration des cartes réseaux :

```powershell
# Idem premier script avec WMI script02.ps1
# mais le résultat d'un objet WMI est transmis à un pipeline

$fichierPoste="c:\temp\listeposte.txt"
$fichierAdresse="c:\temp\adresseposte.txt"
$fichierErreur="c:\temp\erreur.txt"
foreach ($nom in Get-Content $fichierPoste){
    $filtre="address='"+$nom+"'"
    $test=Get-WmiObject Win32_PingStatus -Filter $filtre
    if ($test.StatusCode -eq 0) {
        Get-WmiObject Win32_NetworkAdapterConfiguration -Filter IPEnabled=True -ComputerName $nom |
        ForEach-Object {Add-Content $fichierAdresse ($nom+"/"+$_.MACAddress+"/"+$_.IPAddress)}
    }
    else{
        Add-Content $fichierErreur "pas trouvé le poste : $nom"
    }
}
```

Chaque objet (configuration adaptateur réseau) est transmis au pipeline.

Chaque objet transmis est référencé par $ pour obtenir les propriétés

Et une autre version avec le parcours du fichier via un pipeline :

```powershell
# idem 2ème version du premier script avec WMI script02.ps1
# mais parcours du fichier avec un pipeline

$fichierPoste="c:\temp\listeposte.txt"
$fichierAdresse="c:\temp\adresseposte.txt"
$fichierErreur="c:\temp\erreur.txt"
Get-Content $fichierPoste |
    ForEach-Object{$nom=$_;$test=Get-WmiObject Win32_PingStatus -Filter ("address='"+$nom+"'")
        if ($test.StatusCode -eq 0) {
            Get-WmiObject Win32_NetworkAdapterConfiguration -Filter IPEnabled=True -ComputerName $nom |
            ForEach-Object {Add-Content $fichierAdresse ($nom+"/"+$_.MACAddress+"/"+$_.IPAddress)}
        }
        else{
            Add-Content $fichierErreur "pas trouvé le poste : $nom"
        }
    }
```

Chaque objet (ligne du fichier) est transmis au

$_ représente la ligne (un nom de poste), $nom mémorise ce nom pour être utilisé dans le dernier

Remarque : La lecture du script devient un peu plus complexe.

2. Script WMI à partir d'adresses IP

Il est possible de générer une plage d'adresses IP à partir d'un tableau de numéros construit avec l'instruction 1..254 et un pipeline, ce qui donne pour notre test 'ping' dans le script :

```powershell
1..254 | ForEach-Object{
        $IP="192.168.0."+$_
        $test=Get-WmiObject Win32_PingStatus -Filter ("address='"+$IP+"'")
```

Pour chaque objet (une valeur du tableau), l'IP est construite avec la

Et la construction d'une plage d'adresses IP sans pipeline :

```powershell
$vals=1..254;foreach($ip in $vals){"192.168.0."+$ip}
```