4.12 Exécution de commandes distantes

Une particularité très importante de PS est sa possibilité d'exécuter des commandes sur des machines distantes aussi facilement que sur la machine locale. L'administration à distance devient alors un jeu d'enfant.

4.12.1 Configuration

Pour pouvoir utiliser PowerShell à distance, il faut configurer le *service* « Windows Remote Management » (WinRM) sur l'ordinateur cible. Vous trouverez les informations complètes d'installation et de configuration de WinRM à cet endroit http://msdn.microsoft.com/en-us/library/aa384372 . Votre machine locale communiquera avec les machines distantes via le protocole WS-MAN (Web-Services for Management). C'est un protocole basé sur HTTP(S) qui peut encapsulé différentes variétés de communications (cf. Figure 2). Comme vous l'aurez compris, votre session PS ne pourra dialoguer qu'avec des machines sur lesquelles le service WinRM est correctement configuré.

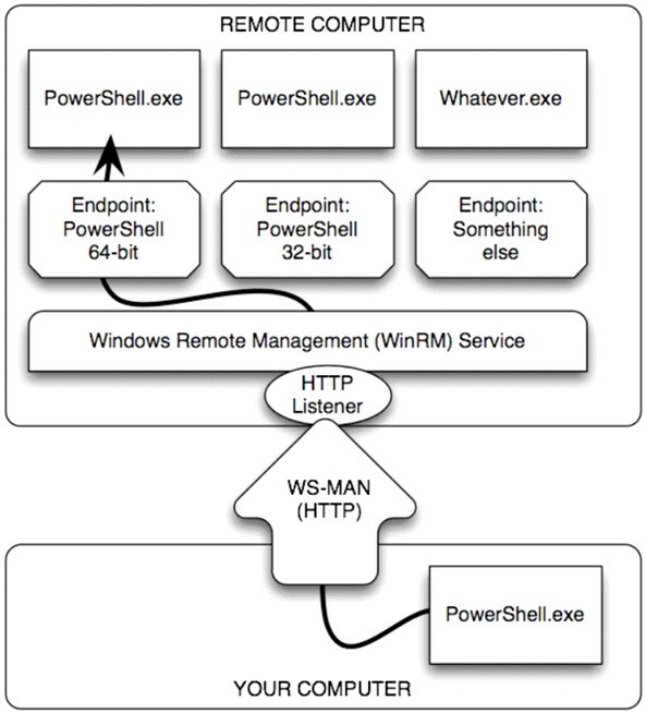

Figure 2: Communication avec un ordinateur distant

Exercice 63 Trouver l'instruction qui permet de récupérer la liste des services qui contient la chaîne 'RM' sur la machine locale. Ensuite, récupérez uniquement l'état du service WinRM.

Réponse :

Pour qu'une machine soit administrable à distance, nous supposons que :

- la machine qui administre et la machine distante soient dans le même domaine ;
- Windows Remote Management soit installé et lancé sur la machine distante. Pour cela, il est nécessaire de lancer PS en tant qu'administrateur et de configurer WinRM.

Exercice 64 Exécutez l'instruction Enable-PSRemoting afin de configurer WinRM (équivalent à `winrm quickconfig` en ligne de commandes windows). Répondez par 'oui' pour que l'ordinateur accepte les

requêtes, ajoute un « listener » (le moyen de recevoir des commandes) et ajoute les règles d'exception du pare-feu. Quelle est l'instruction qui permet de retrouver toutes les cmdlmets contenant 'session' ? Quelle est la commande qui permet de se connecter sur la machine distante que vous venez de configurer en utilisant votre login? Modifier l'instruction précédente en utilisant le nom de login de votre voisin et lui demander de saisir son mot de passe (`Credential`). Vérifier que vous êtes bien connecté à la machine distante en affichant le nom de la machine hôte.

Réponse :

Exercice 65 Il n'est pas nécessaire d'ouvrir une session PS pour exécuter quelques commandes à distance. Une autre cmdlet existe pour cela : Invoke-Command. Récupérer les services arrêtés de la machine distante. Il est possible de lancer plusieurs commandes avec la même session ; pour cela intéressez vous à la cmdlet New-PSSession.

Réponse :

Exercice 66 L'un des plus gros avantages de PS est de lancer des scripts sur un ensemble de machines distantes simultanément. Créez un fichier texte contenant l'ensemble des machines accessibles. Utilisez ce fichier pour récupérer les processus communs à toutes les machines accessibles.

Réponse :

4.12.2 Les machines distantes

Exercice 67 Il est possible de découvrir l'ensemble des machines ("computer" en anglais) distantes de son domaine en utilisant les objets `system.directoryservices.directoryentry` et `system.directoryservices.directorysearcher`. Écrire le script qui permet de récupérer ces machines.

Réponse :

Exercice 68 Quelle est l'instruction qui nous permettrait de tester si une machine distante supporte l'accès distant ?

Réponse :

5 Windows Management Instrumentation

5.1.1 Introduction

WMI est un système générique de gestion des éléments logiques et physiques d'une machine munie d'un système d'exploitation Microsoft. Pour chaque éléments d'une machine utilisant Windows, WMI propose une interface utilisable en vbscript, C++, Javascript , C# et donc Powershell et accessible à distance. De plus WMI détecte et traite les événements survenant sur le système.

Afin d'organiser toutes les informations sur le système, WMI est organisé en espace de nommages ou *namespaces*. Un namespace est une sorte de répertoire contenant des informations sur un produit spécifique ou une technologie. Par exemple, le namespace `root\CIMv2` contient toutes les informations sur Windows et la configuration Hardware de la machine ; le namespace `root\MicrosoftDNS` inclut toutes les informations sur le serveur DNS ; `root\SecurityCenter` contient toutes les informations sur le firewall, l'antivirus, les anti-spywares... Chaque namespace est divisé en une séries de classes à partir desquelles des informations peuvent être obtenues ; par exemple, la classe `Win32_LogicalDisk` du namespace `root\CIMv2` gère les informations sur les disques locaux. Néanmoins, ce n'est pas parce la classe existe sur un ordinateur que celui-ci possède physiquement le composant. Par exemple, la classe `Win32_TapeDrive` est présente sur toutes les versions de Windows même si ce genre de disque n'est pas installé physiquement sur la machine. Pour savoir si le composant existe réellement, il faut vérifier les *instances* de ces classes.

Vous pouvez télécharger un explorateur de classes WMI à cet endroit (à priori, il n'y a pas besoin d'être administrateur sur la machine). Installez le et lancez le.

5.1.2 WMI & Powershell

En PS, vous pouvez récupérer les instances de WMI grâce à la cmdlet `Get-WmiObject`. Cette commande permet de spécifier un espace de nommage, un nom de classe ou encore le nom d'une machine distante pour récupérer toutes les instances de la précédente classe sur la machine spécifiée ; comme par exemple :

```
Get-WmiObject -namespace root\cimv2 -class win32_LogicalDisk
```

Remarquez qu'avec l'instruction précédente, vous récupérez plusieurs instances de la classe spécifiée. Par conséquent, si vous souhaitez utiliser une information d'une instance, il faudra utiliser les opérateurs de collection. Un autre moyen d'accéder aux instances de classes est d'utiliser le cast de type. Avec WMI, il existe plusieurs types comme :

- [WMI] qui accède à une seule instance de classe WMI en précisant un chemin unique

```
[wmi]"win32_LogicalDisk.DeviceID='C:'"
```

- [WMICLASS] qui accède à une classe WMI et en particulier à ses méthodes statiques (méthodes communes à toutes les instances d'une classe).

```
[wmiclass]"win32_LogicalDisk"
```

- [WMISEARCHER] qui exécute une interrogation WMI à l'aide du langage WQL et renvoie un résultat.

```
([wmisearcher]"select * from win32_LogicalDisk").get()
```

Enfin, pour distinguer une classe d'une instance de classe, il suffit de regarder la propriété __GENUS ; la valeur 1 (WBEM_GENUS_CLASS) indique une classe et la valeur 2 (WBEM_GENUS_INSTANCE) indique une instance ou un événement.

Exercice 69 Récupérer toutes les instances de l'espace de nommage root\cimv2. Réduisez cette liste en incluant que les classes qui contiennent 'network'.

Réponse :

Exercice 70 Rechercher les logiciels ("product" en anglais) installés sur la machine. Même chose avec les informations systèmes.

Réponse :

Exercice 71 Afficher uniquement les informations des disques durs physiques (pas de CD-ROM ou de disques distants). Ensuite, envoyer par pipe la précédente commande vers la cmdlet `Get-Member`. Écrivez un script ou une cmdlet qui calcule l'espace libre sur l'ensemble des disques distants (affichez l'information de manière lisible). Écrivez un script qui permet d'appeler la méthode `chkdsk` pour chacun des disques.

Réponse :

Exercice 72 Écrire l'instruction qui permet de rechercher et de fermer toutes les instances de `notepad.exe`. Intéressez vous à Win32_Process. (Il est évident que pour tester votre script, il vous faudra créer plusieurs instances de notepad).

Réponse :

Il est également possible d'utiliser les méthodes proposées par les classes ou instances de classes, mais il faut d'abord vérifier quelles méthodes sont disponibles pour la classe. Prenons l'exemple de la classe Win32_Process et affichons les méthodes applicables avec l'instruction :

```
([wmiclass]"Win32_Process").psbase.methods
```

Exercice 73 Écrire l'instruction PS permettant de créer une instance de `cmd.exe`.

Réponse :

Exercice 74 Il est également possible de modifier les valeurs contenues dans des instances de classe WMI à condition que cela soit autorisé par l'instance ou la classe. Écrire l'instruction qui affiche les actions possibles des classes `Win32_Process`, `Win32_Desktop`, `Win32_OperatingSystem`, `Win32_NTDomain`, `Win32_Environment`. Ensuite, si la classe est modifiable et si les propriétés modifiées sont en lecture/écriture et si le compte utilisé possède les droits de modifications de WMI, alors il faut utiliser la méthode `Put()`. Ajouter une variable d'environnement avec WMI en créant une instance, puis

Réponse :

Exercice 75 Écrire l'instruction reboutant la machine. Il y a de forte chance que l'erreur « Privilège non maintenu » apparaisse. Pour cela, il faut modifier les privilèges de la classe avec les instructions suivantes : `$OS=(gwmi WIN32_OperatingSystem); $OS.PSBase.Scope.Options.EnablePrivileges = $True`.

Réponse :