Comunicaciones remotas de Framework .NET 2.0

Como usted bien sabe, PowerShell se basa plenamente en el Framework .NET 2.0 y por ello se beneficia de las funcionalidades de ejecución remota de este último. Es así como algunos commandlets han heredado el parámetro **-ComputerName**. Algunas de estas funcionalidades pueden ejecutarse en uno o varios ordenadores remotos incluso sin que PowerShell tenga necesidad de estar instalado en estos últimos.

Las comunicaciones remotas de Framework .NET representan la forma más sencilla de actuar en máquinas remotas, pero atención, ya que se basan en el protocolo RPC. Este protocolo suele ser por los routers, por lo que únicamente se puede utilizar en redes de tipo LAN.

1. Requisitos previos

- Ser miembro del grupo Administradores del ordenador remoto o ser miembro del grupo Administradores del dominio,
- Disponer de PowerShell v2 únicamente en su ordenador.

2. Determinar los comandos remotos del Framework .NET 2.0

Para conocer todos los commandlets provistos del parámetro **-ComputerName**, teclee:

```
PS > Get-Help * -parameter ComputerName
```

Posteriormente para garantizar que el commandlet se basa en los mecanismos del Framework .NET, es preciso verificar, mediante la ayuda en línea, que no se basa en «la comunicación remota Windows PowerShell» (veremos qué es esto en la parte siguiente de este capítulo).

Ejemplo:

```
PS C:\Windows\system32> Get-Help Get-Process -parameter ComputerName

-ComputerName <string[]>
    Obtiene los procesos que se ejecutan en los equipos especificados.
El valor predeterminado es el equipo local.

    Escriba el NetBIOS, la dirección IP o el nombre de dominio completo
de uno o varios equipos. Para especificar el equipo local, escriba el
nombre del equipo, un punto (.), o bien, "localhost".

    Este parámetro no se basa en la comunicación remota de Windows
PowerShell. Puede usar el parámetro ComputerName de Get-Process incluso si
el equipo no está configurado para la ejecución de comandos remotos.

    ¿Requerido?                    false
    ¿Posición?                     named
    Valor predeterminado
    ¿Aceptar canalización?         true (ByPropertyName)
    ¿Aceptar caracteres comodín? false
```

Podemos leer, con la ayuda de este comando que este commandlet «no se basa en la comunicación a distancia Windows PowerShell», lo significa que se basa en los mecanismos de comunicación remota del Framework .NET 2.0.

Este enfoque puede parecer un poco especial para determinar cuáles son los comandos que se basan en los

mecanismos de comunicación remota del Framework .NET 2.0 y cuáles se basan en los mecanismos de comunicación a distancia PowerShell; pero desgraciadamente no hay otras soluciones.

La idea del «Team PowerShell» es sin duda que el usuario final no se plantea todas estas cuestiones. En efecto, la finalidad es proponer el mismo parámetro a diferentes comandos, sin importar la tecnología que se esconda detrás. Dicho esto, conocer la tecnología subyacente puede tener su importancia ya que van a tener que pasar muchos años antes de que todas las empresas hayan finalizado el despliegue de PowerShell v2 en todo su parque de máquinas. No sólo debe instalarse PowerShell v2, sino que también debe estar configurado para aceptar las comunicaciones remotas PowerShell. Por ello, nos parece importante poner de relieve y dar a conocer los commandlets que pueden emplearse en remoto con unos requisitos previos mínimos (véase más arriba).

Como es engorroso consultar la ayuda de cada comando para comprobar la presencia de esta cadena de caracteres, este pequeño script nos será de gran ayuda:

```
# Find-DotNetRemoteCmdlets.ps1
# Lista los commandlets que no se basen en las funcionalidades
# de comunicación remota PowerShell v2

$encontrado = @()
$noEncontrado = @()
$pattern = 'no basado en la comunicación remota Windows PowerShell'
$lista = Get-Help * -parameter ComputerName

ForEach ($cmde in $lista)
{
   $description =
    (Get-Help $cmde.name -parameter ComputerName).description | Out-String
   If ($description | Select-String -pattern $pattern)
   {
     $encontrado += $cmde.name
   }
   Else
   {
     $noEncontrado +=$cmde.name
   }
}

$encontrado | sort-object
```

El script empieza por inicializar dos variables de tipo tabla (`$encontrado` y `$noEncontrado`) en las cuales se almacenarán los resultados de las búsquedas. La variable `$pattern` contiene la cadena a buscar y la variable `$lista` contiene la lista de los commandlets que poseen un parámetro denominado **`ComputerName`**. Después se itera cada comando de la lista `$lista`. La iteración consiste en recordar el comando `Get-Help` de cada comando con el fin de buscar por medio de `Select-String` la cadena característica contenida en `$pattern`. Observe la utilización de `Out-String` para convertir la ayuda en una cadena de caracteres necesaria para utilizar `Select-String`. Finalmente si la búsqueda obtiene resultados, se almacena el nombre del comando en la variable `$encontrado`, en caso contrario se almacenará en `$noEncontrado`. Después se devuelve el resultado de la variable `$encontrado` ordenado alfabéticamente.

Lo que nos da el resultado siguiente:

```
Clear-EventLog
Get-Counter
Get-EventLog
Get-HotFix
Get-Process
Get-Service
Get-WinEvent
```

```
Get-WmiObject
Limit-EventLog
New-EventLog
Remove-EventLog
Remove-WmiObject
Restart-Computer
Set-Service
Set-WmiInstance
Show-EventLog
Stop-Computer
Test-Connection
Write-EventLog
```

3. El conjunto de comandos

Vemos un poco más en detalle lo que es posible hacer con cada uno de estos comandos. Los hemos agrupado por temas:

Comando	Descripción
`Clear-EventLog`	Suprime todas las entradas de los registros de eventos especificados.
`Get-EventLog`	Obtiene los eventos de un registro de eventos o la lista de los registros de eventos.
`Limit-EventLog`	Define las propiedades del registro de eventos que limita el tamaño del registro de eventos y la antigüedad de sus entradas.
`Remove-EventLog`	Suprime un registro de eventos o anula la inscripción de un origen de eventos.
`Show-EventLog`	Muestra los registros de eventos.
`New-EventLog`	Crea un registro de eventos y un origen de eventos.
`Write-EventLog`	Escribe un evento en el registro de eventos.
`Get-WinEvent`	Obtiene los eventos a partir de los registros de eventos y de los archivos de registro de seguimiento de eventos.
`Get-Service`	Obtiene la lista de servicios.
`Set-Service`	Arranca, detiene e interrumpe un servicio, y luego modifica sus propiedades.
`Get-HotFix`	Obtiene las revisiones del sistema que se han aplicado a los equipos local y remotos.
`Set-WmiInstance`	Crea o actualiza una instancia de una clase WMI existente.
`Get-WmiObject`	Obtiene las instancias de clases WMI o la información sobre las clases disponibles.
`Remove-WmiObject`	Elimina una instancia de una clase WMI existente.
`Get-Process`	Obtiene la lista de procesos en ejecución.
`Restart-Computer`	Reinicia el sistema operativo.

`Stop-Computer`	Detiene el sistema operativo.
`Test-Connection`	Envía paquetes de solicitud de eco ICMP («pings») a uno o más equipos.
`Get-Counter`	Obtiene los datos del contador de rendimiento.

4. Envió de comandos remotos

El envío de un comando de la lista de la tabla anterior no puede ser más sencillo, ya que no importa si PowerShell está instalado o no en las máquinas remotas. El único requisito previo es estar conectado con una cuenta que sea como mínimo administrador de la máquina remota. En efecto, todos estos commandlets no permiten la transferencia de autorizaciones alternativas.

Ejemplo 1: parada/reinicio del servicio W32Time de un servidor remoto

Parada del servicio:

```
PS > Get-Service -ComputerName W2K8R2VM -Name W32time | Set-Service
-Status stopped
```

Verificación del estado del servicio:

```
PS > Get-Service -ComputerName W2K8R2VM -name W32time

Status   Name               DisplayName
------   ----               -----------
Stopped  W32time            Windows Time
```

Inicio del servicio:

```
PS > Get-Service -ComputerName W2K8R2VM -name W32time | Set-Service
-Status running
```

Ejemplo 2: leer los registros de eventos de una máquina remota

```
PS > Get-EventLog -ComputerName W2K8R2VM -LogName system -Newest 10
```

Esta línea de comando recupera las 10 entradas más recientes del registro del sistema de una máquina remota.

```
 Index Time          EntryType   Source              InstanceID Message
 ----- ----          ---------   ------              ---------- -------
 14046 ago 02 21:41  Information Service Contro...   1073748860 El servicio
Protección de software entró en est...
 14045 ago 02 21:40  Information Service Contro...   1073748860 El servicio
Programador de aplicaciones multime...
 14044 ago 02 21:35  Information Service Contro...   1073748860 El servicio
Información de la aplicación entró ...
 14043 ago 02 21:34  Information Service Contro...   1073748860 El servicio
Servicio de detección automática de...
 14042 ago 02 21:29  Information Microsoft-Wind...          206 El servicio
de compatibilidad de programas inic...
 14041 ago 02 21:28  Information Service Contro...   1073748860 El servicio
Ayuda del Panel de control de Infor...
```

```
 14040 ago 02 21:28  Information Service Contro...   1073748860 El servicio
Ayuda del Panel de control de Infor...
 14039 ago 02 21:21  Information Service Contro...   1073748860 El servicio
Protección de software entró en est...
 14038 ago 02 21:21  Information Microsoft-Wind...           18 Instalación
preparada: las actualizaciones sigu...
 14037 ago 02 21:20  Information Microsoft-Wind...           18 Instalación
preparada: las actualizaciones sigu...
```

Y si se quiere filtrar para mostrar únicamente los 10 últimos errores, es tan sencillo como:

```
PS > Get-EventLog -ComputerName W2K8R2VM -LogName system |
Where {$_.EntryType -eq 'Error'} | Select-Object -First 10

Index Time          EntryType   Source            InstanceID Message
----- ----          ---------   ------            ---------- -------
13717 ago 02 08:16  Error       FW1               3221225473 FW1: -->ng
clock change.
13716 ago 02 08:16  Error       FW1               3221225473 FW1: FW-1:
last packet seen 31475 seconds ago, ...
13695 ago 01 23:29  Error       Disk              3221487627 El controlador
detectó un error de controladora...
13694 ago 01 23:29  Error       Disk              3221487627 El controlador
detectó un error de controladora...
13693 ago 01 23:29  Error       Disk              3221487627 El controlador
detectó un error de controladora...
13691 ago 01 23:29  Error       Disk              3221487627 El controlador
detectó un error de controladora...
13508 ago 01 14:07  Error       Service Cont...   3221232472 El servicio
Dispositivo host de UPnP no pudo in...
13507 ago 01 14:07  Error       Service Cont...   3221232510 El servicio
upnphost no se pudo iniciarse como ...
13504 ago 01 14:07  Error       DCOM              3221235477 No se encontró
la descripción del Id. de evento...
13499 ago 01 14:07  Error       DCOM              3221235482 No se encontró
la descripción del Id. de evento...
```