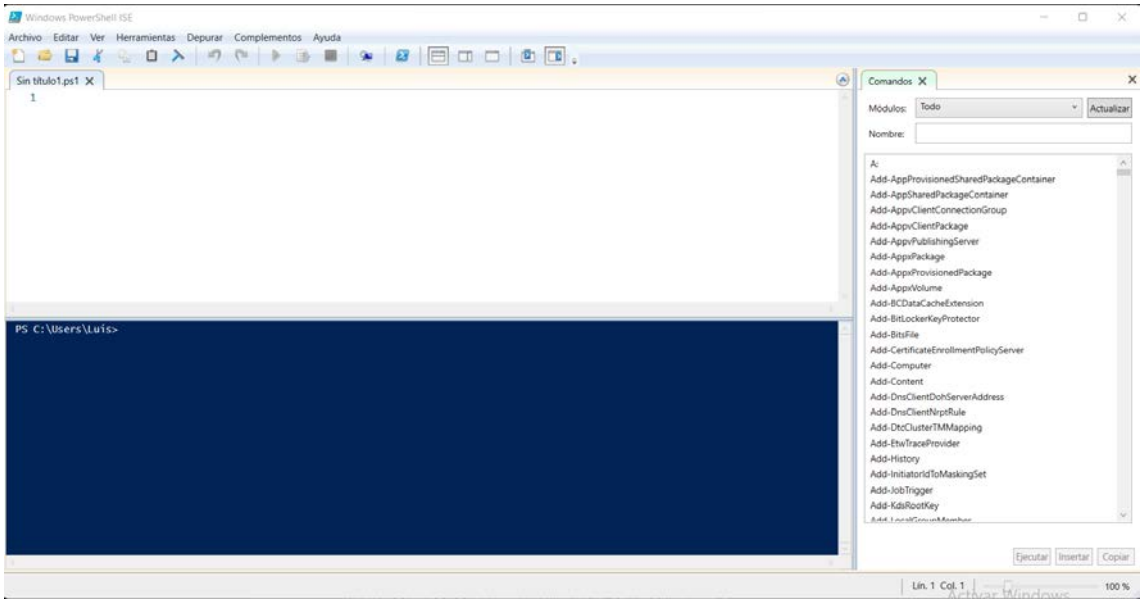
POWERSHELL

PowerShell es el sucesor de la línea de comandos de Windows. Aparecido en noviembre de 2006 y es infinitamente más potente. Se basa en pequeños programas escritos en C# llamados ***cmdlets***. En la actualidad se usa para casi todo, pero es indispensable para gestionar profesionalmente los productos de Microsoft, ya que toda su gestión se basa generalmente en cmdlets de PowerShell.

SCRIPTS

Los scripst los podemos crear, probar y depurar con el entorno de desarrollo que nos ofrece Windows por defecto, llamado ***"Windows PowerShell ISE"***



La extensión de los ficheros scripts de powershell, ha de ser ps1.

MODOS DE EJECUCIÓN

Al igual que en Linux, cuando creábamos un script y de dábamos permiso de ejecución, en Windows powershell, tenemos que estar en el modo correcto para ejecutar el mismo. En relación a ello los comandos que se encargan de eso son:

- Set-ExecutionPolicy
- Get-ExecutionPolicy

Hay diferentes "ámbitos" (scope) sobre los que se puede actuar en el modo de ejecución:

A. MachinePolicy
B. UserPolicy
C. Process
D. CurrentUser
E. LocalMachine

Los modos de ejecución que se pueden especificar son los siguientes:

• ***Restricted (Restringida):*** es la regla por defecto. Permite la ejecución de comandos individuales pero no de archivos de scripts, incluyendo los archivos de configuración y formato (.ps1xml), los archivos de scripts de módulos (.psm1) y los perfiles de Windows PowerShell (.ps1).

• ***Allsigned (Solo firmas):*** permite ejecutar scripts firmados por un editor de confianza, incluyendo los scripts que se escriban en el equipo local. Solicita confirmación antes de ejecutar scripts de publicadores que no hayan sido clasificados como de confianza.

• ***Remotesigned (Firma remota):*** permite la ejecución de scripts descargados de internet firmados digitalmente por un editor de confianza. No requiere firma digital en los scripts que hayan sido escritos en el equipo local.

• ***Unrestricted (Sin restricción):*** permite ejecutar scripts sin firmar. Advierte al usuario antes de ejecutar archivos de configuración y scripts descargados de Internet con el fin de añadir seguridad.

• ***Bypass:*** esta directiva no bloquea nada y no muestra advertencias de seguridad. Pensado para programas que integran un script de Windows PowerShell en una aplicación compleja.

• ***Undefined (Indefinido):*** esta opción indica que no existe ninguna directiva de ejecución establecida. Si la directiva de ejecución en todos los ámbitos es Undefined, la directiva de ejecución será Restricted, que es la directiva de ejecución por defecto en Windows.

Si queremos ejecutar scripts de PowerShell en una máquina, debemos permitir antes su ejecución mediante el comando Set-ExecutionPolicy del siguiente modo:

PS C:\Users\usuario> Set-ExecutionPolicy -Scope LocalMachine unrestricted

PROGRAMADOR DE TAREAS

Crear la tarea que realice la ejecución del script powershell que hemos creado. Para ello solamente tenemos que poner "powershell.exe"como comando y usar **el script como comando mediante el parámetro "-Command"**. Además, conviene decir **que no cargue el perfil del usuario** (salvo que sea estrictamente necesario por algo), decirle **que no es un script interactivo** (de modo que no se vaya a interrumpir nunca la ejecución con la interfaz de usuario) y **que quite las restricciones a la ejecución**.

Es decir, lo que debemos hacer es pasarle estos parámetros:

-NoProfile

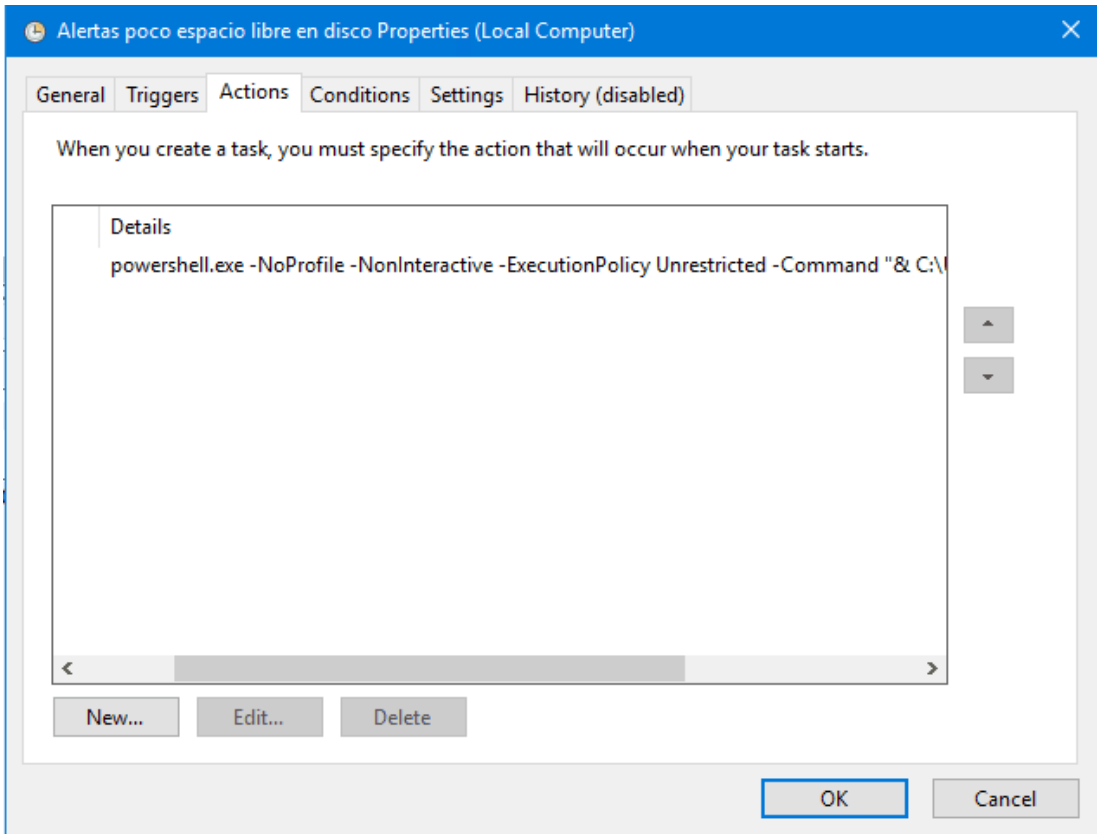
-NonInteractive

-ExecutionPolicy Unrestricted

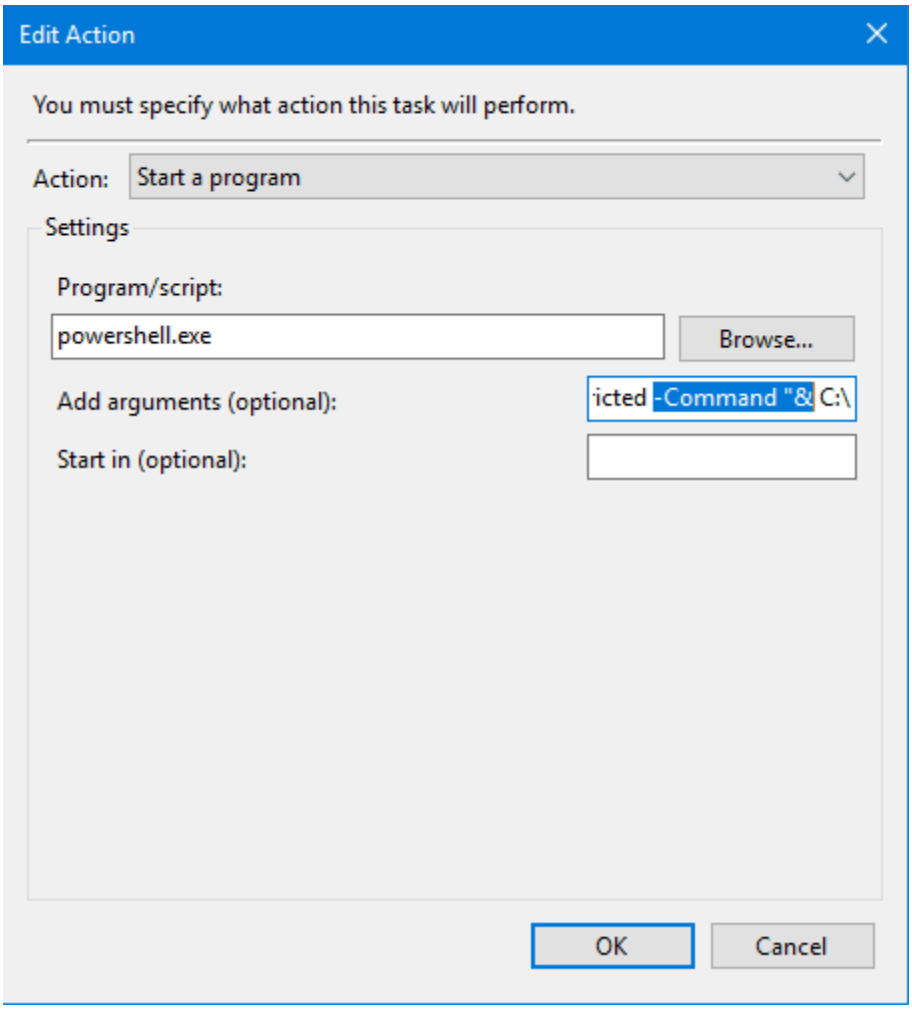
-Command "& ruta al script a ejecutar"

Pero, y aquí hay un gran pero, el verdadero truco está en que para que el script funcione debemos **precederlo con un "et" o "ampersand" (&) o no funcionará**.

Este es el aspecto de una de mis tareas programadas que usa un script .ps1 en un servidor:



Fíjate en cómo tras el -Command empieza la ruta del script, pero tiene el & delante, justo después de las comillas que abren el parámetro. O más en detalle al editarlo:



Es importante que el & esté entre las comillas. Este símbolo es especial para PowerShell. Se llama ***operador de llamada*** y sirve para indicar al parámetro que lo que se va a ejecutar es un script obtenido de un archivo o de un bloque de texto.