POWERSHELL: Interfaz powershell.exe

1. Abreviaciones

Powershell dispone de 14 opciones que pueden establecerse a la hora de iniciar una sesión con su interfaz de consola powershell.exe.
La forma en que Powershell trata estas opciones tiene dos características básicas, se tienen abreviaciones, y adicionalmente a cada opción se expande con el carácter '*'. También se ha de tener en cuenta que los sistemas operativos Microsoft Windows son Case-Insensitive, y Powershell trabaja de igual forma con sus parámetros.

Estas características aportan un numero realmente elevado de especificar una misma opción, y de tratar de esconder los parámetros de dichas opciones.

Teniendo en cuenta exclusivamente las abreviaciones, tendríamos:

Comando	Abreviación
-Command	c
-EncodedArguments	ea, encodeda
-EncodedCommand	e, ec, enc
-ExecutionPolicy	ex, ep
-File	f
-InputFormat	i, if
-Mta	m
-NoExit	noe
-NoLogo	nol
-NoProfile	nop
-NonInteractive	noni
-OutputFormat	o, of
-Sta	s
-WindowsStyle	w

2. Expansión de parámetros

Como se comentaba en los primeros párrafos, también es posible abreviar cualquier opción especificando únicamente sus primeros caracteres, siendo Powershell quien se encargará de expandir dicha opción añadiendo el carácter '*'.
Siempre y cuando esta nueva abreviación no entre en conflicto con otras opciones, será posible su uso.

Un ejemplo de variantes todas correctas puede verse a continuación:

- -EncodedCommand
- -EncodedComman
- -EncodedComma
- -EncodedComm
- -EncodedCom
- -EncodedCo
- -EncodedC
- -Encoded
- -Encode
- -Encod
- -Enco
- -Enc
- -En
- -E

O dicho de otro modo, **-EncodedCommand** puede especificarse de 14 formas diferentes, al igual que ocurre con el resto de parámetros posibles.

3. Insensibilidad a Mayúsculas

El tercer punto es la característica de no establecer diferencia entre mayúsculas y minúsculas, interpretándolas todas de igual forma. Esto hace que cada par de caracteres pueda expresarse de cuatro formas distintas con igual significado, véase como ejemplo el comando **-Sta**:

- -sta
- -stA
- -sTa
- -sTA

Esta característica nos permitirá tener **2^(n-1)** combinaciones para una misma palabra.

4. Carácter de escape

Habitualmente en diversos lenguajes de programación y sistemas operativos se utiliza '\' como carácter de escape, pero en Microsoft Windows este es un carácter utilizado para separación de rutas, por lo que en **CMD** el carácter de escape es '^'.

El echo que hemos de notar, es que al tratar '^o' y escapar dicho carácter quedaría 'o', o lo que es lo mismo, el carácter de escape '^' no tiene ningún efecto sobre caracteres que no necesitan escaparse. Y esto nos aporta un gran número diferente de combinaciones para expresar una misma palabra:

- **p^o^w^e^r^s^h^e^l^l**

Este carácter de escape nos permitirá tener **2^(n-1)** combinaciones para una misma palabra.

5. Combinando posibilidades

Todo lo visto hasta el momento puede combinarse y utilizar todas las características anteriores en un único comando, lo que multiplica las posibilidades y dificulta localizar cadenas potencialmente peligrosas si para ello se utilizan expresiones regulares.

A título de ejemplo vamos a utilizar el propio Powershell para generar combinaciones aleatorias de dichas técnicas para formar una cadena válida.

```
function r-set {
    param(
        [UInt32] $Longitud
```

```
    )
    # n Valores comprendidos entre 0 y $Longitud
    if ($Longitud -gt 1) {
        Get-Random -InputObject (0..$Longitud) -Count (Get-Random -Minimum 1 -Maximum $Longitud) | Sort-Object
    } else {
        @()
    }
}

function case-insensitive {
    param(
        [String] $Palabra
    )
    $Palabra = $Palabra.ToLower()
    $Longitud = $Palabra.Length - 1
    r-set -Longitud $Longitud | %{
        # Mayuscula en posicion aleatoria
        $switch = $Palabra[$_].ToString()
        $Palabra = $Palabra.remove($_, 1).insert($_, $switch.ToUpper())
    }
    $Palabra
}

function caret {
    param(
        [String] $Palabra
    )
    $Longitud = $Palabra.Length - 1
    $inc = 0
    r-set -Longitud $Longitud | %{
        if ($_ -ne 0) {
            # Carácter de escape en posicion aleatoria
            $Palabra = $Palabra.insert($_ + $inc, '^')
            $inc += 1
        }
    }
    $Palabra
}

function expansion {
    param(
        [String] $Palabra
    )
    $Minimos = @{
        "-Command" = 2; "-EncodedArguments" = 9; "-EncodedCommand" = 2;
        "-ExecutionPolicy" = 3; "-File" = 2; "-InputFormat" = 2;
        "-Mta" = 2; "-NoExit" = 4; "-NoLogo" = 4;
        "-NoProfile" = 4; "-NonInteractive" = 5; "-OutputFormat" = 2;
        "-Sta" = 2; "-WindowsStyle" = 2
    }
    # n Caracteres del parámetro desde m hasta $Longitud
    if ($Minimos.ContainsKey($Palabra)) {
        $n = Get-Random -Minimum $Minimos.Get_Item($Palabra) -Maximum $Palabra.Length
        $Palabra.Substring(0, $n)
    }
}

$Comando = "powershell.exe -noprofile -windowstyle hidden -noninteractive -executionpolicy bypass -command `"Write 'It Works!'`""
$Partes = $Comando.Split("-")
$Main = case-insensitive -Palabra $Partes[0] | %{ caret -Palabra $_ }
$Parametros = @()
$Longitud = $Partes.Count - 1
(1..$Longitud) | %{
    $valor = $Partes[$_].split(" ")[(1..($Partes[$_].Length))]
    $clave = $Partes[$_].split(" ")[0]
    $clave = expansion -Palabra "-$clave" | %{ case-insensitive -Palabra $_ | %{ caret -Palabra $_ }}
    $Parametros += "$clave $valor"
}
Write-Host $Main, $Parametros
```

Dando como resultado cadenas similares a la siguiente:

```
PS C:\ P^O^weRS^HeL^l^.^EX^E^ -nOpR -^wl hidden -^N^O^Ni^N^T^e^R -^ex^EcU^tiO^n^pO^L^i bypass -C^oMmaN "Write 'It Works!'"
It Works!
```

6. Constantes

De los parámetros posibles en la interfaz de comandos powershell.exe, algunos de ellos tienen como posibles valores tipos enumerados .NET, como son **-WindowStyle** o **-ExecutionPolicy**

A la hora de especificar el valor para estas opciones, es válido utilizar su representación textual, o su valor numérico, por lo que nuevamente tenemos versatilidad para especificar un mismo valor en diferentes variantes.

Opción	Numérico	Cadena
-WindowStyle	1	Hidden
-ExecutionPolicy	0	Unrestricted
-ExecutionPolicy	4	Bypass