

PowerShell Obfuscation Lab Quick Writeup:

I ran a basic obfuscation test using PowerShell's `EncodedCommand` flag. All I did was base64 encode a `Write-Host` command and execute it with `NoProfile`. The point wasn't to make the payload fancy. I just wanted to see how obvious or hidden the logs would be.

Honestly, it's dumb how easy this is to do. But that's the whole point. Most environments still miss this because the payload stays encoded unless you have decoding in place.

MITRE ATT&CK Techniques

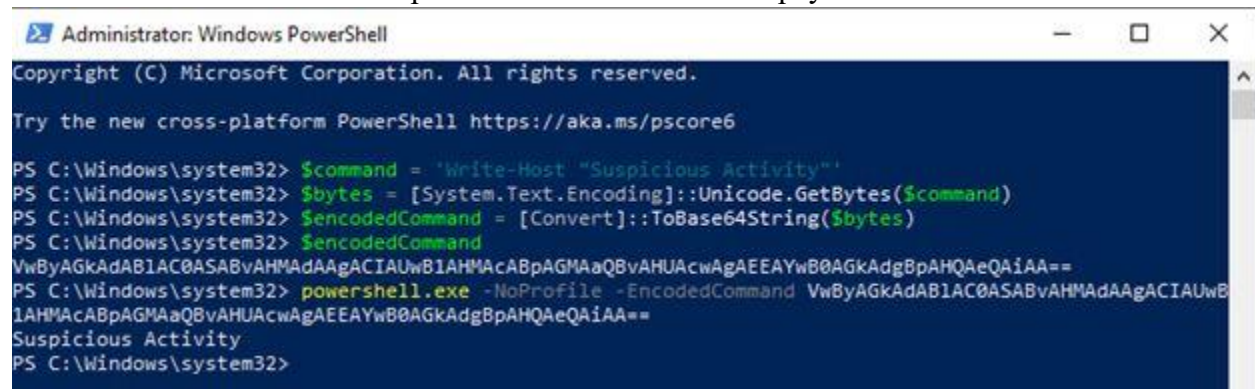
- T1059.001 PowerShell
- T1027 Obfuscated Files or Information

If you see `EncodedCommand` paired with `NoProfile`, that should trigger an alert. Every time.

What I Observed:

Step 1: Command Encoding

This was the basic PowerShell process I used to encode the payload.



```
Administrator: Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> $command = 'Write-Host "Suspicious Activity"'
PS C:\Windows\system32> $bytes = [System.Text.Encoding]::Unicode.GetBytes($command)
PS C:\Windows\system32> $encodedCommand = [Convert]::ToBase64String($bytes)
PS C:\Windows\system32> $encodedCommand
VwByAGkAdABlAC0ASABvAHMAdAAgACIAUwB1AHMacABpAGMAaQBvAHUAcwAgAEEAYwB0AGkAdgBpAHQAeQAiAA==
PS C:\Windows\system32> powershell.exe -NoProfile -EncodedCommand VwByAGkAdABlAC0ASABvAHMAdAAgACIAUwB1AHMacABpAGMAaQBvAHUAcwAgAEEAYwB0AGkAdgBpAHQAeQAiAA==
Suspicious Activity
PS C:\Windows\system32>
```

Figure 1: Encoded command

Step 2: Execution and Logging

Sysmon captured the process creation as Event ID 1. The base64 string is logged, but unless you decode it, you are not seeing what actually ran.

```
Process Create:
RuleName: -
UtcTime: 2025-06-14 20:24:51.050
ProcessGuid: {2c69427c-da93-684d-6506-000000000900}
ProcessId: 372
Image: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
FileVersion: 10.0.19041.546 (WinBuild.160101.0800)
Description: Windows PowerShell
Product: Microsoft® Windows® Operating System
Company: Microsoft Corporation
OriginalFileName: PowerShell.EXE
CommandLine: "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -EncodedCommand
VwByAGkAdABIAC0ASABvAHMAdAAgACIAUwB1AHMAcABpAGMAaQBvAHUAcwAgAEEAYwB0AGkAdgBpAHQAeQAIAA==
CurrentDirectory: C:\Windows\system32\
User: DESKTOP-QOB2LE4\admin
LogonGuid: {2c69427c-6c6f-682d-32aa-050000000000}
LogonId: 0x5AA32
```

Figure 2: Event catching the encoded command

Defensive Thoughts:

- Alert on any use of EncodedCommand.
- Decode base64 in your detection pipeline before the logs hit your analysts.
- Enable script block logging so you can see the decoded script content.
- Flag any obfuscated PowerShell that runs under elevated accounts.

My Takeaway:

Defenders waste too much time chasing advanced techniques when most attackers are using simple stuff like this. If your SOC can't catch base64 encoded PowerShell, you have bigger problems. Behavior-based detection should be the first priority.