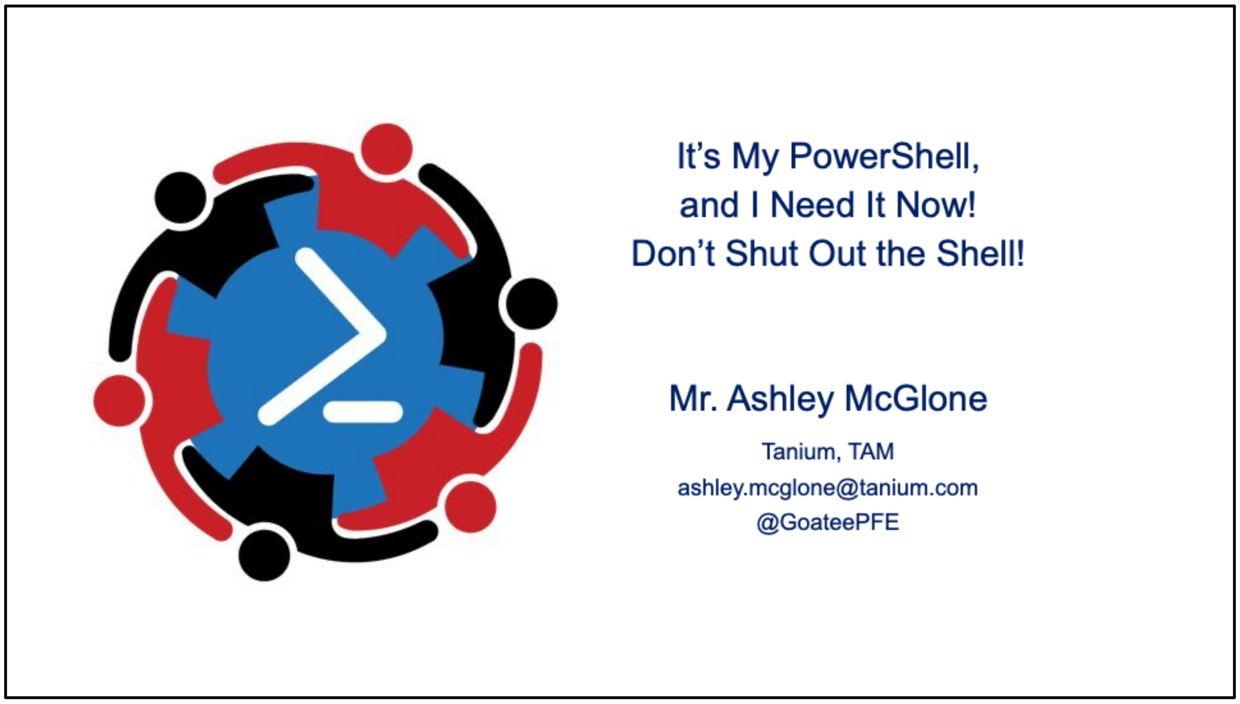


It’s My PowerShell, and I Need It Now! Don’t Shut Out the Shell!

Mr. Ashley McGlone
Tanium, TAM
ashley.mcglone@tanium.com
@GoateePFE

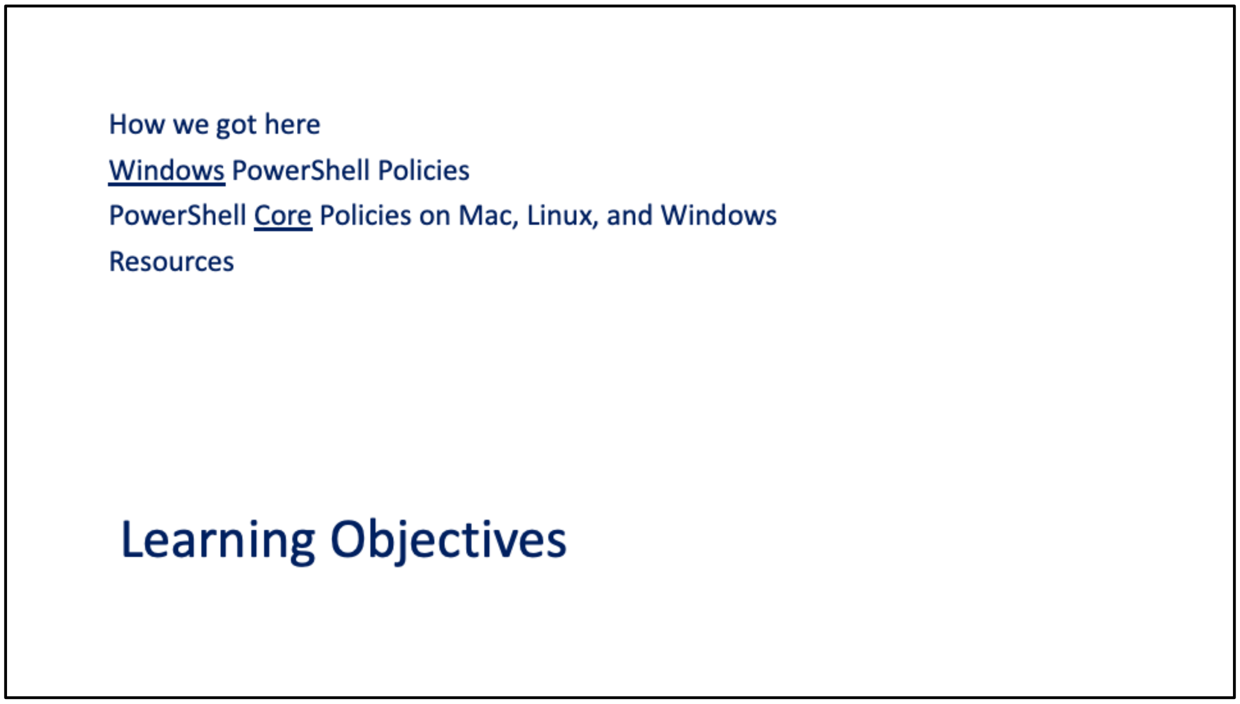
It’s My PowerShell, and I Need It Now! Don’t Shut Out the Shell!

Abstract

Tired of infosec complaining about commodity PowerShell-based malware? Are they trying to shut you down? Show them instead how to track every rogue script with PowerShell features like transcription, module logging, script block logging, and a few other tricks. Do this even in the latest PowerShell Core 7 on Windows, MacOS and Linux. Find out about a few gotchas before implementing enterprise-wide. Learn it directly from a former Microsoft insider. Take away free techniques you can use today.

Bio

Ashley McGlone is a former Microsoft Premier Field Engineer and now a Technical Account Manager at Tanium. He has a familiar face (or goatee) in the PowerShell community as a blogger, tweeter, and speaker. You can find his content on YouTube and TechNet. While at Microsoft Ashley created and delivered PowerShell training to customers around the world. Now at Tanium he is helping companies catch PowerShell malware at scale. Previous attendees of his sessions have said that he is both "informative and entertaining". Ashley's goal is to help people use PowerShell securely in the enterprise.



How we got here

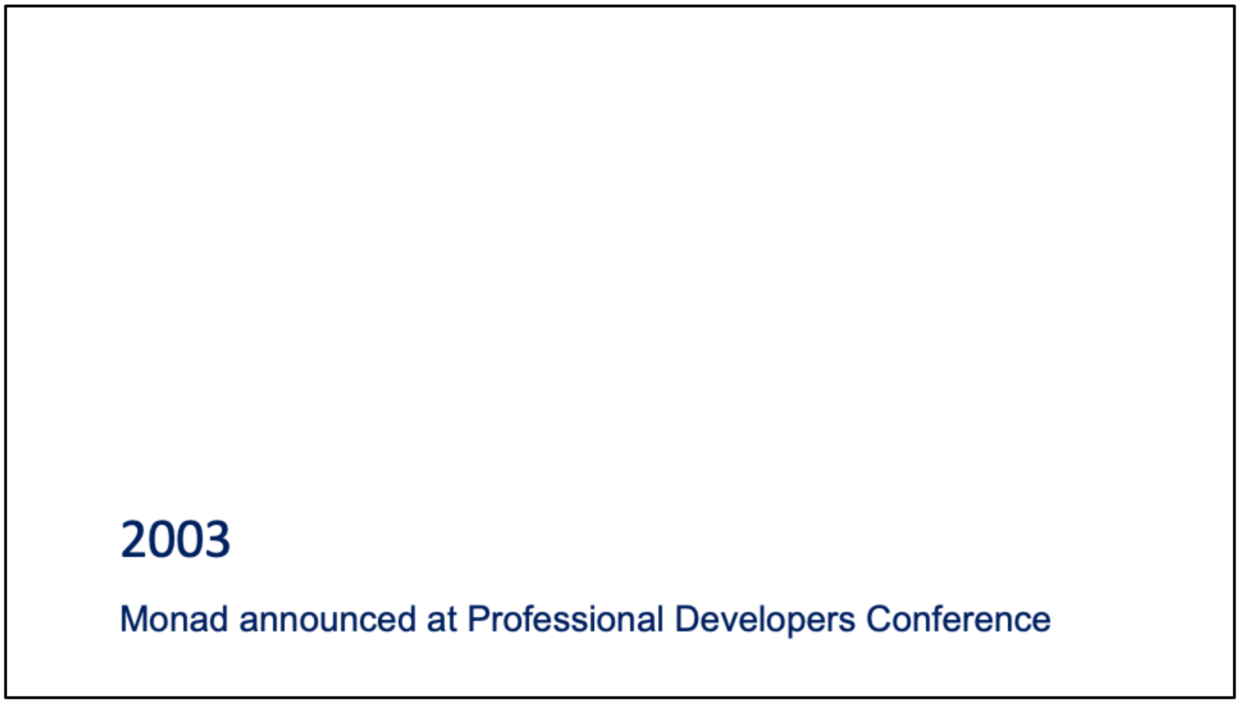
Windows PowerShell Policies

PowerShell Core Policies on Mac, Linux, and Windows

Resources

Learning Objectives

A *Brief* History of PowerShell Security



2003

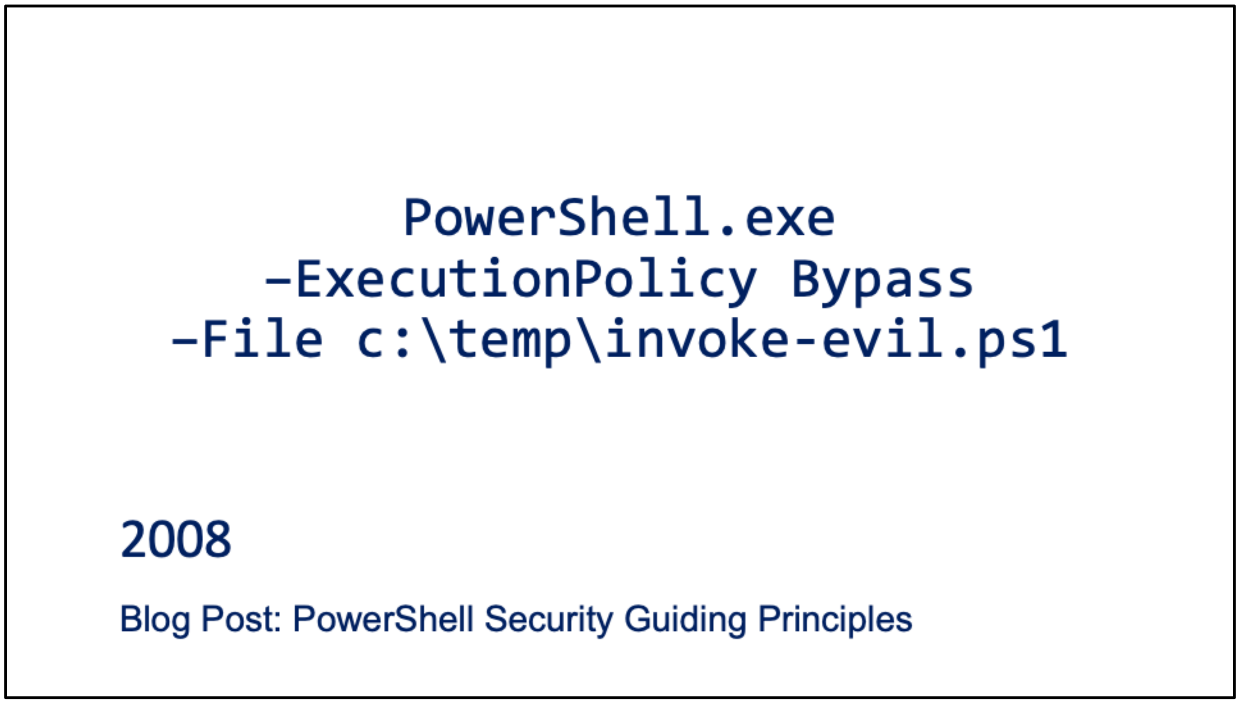
Monad announced at Professional Developers Conference

https://en.wikipedia.org/wiki/PowerShell

2006

PowerShell 1.0 released

https://en.wikipedia.org/wiki/PowerShell

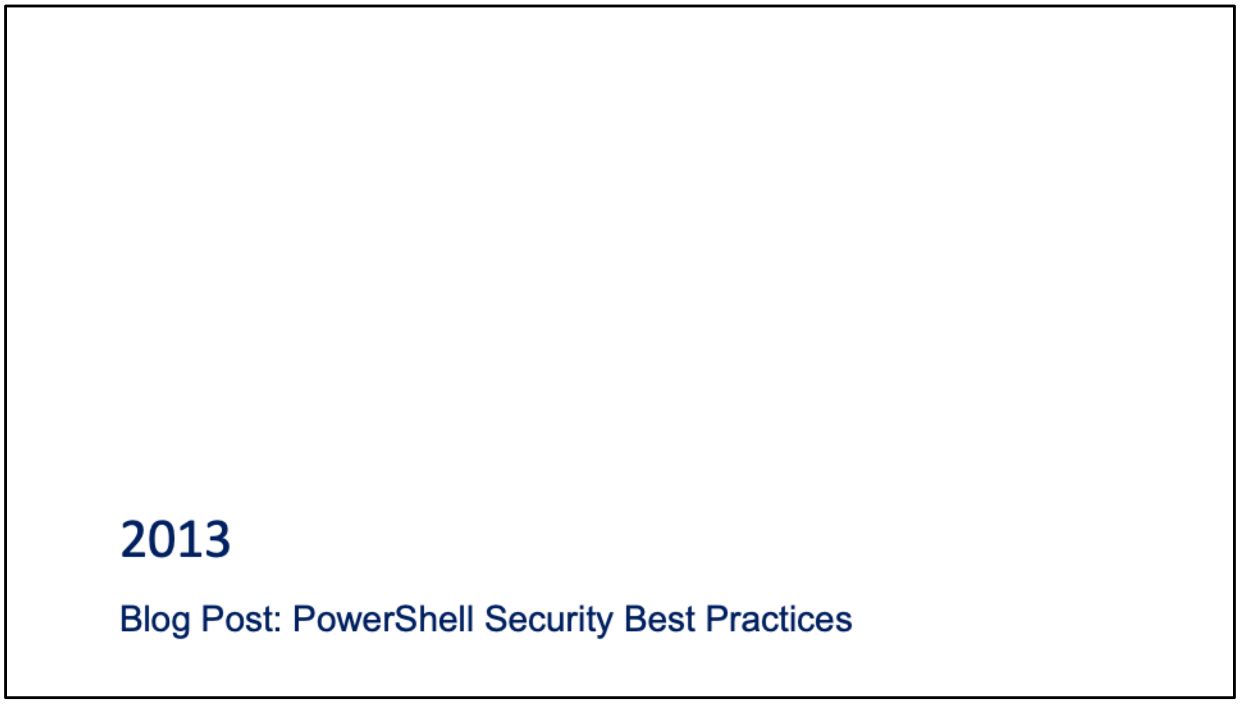


```
PowerShell.exe
-ExecutionPolicy Bypass
-File c:\temp\invoke-evil.ps1
```

2008

Blog Post: PowerShell Security Guiding Principles

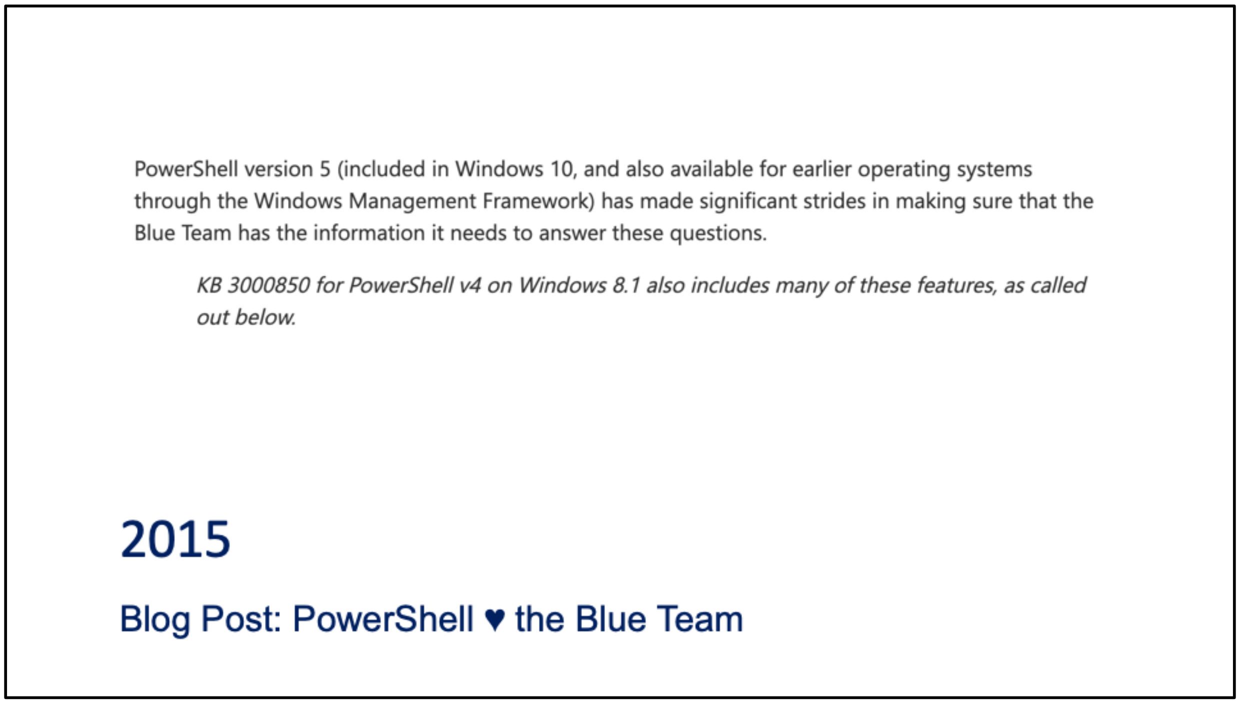
https://blogs.msdn.microsoft.com/powershell/2008/09/30/powershells-security-guiding-principles/



2013

Blog Post: PowerShell Security Best Practices

https://devblogs.microsoft.com/powershell/powershell-security-best-practices/



PowerShell version 5 (included in Windows 10, and also available for earlier operating systems through the Windows Management Framework) has made significant strides in making sure that the Blue Team has the information it needs to answer these questions.

> *KB 3000850 for PowerShell v4 on Windows 8.1 also includes many of these features, as called out below.*

2015

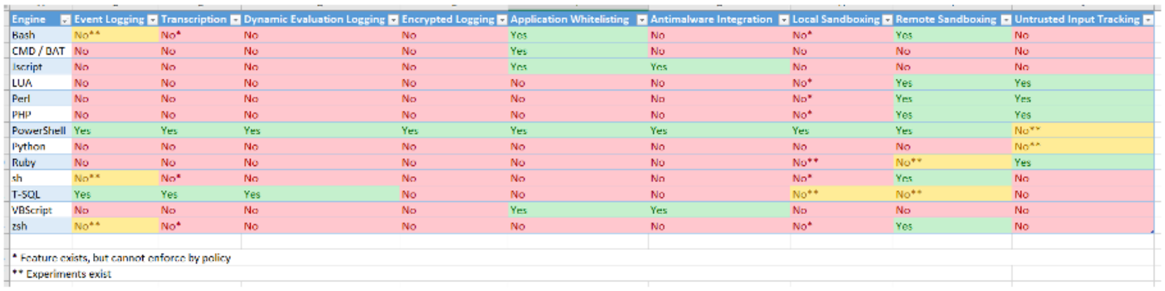
Blog Post: PowerShell ♥ the Blue Team

https://devblogs.microsoft.com/powershell/powershell-the-blue-team/

2015

PowerShell Empire premiered at BSidesLV

https://www.youtube.com/watch?v=Pq9t59w0mUI



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

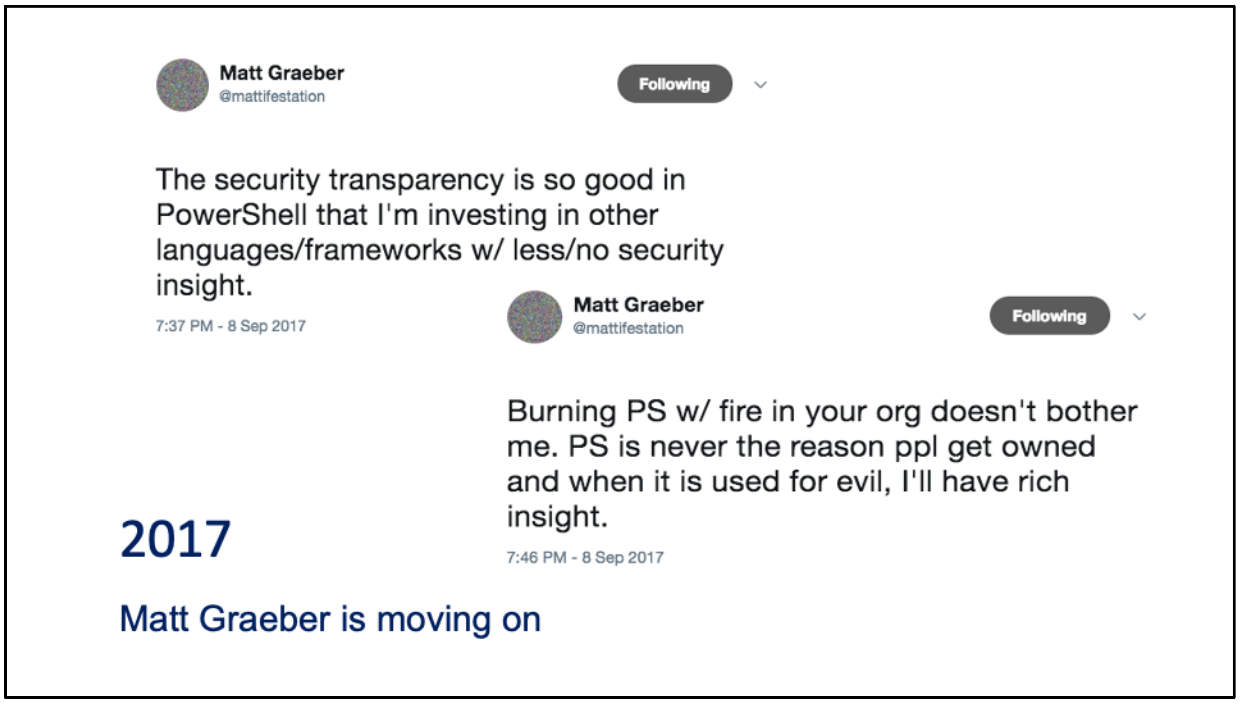
* Feature exists, but cannot enforce by policy

** Experiments exist

2017

Blog Post: A Comparison of Shell and Scripting Language Security

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



Matt Graeber
@mattifestation

Following

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

7:37 PM - 8 Sep 2017

Matt Graeber
@mattifestation

Following

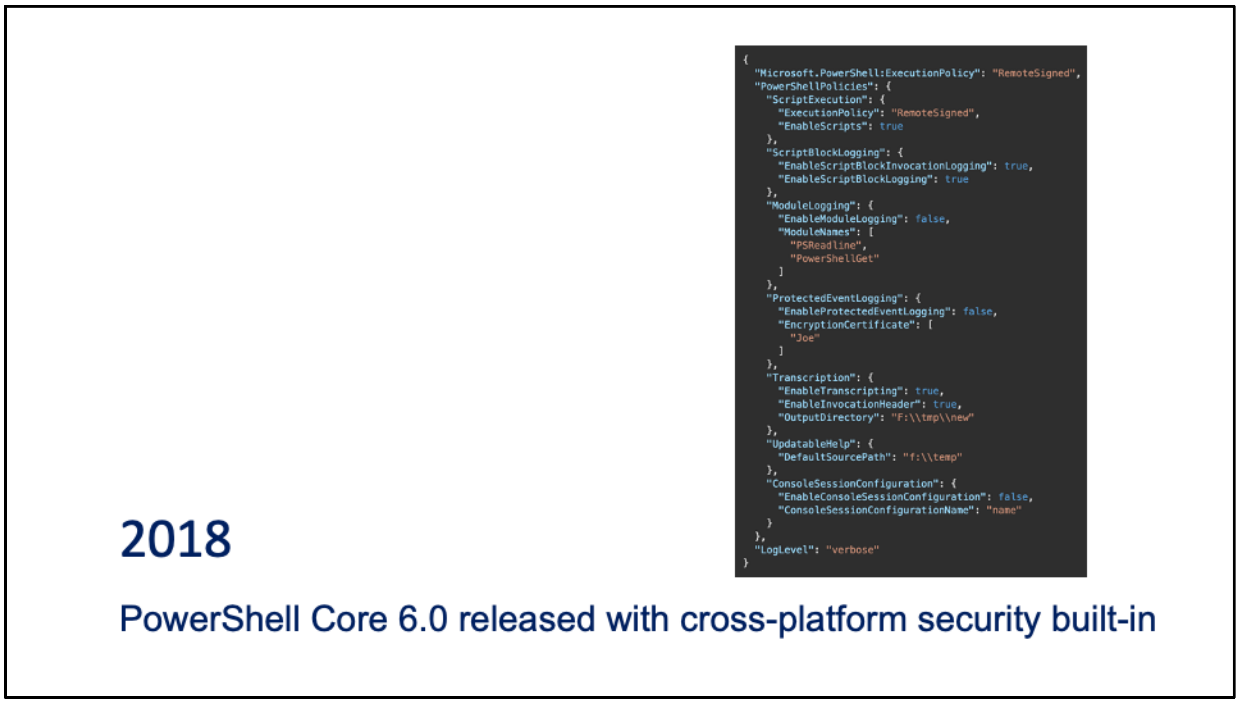
Burning PS w/ fire in your org doesn't bother me. PS is never the reason ppl get owned and when it is used for evil, I'll have rich insight.

7:46 PM - 8 Sep 2017

2017

Matt Graeber is moving on

https://twitter.com/mattifestation/status/906315527609507840

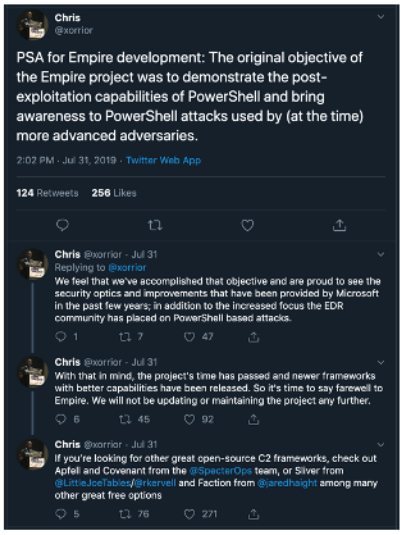


```json
{
  "Microsoft.PowerShell:ExecutionPolicy": "RemoteSigned",
  "PowerShellPolicies": {
    "ScriptExecution": {
      "ExecutionPolicy": "RemoteSigned",
      "EnableScripts": true
    },
    "ScriptBlockLogging": {
      "EnableScriptBlockInvocationLogging": true,
      "EnableScriptBlockLogging": true
    },
    "ModuleLogging": {
      "EnableModuleLogging": false,
      "ModuleNames": [
        "PSReadline",
        "PowerShellGet"
      ]
    },
    "ProtectedEventLogging": {
      "EnableProtectedEventLogging": false,
      "EncryptionCertificate": [
        "Joe"
      ]
    },
    "Transcription": {
      "EnableTranscripting": true,
      "EnableInvocationHeader": true,
      "OutputDirectory": "F:\\tmp\\new"
    },
    "UpdatableHelp": {
      "DefaultSourcePath": "f:\\temp"
    },
    "ConsoleSessionConfiguration": {
      "EnableConsoleSessionConfiguration": false,
      "ConsoleSessionConfigurationName": "name"
    }
  },
  "LogLevel": "verbose"
}
```

2018

PowerShell Core 6.0 released with cross-platform security built-in

https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_logging_non-windows?view=powershell-7.1#configuring-logging-on-a-non-windows-system
The file powershell.config.json is a JSON formatted file residing in the PowerShell $PSHOME directory.


Chris
@xorrior

PSA for Empire development: The original objective of the Empire project was to demonstrate the post-exploitation capabilities of PowerShell and bring awareness to PowerShell attacks used by (at the time) more advanced adversaries.

2:02 PM · Jul 31, 2019 · Twitter Web App

124 Retweets **256** Likes

Chris @xorrior · Jul 31
Replying to @xorrior
We feel that we've accomplished that objective and are proud to see the security optics and improvements that have been provided by Microsoft in the past few years; in addition to the increased focus the EDR community has placed on PowerShell based attacks.
1 7 47

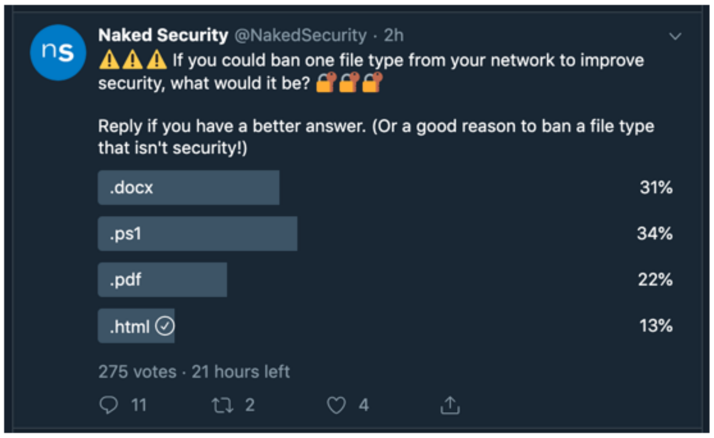
Chris @xorrior · Jul 31
With that in mind, the project's time has passed and newer frameworks with better capabilities have been released. So it's time to say farewell to Empire. We will not be updating or maintaining the project any further.
6 45 92

Chris @xorrior · Jul 31
If you're looking for other great open-source C2 frameworks, check out Apfell and Covenant from the @SpecterOps team, or Silver from @LittleJoeTables/@rkervell and Faction from @jaredhaight among many other great free options
5 76 271

2019

PowerShell Empire is dead. New tools emerge.

https://twitter.com/xorrior/status/1156626181107736576



Naked Security @NakedSecurity · 2h

⚠⚠⚠ If you could ban one file type from your network to improve security, what would it be? 🔐🔐🔐

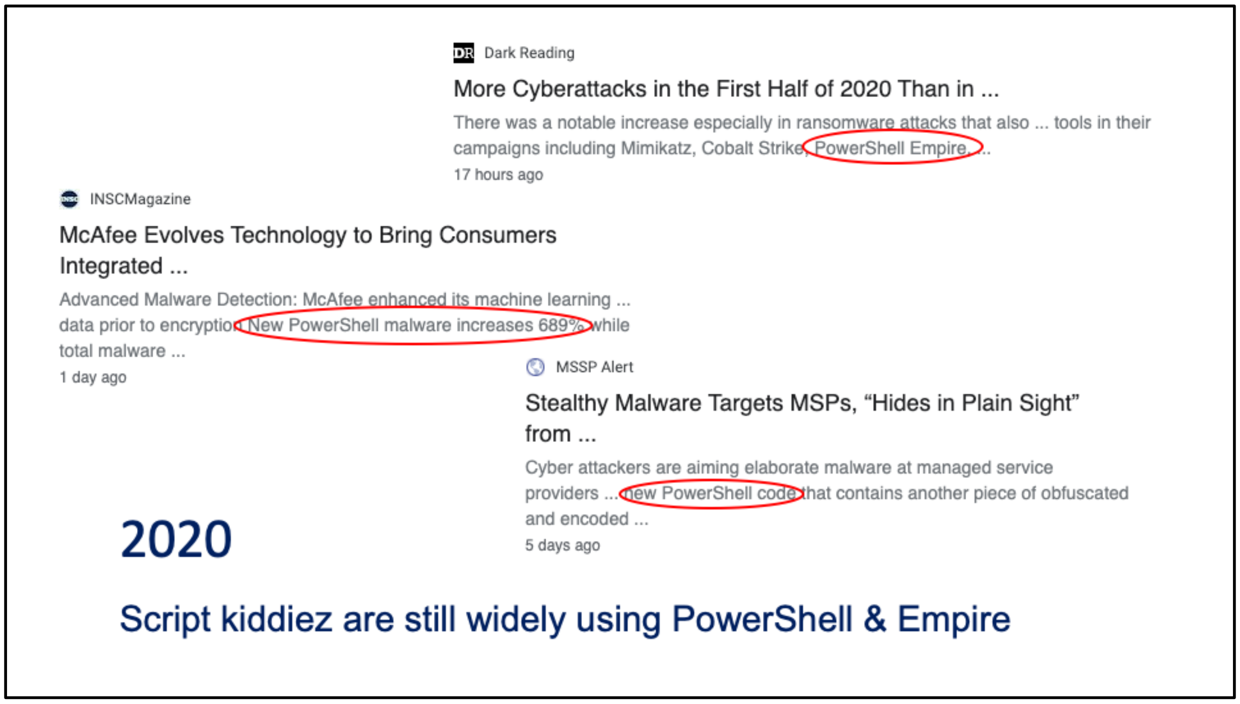
Reply if you have a better answer. (Or a good reason to ban a file type that isn't security!)

.docx	31%
.ps1	34%
.pdf	22%
.html ✓	13%

275 votes · 21 hours left

2019

Enterprises and vendors struggle to identify malicious PowerShell



Dark Reading

More Cyberattacks in the First Half of 2020 Than in ...

There was a notable increase especially in ransomware attacks that also ... tools in their campaigns including Mimikatz, Cobalt Strike, PowerShell Empire, ...

17 hours ago

INSCMagazine

McAfee Evolves Technology to Bring Consumers Integrated ...

Advanced Malware Detection: McAfee enhanced its machine learning ... data prior to encryption New PowerShell malware increases 689% while total malware ...

1 day ago

MSSP Alert

Stealthy Malware Targets MSPs, "Hides in Plain Sight" from ...

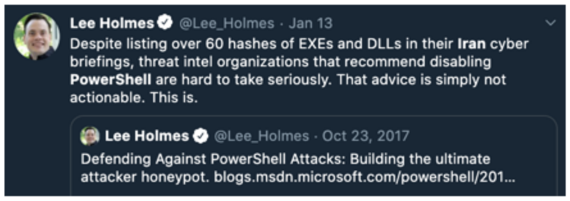
Cyber attackers are aiming elaborate malware at managed service providers ... new PowerShell code that contains another piece of obfuscated and encoded ...

5 days ago

2020

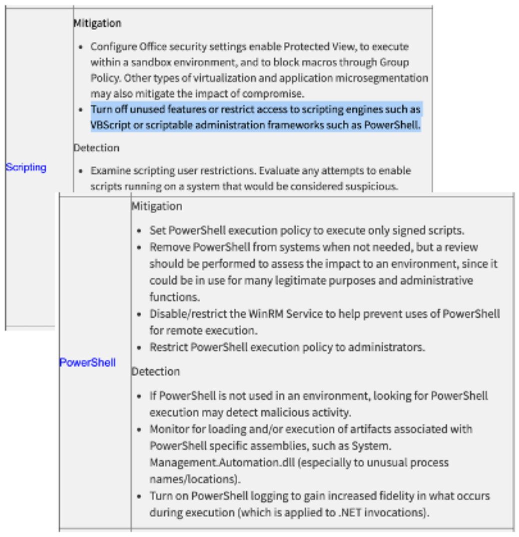
Script kiddiez are still widely using PowerShell & Empire

2020-09-16 screenshots of headlines



Lee Holmes @Lee_Holmes · Jan 13
Despite listing over 60 hashes of EXEs and DLLs in their **Iran** cyber briefings, threat intel organizations that recommend disabling **PowerShell** are hard to take seriously. That advice is simply not actionable. This is.

Lee Holmes @Lee_Holmes · Oct 23, 2017
Defending Against PowerShell Attacks: Building the ultimate attacker honeypot. blogs.msdn.microsoft.com/powershell/201...

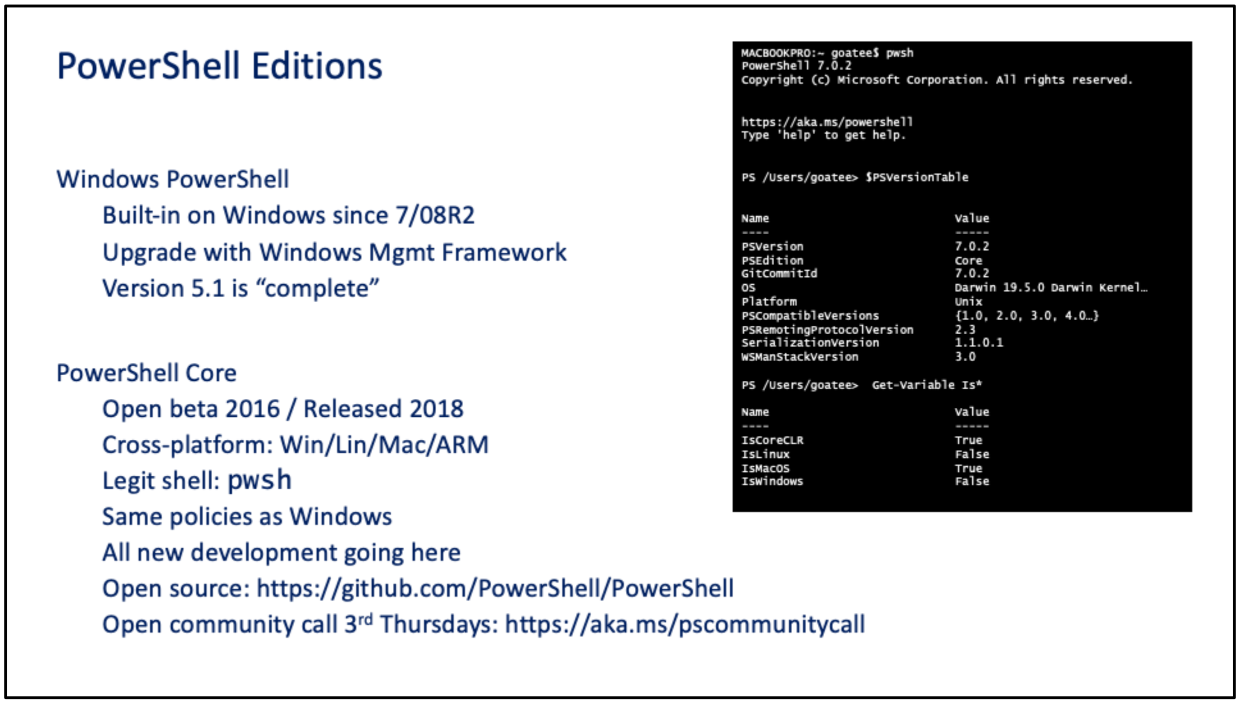


Scripting	Mitigation • Configure Office security settings enable Protected View, to execute within a sandbox environment, and to block macros through Group Policy. Other types of virtualization and application microsegmentation may also mitigate the impact of compromise. • Turn off unused features or restrict access to scripting engines such as VBScript or scriptable administration frameworks such as PowerShell. Detection • Examine scripting user restrictions. Evaluate any attempts to enable scripts running on a system that would be considered suspicious.
PowerShell	Mitigation • Set PowerShell execution policy to execute only signed scripts. • Remove PowerShell from systems when not needed, but a review should be performed to assess the impact to an environment, since it could be in use for many legitimate purposes and administrative functions. • Disable/restrict the WinRM Service to help prevent uses of PowerShell for remote execution. • Restrict PowerShell execution policy to administrators. Detection • If PowerShell is not used in an environment, looking for PowerShell execution may detect malicious activity. • Monitor for loading and/or execution of artifacts associated with PowerShell specific assemblies, such as System.Management.Automation.dll (especially to unusual process names/locations). • Turn on PowerShell logging to gain increased fidelity in what occurs during execution (which is applied to .NET invocations).

2020

Iran's cyberattacks included commodity PowerShell malware.
US CERT guidance was both helpful and not helpful.

https://twitter.com/Lee_Holmes/status/1216703900792655875
Iran CERT Bulletin Calling Out PowerShell
https://www.us-cert.gov/ncas/alerts/aa20-006a



PowerShell Editions

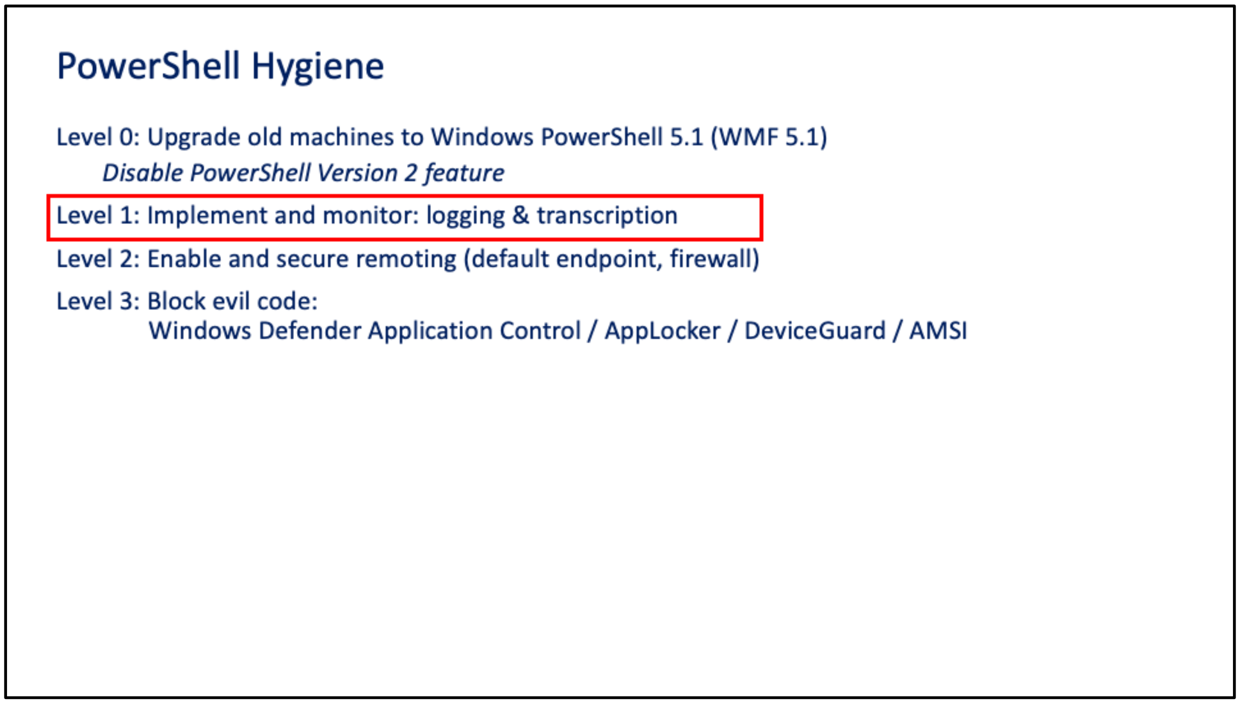
Windows PowerShell
- Built-in on Windows since 7/08R2
- Upgrade with Windows Mgmt Framework
- Version 5.1 is “complete”

PowerShell Core
- Open beta 2016 / Released 2018
- Cross-platform: Win/Lin/Mac/ARM
- Legit shell: `pwsh`
- Same policies as Windows
- All new development going here
- Open source: https://github.com/PowerShell/PowerShell
- Open community call 3rd Thursdays: https://aka.ms/pscommunitycall

```
MACBOOKPRO:~ goatee$ pwsh
PowerShell 7.0.2
Copyright (c) Microsoft Corporation. All rights reserved.


https://aka.ms/powershell
Type 'help' to get help.


PS /Users/goatee> $PSVersionTable


Name                           Value
----                           -----
PSVersion                      7.0.2
PSEdition                      Core
GitCommitId                    7.0.2
OS                             Darwin 19.5.0 Darwin Kernel…
Platform                       Unix
PSCompatibleVersions           {1.0, 2.0, 3.0, 4.0…}
PSRemotingProtocolVersion      2.3
SerializationVersion           1.1.0.1
WSManStackVersion              3.0

PS /Users/goatee>  Get-Variable Is*

Name                           Value
----                           -----
IsCoreCLR                      True
IsLinux                        False
IsMacOS                        True
IsWindows                      False
```

https://devblogs.microsoft.com/powershell/getting-started-with-powershell-core-on-windows-mac-and-linux/
https://aka.ms/pscommunitycall
https://github.com/PowerShell/PowerShell

How to PowerShell Security


PowerShell Hygiene

Level 0: Upgrade old machines to Windows PowerShell 5.1 (WMF 5.1)

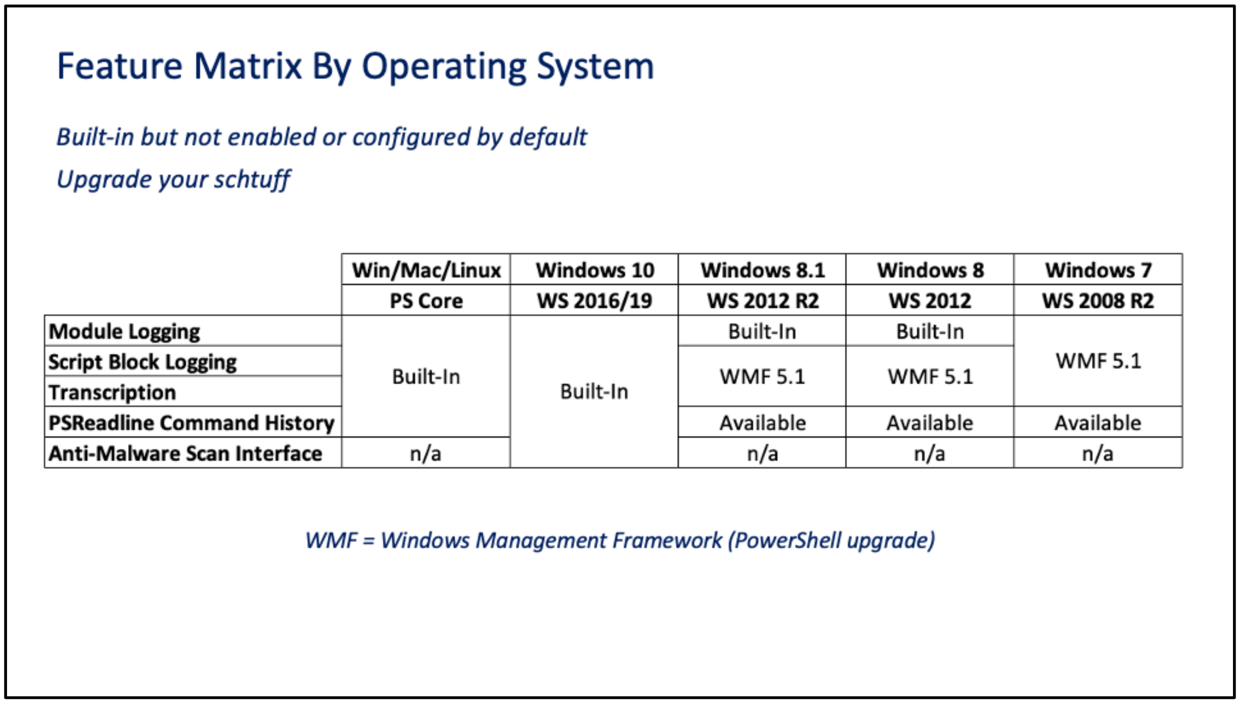
Disable PowerShell Version 2 feature

Level 1: Implement and monitor: logging & transcription

Level 2: Enable and secure remoting (default endpoint, firewall)

Level 3: Block evil code:
Windows Defender Application Control / AppLocker / DeviceGuard / AMSI

https://docs.microsoft.com/en-us/windows/security/threat-protection/windows-defender-application-control/wdac-and-applocker-overview
https://devblogs.microsoft.com/powershell/defending-against-powershell-attacks/



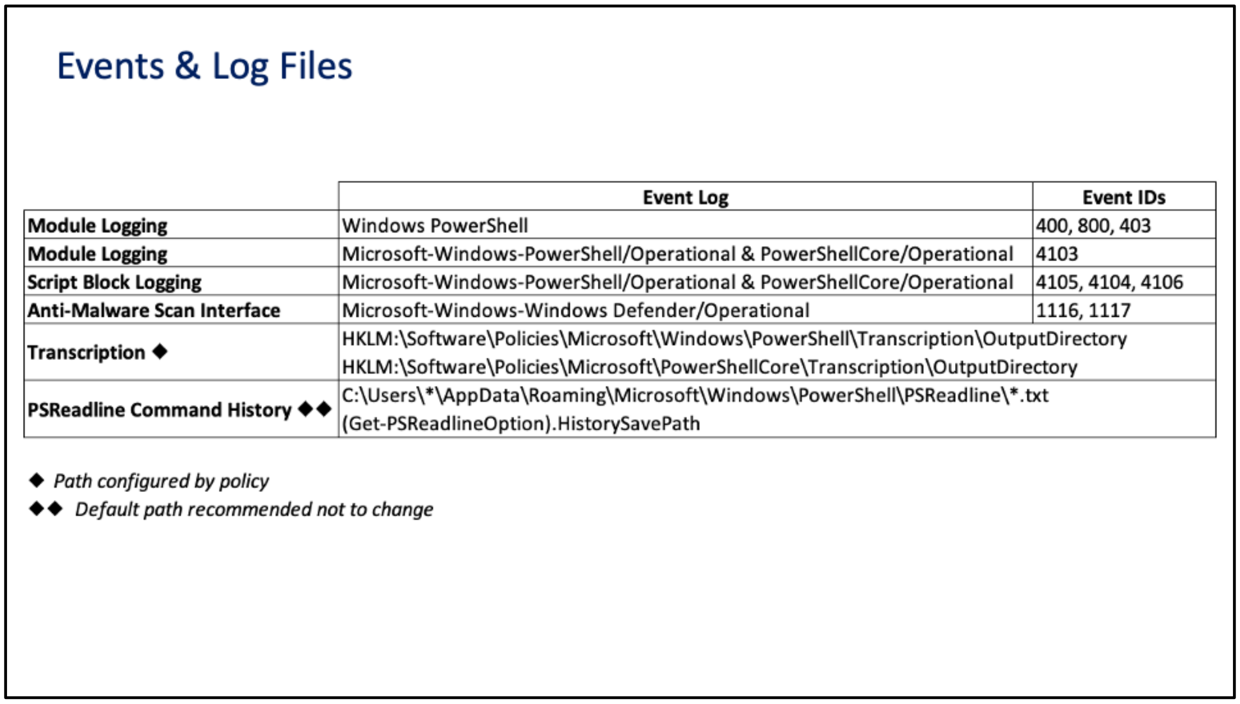
Feature Matrix By Operating System

Built-in but not enabled or configured by default

Upgrade your schtuff

	Win/Mac/Linux	**Windows 10**	**Windows 8.1**	**Windows 8**	**Windows 7**
	PS Core	**WS 2016/19**	**WS 2012 R2**	**WS 2012**	**WS 2008 R2**
Module Logging	Built-In	Built-In	Built-In	Built-In	WMF 5.1
Script Block Logging			WMF 5.1	WMF 5.1	
Transcription					
PSReadline Command History			Available	Available	Available
Anti-Malware Scan Interface	n/a		n/a	n/a	n/a

WMF = Windows Management Framework (PowerShell upgrade)



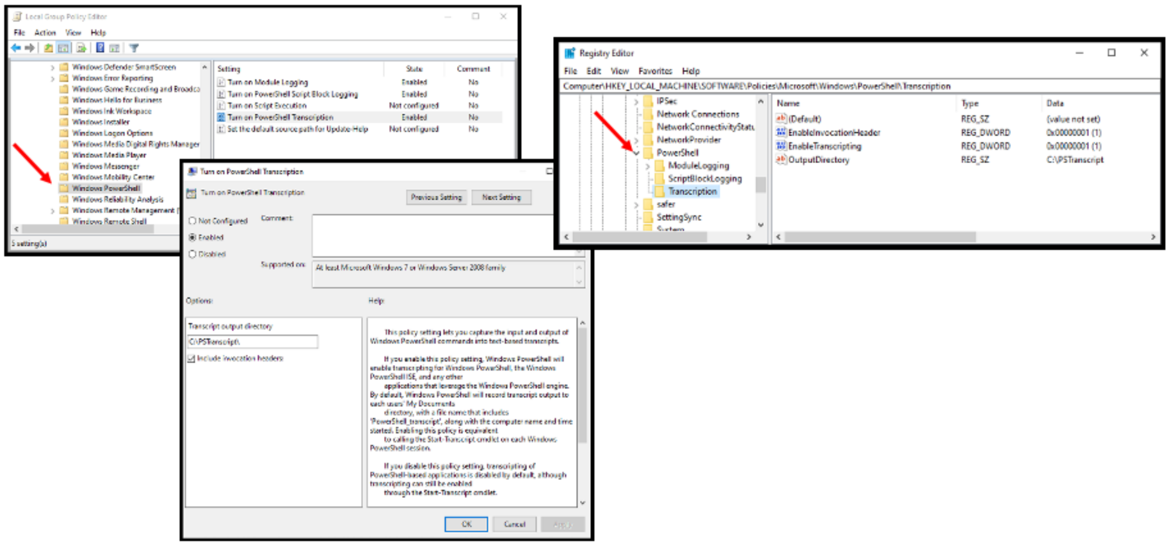
Events & Log Files

	Event Log	Event IDs
Module Logging	Windows PowerShell	400, 800, 403
Module Logging	Microsoft-Windows-PowerShell/Operational & PowerShellCore/Operational	4103
Script Block Logging	Microsoft-Windows-PowerShell/Operational & PowerShellCore/Operational	4105, 4104, 4106
Anti-Malware Scan Interface	Microsoft-Windows-Windows Defender/Operational	1116, 1117
Transcription ◆	HKLM:\Software\Policies\Microsoft\Windows\PowerShell\Transcription\OutputDirectory HKLM:\Software\Policies\Microsoft\PowerShellCore\Transcription\OutputDirectory	
PSReadline Command History ◆◆	C:\Users\*\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline\*.txt (Get-PSReadlineOption).HistorySavePath	

◆ *Path configured by policy*

◆◆ *Default path recommended not to change*

Policies on Windows – Windows PowerShell



Setting	State	Comment
Turn on Module Logging	Enabled	No
Turn on PowerShell Script Block Logging	Enabled	No
Turn on Script Execution	Not configured	No
Turn on PowerShell Transcription	Enabled	No
Set the default source path for Update-Help	Not configured	No

Turn on PowerShell Transcription

Not Configured [✓ Enabled] Disabled

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

Options:

Transcript output directory: C:\PSTranscript\

[✓] Include invocation headers

Help:

This policy setting lets you capture the input and output of Windows PowerShell commands into text-based transcripts.

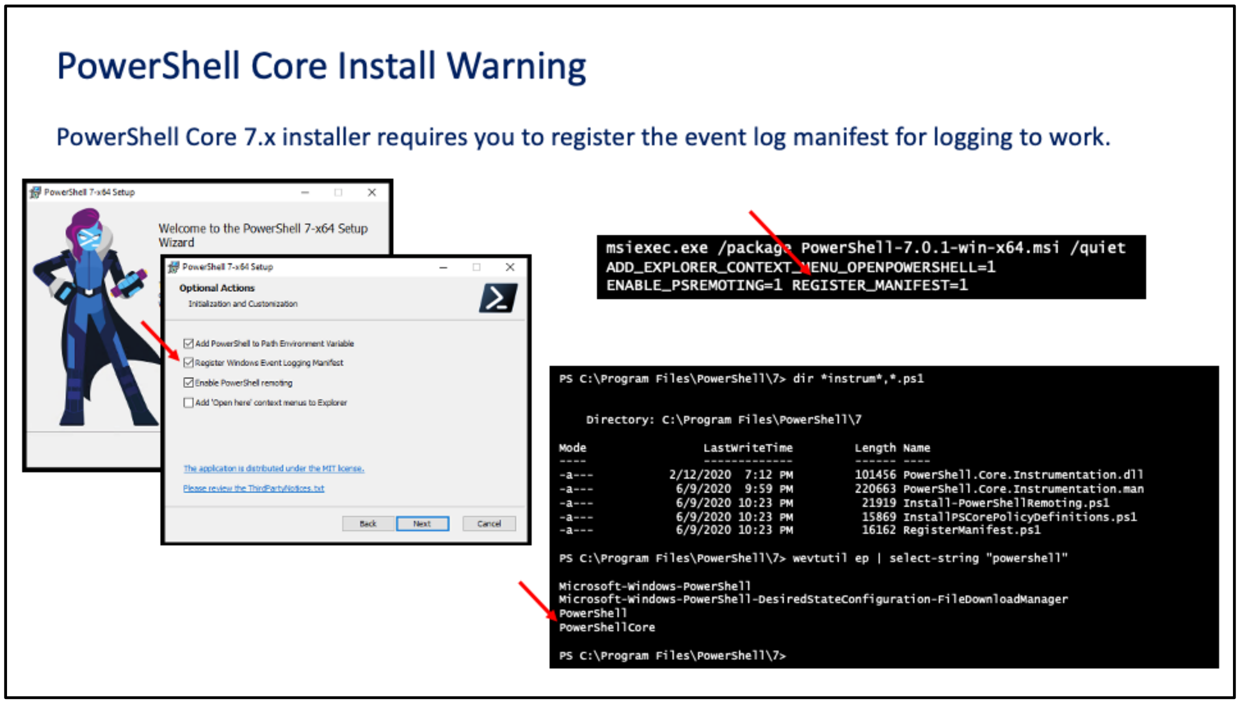
If you enable this policy setting, Windows PowerShell will enable transcripting for Windows PowerShell, the Windows PowerShell ISE, and any other applications that leverage the Windows PowerShell engine. By default, Windows PowerShell will record transcript output to each users' My Documents directory, with a file name that includes 'PowerShell_transcript', along with the computer name and time started. Enabling this policy is equivalent to calling the Start-Transcript cmdlet on each Windows PowerShell session.

If you disable this policy setting, transcripting of PowerShell-based applications is disabled by default, although transcripting can still be enabled through the Start-Transcript cmdlet.

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

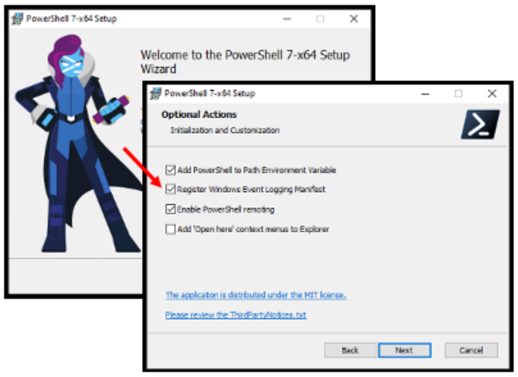
Name	Type	Data
(Default)	REG_SZ	(value not set)
EnableInvocationHeader	REG_DWORD	0x00000001 (1)
EnableTranscripting	REG_DWORD	0x00000001 (1)
OutputDirectory	REG_SZ	C:\PSTranscript

https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_group_policy_settings?view=powershell-7.1



PowerShell Core Install Warning

PowerShell Core 7.x installer requires you to register the event log manifest for logging to work.



PowerShell 7-x64 Setup

Welcome to the PowerShell 7-x64 Setup Wizard

PowerShell 7-x64 Setup

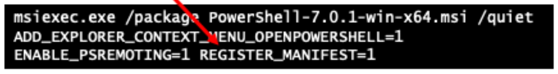
Optional Actions

Initialization and Customization

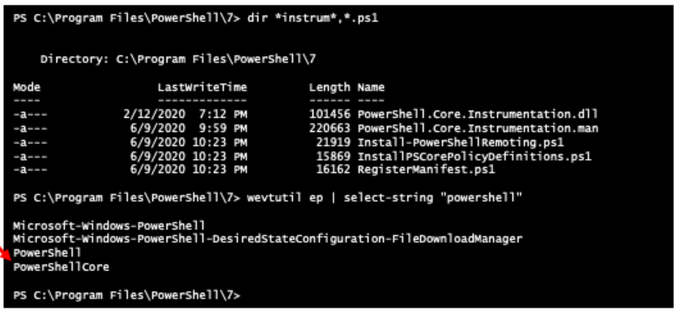
- [✓] Add PowerShell to Path Environment Variable
- [✓] Register Windows Event Logging Manifest
- [✓] Enable PowerShell remoting
- [ ] Add 'Open here' context menus to Explorer

The application is distributed under the MIT license.

Please review the ThirdPartyNotices.txt

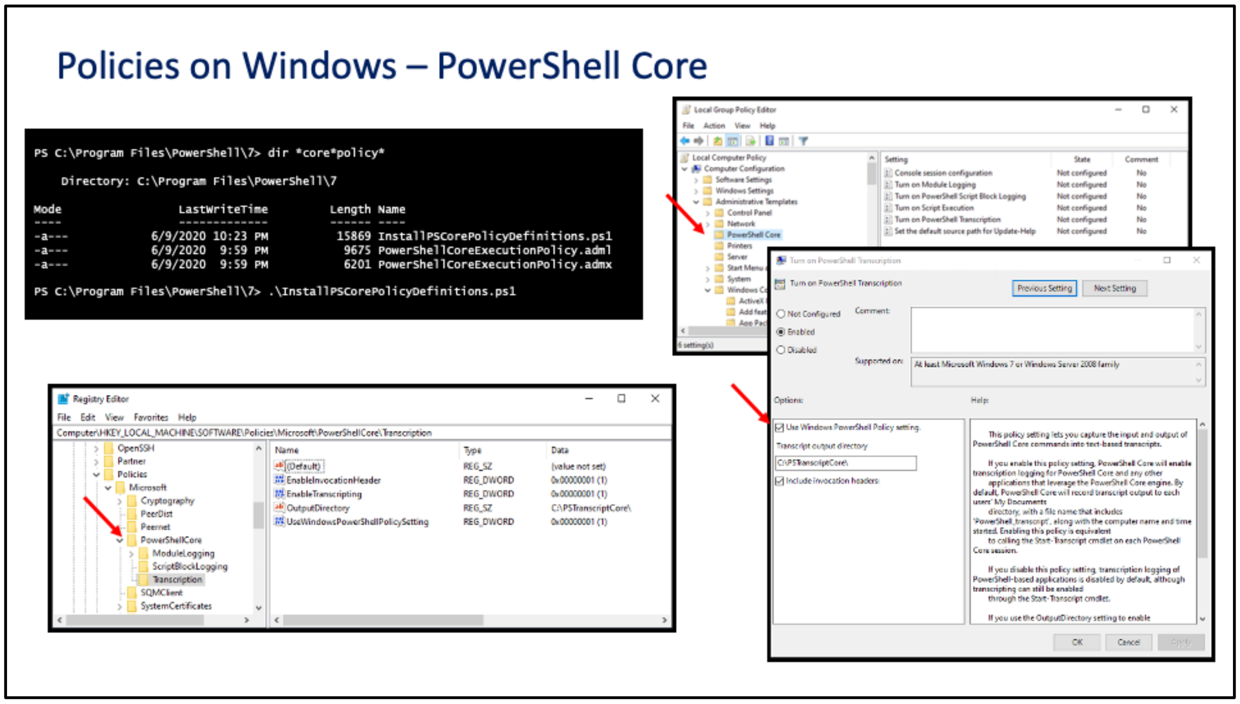


```
msiexec.exe /package PowerShell-7.0.1-win-x64.msi /quiet
ADD_EXPLORER_CONTEXT_MENU_OPENPOWERSHELL=1
ENABLE_PSREMOTING=1 REGISTER_MANIFEST=1
```



```
PS C:\Program Files\PowerShell\7> dir *instrum*,*.ps1


    Directory: C:\Program Files\PowerShell\7

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---           2/12/2020   7:12 PM         101456 PowerShell.Core.Instrumentation.dll
-a---            6/9/2020   9:59 PM         220663 PowerShell.Core.Instrumentation.man
-a---            6/9/2020  10:23 PM          21919 Install-PowerShellRemoting.ps1
-a---            6/9/2020  10:23 PM          15869 InstallPSCorePolicyDefinitions.ps1
-a---            6/9/2020  10:23 PM          16162 RegisterManifest.ps1

PS C:\Program Files\PowerShell\7> wevtutil ep | select-string "powershell"

Microsoft-Windows-PowerShell
Microsoft-Windows-PowerShell-DesiredStateConfiguration-FileDownloadManager
PowerShell
PowerShellCore

PS C:\Program Files\PowerShell\7>
```

https://docs.microsoft.com/en-us/powershell/scripting/install/installing-powershell-core-on-windows?view=powershell-7#installing-the-msi-package



Policies on Windows – PowerShell Core

```
PS C:\Program Files\PowerShell\7> dir *core*policy*

    Directory: C:\Program Files\PowerShell\7

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---           6/9/2020 10:23 PM          15869 InstallPSCorePolicyDefinitions.ps1
-a---           6/9/2020  9:59 PM           9675 PowerShellCoreExecutionPolicy.adml
-a---           6/9/2020  9:59 PM           6201 PowerShellCoreExecutionPolicy.admx

PS C:\Program Files\PowerShell\7> .\InstallPSCorePolicyDefinitions.ps1
```

Local Group Policy Editor

Local Computer Policy > Computer Configuration > Administrative Templates > PowerShell Core

Setting	State	Comment
Console session configuration	Not configured	No
Turn on Module Logging	Not configured	No
Turn on PowerShell Script Block Logging	Not configured	No
Turn on Script Execution	Not configured	No
Turn on PowerShell Transcription	Not configured	No
Set the default source path for Update-Help	Not configured	No

6 setting(s)

Turn on PowerShell Transcription

○ Not Configured [✓ Enabled] ○ Disabled

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

Options:

[✓] Use Windows PowerShell Policy setting.

Transcript output directory: C:\PSTranscriptCore\

[✓] Include invocation headers:

Help:

This policy setting lets you capture the input and output of PowerShell Core commands into text-based transcripts.

If you enable this policy setting, PowerShell Core will enable transcription logging for PowerShell Core and any other applications that leverage the PowerShell Core engine. By default, PowerShell Core will record transcript output to each users' My Documents directory, with a file name that includes 'PowerShell_transcript', along with the computer name and time started. Enabling this policy is equivalent to calling the Start-Transcript cmdlet on each PowerShell Core session.

If you disable this policy setting, transcription logging of PowerShell-based applications is disabled by default, although transcripting can still be enabled through the Start-Transcript cmdlet.

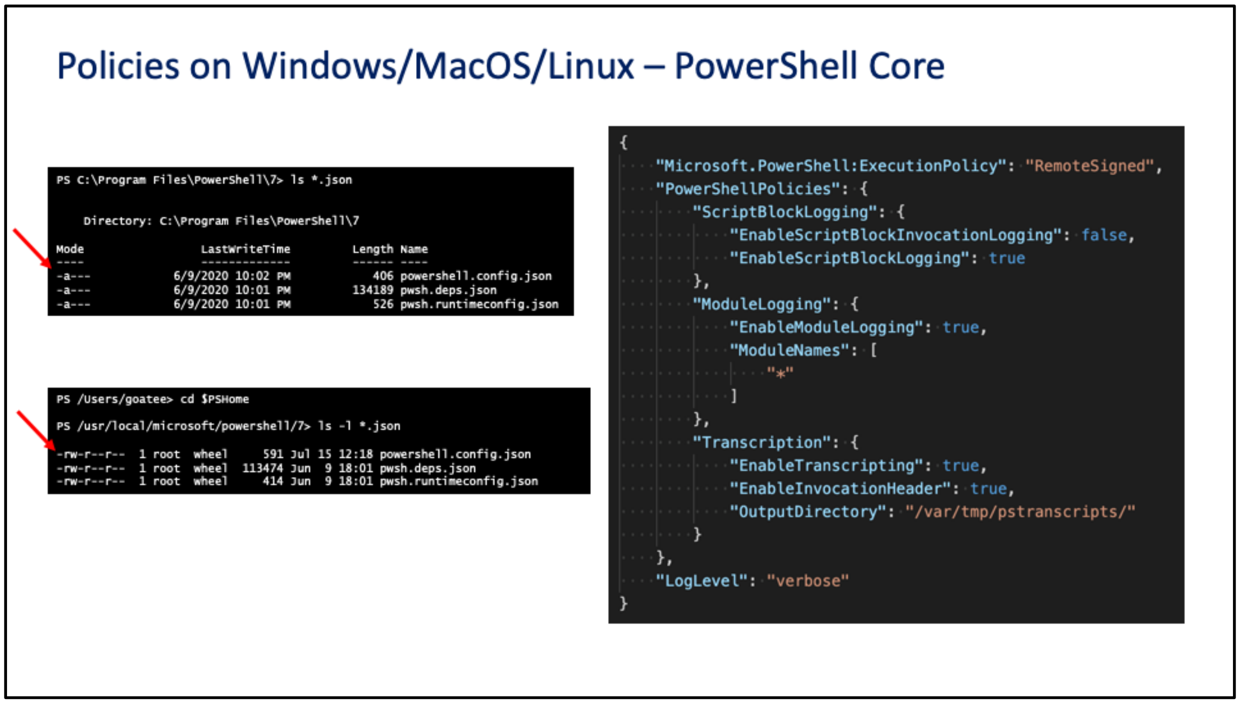
If you use the OutputDirectory setting to enable

Registry Editor

Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\PowerShellCore\Transcription

Name	Type	Data
(Default)	REG_SZ	(value not set)
EnableInvocationHeader	REG_DWORD	0x00000001 (1)
EnableTranscripting	REG_DWORD	0x00000001 (1)
OutputDirectory	REG_SZ	C:\PSTranscriptCore\
UseWindowsPowerShellPolicySetting	REG_DWORD	0x00000001 (1)

https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_group_policy_settings?view=powershell-7.1



Policies on Windows/MacOS/Linux – PowerShell Core

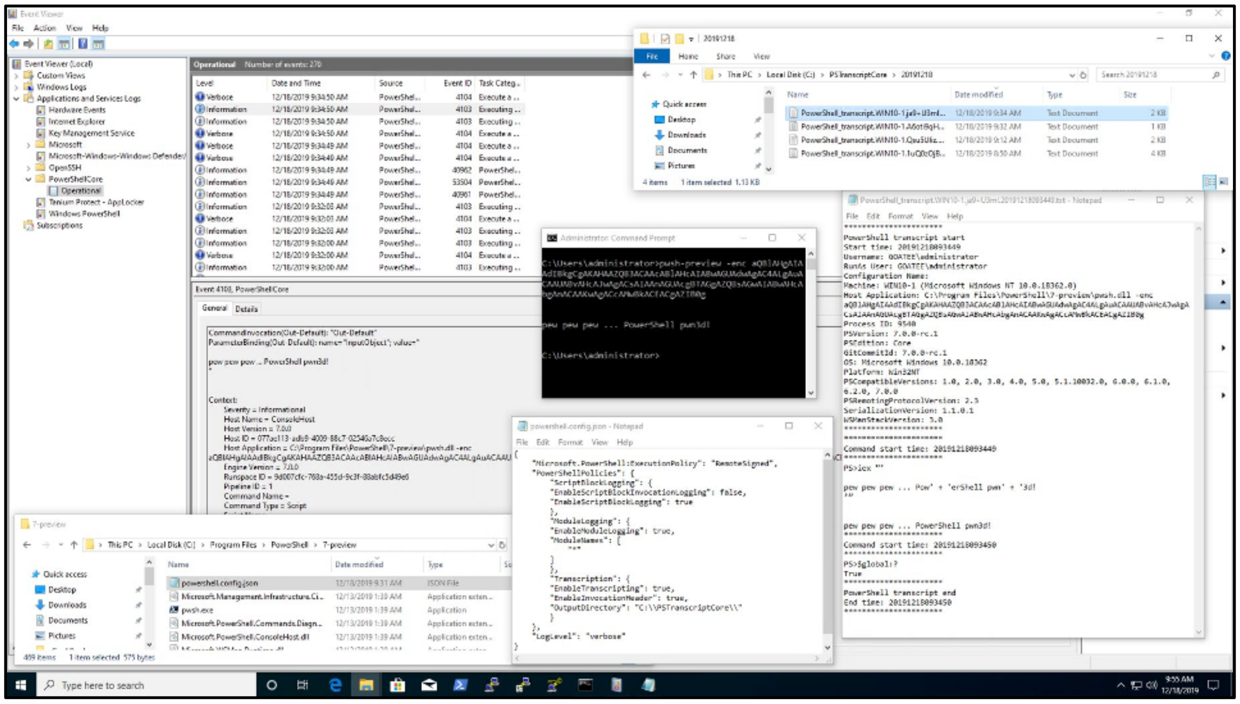
```
PS C:\Program Files\PowerShell\7> ls *.json


    Directory: C:\Program Files\PowerShell\7

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---           6/9/2020 10:02 PM             406 powershell.config.json
-a---           6/9/2020 10:01 PM          134189 pwsh.deps.json
-a---           6/9/2020 10:01 PM             526 pwsh.runtimeconfig.json
```

```
PS /Users/goatee> cd $PSHome

PS /usr/local/microsoft/powershell/7> ls -l *.json

-rw-r--r--  1 root  wheel     591 Jul 15 12:18 powershell.config.json
-rw-r--r--  1 root  wheel  113474 Jun  9 18:01 pwsh.deps.json
-rw-r--r--  1 root  wheel     414 Jun  9 18:01 pwsh.runtimeconfig.json
```

```
{
    "Microsoft.PowerShell:ExecutionPolicy": "RemoteSigned",
    "PowerShellPolicies": {
        "ScriptBlockLogging": {
            "EnableScriptBlockInvocationLogging": false,
            "EnableScriptBlockLogging": true
        },
        "ModuleLogging": {
            "EnableModuleLogging": true,
            "ModuleNames": [
                "*"
            ]
        },
        "Transcription": {
            "EnableTranscripting": true,
            "EnableInvocationHeader": true,
            "OutputDirectory": "/var/tmp/pstranscripts/"
        }
    },
    "LogLevel": "verbose"
}
```

https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_logging_non-windows?view=powershell-7.1#configuring-logging-on-a-non-windows-system
The file powershell.config.json is a JSON formatted file residing in the PowerShell $PSHOME directory.

```
{
    "Microsoft.PowerShell:ExecutionPolicy": "RemoteSigned",
    "PowerShellPolicies": {
        "ScriptBlockLogging": {
        "EnableScriptBlockInvocationLogging": false,
        "EnableScriptBlockLogging": true
        },
        "ModuleLogging": {
        "EnableModuleLogging": true,
        "ModuleNames": [
            "*"
        ]
        },
        "Transcription": {
        "EnableTranscripting": true,
        "EnableInvocationHeader": true,
        "OutputDirectory": "/var/tmp/pstranscripts/"
        }
    },
    "LogLevel": "verbose"
}
```



```
Operational   Number of events: 270
Level         Date and Time            Source        Event ID  Task Categ...
Verbose       12/18/2019 9:34:50 AM    PowerShel...      4104  Execute a ...
Information   12/18/2019 9:34:50 AM    PowerShel...      4103  Executing ...
Information   12/18/2019 9:34:50 AM    PowerShel...      4103  Executing ...
Verbose       12/18/2019 9:34:50 AM    PowerShel...      4104  Execute a ...
Verbose       12/18/2019 9:34:49 AM    PowerShel...      4104  Execute a ...
Verbose       12/18/2019 9:34:49 AM    PowerShel...      4104  Execute a ...
Information   12/18/2019 9:34:49 AM    PowerShel...     40962  PowerShel...
Information   12/18/2019 9:34:49 AM    PowerShel...     53504  PowerShel...
Information   12/18/2019 9:34:49 AM    PowerShel...     40961  PowerShel...
Information   12/18/2019 9:32:03 AM    PowerShel...      4103  Executing ...
Verbose       12/18/2019 9:32:03 AM    PowerShel...      4104  Execute a ...
Information   12/18/2019 9:32:03 AM    PowerShel...      4103  Executing ...
Information   12/18/2019 9:32:00 AM    PowerShel...      4103  Executing ...
Verbose       12/18/2019 9:32:00 AM    PowerShel...      4104  Execute a ...
Information   12/18/2019 9:32:00 AM    PowerShel...      4103  Executing ...
```

Event 4103, PowerShellCore

```
CommandInvocation(Out-Default): "Out-Default"
ParameterBinding(Out-Default): name="InputObject"; value="

pew pew pew ... PowerShell pwn3d!
"

Context:
        Severity = Informational
        Host Name = ConsoleHost
        Host Version = 7.0.0
        Host ID = 077ae113-adb9-4009-88c7-02546a7c8ecc
        Host Application = C:\Program Files\PowerShell\7-preview\pwsh.dll -enc
aQBlAHgAIAAdIBkgCgAKAHAAZQB3ACAAcABlAHcAIABwAGUAdwAgAC4ALgAuACAAU
        Engine Version = 7.0.0
        Runspace ID = 9d007cfc-768a-455d-9c3f-88abfc5d49e6
        Pipeline ID = 1
        Command Name =
        Command Type = Script
```

```
C:\Users\administrator>pwsh-preview -enc aQBlAHgAIAAdIBkgCgAKAHAAZQB3ACAAcABlAHcAIABwAGUAdwAgAC4ALgAuACAAUABvAHcAJwAgACsAIAAnAGUAcgBTAGgAZQBsAGwAIABwAHcAbgAnACAAKwAgACcAMwBkACEACgAZIB0g


pew pew pew ... PowerShell pwn3d!


C:\Users\administrator>
```

```
PowerShell_transcript.WIN10-1.ja9+U3ml...  12/18/2019 9:34 AM  Text Document  2 KB
PowerShell_transcript.WIN10-1.A6ot8qH...   12/18/2019 9:32 AM  Text Document  1 KB
PowerShell_transcript.WIN10-1.Qau5Ukz...   12/18/2019 9:12 AM  Text Document  2 KB
PowerShell_transcript.WIN10-1.1uQ0zOjB...  12/18/2019 8:50 AM  Text Document  4 KB
```

PowerShell_transcript.WIN10-1.ja9+U3ml.20191218093449.txt - Notepad

```
**********************
PowerShell transcript start
Start time: 20191218093449
Username: GOATEE\administrator
RunAs User: GOATEE\administrator
Configuration Name:
Machine: WIN10-1 (Microsoft Windows NT 10.0.18362.0)
Host Application: C:\Program Files\PowerShell\7-preview\pwsh.dll -enc
aQBlAHgAIAAdIBkgCgAKAHAAZQB3ACAAcABlAHcAIABwAGUAdwAgAC4ALgAuACAAUABvAHcAJwAgA
CsAIAAnAGUAcgBTAGgAZQBsAGwAIABwAHcAbgAnACAAKwAgACcAMwBkACEACgAZIB0g
Process ID: 9540
PSVersion: 7.0.0-rc.1
PSEdition: Core
GitCommitId: 7.0.0-rc.1
OS: Microsoft Windows 10.0.18362
Platform: Win32NT
PSCompatibleVersions: 1.0, 2.0, 3.0, 4.0, 5.0, 5.1.10032.0, 6.0.0, 6.1.0,
6.2.0, 7.0.0
PSRemotingProtocolVersion: 2.3
SerializationVersion: 1.1.0.1
WSManStackVersion: 3.0
**********************
**********************
Command start time: 20191218093449
**********************
PS>iex "'

pew pew pew ... Pow' + 'erShell pwn' + '3d!
'"


pew pew pew ... PowerShell pwn3d!
**********************
Command start time: 20191218093450
**********************
PS>$global:?
True
**********************
PowerShell transcript end
End time: 20191218093450
**********************
```

```
powershell.config.json                      12/18/2019 9:31 AM  JSON File
Microsoft.Management.Infrastructure.Ci...   12/13/2019 1:39 AM  Application exten...
pwsh.exe                                    12/13/2019 1:39 AM  Application
Microsoft.PowerShell.Commands.Diagn...      12/13/2019 1:39 AM  Application exten...
Microsoft.PowerShell.ConsoleHost.dll        12/13/2019 1:39 AM  Application exten...
```

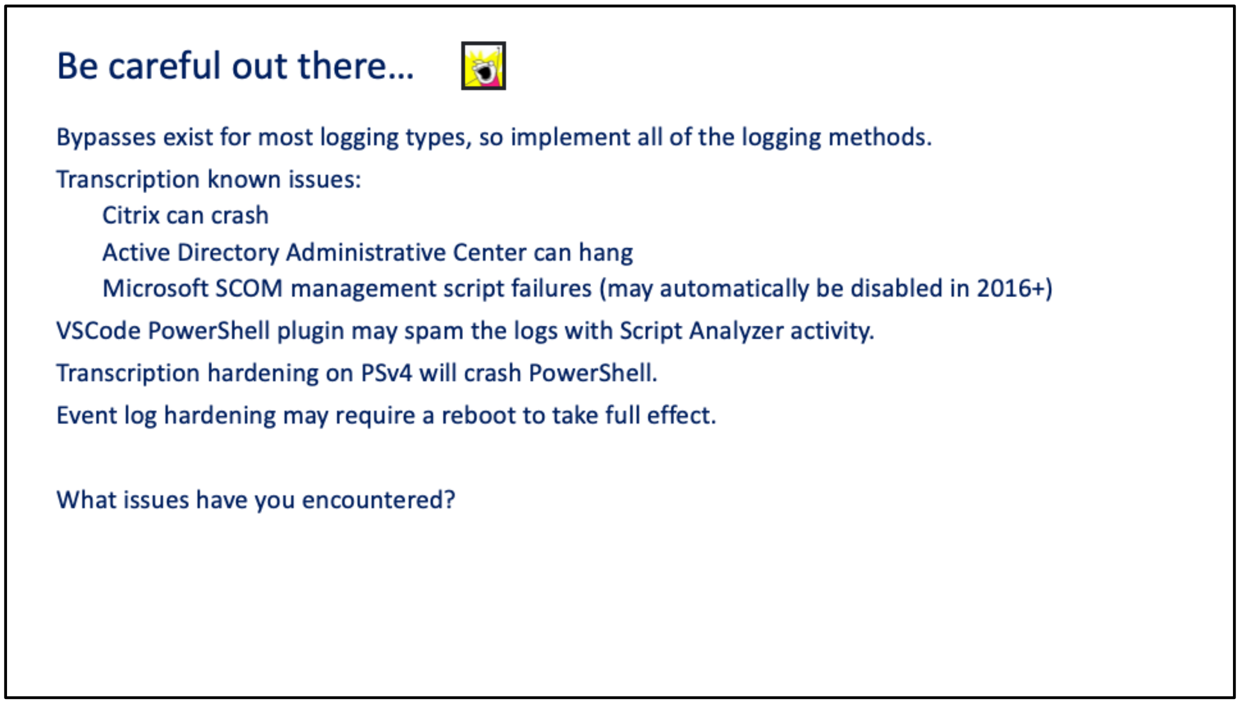
powershell.config.json - Notepad

```
{
    "Microsoft.PowerShell:ExecutionPolicy": "RemoteSigned",
    "PowerShellPolicies": {
        "ScriptBlockLogging": {
        "EnableScriptBlockInvocationLogging": false,
        "EnableScriptBlockLogging": true
        },
        "ModuleLogging": {
        "EnableModuleLogging": true,
        "ModuleNames": [
            "*"
        ]
        },
        "Transcription": {
        "EnableTranscripting": true,
        "EnableInvocationHeader": true,
        "OutputDirectory": "C:\\PSTranscriptCore\\"
        }
    },
    "LogLevel": "verbose"
}
```

https://twitter.com/GoateePFE/status/1207320954264784897

Pro Tip: PowerShell policy changes will take effect in a <u>new</u> session.

DEMO



Be careful out there…

Bypasses exist for most logging types, so implement all of the logging methods.

Transcription known issues:

- Citrix can crash
- Active Directory Administrative Center can hang
- Microsoft SCOM management script failures (may automatically be disabled in 2016+)

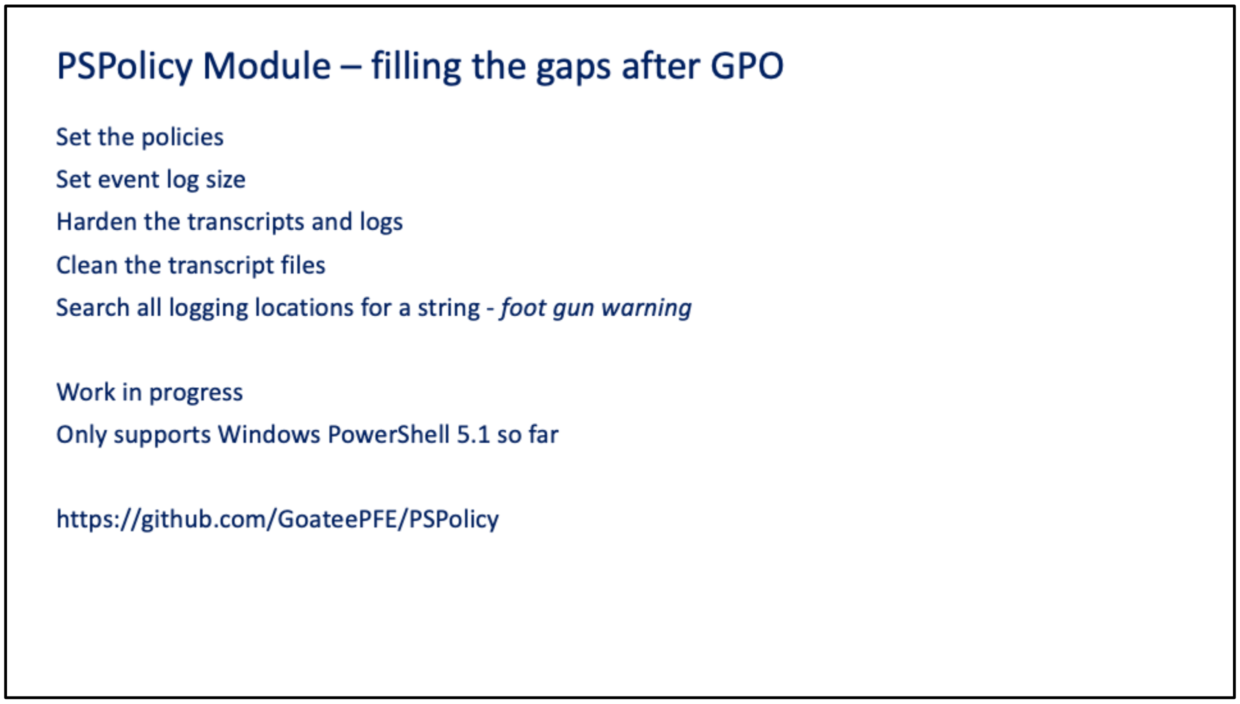
VSCode PowerShell plugin may spam the logs with Script Analyzer activity.

Transcription hardening on PSv4 will crash PowerShell.

Event log hardening may require a reboot to take full effect.

What issues have you encountered?

Resources



PSPolicy Module – filling the gaps after GPO

Set the policies

Set event log size

Harden the transcripts and logs

Clean the transcript files

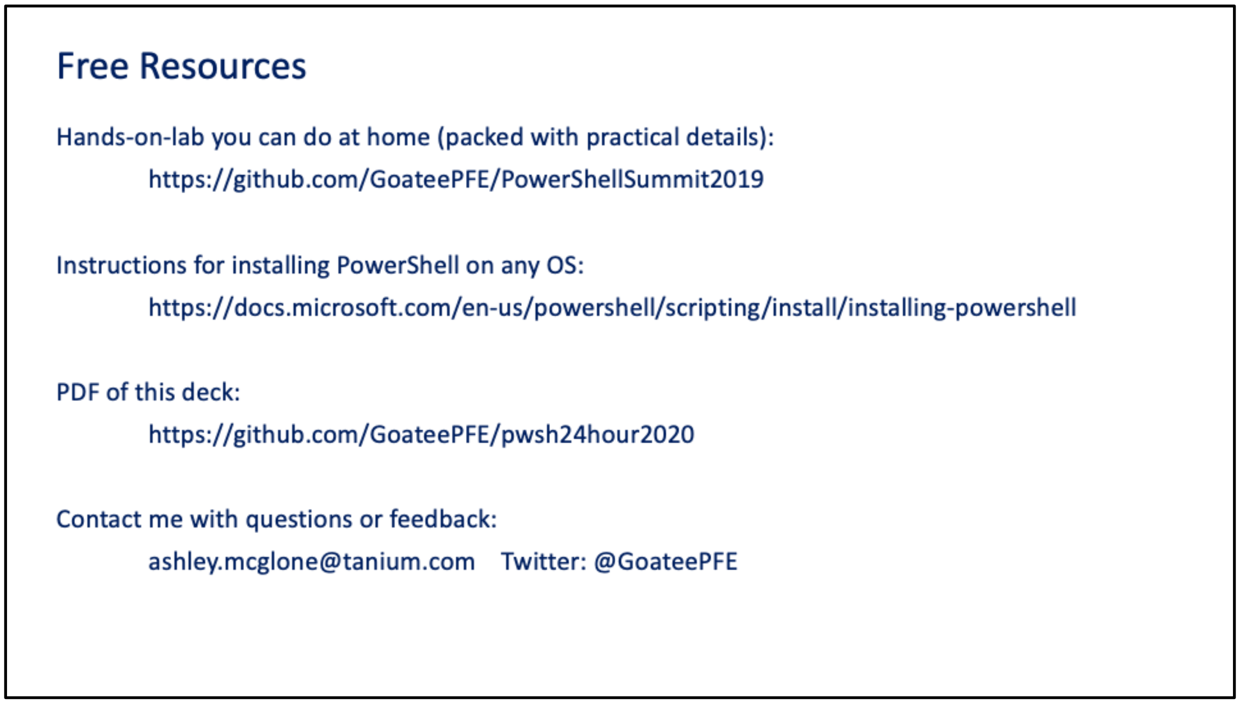
Search all logging locations for a string - *foot gun warning*

Work in progress

Only supports Windows PowerShell 5.1 so far

https://github.com/GoateePFE/PSPolicy

https://github.com/GoateePFE/PSPolicy


Free Resources

Hands-on-lab you can do at home (packed with practical details):

https://github.com/GoateePFE/PowerShellSummit2019

Instructions for installing PowerShell on any OS:

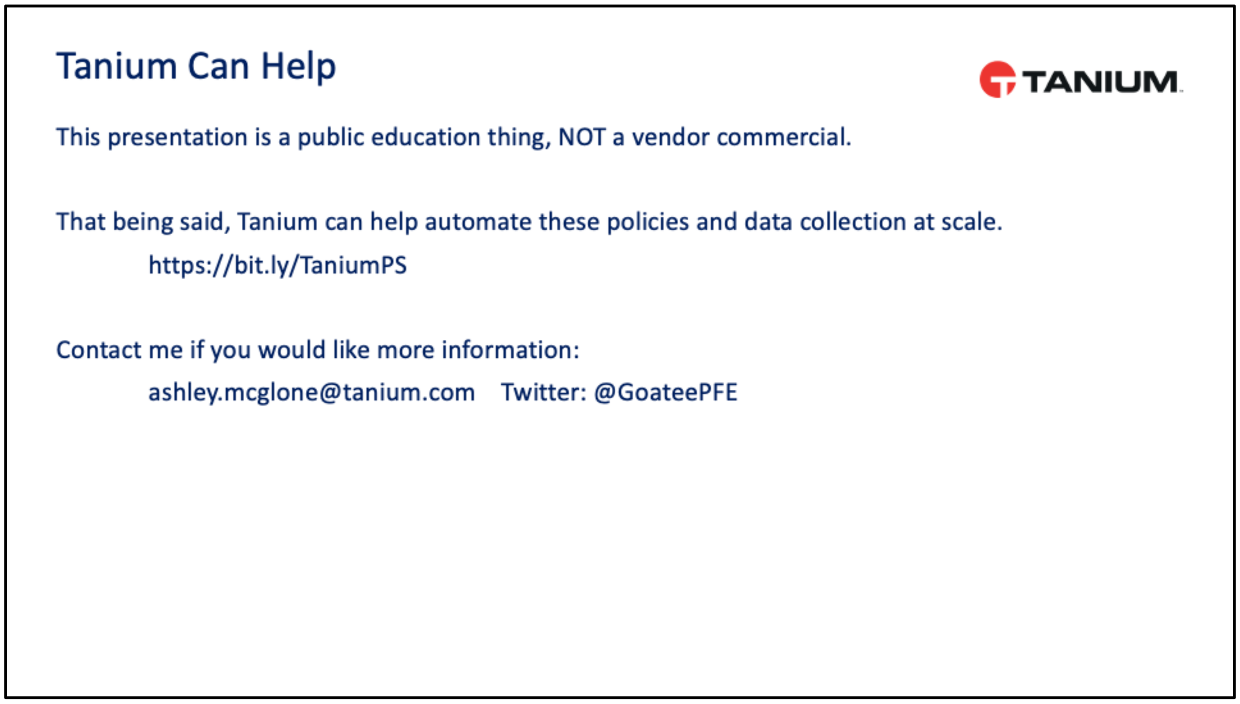
https://docs.microsoft.com/en-us/powershell/scripting/install/installing-powershell

PDF of this deck:

https://github.com/GoateePFE/pwsh24hour2020

Contact me with questions or feedback:

ashley.mcglone@tanium.com Twitter: @GoateePFE



Tanium Can Help

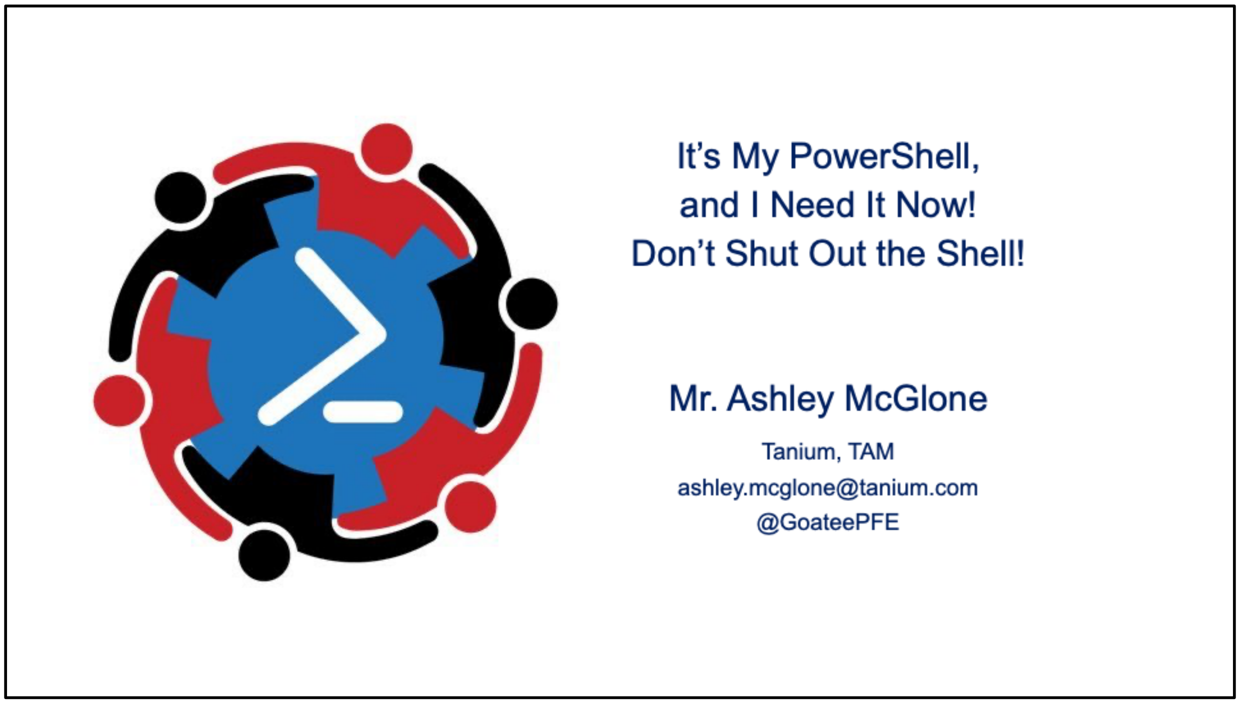
This presentation is a public education thing, NOT a vendor commercial.

That being said, Tanium can help automate these policies and data collection at scale.

https://bit.ly/TaniumPS

Contact me if you would like more information:

ashley.mcglone@tanium.com   Twitter: @GoateePFE



It's My PowerShell,
and I Need It Now!
Don't Shut Out the Shell!

Mr. Ashley McGlone

Tanium, TAM
ashley.mcglone@tanium.com
@GoateePFE

In the words of Mark Minasi, “Use your powers for good and not for evil.”