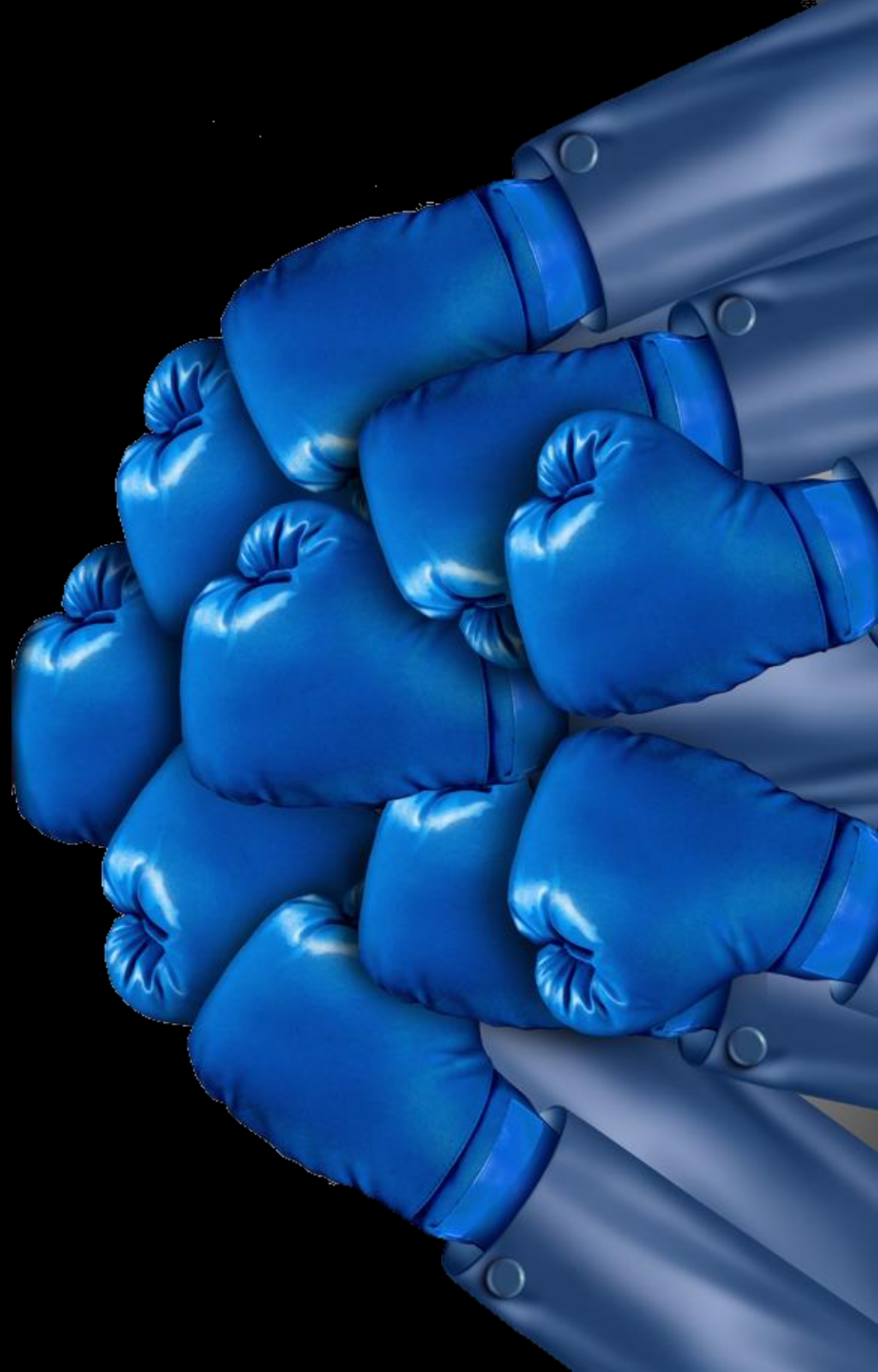
PowerShell for the BlueTeam



```
PS C:\> Get-Presenter
```


```
Name      : Martin Schvartzman
Email     : Martin.Schvartzman@microsoft.com
Blog      : http://aka.ms/pstips
```

ExecutionPolicy

- Restricted = Never
- AllSigned = Only if signed
- Remotesigned = If downloaded, only if signed
- Unrestricted = Always, but prompt if downloaded
- Bypass =Always

- If an attacker is changing your execution policy, he already has arbitrary code running on your machine

Transcription and logging

- **Transcription** (v2, Improved in v5)

  Record the contents of a PowerShell session

- **Module Logging** (v3)

  Capture execution details (too much information)

- **Deep ScriptBlock logging** (v5)

  Record scriptblocks as they are executed

ScriptBlock Logging

- All scriptblocks processed by PowerShell are logged the first time. To minimize log noise, repeated commands within a session are not logged.
- Includes dynamically generated code (**Invoke-Expression**)
- Includes any host: command line, console, ISE, etc.
- Optionally includes **Invocation Start/Stop** events (high log traffic).

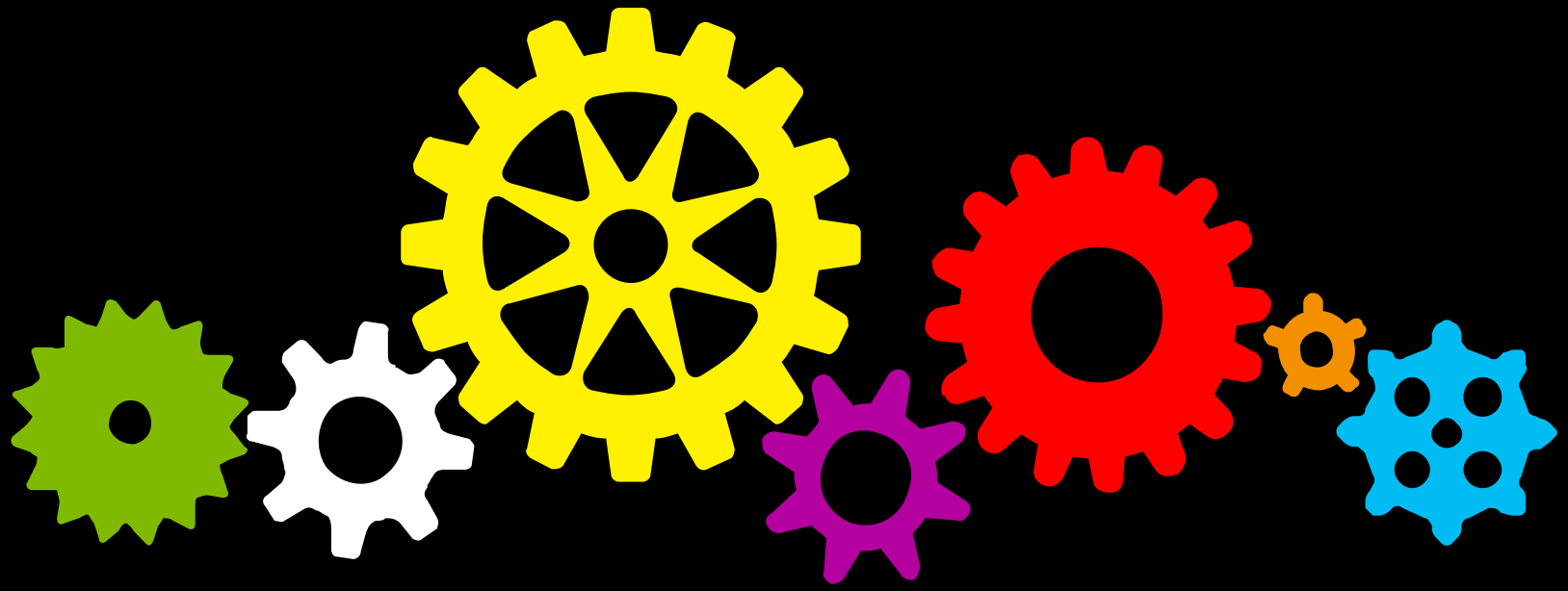
- *PowerShell hosts cache Group Policy settings at startup. Only hosts launched after enabling the logging will be affected.*

Transcription Guidance

- Secure the transcript folder from tampering (local or UNC):
  - Remove inherited permissions
  - Allow: Administrators – Full Control
  - Allow: Everyone – Write
  - Deny: Creator Owner – Full Control
- For UNC transcript paths make sure the path is accessible
- Default Windows Firewall will have SMB In/Out disabled, enable for UNC transcription
- Use GPO script or scheduled task to remove transcripts older than X days

PowerShell-6.0.0

About_Demo



Enable-PSTranscription.ps1

PowerShell-6.0.0

Language Modes

```
Get-Help about_Language_Modes
```

- $ExecutionContext.SessionState.LanguageMode
- $ExecutionContext.SessionState.LanguageMode = [System.Management.Automation.PSLanguageMode]::ConstrainedLanguage
- $ExecutionContext.SessionState.LanguageMode = [System.Management.Automation.PSLanguageMode]::RestrictedLanguage

Constrained Language

- The Constrained Language language mode (PSv3 and above) permits all Windows cmdlets and all Windows PowerShell language elements, but it limits permitted types.
- Once Constrained Language Mode is enabled, many PowerShell attack tools don’t work since they rely on components blocked by constrained language
- The Constrained Language mode was designed to work with system-wide application control solutions such as **AppLocker** and **Device Guard User Mode Code Integrity (UMCI)**

Constrained PowerShell & AppLocker

- AppLocker uses rules to allow/deny applications/scripts from running based on unique identities of files and to specify which users or groups can run those applications.
- With PSv5+ it enforces PowerShell **Constrained Language mode** when allowing scripts with AppLocker.
- Scripts allowed by the AppLocker policy (for example: signed by the enterprise's trusted code signing certificate, or in a trusted directory) are not subject to Constrained Language.
- *AppLocker requires the AppIDSvc service to be running.*

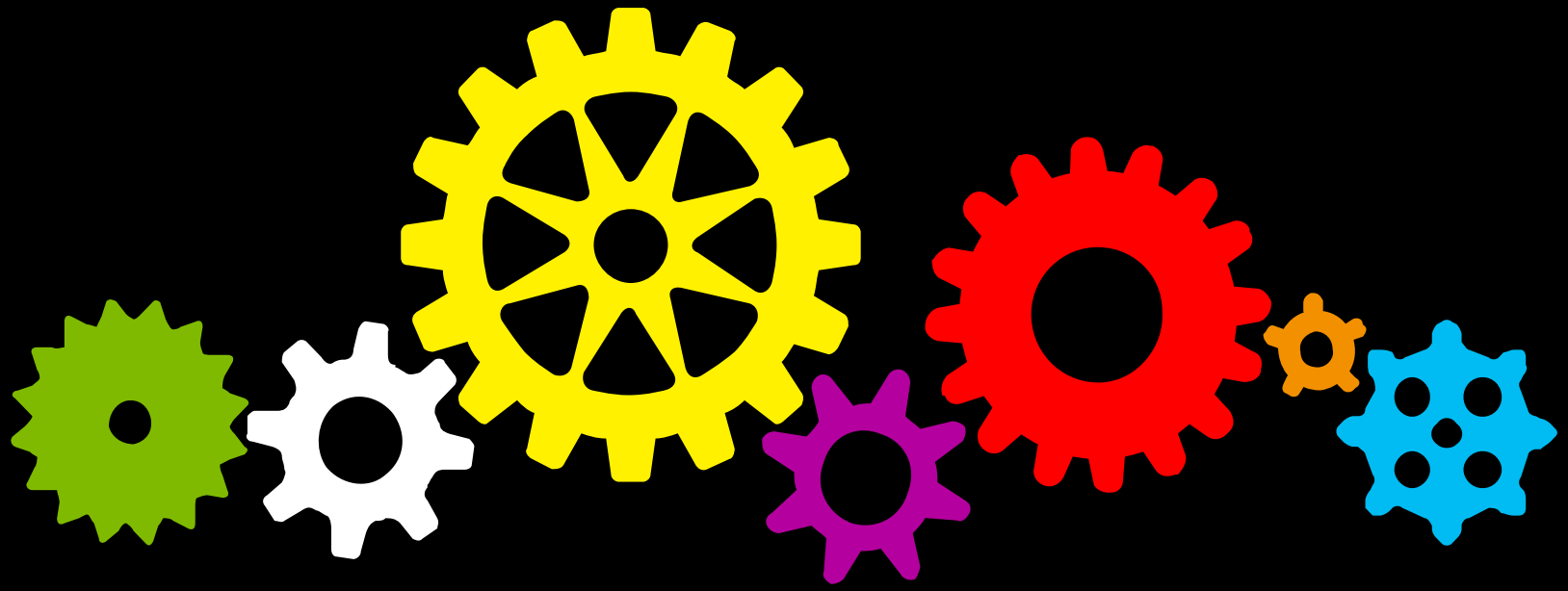
PowerShell-6.0.0

PowerShell without PowerShell.exe

- Referencing System.Managemet.Automation

PowerShell-6.0.0

About_Demo



Get-PSHosts.ps1

Engine Lifecycle Events

- Log
  - Microsoft-Windows-PowerShell/Operational
    - Increase the event log size to 1GB
- Events
  - **4104** – Script block logging, long script blocks span multiple events
  - Optional invocation info (includes Runspace ID)
    - 4105 – Start
    - 4106 – Stop

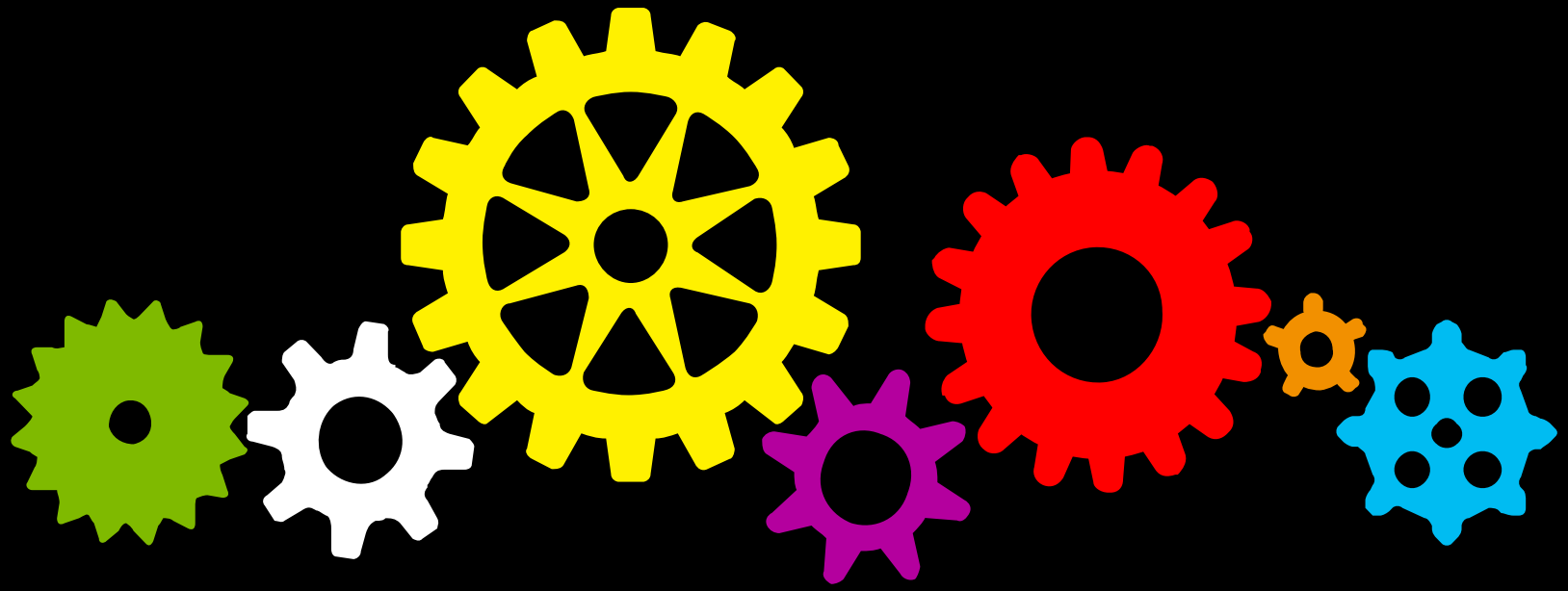
Downgrade attacks

- “Unfortunately, the PowerShell v5 enhancements did NOT include time travel, so the v2 binaries that were shipped in 2008 did NOT include the code we wrote in 2014..”
- Lee Holmes,
- Principal Technical Lead, PowerShell team
- http://www.leeholmes.com/blog/2017/03/17/detecting-and-preventing-powershell-downgrade-attacks/

- **Prevention:**
- Remove the PowerShell 2.0 Engine from the OS
- Apply application blacklisting (using AppLocker) to deny access to PowerShell 2.0 Engine specific .NET assemblies.

PowerShell-6.0.0

About_Demo



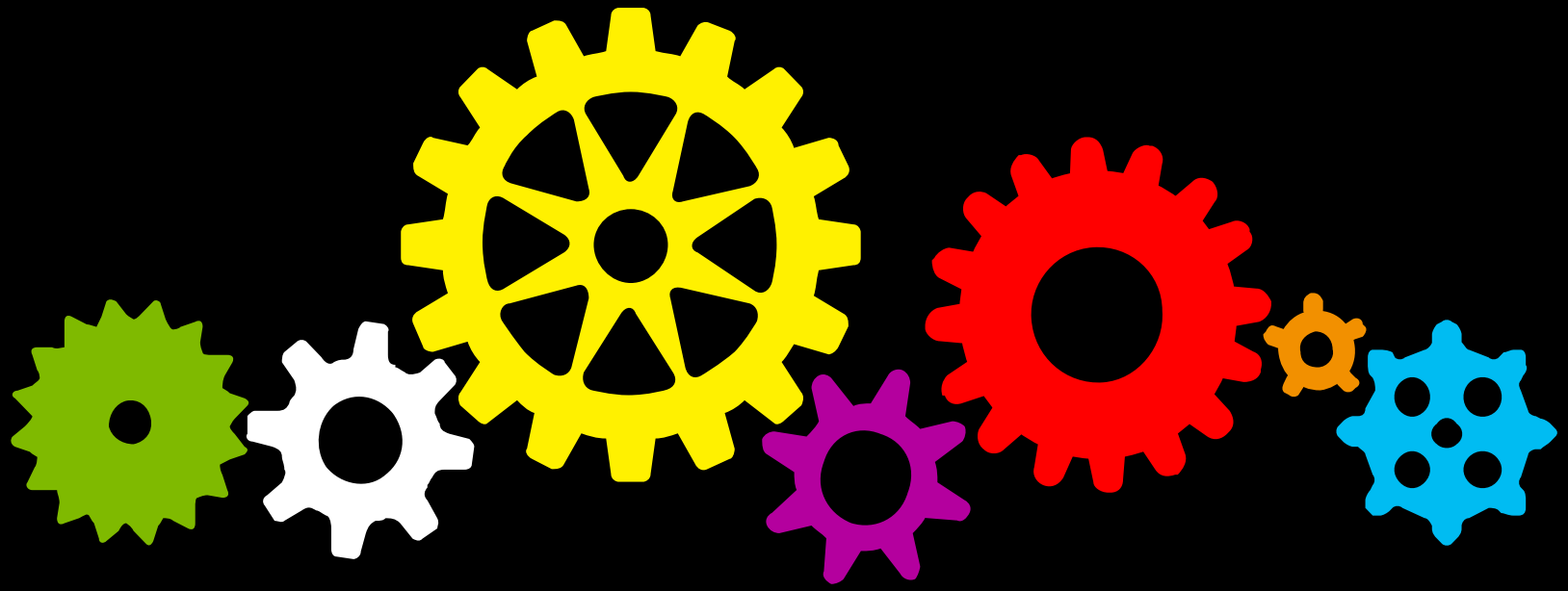
Get-DowngradeEvent.ps1

Anti-Malware Scan Interface (AMSI)

- Available on Windows 10 and Windows Server 2016
- Protects customers from dynamic, script-based malware ("download cradle", encoded commands)
- For example, an otherwise "safe" script file may actually download and execute malware using **Invoke-Expression**
- Scans PowerShell, VBScript, and JScript
- Supported in Windows Defender
- Open standard for anti-virus vendors to use in third party products

PowerShell-6.0.0

About_Demo



Test-AMSI.ps1

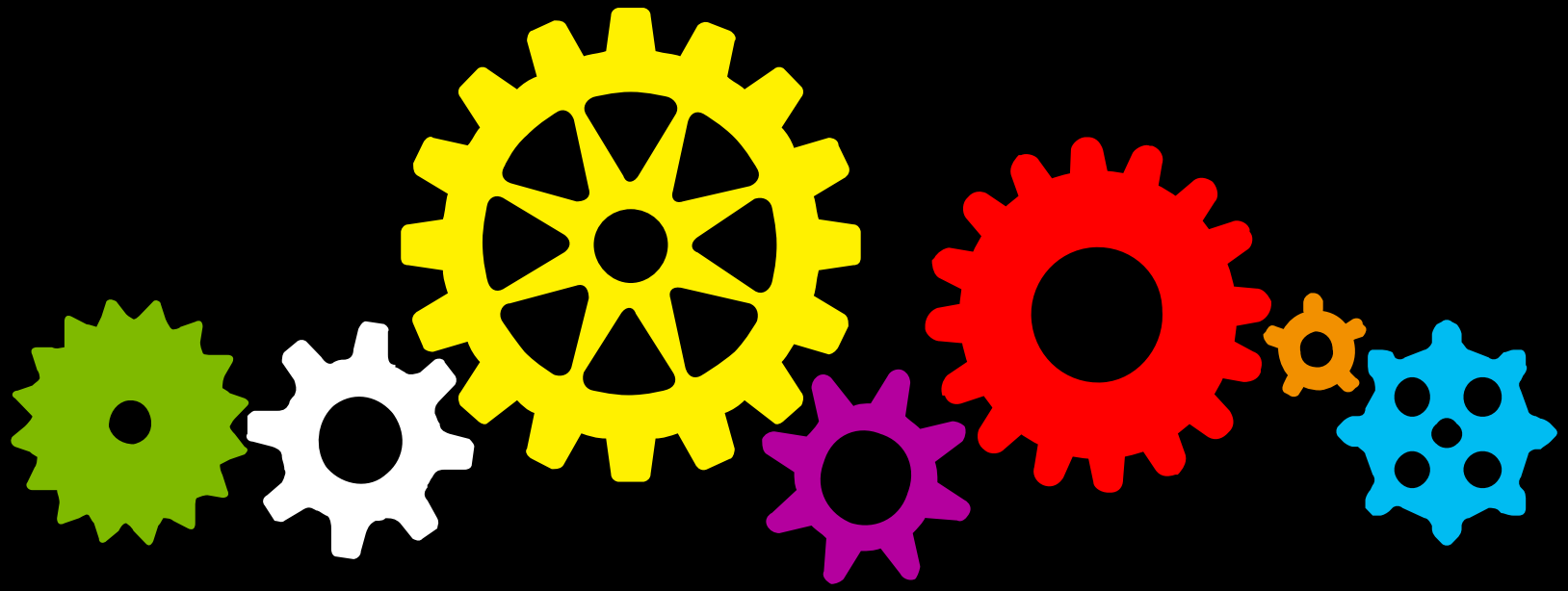
PowerShell-6.0.0

Listening to PSHosts starting

- Use sysmon
- Use WMI events

PowerShell-6.0.0

About_Demo



Register-PSHostStartEvent.ps1


Microsoft