Microsoft

Nov. 2019

# PowerShell Security

```
PS C:\> Get-Presenter

Name      : Shay Amar
Email     : Shay.Amar@microsoft.com

Name      : Martin Schvartzman
Email     : Martin.Schvartzman@microsoft.com
```

# Conditions and Terms of Use

Microsoft Confidential

This training package is proprietary and confidential, and is intended only for uses described in the training materials. Content and software is provided to you under a Non-Disclosure Agreement and cannot be distributed. Copying or disclosing all or any portion of the content and/or software included in such packages is strictly prohibited.

The contents of this package are for informational and training purposes only and are provided "as is" without warranty of any kind, whether express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

Training package content, including URLs and other Internet website references, is subject to change without notice. Because Microsoft must respond to changing market conditions, the content should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

# Copyright and Trademarks



Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

For more information, see **Use of Microsoft Copyrighted Content** at
http://www.microsoft.com/about/legal/permissions/

Microsoft®, Internet Explorer®, Outlook®, SkyDrive®, Windows Vista®, Zune®, Xbox 360®, DirectX®, Windows Server® and Windows® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other Microsoft products mentioned herein may be either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. All other trademarks are property of their respective owners.

# Logistics

**Schedule**

**Lunch**

**Rest Rooms**

**Phones**

**Smoking**

**Content**

http://aka.ms/PSSec-201911

# Introductions

## About You:

- Name
- Organization
- Title/Function/Area of Responsibility
- Product experience
- Expectations

# Show of hands

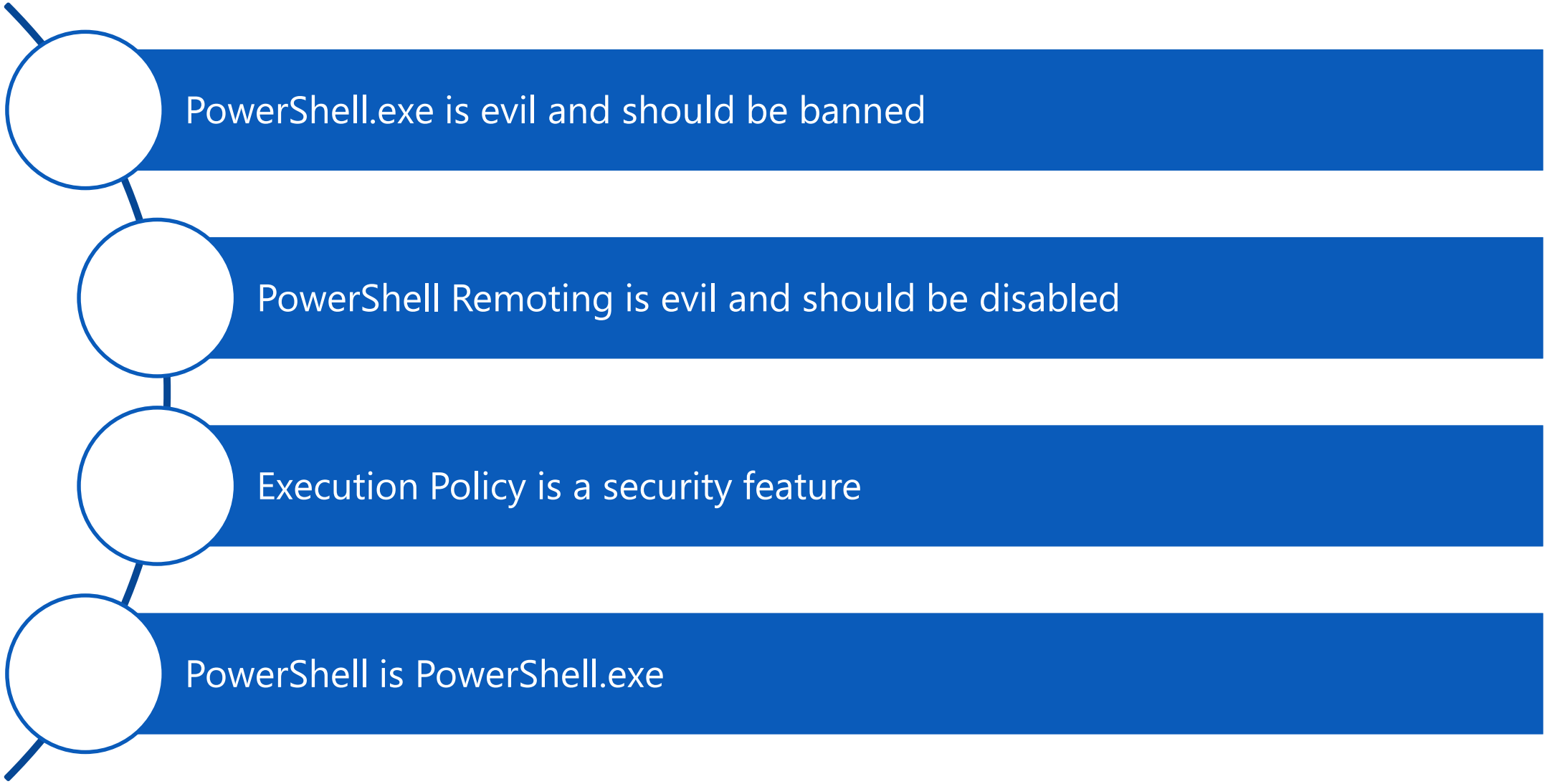
Raise your hand if (any of the below):

- I have Windows 7 (or older) in my org.
- I have Windows 2008 (or older) in my org.
- I don’t have PowerShell 5.1 everywhere
- I have SMBv1 enabled

# Agenda

- Module 1: Introduction & Context
- Module 2: Execution Policy
- Module 3: PowerShell without PowerShell.exe
- Module 4: Obfuscation
- Module 5: Transcription & ScriptBlock Logging
- Module 6: Working with Credentials
- Module 7: PSRemoting
- Module 8: Constrained Language, AppLocker & AMSI
- Module 9: Just Enough Administration
- Module 10: Windows Defender ATP

# Common Misconceptions



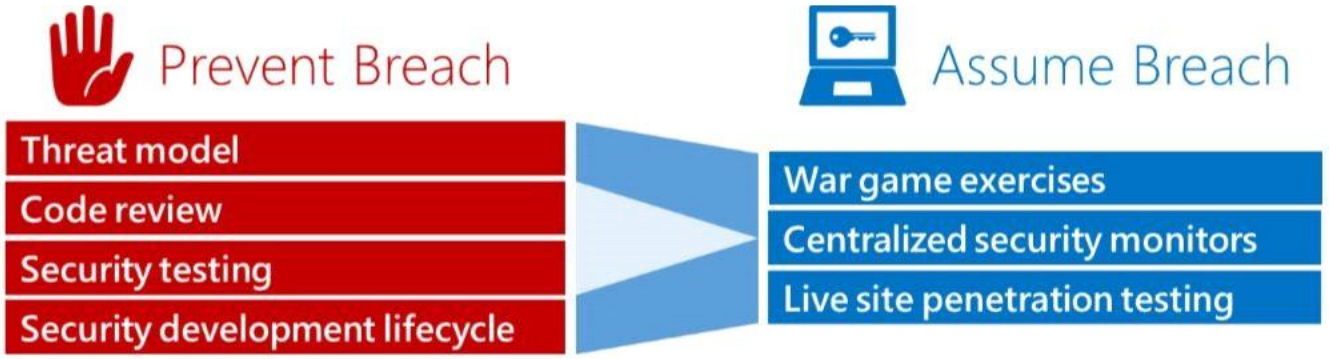
# Traditional defenses

Traditional defenses are no match for today’s challenges

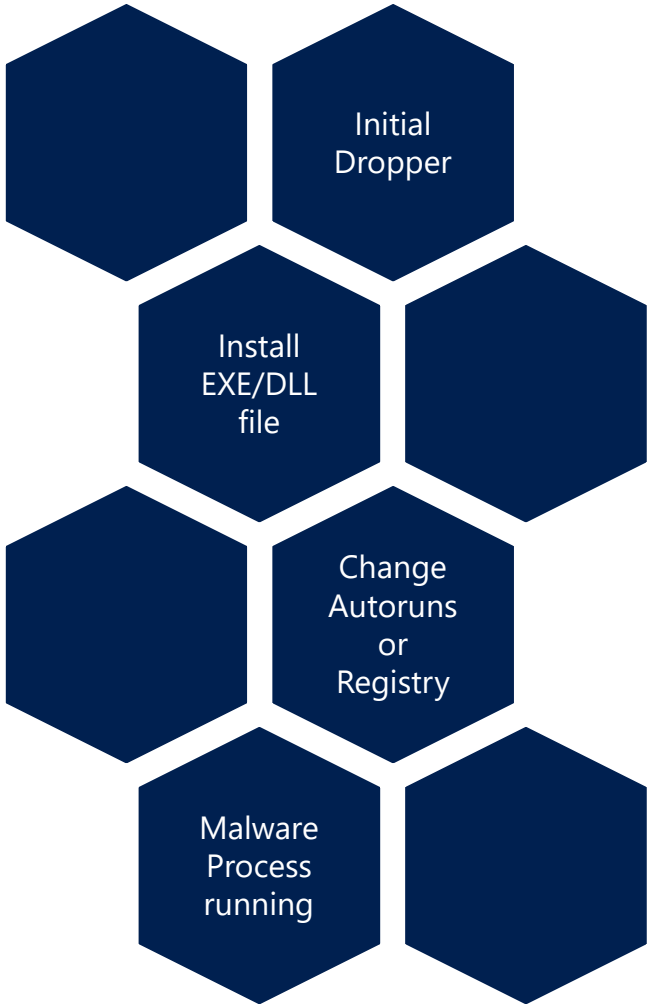
# Assume Breach

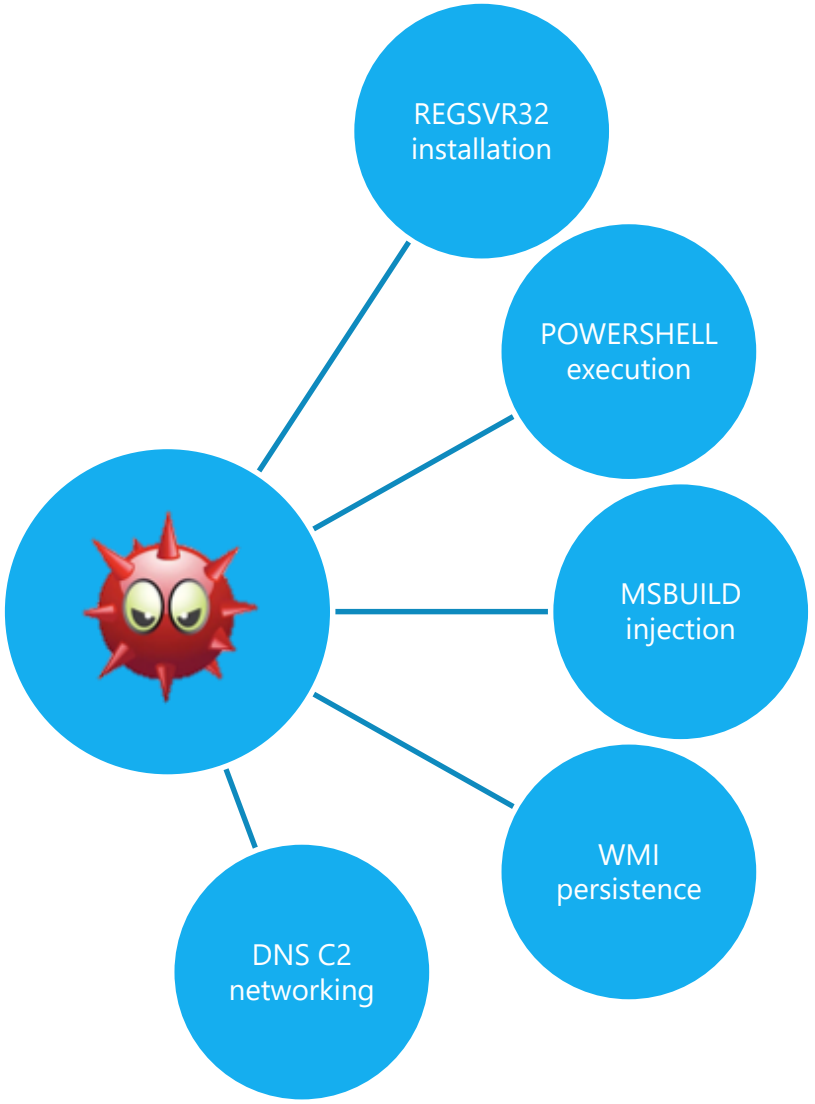
*"Fundamentally, **if somebody wants to get in, they're getting in**…accept that. What we tell clients is:*
*Number one, **you're in the fight**, whether you thought you were or not.*
*Number two, **you almost certainly are penetrated**."*

Michael Hayden
Former Director of NSA & CIA

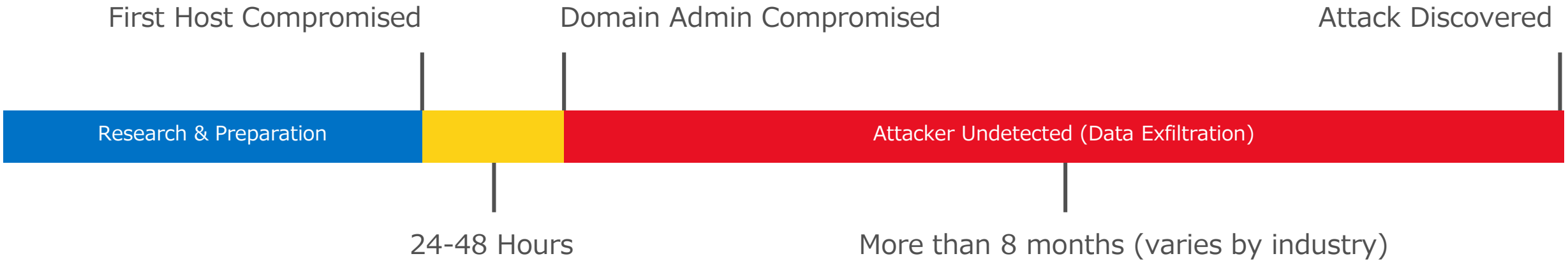


# Old Malware vs Modern Malware



# Typical Attack Timeline & Observations




## Attack Sophistication

Attack operators exploit any weakness

Target information on any device or service

## Target AD & Identities

Active Directory controls access to business assets

Attackers commonly target AD and IT Admins

## Attacks not detected

Current detection tools miss most attacks

You may be under attack (or compromised)

## Response and Recovery

Response requires advanced expertise and tools

Expensive and challenging to successfully recover

# LOL: Living Off the Land

wmic.exe /node:[IP Address] /user:[USERNAME] /password:[PASSWORD] process call create "C:\Windows\System32\rundll32.exe \"C:\Windows\perfc.dat\" #1 60”

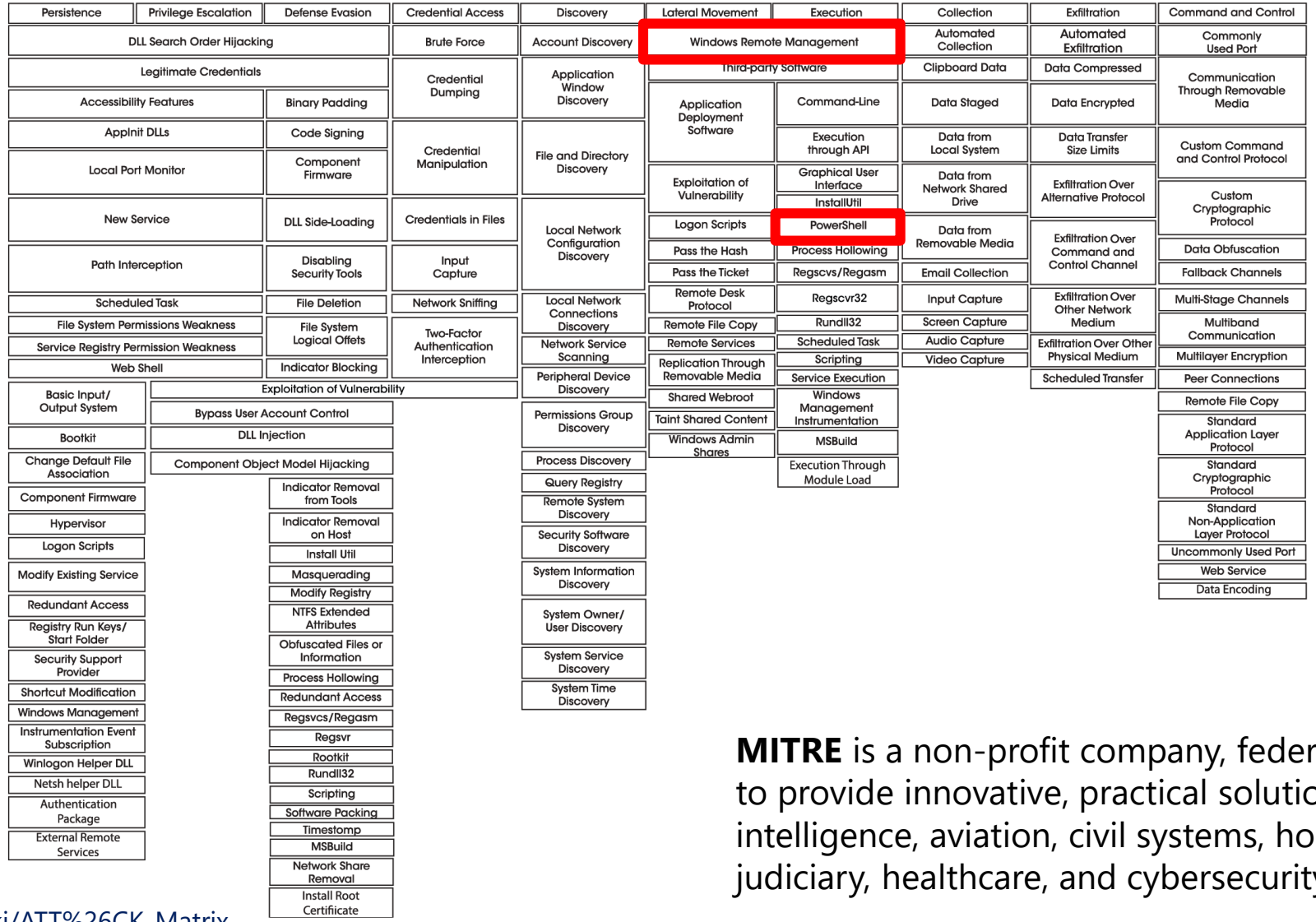
wevtutil cl Setup & wevtutil cl System & wevtutil cl Security & wevtutil cl Application & fsutil usn deletejournal /D %c:

schtasks /RU "SYSTEM" /Create /SC once /TN "" /TR "C:\Windows\system32\shutdown.exe /r /f" /ST 14:42

The above commands were used in Petya. Notice there’s no use of PowerShell

VBA Macros, VBScript, JScript, csc.exe, HTA, SQL, Perl, Python, Ruby, Bash …

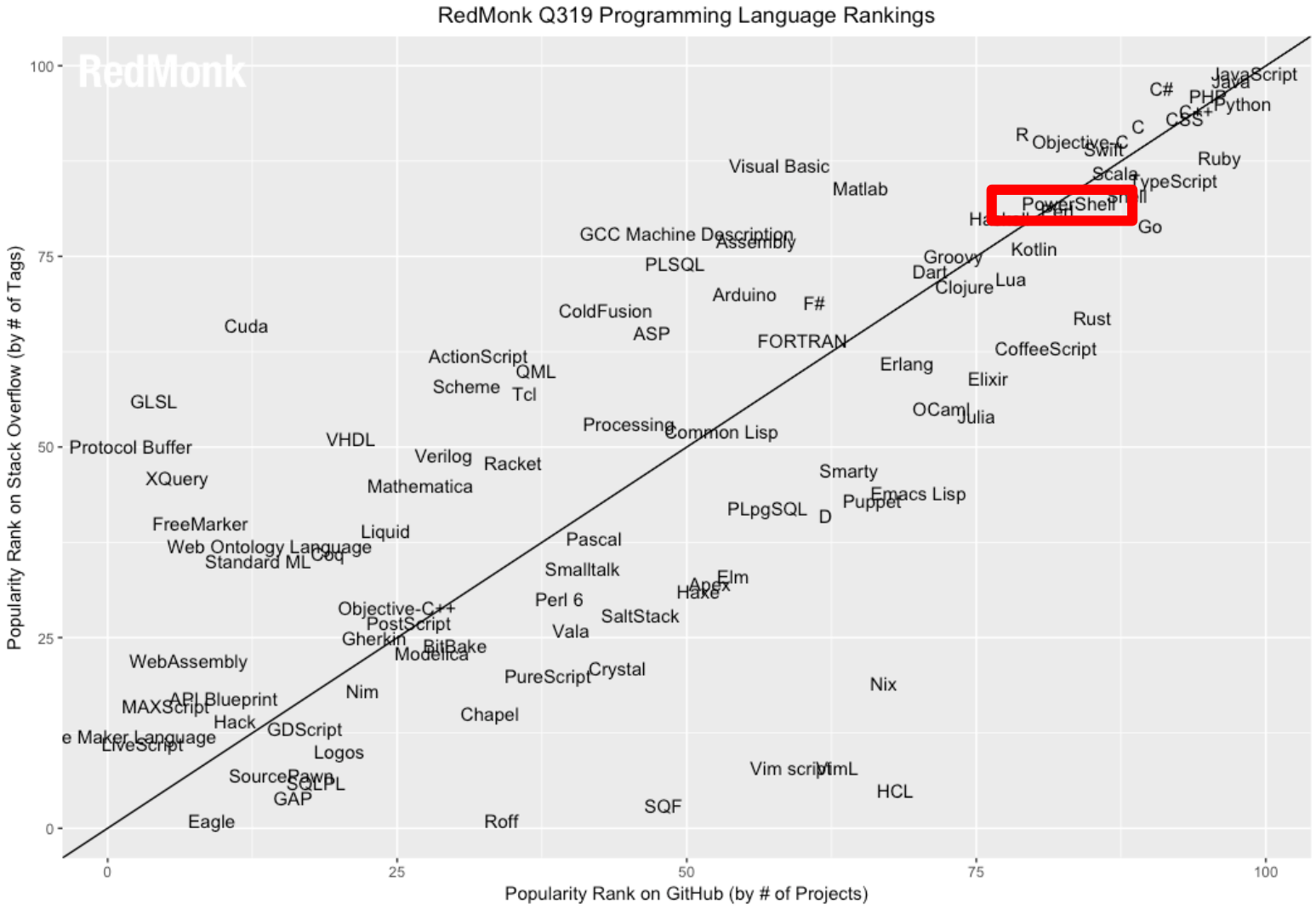
# MITRE attack matrix




**MITRE** is a non-profit company, federally funded research, to provide innovative, practical solutions in the defense and intelligence, aviation, civil systems, homeland security, judiciary, healthcare, and cybersecurity spheres

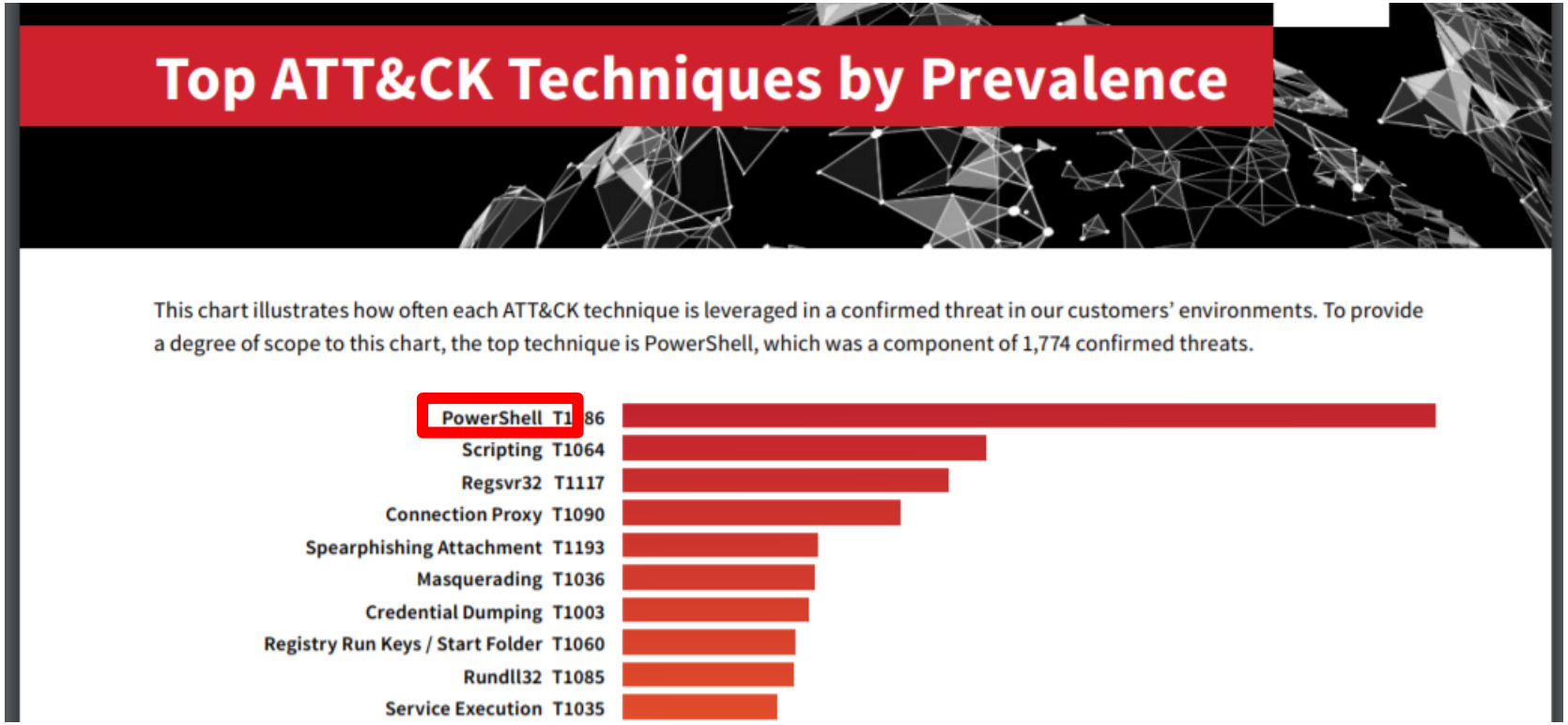
https://attack.mitre.org/wiki/ATT%26CK_Matrix

# Programming Language Rankings




https://redmonk.com/sogrady/files/2019/07/lang.rank_.619.png

# Top ATT&CK Techniques

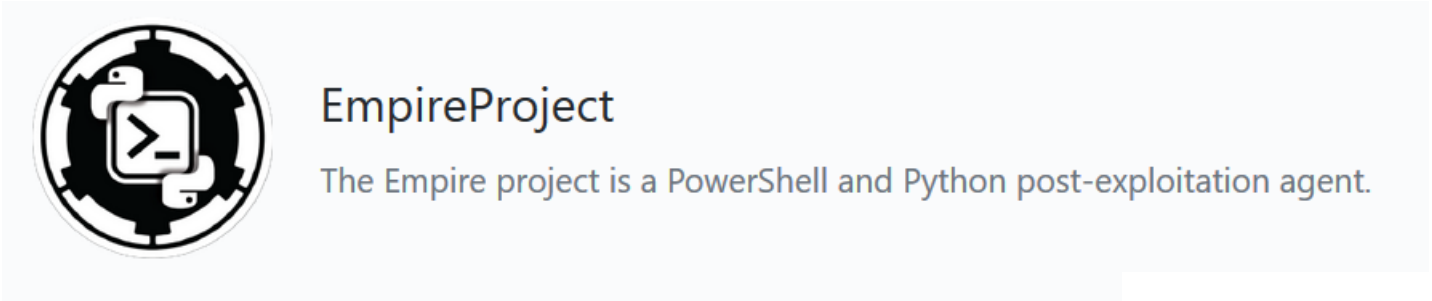



https://resources.redcanary.com/hubfs/ThreatDetectionReport-2019.pdf

# Post Exploitation

**Post exploitation** basically means the phases of operation once a victim's system has been compromised by the attacker.

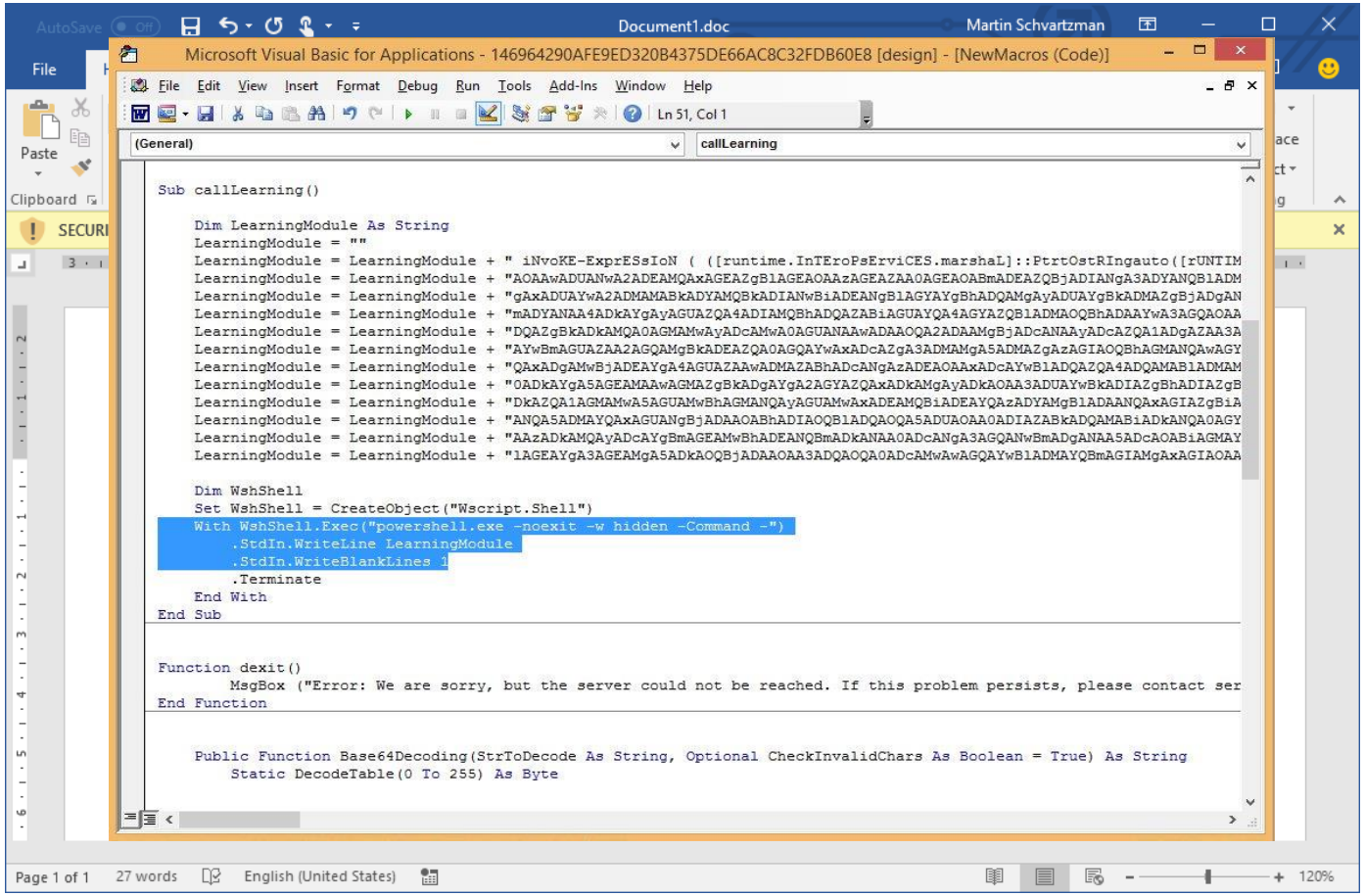
The value of the compromised system is determined by the value of the actual data stored in it and how an attacker may make use of it for malicious purposes.




PowerSploit - A PowerShell Post-Exploitation Framework

# Initial point of entry

## Remember, we're talking POST Breach



# Security in PowerShell

## Macro

# Blocking PowerShell

PCWorld FROM IDG

SUBSCRIBE

NEWS | REVIEWS | HOW-TO | VIDEO | BUSINESS | LAPTOPS | TABLETS | PHONES | HARDWARE | SECURITY | SOFTWARE | GADGETS

Security | Games | Productivity Software | Home Audio | Business Software | Photography | **Utility Software**

Ads by Google | Computer Software | Best AntiVirus | Computer Virus

Home / Utility Software

NEWS

## Longhorn Component to Spur Old-School Viruses?

By Paul Roberts

IDG News Service | OCT 4, 2004 5:00 PM PT



A planned component for Microsoft's next version of Windows is causing consternation among antivirus experts, who say that the new module, a scripting platform called Microsoft Shell, could give birth to a whole new generation of viruses and remotely exploitable attacks.

**MORE LIKE THIS**

The Life and Times of Windows XP

Windows 7 First Look: A Big Fix for Vista

Microsoft Shell, code-named "Monad," is still in development and is planned for release with the next version of Windows, known as "Longhorn." Monad will allow developers or administrators to configure Windows systems using text commands or scripts containing multiple commands. **But the flexibility of the new platform and its support for remote execution of commands could spawn a whole new generation of "script viruses,"** like the "Melissa" script virus of 1999; e-mail worms; and remote attacks

https://www.pcworld.com/article/118045/article.html

# Blocking PowerShell

“It’s unsecure, I heard about it in the news”

“After the CIO went to a security conference, he asked to remove PowerShell from the environment”

“Our last audit said that PowerShell needs to be locked down on all servers”

“InfoSec will not let us turn on PowerShell remoting”

“The NCSA recommends to do so“

“Let’s block PowerShell.exe and be done with it”

# Blocking PowerShell

Systems were attacked before PowerShell was invented, they will continue to be so with or without it

Blocking PowerShell doesn't address the real security problem, it just removes your **most secured management tool**

# Shell and Scripting Language Security Comparison

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

* Feature exists, but cannot enforce by policy

** Experiments exist

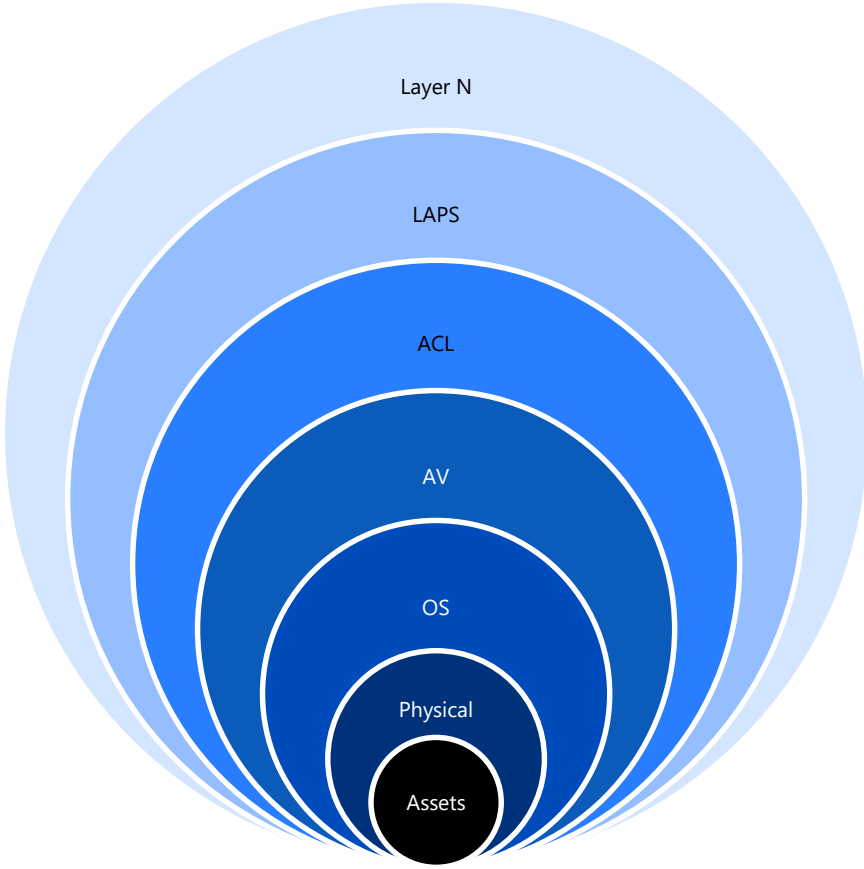
https://blogs.msdn.microsoft.com/PowerShell/2017/04/10/a-comparison-of-shell-and-scripting-language-security/
Lee Holmes, Azure Management Security, April 10, 2017

# Layered approach to security

Organizations need to widen their **security** nets to protect and defend against opportunistic attacks and infections

The term describes a defensive strategy featuring multiple defensive **layers** that are designed to slow down an attacker

The military calls this ***defense in depth***



# Agenda

- Module 1: Introduction & Context
- Module 2: Execution Policy
- Module 3: PowerShell without PowerShell.exe
- Module 4: Obfuscation
- Module 5: Transcription & ScriptBlock Logging
- Module 6: Working with Credentials
- Module 7: PSRemoting
- Module 8: Constrained Language, AppLocker & AMSI
- Module 9: Just Enough Administration
- Module 10: Windows Defender ATP

# Script Execution

The Execution Policy is NOT a security boundary or mechanism!

'Restricted' by default, just prevents ps1 scripts from running accidently

Execution Policy is saved in the registry

# Execution Policy Levels

## Restricted

- Default
- Scripts cannot be run
- PowerShell interactive-mode only

## AllSigned

- Runs a script only if signed
- Signature must be trusted on local machine

## RemoteSigned

- Recommended Minimum
- Runs all local scripts
- Downloaded scripts must be signed by trusted source

## Unrestricted

- All scripts from all sources can be run without signing

# Execution Policy Scope

Apply Execution Policy Levels at one or more of these 5 scopes

## AD Group Policy – Computer

- Affects all users on targeted computer
- Edited through GPO Tools

## AD Group Policy – User

- Affects users targeted only
- Edited through GPO Tools

## Process

- Console or ISE Command-line Parameter:
  i.e.: *c:\> powershell.exe –executionpolicy remotesigned*
- Affects current PowerShell Host session only
- Lost upon exit of session (i.e. host process)

## Registry – User

- Affects current user only
- Stored in HKCU registry subkey

## Registry – Computer

- Affects all users on computer
- Stored in HKLM registry subkey (Admin access needed to change)

# Security in PowerShell

## Execution Policy

# Bypassing Execution Policy

Change the execution context by resetting the authorization manager

```
$context = $executioncontext.GetType().GetField(
        '_context','nonpublic,instance').GetValue($executioncontext)

$field = $context.GetType().GetField(
        '_authorizationManager','nonpublic,instance')

$field.SetValue($context,
        (New-Object -TypeName Management.Automation.AuthorizationManager `
                -ArgumentList Microsoft.PowerShell))
```

Or simply use the command or console

# Security in PowerShell

## Bypassing Execution Policy

# Agenda

- Module 1: Introduction & Context
- Module 2: Execution Policy
- Module 3: PowerShell without PowerShell.exe
- Module 4: Obfuscation
- Module 5: Transcription & ScriptBlock Logging
- Module 6: Working with Credentials
- Module 7: PSRemoting
- Module 8: Constrained Language, AppLocker & AMSI
- Module 9: Just Enough Administration
- Module 10: Windows Defender ATP

# PowerShell without PowerShell.exe

System.Management.Automation.dll hosted in non-standard processes

https://docs.microsoft.com/en-us/dotnet/api/system.management.automation.powershell

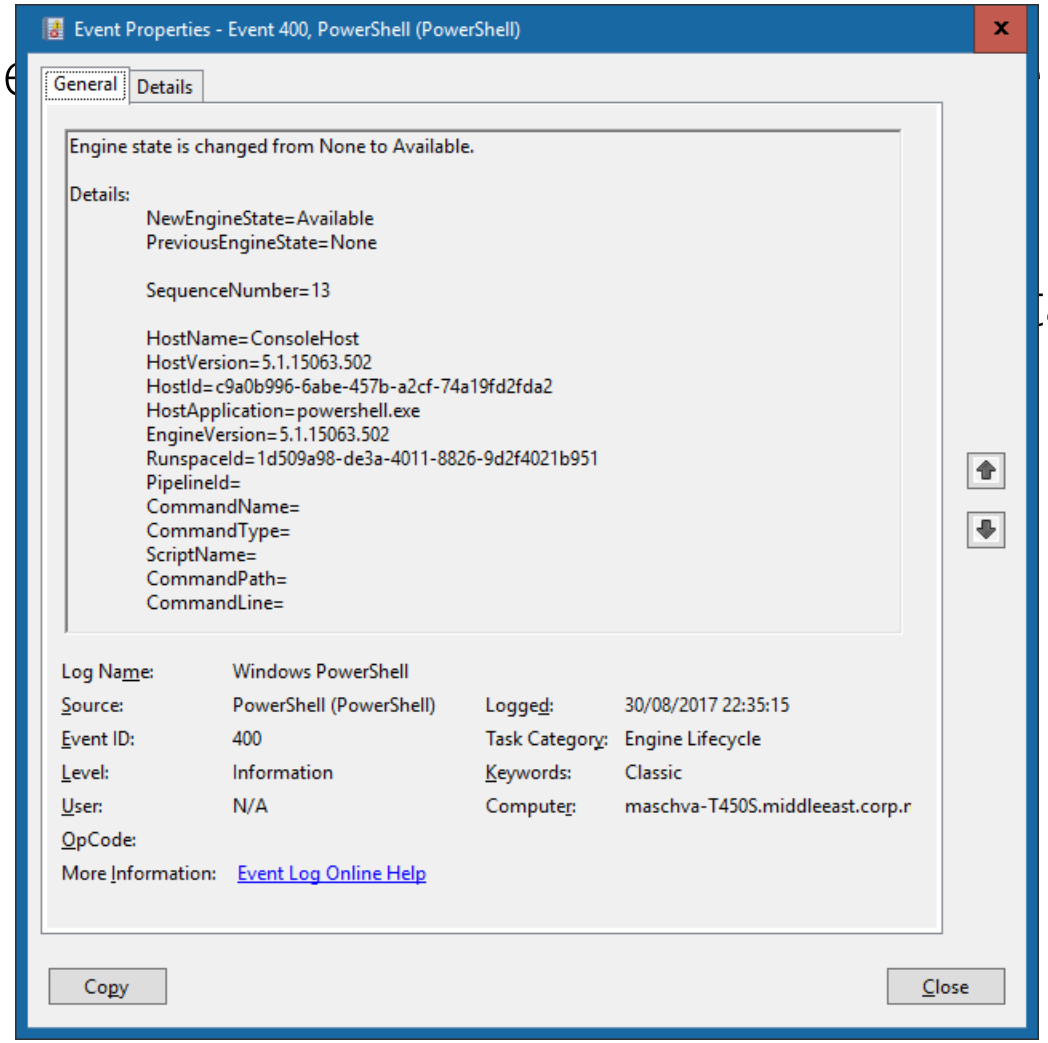
```
PS C:\> Get-Process | Where-Object { $_.Modules -like '*System.Management.Automation*' }

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)      Id  SI ProcessName
-------  ------    -----      -----     ------      --  -- -----------
    337      35    50108      49896       0.38   18732   1 akada
    788      73   149288     164804      10.64   16916   1 powershell
    913      64   178988     205484      13.81   16360   1 powershell_ise
```

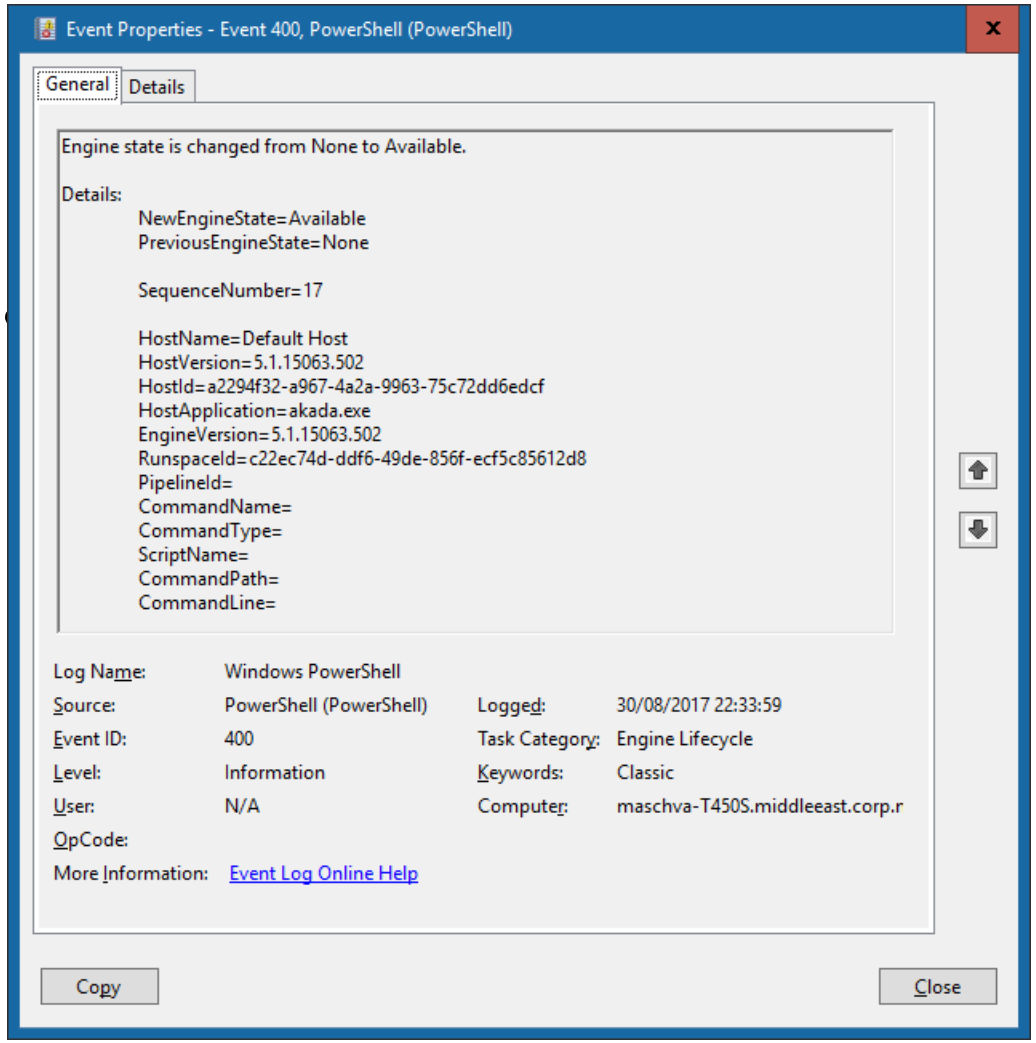
Note that custom EXEs can still natively call .NET and Windows APIs directly

# PowerShell without PowerShell.exe

De ll:

tand



# PowerShell without PowerShell.exe

## Referencing **System.Managemet.Automation** in .NET

```
using System.Management.Automation;
...
namespace PSMartin {
    internal class Program {
        private static void Main(string[] args) {
            ...
            using (Runspace runspace = RunspaceFactory.CreateRunspace()) {
                runspace.Open();
                using (Pipeline pipeline = runspace.CreatePipeline()) {
                    string @cmd = "Get-Service BITS";
                    pipeline.Commands.AddScript(@cmd);
                    pipeline.Commands.Add("Out-String");
                    Collection<PSObject> collection = pipeline.Invoke();
                    ...
                }
            }
        }
    }
}
```

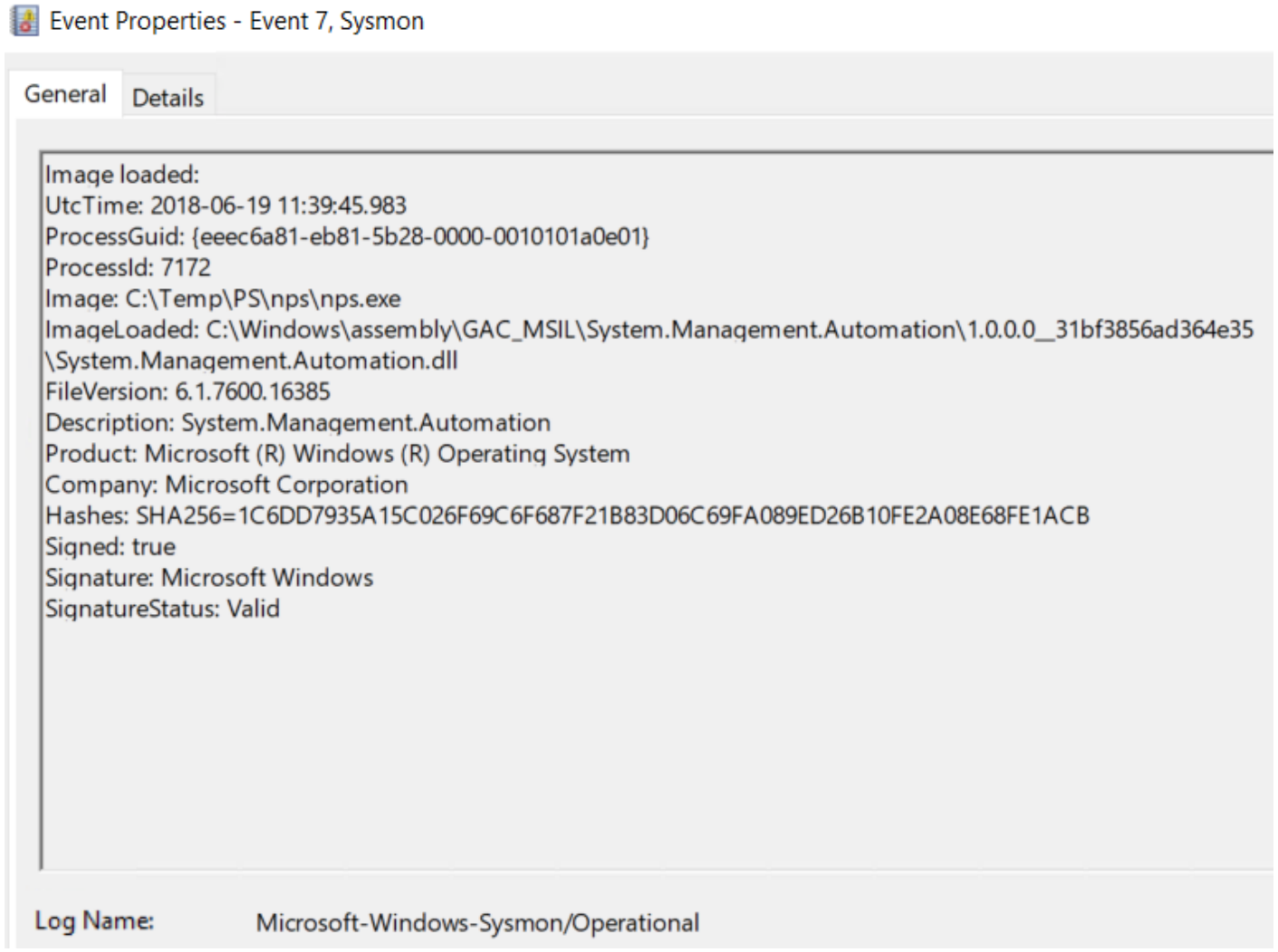
# Security in PowerShell

## PowerShell w/o PowerShell.exe

# Security in PowerShell

## Detecting Hosts Startup

# Using Sysmon to hunt rouge PowerShell hosts



# Agenda

- Module 1: Introduction & Context
- Module 2: Execution Policy
- Module 3: PowerShell without PowerShell.exe
- Module 4: Obfuscation
- Module 5: Transcription & ScriptBlock Logging
- Module 6: Working with Credentials
- Module 7: PSRemoting
- Module 8: Constrained Language, AppLocker & AMSI
- Module 9: Just Enough Administration
- Module 10: Windows Defender ATP

# Obfuscation

"The action of making something obscure, unclear, or unintelligible"

((47,65,74,'2d',53,65,72,76,69,63,65,20,42,49,54,53)|fOREach{([cONvERT]::Toint16(([sTRING]$_),16)-as[ChaR])})-jOin''|&( $eNV:coMSpec[4,24,25]-JOIn'')

inVOkE-ExpREssIOn(-joIn( '71!101x116n45&83x101n114&118V105V99V101V32&66W73V84z83'-SPLit'W' -spLiT'!'-spLiT '&'-sPliT 'J' -spLit'V' -spliT 'x'-SpIlt 'n'-SpliT 't'-SpLiT'Y' -SpliT'z' | fOReAcH {([cHaR][Int] $_) } ) )

([RUntiMe.interOpsERvICes.MARshaL]::([RunTiMe.iNtErOPseRvIceS.mARShal].gETmemBerS()[4].NAmE).INvOKe([RUNTImE.inTeRoPSERvICes.mARshaL]::secUREstriNgtoGLObaLALLOcUniCoDE($('76492d1116743f0423413b16050a5345MgB8ADYAYgBIAGQAWABwAGcAbgBYAG0ARABSAFoASQBkAEgAcwAzAFgAWQAxAFEAPQA9AHwANgA2AGQAMwBiADIAOQA2ADgAOABmADkAYQA2ADMAOQAwAGQAOAA1AGIAMQA1AGQAYQAxADcAYQA2AGMAOQA1ADMAYQA0AGQAYwA3ADgAZABjAGUAMgAwAGYANwAyADAAMQA1ADcAZQAyAGEAYQBmADkAZQAxADMAOQAyADgAOAAxAGUAZQAyADEAOQBmADkAMQBkADkANgAzADAAOAA3ADQAZQA0AGIANgA0AGIAZgBkADIAMwA4ADgAYgA3ADUA'|convERTTo-SecuREsTRINg -KEy 203,44,105,6,209,29,205,9,226,126,34,92,19,130,12,164,96,90,234,90,131,187,101,158,136,170,213,99,230,174,102,106) ) ))|IeX

# Security in PowerShell

## Obfuscation

# Agenda

- Module 1: Introduction & Context
- Module 2: Execution Policy
- Module 3: PowerShell without PowerShell.exe
- Module 4: Obfuscation
- Module 5: Transcription & ScriptBlock Logging
- Module 6: Working with Credentials
- Module 7: PSRemoting
- Module 8: Constrained Language, AppLocker & AMSI
- Module 9: Just Enough Administration
- Module 10: Windows Defender ATP

# Security Transparency Enhancements

**Transcription** (v2) → **Over-the-shoulder system-wide transcription** (v5)

- Record the contents of a PowerShell session

**Module Logging** (v3) → **Deep script block logging** (v5)

- Record evaluated scriptblocks as they are executed
- By default, logs suspicious code ("naughty-list" of words): EventId 4104 Warning

# Transcription in PowerShell v2

Start-Transcript & Stop-Transcript cmdlets

Needed to be called in one of the profile script files

Implemented in the PowerShell.exe host

Not Available in other hosts (e.g. ISE)

# Transcription Enhancements

Transcription everywhere: remoting sessions, ISE, background programs, etc.

**Start-Transcript** returns an object with **Path** property

Nested transcription

Automatic file name is unique, with computer name, time stamp, and randomization

New **-OutputDirectory** parameter for local or UNC path

Rich file header data (host, process ID, versions, etc.)

Header fields **UserName** and **RunAsUser** differentiate users in PSSessionConfigurations

**-IncludeInvocationHeader** parameter adds header for each command

Automatic transcription with GPO **Turn on PowerShell Transcription** (or DSC registry)

# Transcription Guidance

Secure the transcript folder from tampering (local or UNC)

- Administrators – Full Control – Allow
- Everyone – Write – Allow
- Creator Owner – Full Control – Deny

For UNC transcript paths make sure the path is accessible

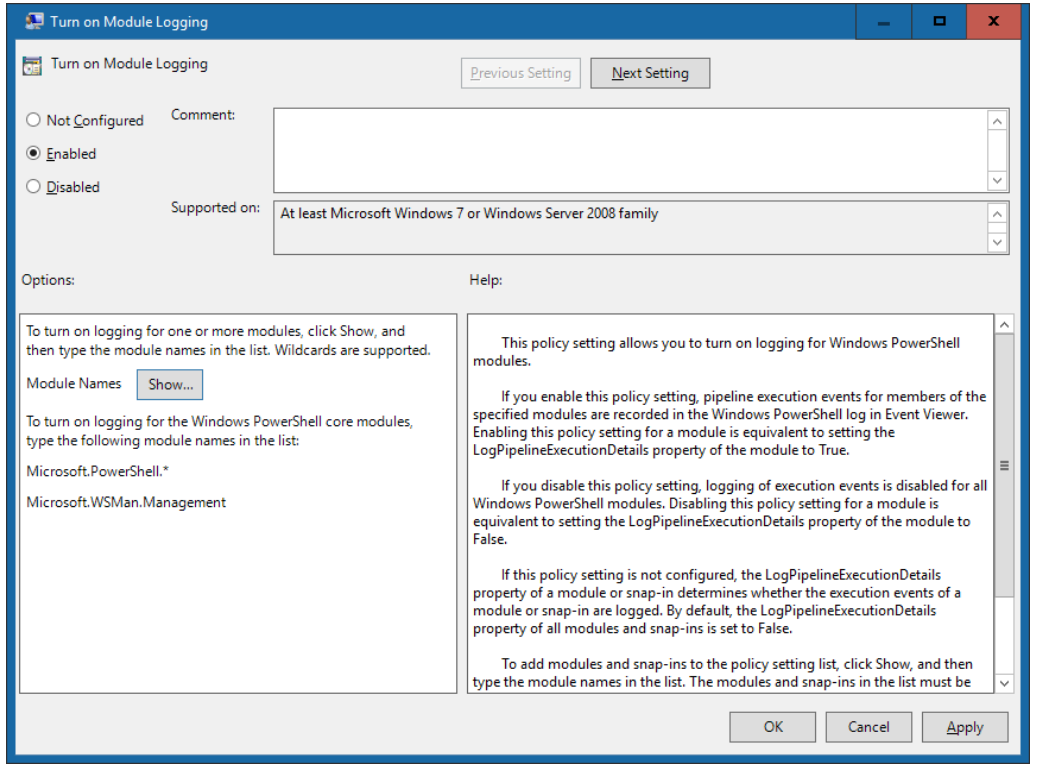
Default Windows Firewall will have SMB In/Out disabled, enable for UNC transcription

Use GPO script or scheduled task to remove transcripts older than X days

# Module Logging

The "Turn on Module Logging" policy setting turns on logging for selected Windows PowerShell modules

The pipeline execution events for the specified modules are recorded in the Windows PowerShell log in Event Viewer



# ScriptBlock Logging

All scriptblocks processed by PowerShell are logged the first time. To minimize log noise, repeated commands within a session are not logged.

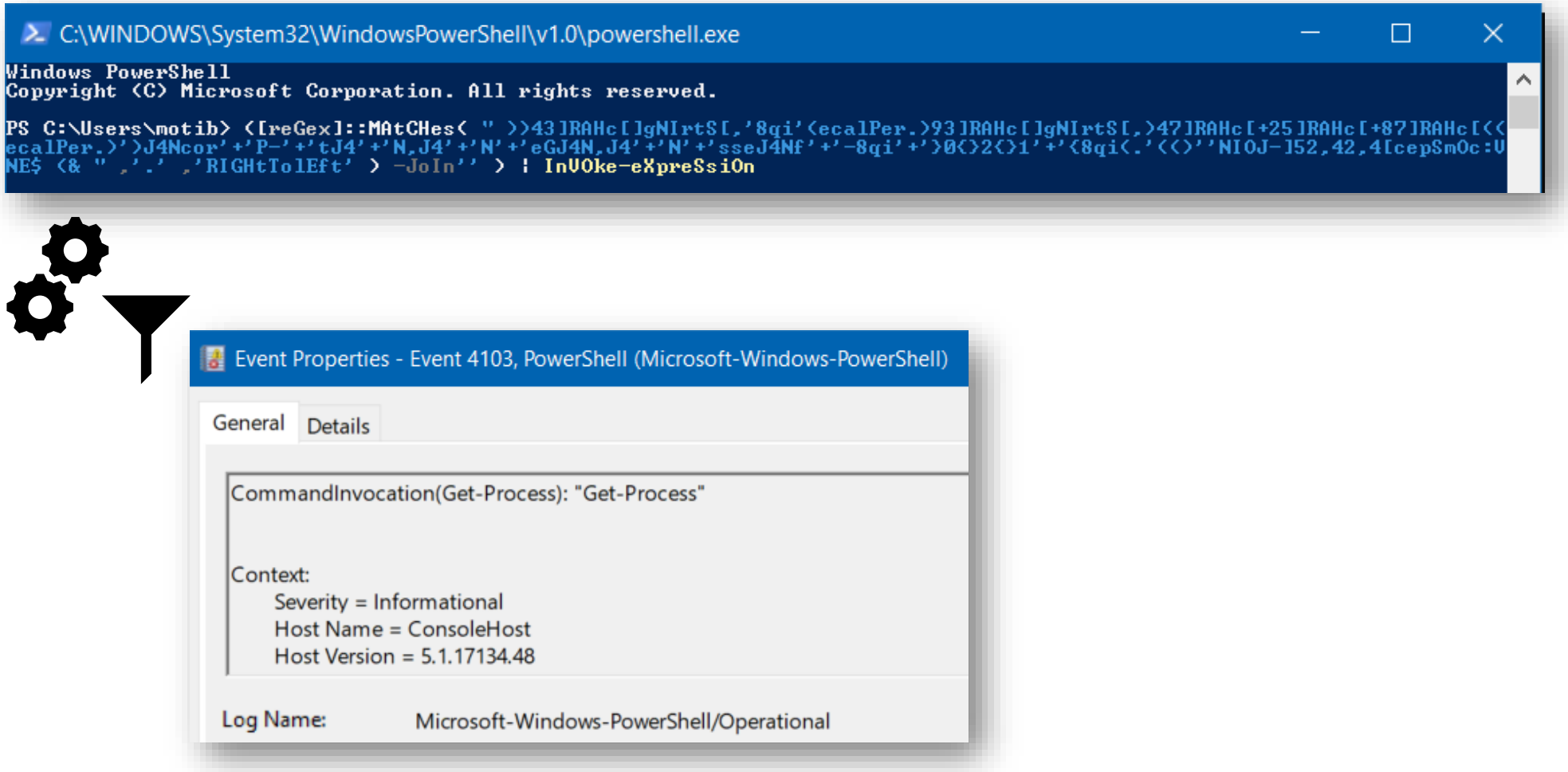
Includes dynamically generated code (Invoke-Expression)

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.*

# ScriptBlock Logging



# Security in PowerShell

## Transcription and logging

# Event IDs

Operational Log

- Microsoft-Windows-PowerShell/Operational
  - Increase the event log size to 1GB
  - Harden the security permissions

Events

- **4104** – Script block logging, **long script blocks span multiple events**

Optional invocation info (includes Runspace ID)

- 4105 – Start
- 4106 – Stop

# Important Event IDs

| Log Name | Event Id | Purpose |
|---|---|---|
| System | 104 | An event log was cleared |
| Security | 4656<br>4657 | Auditing of configured files or registry keys:<br>PowerShell profiles (*profile*.ps1)<br>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<br>**Warning** | ScriptBlock *automatic* logging – used APIs or techniques commonly associated with malware |
| Microsoft-Windows-PowerShell/Operational | 4104<br>**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 |

# Operational logs: Size and Permissions

```
(Get-WinEvent -ListLog Security).SecurityDescriptor
(Get-WinEvent -ListLog `
      Microsoft-Windows-PowerShell/Operational).SecurityDescriptor

$evt = Get-WinEvent -ListLog 'Microsoft-Windows-PowerShell/Operational'
$evt.MaximumSizeInBytes = 100mb
$evt.SecurityDescriptor = (
      Get-WinEvent -ListLog Security).SecurityDescriptor
$evt.SaveChanges()
```

The CustomSD can prevent attackers from harvesting credentials or other secrets from the PowerShell operational event logs

# Protected Logging

Encrypts PowerShell script block logging using Cryptographic Message Syntax (CMS).

Certificate requirements are the same as CMS (document encryption capable).

Must be enabled **in addition** to script block logging. (Same event ID 4104)

If the certificate configured cannot be located, then a warning event will be logged and logging will proceed unencrypted.

Use **Unprotect-CmsMessage** to view the decrypted event message body.

Use **Unprotect-CmsMessage -IncludeContext** to view the entire event object.

# Enable Protected Logging

In Windows 10 / Server 2016:
GPO → Windows Components → Administrative Templates → Event Logging

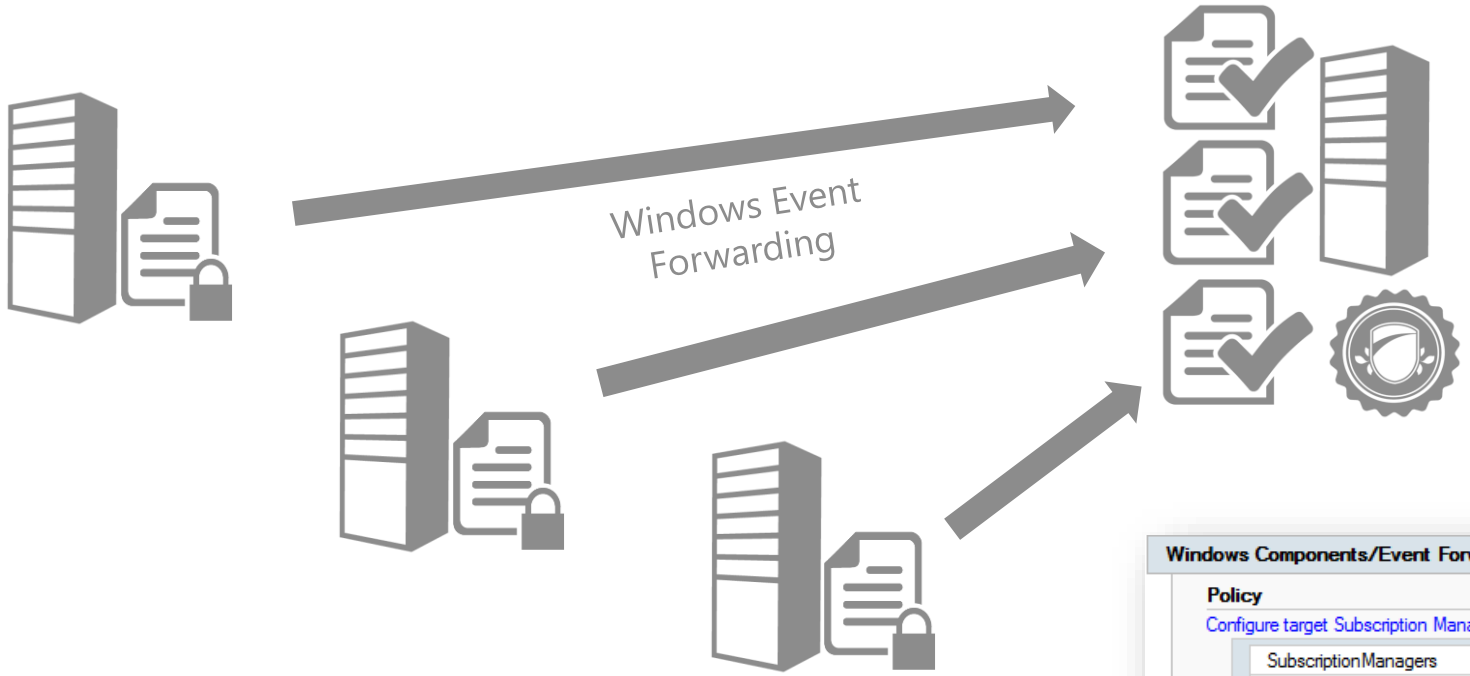
In PowerShell 5.x (Non Windows 10/2016):
Set the **EncryptionCertificate** & **EnableProtectedEventLogging**
Under **HKLM:\Software\Policies\Microsoft\Windows\EventLog\ProtectedEventLogging**

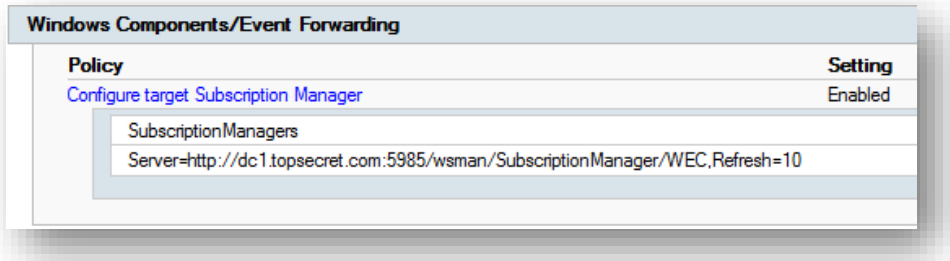
Requires an encryption certificate:

- The content of a base-64 encoded X.509 certificate
- The thumbprint of a certificate (present in the Local Machine certificate store)
- The subject name of a certificate (present in the Local Machine certificate store)
- The full path to a certificate file (can be local, or a remote share)

# Windows Event Forwarding

Use Windows Event Forwarding to archive logs to a central location




https://aka.ms/WEF

# Windows Event Forwarding

Windows Event Forwarding (WEF) reads any event log on a device in your organization and forwards the events you choose to a Windows Event Collector (WEC) server

- Built-in on Windows 2008 and above
- Configured via GPO
- Uses Windows Remote Management (Kerberos)
- Can (and should be) targeted to specific events
- Native EVTX (xml) log format
- "Push" log mode option available – reduce attack surface
- Control logging destiny

# Don’t just log

- So you logged everything and now have 5TB of logs. Daily.

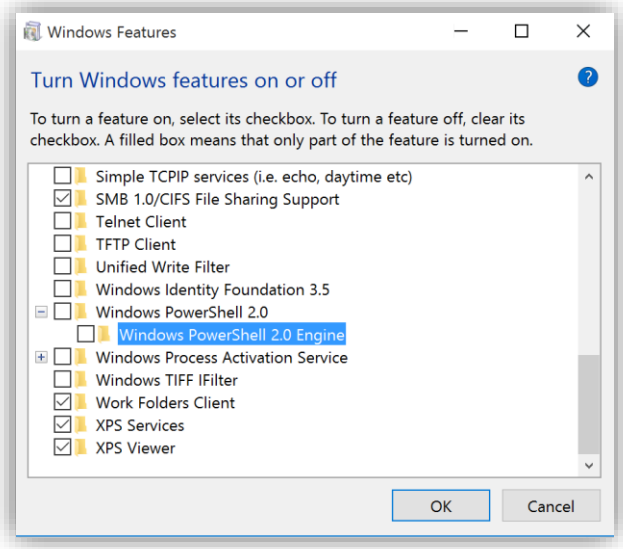
  Congratulations! Buy more storage

- Create alerts
- Cleanup false positives
- Create playbooks
- Automate playbooks where possible

# 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 to deny access to PowerShell 2.0 Engine specific .NET assemblies.

# Security in PowerShell

## Downgrade attacks

# PSv2 Engine Lifecycle Events

```
Get-WinEvent -LogName "Windows PowerShell" |
    Where-Object Id -eq 400 | Foreach-Object {
        $version = [Version] ($_.Message `
            -replace '(?s).*EngineVersion=([\d\.]+)*.*','$1')
        if($version -eq ([Version] "2.0")) { $_ }
}
```

# Agenda

- Module 1: Introduction & Context
- Module 2: Execution Policy
- Module 3: PowerShell without PowerShell.exe
- Module 4: Obfuscation
- Module 5: Transcription & ScriptBlock Logging
- Module 6: Working with Credentials
- Module 7: PSRemoting
- Module 8: Constrained Language, AppLocker & AMSI
- Module 9: Just Enough Administration
- Module 10: Windows Defender ATP

# Working with Credentials: Don't #1

```powershell
# Clear-text username and password
$userName = 'CONTOSO\Administrator'
$clearTextPassword = 'my$3cur3dP@55w0rd!'

# Convert clear-text password to a SecureString
$SecurePassword = $clearTextPassword | ConvertTo-SecureString -AsPlainText -Force

# Create credential object (same object that Get-Credential outputs)
$Cred = New-Object PSCredential -ArgumentList $Username, $SecurePassword
```

Warning: Anyone who gains access to this script gets the username and password

# Working with Credentials: Don't #2

```
# Convert the password to a SecureString

PS C:\> $cred = Get-Credential CONTOSO\Administrator
PS C:\> $EncryptedString = $cred.Password | ConvertFrom-SecureString
PS C:\> $EncryptedString
01000000d08c9ddf0115d1118c7a00c04fc297eb0100000071f193d84290394f8...
```

```
# It will only decrypt on machine/user that ran above lines
$EncryptedPassword = '01000000d08c9ddf0115d1118c7a00c04fc297eb0100000071...
$SecurePassword = $EncryptedPassword | ConvertTo-SecureString

$Username = 'CONTOSO\Administrator'
$Cred = New-Object PSCredential -ArgumentList $Username, $SecurePassword
```

Warning: Anyone who gains access to this script gets to use the credentials (and the username and password)

# Security in PowerShell

## Decrypting Passwords

# Decrypting Passwords

```
function Decrypt-SecureString {
    param([System.Security.SecureString]$SecureString)
    return [System.Runtime.InteropServices.Marshal]::PtrToStringUni(
        [System.Runtime.InteropServices.Marshal]::SecureStringToGlobalAllocUnicode(
            $SecureString)
    )
}

$pass = Read-Host 'Enter password' -AsSecureString
Decrypt-SecureString -SecureString $pass

$creds = Get-Credential
$creds.GetNetworkCredential().Password


function ConvertFrom-Base64Encoding {
    param([string]$string)
    [System.Text.Encoding]::Ascii.GetString([Convert]::FromBase64String($string))
}

ConvertFrom-Base64Encoding -string UzNjdXIzZFBANTV3MHJk
```

# Cryptographic Message Syntax (CMS)

WMF 5.0 and KB3000850 add cmdlets to encrypt and decrypt text files.

Based on RFC 5652: http://tools.ietf.org/html/rfc5652

Encryption certificates require a unique key usage identifier (EKU) to identify them as data encryption certificates (like the identifiers for 'Code Signing', 'Encrypted Mail').

```
PS C:\> Get-Command -Noun CmsMessage

CommandType Name                Version Source
----------- ----                ------- ------
Cmdlet      Get-CmsMessage      3.0.0.0 Microsoft.PowerShell.Security
Cmdlet      Protect-CmsMessage  3.0.0.0 Microsoft.PowerShell.Security
Cmdlet      Unprotect-CmsMessage 3.0.0.0 Microsoft.PowerShell.Security
```

# CMS: Protect the Message

```
# Get the script content as string:
$scriptContent = Get-Content -Path C:\Temp\plainTextScript.ps1

# Create the certificate:
$cert = New-SelfSignedCertificate -DnsName PSCms `
        -CertStoreLocation Cert:\CurrentUser\my `
            -KeyUsage KeyEncipherment, DataEncipherment, KeyAgreement `
                      -Type DocumentEncryptionCert

# Encrypt message:
$scriptContent | Protect-CmsMessage -To $cert -OutFile C:\Temp\encryptedScript.txt

# Verify the encrypted contents:
notepad C:\Temp\encryptedScript.txt

# Export the certificate:
$certPassword = 'Password1' | ConvertTo-SecureString -AsPlainText -Force
Export-PfxCertificate -Cert $cert -FilePath C:\Temp\CMS.pfx -Password $certPassword
```

# CMS: Unprotect the Message

```
# Import the certificate:
$certPassword = 'Password1' |
       ConvertTo-SecureString -AsPlainText -Force
Import-PfxCertificate -FilePath C:\Temp\CMS.pfx `
       -CertStoreLocation Cert:\CurrentUser\My -Password $certPassword


# Decrypt message (The cert needs to be installed on the machine):
$plainTextcode = Unprotect-CmsMessage -Path C:\Temp\encryptedScript.txt


# Invoke the code:
Invoke-Expression -Command $plainTextcode
```

# Working with Credentials: Options

- Use Scheduled Tasks
  - Protect the script with ACLs + Consider gMSA

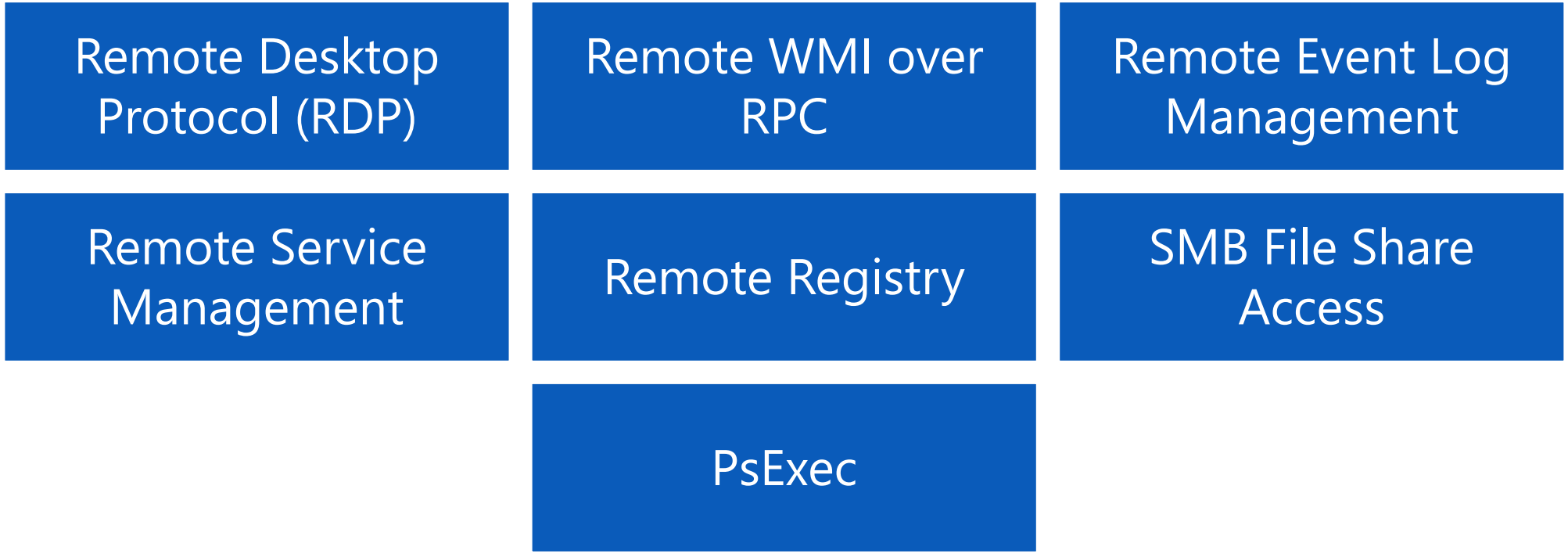
```
$principal = @{
    LogonType = 'Password'
    UserID    = 'CONTOSO\gMSA$'
    RunLevel  = 'Highest'
}
$task = @{
    TaskName  = 'myTask'
    Action    = $action
    Trigger   = $trigger
    Principal = New-ScheduledTaskPrincipal @principal
}
Register-ScheduledTask @task
```

- Use JEA
  - Covered in a later section

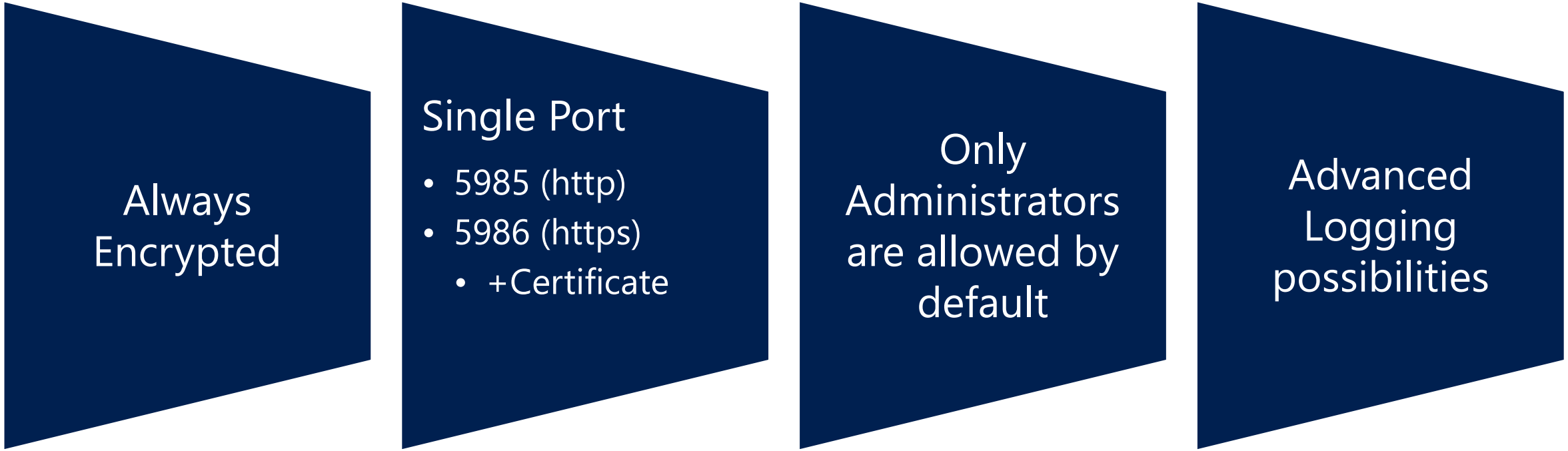
# Agenda

- Module 1: Introduction & Context
- Module 2: Execution Policy
- Module 3: PowerShell without PowerShell.exe
- Module 4: Obfuscation
- Module 5: Transcription & ScriptBlock Logging
- Module 6: Working with Credentials
- Module 7: PSRemoting
- Module 8: Constrained Language, AppLocker & AMSI
- Module 9: Just Enough Administration
- Module 10: Windows Defender ATP

# Remote Management Tools



# PowerShell Remoting




- Regardless of the transport (HTTP or HTTPS), PowerShell Remoting always encrypts all communication after initial authentication with a per-session AES-256 symmetric key
- Hackers will leave fingerprints everywhere, unlike other tools and utilities
- PowerShell should be the only tool you allow for remote administration

# HTTPS Listener

```
 # Create the HTTPS Listener
$listnerParams = @{
    ItemType                = 'Listener'
    Path                    = 'WSMan:\LocalHost\Listener'
    Address                 = '*'
    Transport               = 'HTTPS'
    CertificateThumbPrint = '73DCBA53D8EC5EBF2702F7093222269AE6A0D549'
}
New-Item @listnerParams



# Copy the HTTP Rule to create HTTPS Rule
Get-NetFirewallRule -Name WINRM-HTTP-In-TCP |
   Copy-NetFirewallRule -NewName WINRM-HTTPS-In-TCP

Get-NetFirewallRule -Name WINRM-HTTPS-In-TCP |
   Set-NetFirewallRule -LocalPort 5986 `
      -NewDisplayName 'Inbound rule for Windows Remote Management [TCP 5986]'
```

# Security in PowerShell

## Securing PSRemoting

# Agenda

- Module 1: Introduction & Context
- Module 2: Execution Policy
- Module 3: PowerShell without PowerShell.exe
- Module 4: Obfuscation
- Module 5: Transcription & ScriptBlock Logging
- Module 6: Working with Credentials
- Module 7: PSRemoting
- Module 8: Constrained Language, AppLocker & AMSI
- Module 9: Just Enough Administration
- Module 10: Windows Defender ATP

# Constrained Language

Restricts language elements that provide access to

- Win32 APIs
- COM objects
- .NET methods, property setters, types, and conversions
- Add-Type
- XAML-based workflows
- Jobs
- PowerShell Classes (because they create .NET classes)
- DSC configuration declarations
- Dot sourcing

*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.*

# Constrained Language

The ConstrainedLanguage language mode (PSv3 and above) permits all Windows cmdlets and all Windows PowerShell language elements, but it limits permitted types.

The Constrained Language mode was designed to work with system-wide application control solutions such as Windows Defender Application Control (WDAC) (formerly known as Device Guard User Mode Code Integrity, or UMCI)

Once Constrained Language Mode is enabled, many PowerShell attack tools don't work since they rely on components blocked by constrained language

```
Get-Help about_Language_Modes
```

# Constrained PowerShell & AppLocker

Available on WMF 5.0

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.

Now 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.*

# Device Guard

Starting Windows 10 / Windows Server 2016 you can enforce constrained language mode using:

- AppLocker
- Device Guard User Mode Code Integrity (UMCI)

Device Guard is preferred, because it is much harder to disable (even if you are an administrator), but AppLocker enforcement is better than nothing

There are constrained language mode bypasses, most of them don’t work when you use Application whitelisting

# Anti-Malware Scan Interface (AMSI)

Available on Windows 10 and Windows Server 2016

Protects 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

# Anti-Malware Scan Interface (AMSI)

Disk, Memory or Interactive (Input method doesn't matter)

Scripts are submitted to the AntiVirus/AntiMalware product by AMSI when the de-obfuscated plain code is presented to the script host. This means that obfuscation should help only to a limited extent.

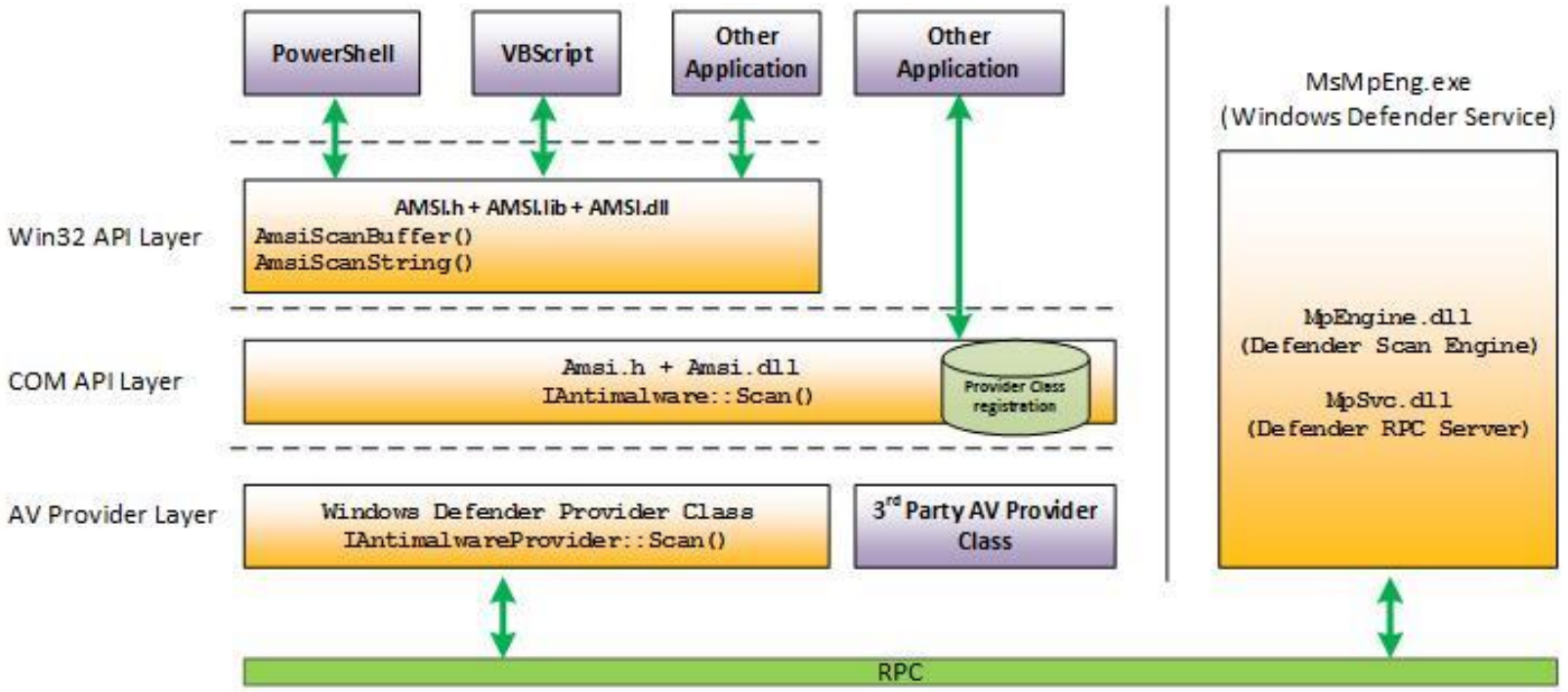
Code is "picked up" when submitted to the scripting host. So even in PowerShell without powershell.exe when referencing to System.Management.Automation the code will be analyzed.

# Vendors that support AMSI

| Vendor | Support |
|---|---|
| Windows Defender | https://blogs.technet.microsoft.com/mmpc/2015/06/09/windows-10-to-offer-application-developers-new-malware-defenses/ |
| AVG | https://support.avg.com/answers?id=906b00000008oUTAAY |
| BitDefender | https://forum.bitdefender.com/index.php?/topic/72455-antimalware-scan-service/ |
| ESET | https://forum.eset.com/topic/11645-beta-eset-endpoint-security-66-is-available-for-evaluation/ |
| Dr. Web | https://news.drweb.com/show/?i=11272&lng=en |
| Avast | https://forum.avast.com/index.php?topic=184491.msg1300884#msg1300884 |
| Kaspersky | https://help.kaspersky.com/KIS/2019/en-US/119653.htm |
| CrowdStrike Falcon | Yes |
| | |
| Trend Micro | N/A |
| Symantec | N/A |
| McAfee | N/A |
| Sophos | N/A |
| F-Secure | N/A |
| Avira | N/A |
| Panda | N/A |
| ThreatTrack Vipre | N/A |

N/A = Searches of the internet or company's site returned no evidence of implementation, although usually returned forum posts requesting implementation.

# Anti-Malware Scan Interface (AMSI)



# Anti-Malware Scan Interface (AMSI)

"EICAR" test for AMSI:

```
$base64 =
"FHJ+YHoTZ1ZARxNgUl5DX1YJEwRWBAFQAFBWHgsFAlEeBwAACh4LBAcDHgNSUAIHCwdQAgALBRQ="
$bytes = [Convert]::FromBase64String($base64)
$string = -join ($bytes | % { [char] ($_ -bxor 0x33) })

iex $string
```

# Security in PowerShell

## AMSI

# Disabling AMSI

## Disable real time monitoring

## Set-MpPreference -DisableRealtimeMonitoring $true

- A notification is shown to the user
- Needs to be run from an elevated shell
- Event ID 5001 (Microsoft-Windows-Windows Defender/Operational) - Windows Defender Real-Time Protection was disabled.

## Avoiding detection of in-memory download:

## Set-MpPreference -DisableIOAVProtection $true

- An elevated shell is still required and;
- Event ID 5004 (Microsoft-Windows-Windows Defender/Operational) - Windows Defender Real-Time Protection feature (IE Downloads and Outlook Express attachments) configuration has changed.

# Security in PowerShell

## Trying to disable AMSI

# Disabling AMSI is disabled by AMSI

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

```
At line:1 char:1
+ [Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetF
...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
This script contains malicious content and has been blocked by your
antivirus software.
    + CategoryInfo          : ParserError: (:) [],
ParentContainsErrorRecordException
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent
```

# Agenda

- Module 1: Introduction & Context
- Module 2: Execution Policy
- Module 3: PowerShell without PowerShell.exe
- Module 4: Obfuscation
- Module 5: Transcription & ScriptBlock Logging
- Module 6: Working with Credentials
- Module 7: PSRemoting
- Module 8: Constrained Language, AppLocker & AMSI
- Module 9: Just Enough Administration
- Module 10: Windows Defender ATP

# Scenario Overview

Too many people have too much access on too many servers.

If a hacker gets the password to a service account or admin account, then they have access to hundreds of servers (lateral traversal, pass-the-hash).

Role Based Access Control (RBAC) is usually application specific.

Role Based Access Control (RBAC) does not work when everyone has every role.

Admin permissions should not be binary.

How do I delegate server-level administration on an Active Directory Domain Controller without granting Domain Administrator access?

# Goals for JEA

Reduce the number of full admins

Limit movement of full admin accounts

Granular control over admin actions

Log all admin actions

# Definition

JEA – Just Enough Administration

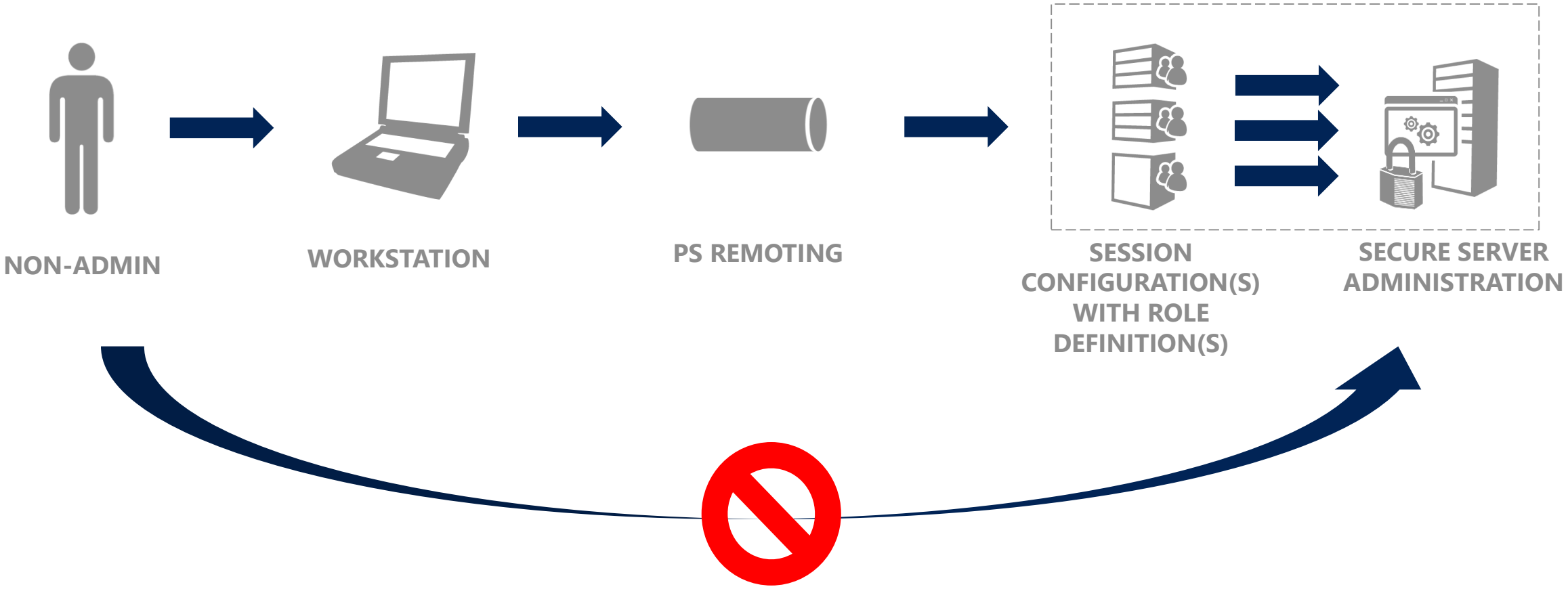
Role-based access control (RBAC) platform through Windows PowerShell

Allows **specific users** to perform **specific tasks** on **specific servers** without giving them full administrator rights

Removes administration privilege from as many user accounts as possible while still allowing them to do their job (*principle of least privilege*)

Uses PowerShell **constrained endpoints** to limit administration

# JEA Overview



# Use Cases

Auditors needing read-only access to systems

Service desk tasks (account resets, etc.)

Hosted environments with multi-tenant administration

Patching sensitive servers while protecting data

Delegated administrative tasks

# Prerequisites

WMF 5.0

- Windows Server 2016
- Windows Server 2012 or 2012R2 with WMF 5.0 RTM

Administrative rights on the server

Elevated PowerShell console

Server is domain joined

Remoting is enabled (default on 2012+)

(The same functionality can be accomplished on Server 2008 with PowerShell 3.0)

# History

Originally announced at TechEd 2014 as **xJEA**, a PowerShell DSC experimental resource.

Now built into WMF 5.0.

| **Old Way** | **New Way** |
| --- | --- |
| xJEA DSC module | Microsoft.PowerShell.Core<br>*New DSC resources planned* |
| CSV format role capabilities | New-PSRoleCapabilityFile |
| Local user account | Virtual account<br>Domain account |

# Remoting Review

PowerShell remoting connects to endpoints called **session configurations**.

Default endpoints specify local Administrator access.

Create your own **constrained endpoints** with restricted access and commands.

```
PS C:\> Get-PSSessionConfiguration | Select-Object Name

Name
----
microsoft.powershell
microsoft.powershell.workflow
microsoft.powershell32
microsoft.windows.servermanagerworkflows
MySessionConfig

PS C:\> Enter-PSSession -ComputerName MS1 -ConfigurationName MySessionConfig
```

# New Features

Session Configurations can now:

- Specify an automatic transcript folder
- Run as a domain or virtual account
- Specify visible external commands (in addition to cmdlets, functions, aliases, etc.)
- Contain one or more role definitions

Role definitions

- Grant specific access to specific users or groups

# Cmdlets

PowerShell core module now includes JEA (*Microsoft.PowerShell.Core*).

Existing / Updated

- New-PSSessionConfigurationFile
- Register-PSSessionConfiguration
- Get-PSSessionConfiguration
- New-PSSession, Invoke-Command, or Enter-PSSession

New

- New-PSRoleCapabilityFile
- Get-PSSessionCapability

# Key Concepts

Users and/or groups are granted access to a custom **endpoint** (session configuration).

PS session configurations **RunAs** a different user.

RunAs user can be a domain account or virtual account.

**SessionType** and **LanguageMode** restrict PowerShell capabilities.

**Role definitions** determine the variables, aliases, functions, providers, modules, cmdlets, parameters, etc. available in the session.

A server can have multiple session configurations and role definitions.

Logging and transcription **audit** session activities.

# Virtual Accounts

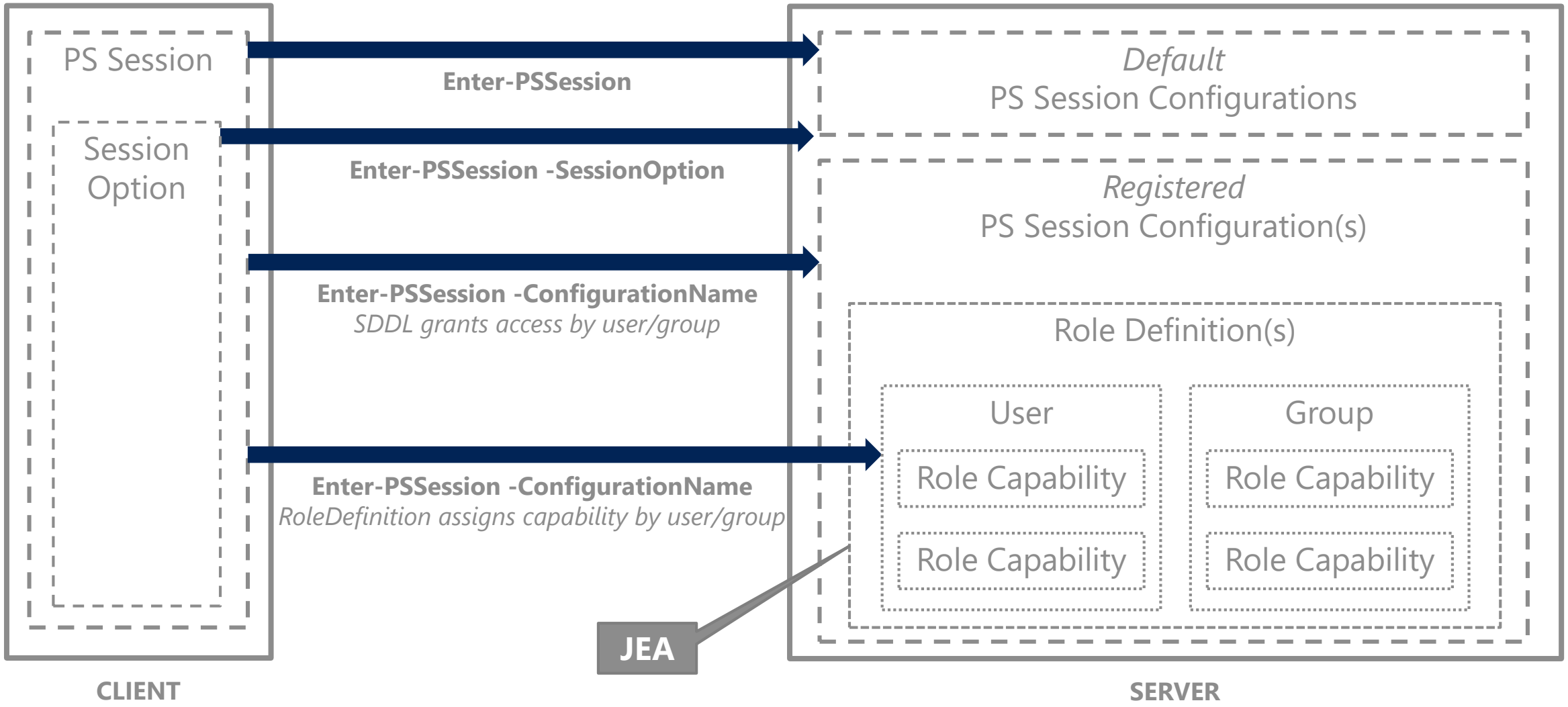
Previous **xJEA** used automatically-generated **local** accounts.

WMF 5.0 **JEA** uses either domain accounts or **virtual** accounts.

None of the following account types require password management:

- ***LocalSystem*** *- extensive privileges on the local computer and acts as the computer on the network, shared by many services, not used by JEA*
- ***NetworkService*** *- minimum privileges on the local computer and acts as the computer on the network, shared by many services, not used by JEA*
- **Virtual Accounts** – like NetworkService, but isolated per service, uniquely-generated name, recommended choice for JEA use

# Remoting Configuration Layers and JEA



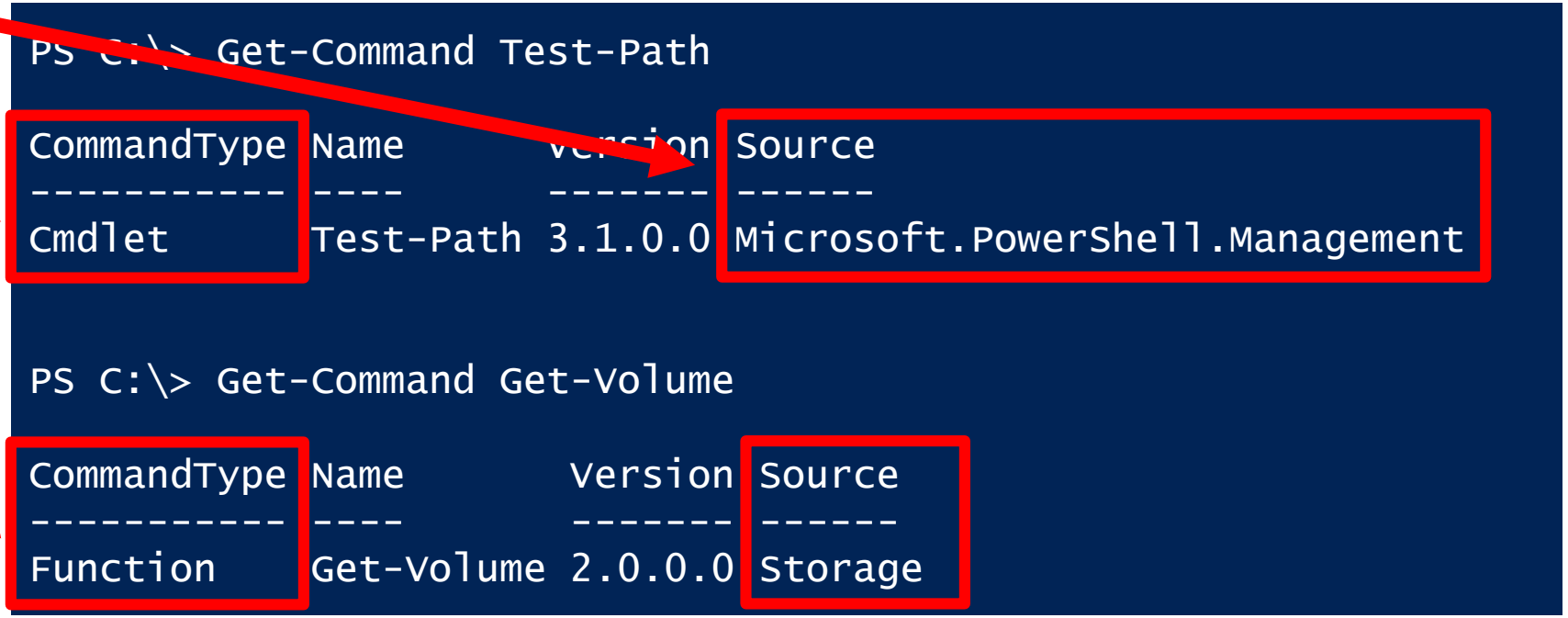
# Identify Users, Groups, and Tasks

Who needs access to the server?

What do they need to do?

What PowerShell resources are required?

- Modules
- Providers
- Cmdlets
- Functions
- Parameters
- Etc.



# Create Role Capability File

Defines toolset available to the role definition by **white-listing** PowerShell items

Lives in a custom module, usually empty

Assigned by domain user or group

**New-PSRoleCapabilityFile**

"$env:ProgramFiles\WindowsPowerShell\Modules

*ModuleName*\RoleCapabilities*RoleName*.psrc"

# Role Capabilities

**Meta Data**
Guid
Author
Description
CompanyName
Copyright

**Environment Setup**
ScriptsToProcess
TypesToProcess
FormatsToProcess
ModulesToImport
AssembliesToLoad

**Custom Items**
AliasDefinitions
FunctionDefinitions
VariableDefinitions
EnvironmentVariables

**Visible Items**
VisibleAliases
VisibleCmdlets
VisibleFunctions
VisibleProviders
VisibleExternalCommands

# Role Example

Grant access to basic file system and share tools

```
$StorageRoleCapability = @{
    Path                = "$ModPath\$ModName\RoleCapabilities\Storage.psrc"
    CompanyName         = 'Contoso'
    Description         = 'Read access to storage information'
    ModulesToImport     = 'Storage', 'SMBShare', 'Microsoft.PowerShell.Management',
                          'Microsoft.PowerShell.Core'
    VisibleAliases      = 'cd','dir'
    VisibleCmdlets      = 'Get-ChildItem','Get-Item','Set-Location'
    VisibleFunctions    = 'Get-*Disk*','Get-*Volume*','Get-*Partition*','Get-SMB*'
    VisibleProviders    = 'FileSystem'
}

New-PSRoleCapabilityFile @StorageRoleCapability
```

# Create and Register Session Configuration

Extracts users and/or groups from role definition(s) and adds their SIDs to the SDDL

```
$SessionConfig = @{
    CompanyName          = 'Contoso'
    SessionType          = 'RestrictedRemoteServer'
    RunAsVirtualAccount = $true
    RoleDefinitions      = @{ 'CONTOSO\StorageAdmins' = @{RoleCapabilities = 'Storage'} }
    TranscriptDirectory = 'C:\Transcripts'
}

New-PSSessionConfigurationFile -Path ".\StorageEndpoint.pssc" @SessionConfig

Register-PSSessionConfiguration -Name StorageEndpoint -Path ".\StorageEndpoint.pssc"
```

*SDDL (Security Descriptor Definition Language) is a string representation of the endpoint ACL (Access Control List) containing SIDs (Security IDentifiers) of accounts and their level of access.*

# Validate and Connect

Validate functionality in restricted experience using **Get-PSSessionCapability**

Connect with **Enter-PSSession**, **New-PSSession**, or **Invoke-Command**

**ConfigurationName** parameter matches the session configuration name on the target

**Credential** parameter matches the role definition user or group

```
Get-PSSessionCapability -ConfigurationName StorageEndpoint -Username CONTOSO\Alice


$cred = Get-Credential -UserName CONTOSO\Alice -Message 'Enter password'
Enter-PSSession -ComputerName SERVER1 -ConfigurationName StorageEndpoint -Credential $cred
```

# Auditing

Be sure event logs and transcripts are secured locally

- Use Windows Event Forwarding for greater security
- Set permissions on the transcript directory

View transcripts for commands and output

Audit RunAs authentications in **Microsoft-Windows-WinRM/Operational**, EventID **193**

```
Request for user S-1-5-21-3841426147-920288206-1542301449-1611 (CONTOSO\Alice)
will be executed using WinRM virtual account S-1-5-94-1452785133
```

# Example use case: DNS admin

Many Windows environments co-locate DNS Server on Active Directory Domain Controllers

Someone managing DNS needs access to DNS tools + some standard Windows tools like service manager and event logs

To manage those extra tools, DNS admins require local admin permission (aka domain admin on a DC)

Should a DNS admin be in control of all of your identities?

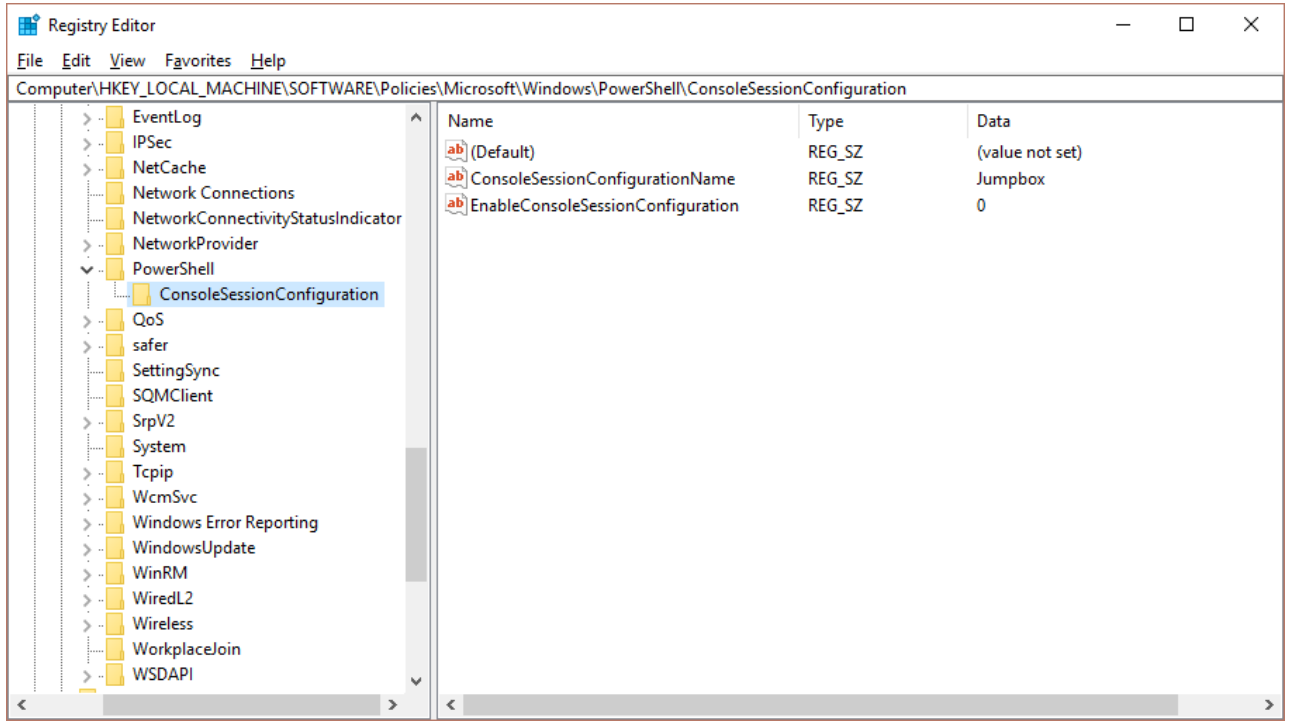
# Security in PowerShell

## JEA

# Local Sandboxing

Configure all local sessions to run in the constraints of a JEA session.

Useful on 'Jumpbox' servers



# Local Sandboxing

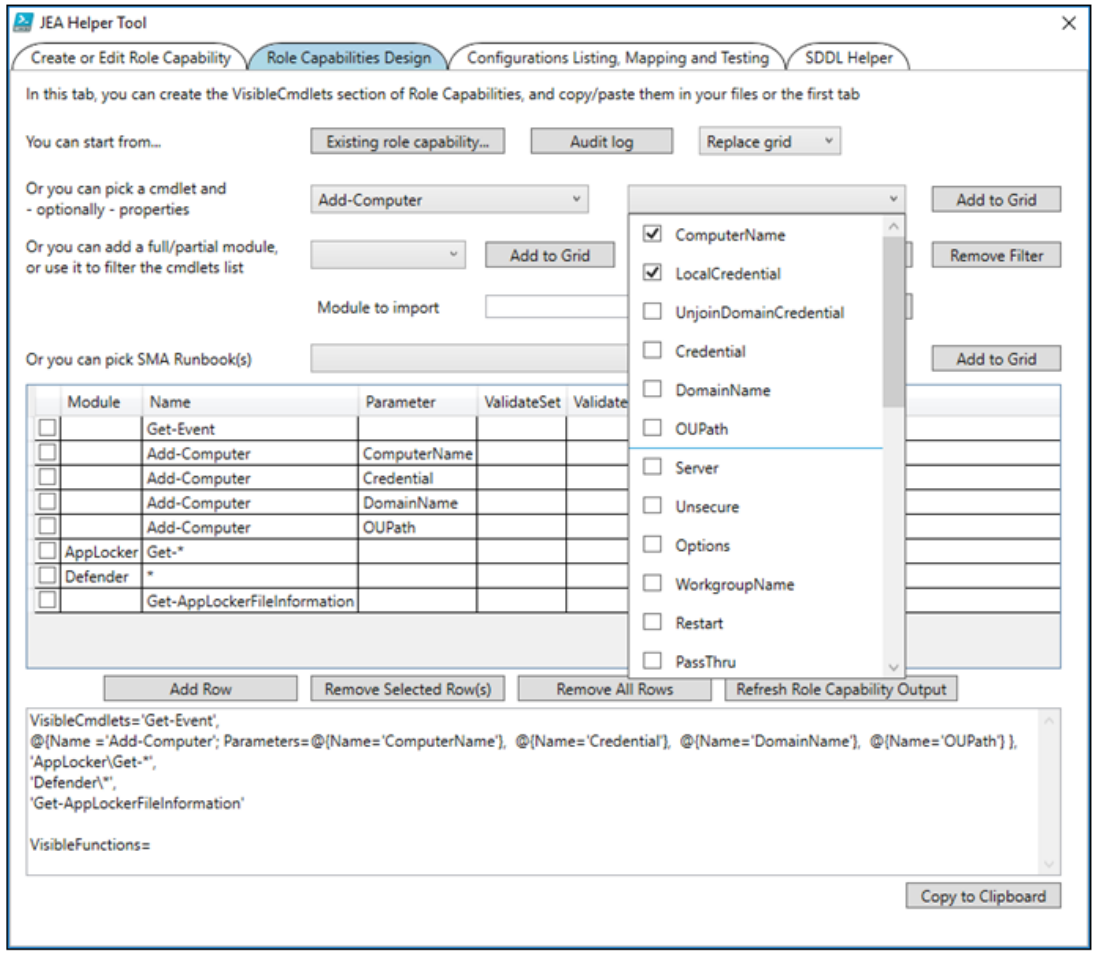
```
New-PSSessionConfigurationFile -Path C:\Temp\sandbox.pssc `
       -SessionType RestrictedRemoteServer `
              -RunAsVirtualAccount -VisibleFunctions TabExpansion2 `
                     -VisibleExternalCommands C:\Windows\System32\whoami.exe

Register-PSSessionConfiguration -name sandbox -Path C:\temp\sandbox.pssc


$key = 'HKLM:\SOFTWARE\Policies\Microsoft\Windows\PowerShell\ConsoleSessionConfiguration'
New-Item -Path $key –Force
New-ItemProperty -Path $key -Name ConsoleSessionConfigurationName -Value sandbox
New-ItemProperty -Path $key -Name EnableConsoleSessionConfiguration -Value 1
```

# JEA Helper Tool

GUI tool to help creating and testing Role Capabilities and Session Configurations files

Written in PowerShell, so you can look at the script underneath, to understand how it creates, maps, tests, registers, unregisters sessions on the local machine

Helps generate the “Security Descriptor Definition Language” (SDDL) syntax




https://gallery.technet.microsoft.com/JEA-Helper-Tool-20-6f9c49dd

# Security in PowerShell

## JEA Helper

# Deploying JEA

**Identify**

Session capability file (Identify users, groups, etc.)

**Restrict**

Role capability file (scope of cmdlets, custom functions and parameters)

**Deploy**

Preferably via DSC

**Test & Correct**

As needed

# Tips

**Transcript directories** must exist already, or transcription will fail

**Secure** transcript files and event logs

Audit RunAs authentications in **Microsoft-Windows-WinRM/Operational**, EventID **193**

Add the visible function **TabExpansion2** for user convenience

Set the LanguageMode to **NoLanguage**

**Wildcard** configuration is evil

# Tips cont'd

Beware of **dangerous commands** to allow:

- Granting the connecting user admin privileges to bypass JEA
  - Add-ADGroupMember, Add-LocalGroupMember, net.exe, dsadd.exe
- Running arbitrary code to bypass protections
  - Start-Process, New-Service, Invoke-Item, Invoke-WmiMethod, Invoke-CimMethod, Invoke-Expression, Invoke-Command, New-ScheduledTask, Register-ScheduledJob

Use **signing** to protect role capability files from tampering

None of this matters if you do not **take away admin rights** and remote desktop access to the servers

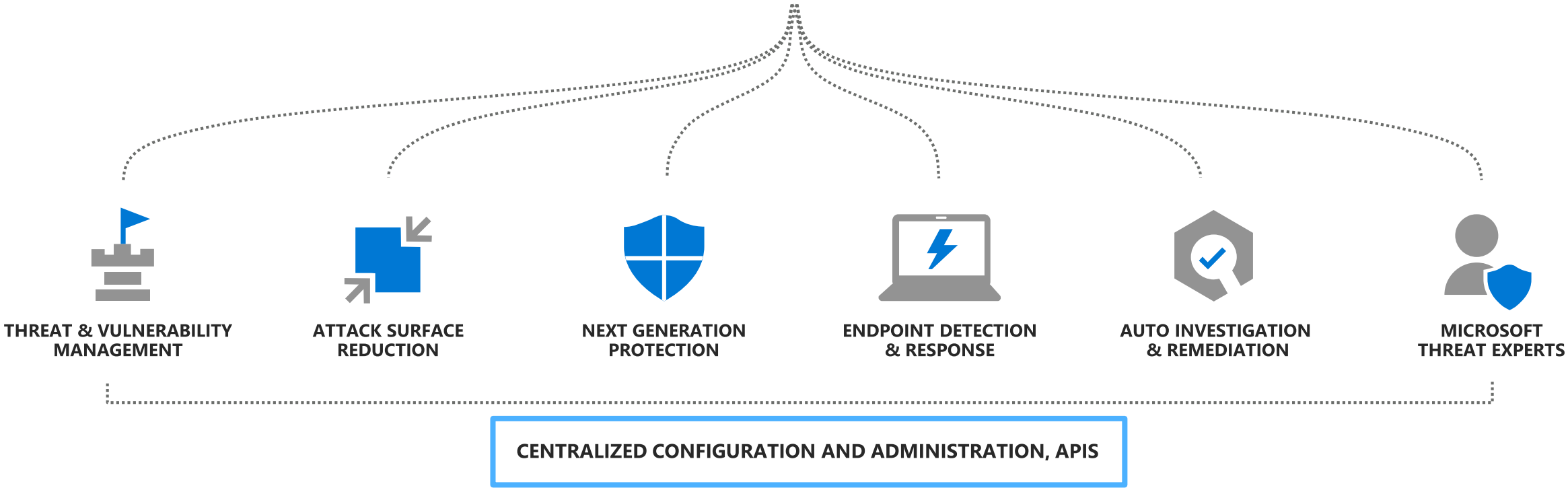
**Read the full JEA whitepaper at** http://aka.ms/jea.

# Agenda

- Module 1: Introduction & Context
- Module 2: Execution Policy
- Module 3: PowerShell without PowerShell.exe
- Module 4: Obfuscation
- Module 5: Transcription & ScriptBlock Logging
- Module 6: Working with Credentials
- Module 7: PSRemoting
- Module 8: Constrained Language, AppLocker & AMSI
- Module 9: Just Enough Administration
- Module 10: Windows Defender ATP

# Microsoft Defender
## Advanced Threat Protection

**Built-in. Cloud-powered.**



## WINDOWS DEFENDER SECURITY CENTER

Alerting

Investigation

Response

## CENTRALIZED MANAGEMENT

Security Analytics

Reporting

Configuration

## CLOUD

URL/IP Reputation

File Reputation

Sandbox Service

Machine Learning

Anomaly/ Behavioral Detection

Automated Investigation & Response

Threat Intelligence

## ENDPOINT

**Attack Surface Reduction**

HW Based Isolation

Application Control

Exploit Protection

**Next Generation Protection**

Web Protection

Machine Learning

Runtime Emulator

Script And Memory Attack Protection

Anti-virus

**Endpoint Detection & Response**

Behavioral Monitoring

Automatic Investigation Remediation

**3rd Party Platform Partners (Cross Os)**

**Graph APIs and SIEM connectors**

# WINDOWS DEFENDER ATP

## Detect, investigate & respond to advanced attacks

### Built in to Windows 10, not bolted on

Sensors are built deep into Windows 10.
No additional deployment and infrastructure.

### Behavior-based, cloud-powered breach detection

Signature-less, intelligent, behavioral, machine learning and past attack detections.
Actionable, correlated alerts for known and unknown adversaries.

### Best of breed investigation experience

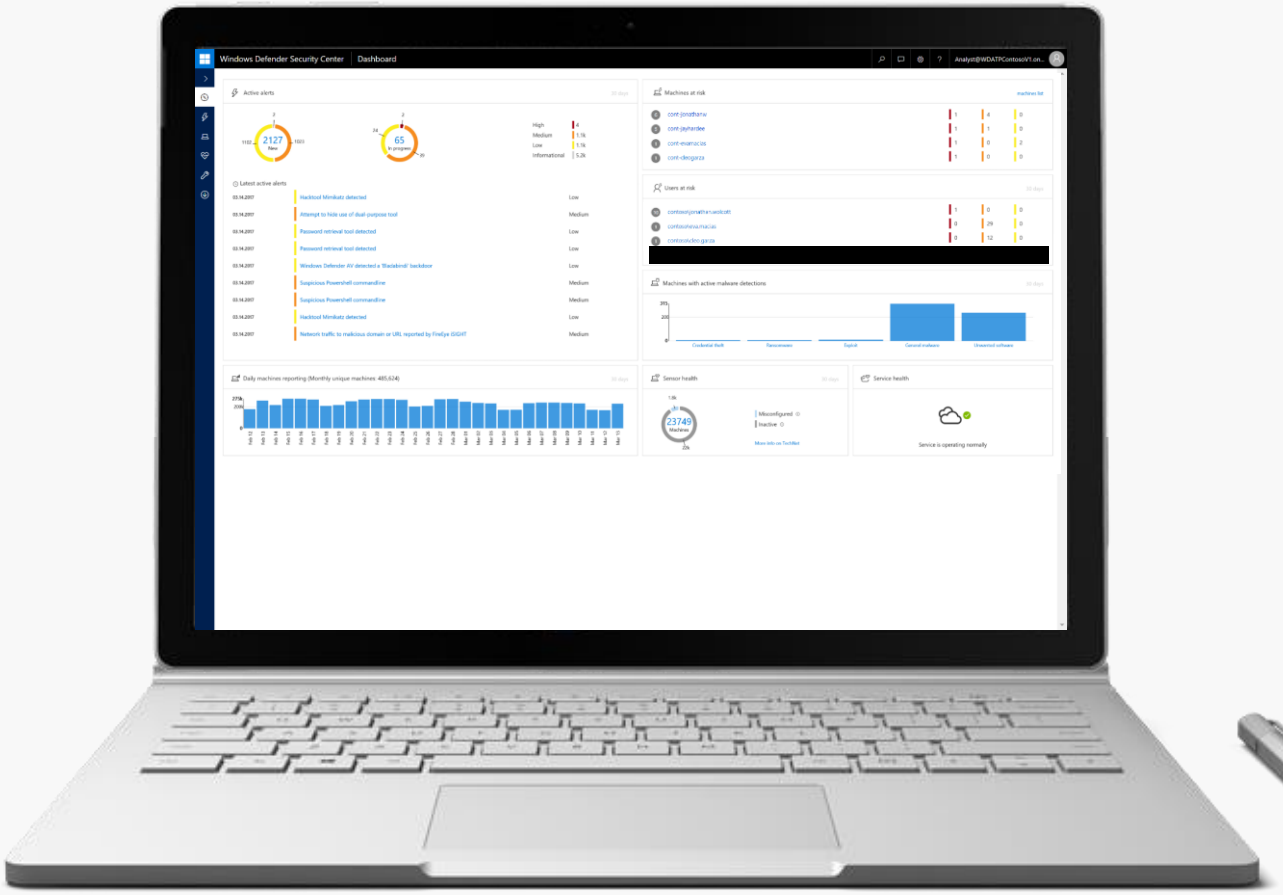
Up to 6 months of historical data for every endpoint, global search across machines, files, processes registry, users, IPs, URLs

### Unique threat intelligence knowledge base

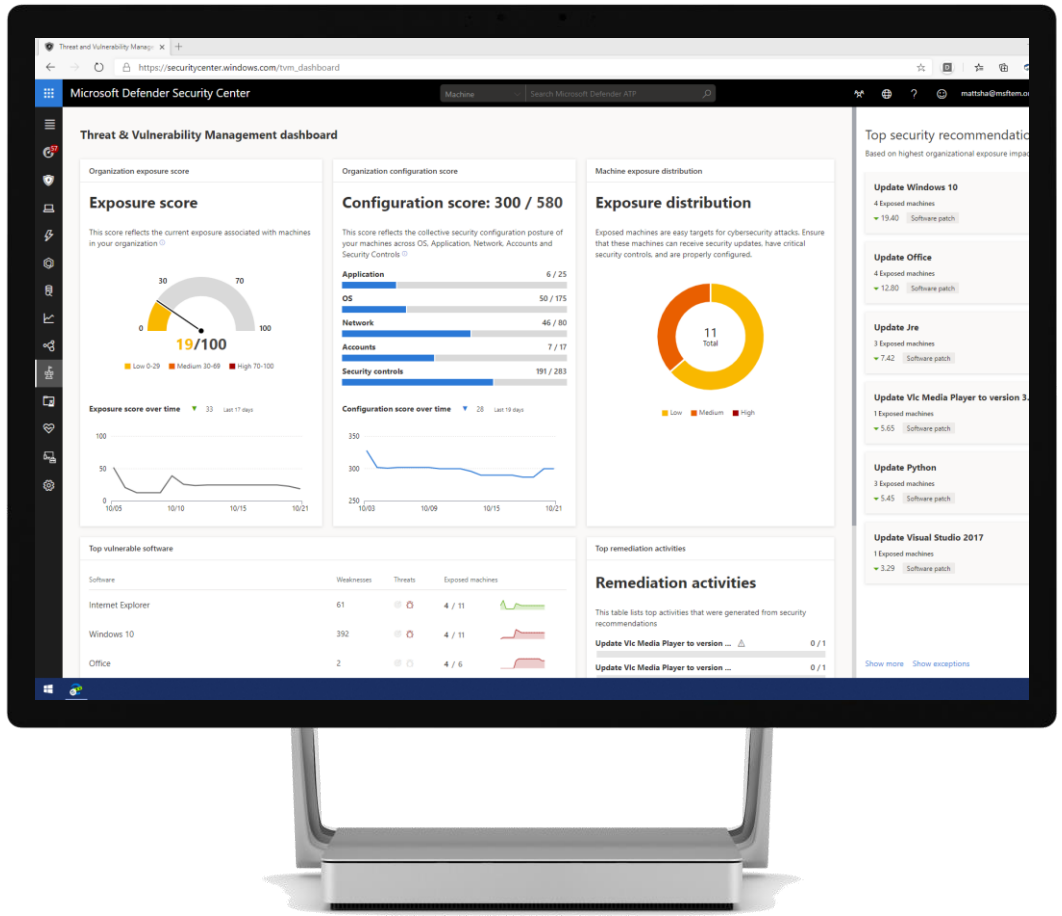
Unparalleled threat optics provide detailed actor profiles
1st and 3rd party threat intelligence data.

### Response based on the Windows stack

Rich SOC toolset ranging from machine-specific intervention or forensic actions to cross-machine blacklisting

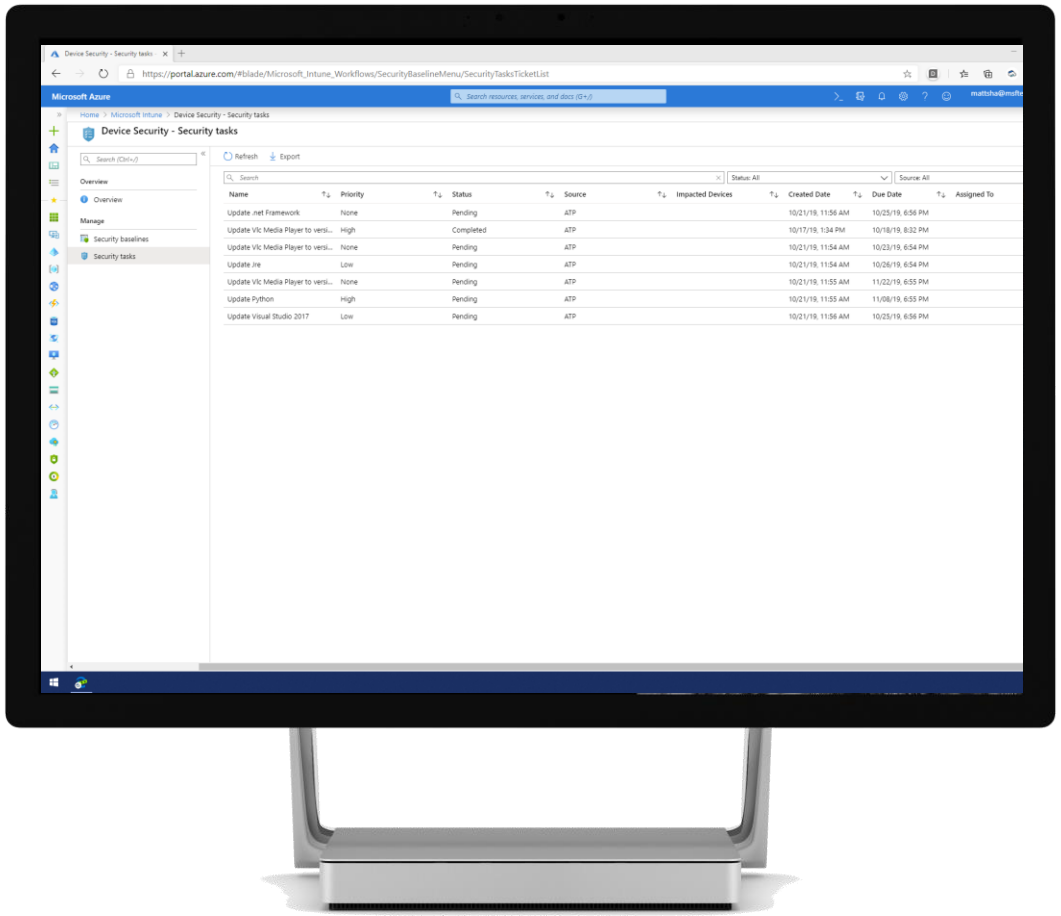


# Microsoft Defender ATP




SecOps

# Microsoft Endpoint Manager




SecAdmin

# Endpoint detection and response

Detect. Investigate. Respond to advanced attacks.


## Client

DEEP OS RECORDING SENSOR

## Cloud

MACHINE LEARNING, BEHAVIORAL & ANOMALY DETECTION

RESPONSE AND CONTAINMENT

SANDBOX ANALISYS

RICH INVESTIGATION ACROSS MACHINES, FILES, USERS, IPS, URLS

REALTIME AND HISTORICAL THREAT HUNTING

THREAT INTELLIGENCE AND CUSTOM DETECTIONS

## Discover and respond to 0-day and Advanced Attacks

**Cloud driven**

**Deep optics**

**Realtime search**

**6 months of historical data**

**Custom Detections and TI feeds**

**Can run side-by-side with 3rd party solutions**

# Platform coverage for the EDR capabilities

## Integrated configuration management in the cloud

Natively managed by Microsoft Intune and System Center Configuration Manager (SCCM)

Support for 3rd party MDM platforms

Provides coverage for the modern diverse network

| CLIENT | SERVER | CROSS-PLATFORM |
| --- | --- | --- |
| | | Bitdefender, SentinelOne, ziften |
| NEW! | | Bitdefender, SentinelOne, ziften |
| NEW! | | Lookout |

Manage Tags | Initiate Automated Investigation | Initiate Live Response Session | Collect investigation package | Run antivirus scan | Restrict app execution | Isolate machine | Consult a threat expert | Action center

## apt29-client3

Domain
apt29.org

OS
Windows 10 x64
Version 1903
Build 18362

Risk level
High

Exposure level
Medium

Health state
Inactive

Active alerts
27

Active incidents
32

Azure ATP alerts
Machine not found at Azure ATP

First seen
Aug 13, 2019, 1:34:05 PM

Last seen
Aug 21, 2019, 5:08:29 PM

IP addresses
See IP addresses info

### Active alerts
180 days

**Risk level: High**

**27 active alerts in 2 incidents**

High (4) Medium (18) Low (4) Informational (1)

### Logged on users
30 days

**No users logged on**

There were no logged on users during the given time range

### Security assessments

**Exposure level: Medium (53)**

**1 Security recommendation**

**16 Installed software**

**129 Discovered vulnerabilities**

Alerts | Timeline | Security recommendations | Software inventory | Discovered vulnerabilities

Customize columns | 30 items per page | Page 1

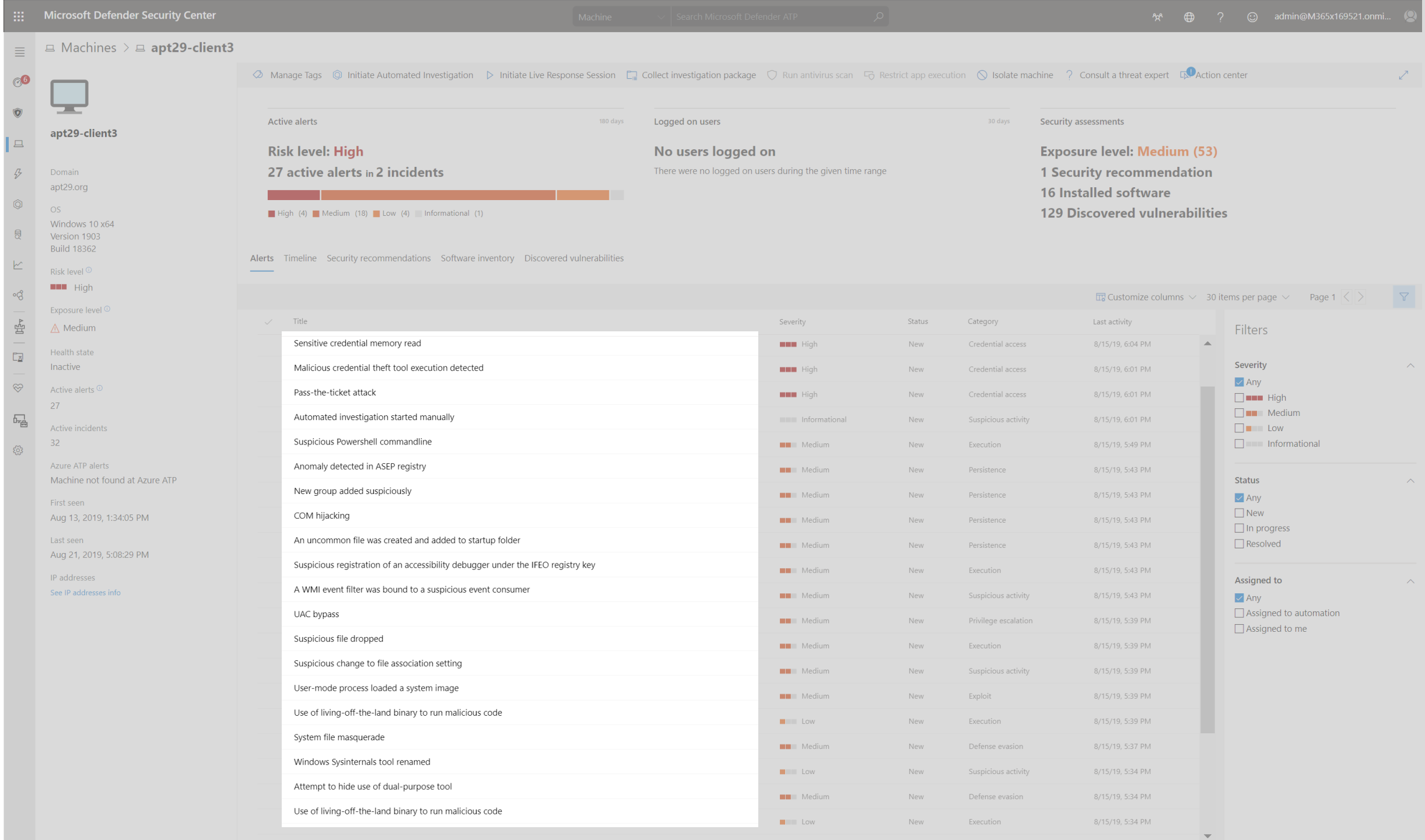
| Title | Severity | Status | Category | Last activity |
|---|---|---|---|---|
| Sensitive credential memory read | High | New | Credential access | 8/15/19, 6:04 PM |
| Malicious credential theft tool execution detected | High | New | Credential access | 8/15/19, 6:01 PM |
| Pass-the-ticket attack | High | New | Credential access | 8/15/19, 6:01 PM |
| Automated investigation started manually | Informational | New | Suspicious activity | 8/15/19, 6:01 PM |
| Suspicious Powershell commandline | Medium | New | Execution | 8/15/19, 5:49 PM |
| Anomaly detected in ASEP registry | Medium | New | Persistence | 8/15/19, 5:43 PM |
| New group added suspiciously | Medium | New | Persistence | 8/15/19, 5:43 PM |
| COM hijacking | Medium | New | Persistence | 8/15/19, 5:43 PM |
| An uncommon file was created and added to startup folder | Medium | New | Persistence | 8/15/19, 5:43 PM |
| Suspicious registration of an accessibility debugger under the IFEO registry key | Medium | New | Execution | 8/15/19, 5:43 PM |
| A WMI event filter was bound to a suspicious event consumer | Medium | New | Suspicious activity | 8/15/19, 5:43 PM |
| UAC bypass | Medium | New | Privilege escalation | 8/15/19, 5:39 PM |
| Suspicious file dropped | Medium | New | Execution | 8/15/19, 5:39 PM |
| Suspicious change to file association setting | Medium | New | Suspicious activity | 8/15/19, 5:39 PM |
| User-mode process loaded a system image | Medium | New | Exploit | 8/15/19, 5:39 PM |
| Use of living-off-the-land binary to run malicious code | Low | New | Execution | 8/15/19, 5:39 PM |
| System file masquerade | Medium | New | Defense evasion | 8/15/19, 5:37 PM |
| Windows Sysinternals tool renamed | Low | New | Suspicious activity | 8/15/19, 5:34 PM |
| Attempt to hide use of dual-purpose tool | Medium | New | Defense evasion | 8/15/19, 5:34 PM |
| Use of living-off-the-land binary to run malicious code | Low | New | Execution | 8/15/19, 5:34 PM |

### Filters

**Severity**
- [x] Any
- [ ] High
- [ ] Medium
- [ ] Low
- [ ] Informational

**Status**
- [x] Any
- [ ] New
- [ ] In progress
- [ ] Resolved

**Assigned to**
- [x] Any
- [ ] Assigned to automation
- [ ] Assigned to me


Microsoft Defender Security Center
Machine
Search Microsoft Defender ATP
admin@M365x169521.onmi...
Machines
apt29-client3
Manage Tags
Initiate Automated Investigation
Initiate Live Response Session
Collect investigation package
Run antivirus scan
Restrict app execution
Isolate machine
Consult a threat expert
Action center
apt29-client3
Domain
apt29.org
OS
Windows 10 x64
Version 1903
Build 18362
Risk level
High
Exposure level
Medium
Health state
Inactive
Active alerts
27
Active incidents
32
Azure ATP alerts
Machine not found at Azure ATP
First seen
Aug 13, 2019, 1:34:05 PM
Last seen
Aug 21, 2019, 5:08:29 PM
IP addresses
See IP addresses info
Active alerts
180 days
Risk level: High
27 active alerts in 2 incidents
High (4) Medium (18) Low (4) Informational (1)
Logged on users
30 days
No users logged on
There were no logged on users during the given time range
Security assessments
Exposure level: Medium (53)
1 Security recommendation
16 Installed software
129 Discovered vulnerabilities
Alerts
Timeline
Security recommendations
Software inventory
Discovered vulnerabilities
Customize columns
30 items per page
Page 1
Title
Severity
Status
Category
Last activity
Sensitive credential memory read
High
New
Credential access
8/15/19, 6:04 PM
Malicious credential theft tool execution detected
High
New
Credential access
8/15/19, 6:01 PM
Pass-the-ticket attack
High
New
Credential access
8/15/19, 6:01 PM
Automated investigation started manually
Informational
New
Suspicious activity
8/15/19, 6:01 PM
Suspicious Powershell commandline
Medium
New
Execution
8/15/19, 5:49 PM
Anomaly detected in ASEP registry
Medium
New
Persistence
8/15/19, 5:43 PM
New group added suspiciously
Medium
New
Persistence
8/15/19, 5:43 PM
COM hijacking
Medium
New
Persistence
8/15/19, 5:43 PM
An uncommon file was created and added to startup folder
Medium
New
Persistence
8/15/19, 5:43 PM
Suspicious registration of an accessibility debugger under the IFEO registry key
Medium
New
Execution
8/15/19, 5:43 PM
A WMI event filter was bound to a suspicious event consumer
Medium
New
Suspicious activity
8/15/19, 5:43 PM
UAC bypass
Medium
New
Privilege escalation
8/15/19, 5:39 PM
Suspicious file dropped
Medium
New
Execution
8/15/19, 5:39 PM
Suspicious change to file association setting
Medium
New
Suspicious activity
8/15/19, 5:39 PM
User-mode process loaded a system image
Medium
New
Exploit
8/15/19, 5:39 PM
Use of living-off-the-land binary to run malicious code
Low
New
Execution
8/15/19, 5:39 PM
System file masquerade
Medium
New
Defense evasion
8/15/19, 5:37 PM
Windows Sysinternals tool renamed
Low
New
Suspicious activity
8/15/19, 5:34 PM
Attempt to hide use of dual-purpose tool
Medium
New
Defense evasion
8/15/19, 5:34 PM
Use of living-off-the-land binary to run malicious code
Low
New
Execution
8/15/19, 5:34 PM
Filters
Severity
Any
High
Medium
Low
Informational
Status
Any
New
In progress
Resolved
Assigned to
Any
Assigned to automation
Assigned to me

Alerts > **Suspicious file dropped**

## Suspicious file dropped

**This alert is part of incident (1)**

Actions

Automated investigation is not applicable to alert type

Severity: Medium
Category: Execution
Detection source: EDR
Detection technology: Behavioral

### Alert context

apt29-client3
apt29\lily.jarvis

First activity: 08.13.2019 | 16:05:51
Last activity: 08.15.2019 | 17:39:22

### Description

Certain characteristics—reputation ratings, prevalence, signer information, or name—or a combination of these characteristics, can indicate that a file is malicious. Such files are typically dropped when moving laterally or establishing persistence.

### Recommended actions

1. Review the file and the process that dropped the file. If necessary, subm
2. Review the machine timeline for any suspicious activities that have occu
3. If confirmed malicious, contain and mitigate the breach. Stop suspicious passwords, block IP addresses and URLs, and install security updates.

### Alert process tree

- wininit.exe
  - services.exe
    - svchost.exe
      - cleanup.exe
        *svchost.exe created file cleanup.exe*
- userinit.exe
  - explorer.exe
    - powershell.exe
      - powershell.exe
        - powershell.exe
          - svchost.exe
            *powershell.exe created file svchost.exe*
          - svchost.exe
          - notepad.exe
            - powershell.exe
              - powershell.exe
                - SkypeUpdateChecker.exe
                  *powershell.exe created file SkypeUpdateChecker.exe*
                  *Detected as HackTool:Win32/Yoasimee.A by Windows Defender AV*
                - rundll32.exe
                  - SkypeUpdateChecker.exe
                    *Detected as HackTool:Win32/Yoasimee.A by Windows Defender AV*
                    - sdclt.exe

## powershell.exe

Open File page | Stop and Quarantine File | Add Indicator | Collect file | ...

### Execution details

| | |
|---|---|
| Process name | powershell.exe |
| Execution time | Aug 13, 2019, 4:05:50.353 PM |
| Path | C:\Windows\System32\WindowsPowerShell\v1.0 |
| Integrity level | Medium |
| Access privileges (UAC) | Restricted |
| Process ID | 8568 |
| Command line | "powershell.exe" -ExecutionPolicy Bypass -Window Hidden -File C:\Users\lily.jarvis\AppData\Local\Temp\amuPyhJZpH.ps1 |
| SHA1 | 36c5d12033b2eaf251bae61c00690ffb17fddc87 |

### File details

| | |
|---|---|
| SHA1 | 36c5d12033b2eaf251bae61c00690ffb17fddc87 |
| SHA256 | 908b64b1971a979c7e3e8ce4621945cba84854cb98d76367b791a6e |
| MD5 | cda48fc75952ad12d99e526d0b6bf70a |
| Size | 451.58 KB |
| Signer | Microsoft Windows |
| Issuer | Microsoft Windows Production PCA 2011 |

### File Detections

| Alerts | High | Medium | Low | Informational |
|---|---|---|---|---|
| | 1 | 17 | 4 | 1 |

VirusTotal detection ratio: 0/71

Malware detected
None

### Observed worldwide

| | |
|---|---|
| Count | 460k |
| First seen | 8 months ago |
| Last seen | 10 hours ago |

### Observed in organization (last 30 days)

| | |
|---|---|
| Count | 0 |

Machines > **apt29-client3**

Manage Tags | Initiate Automated Investigation | Initiate Live Response Session | Collect investigation package | Run antivirus scan | Restrict app execution | Isolate machine | Consult a threat expert

Alerts | **Timeline** | Security recommendations | Software inventory | Discovered vulnerabilities


**Highlighted alert:** Suspicious Powershell commandline

Custom range... | From 8/9/2019 | To 8/16/2019 | Search events

| Event time | Event | Additional information | User | Entities |
|---|---|---|---|---|
| Aug 15, 2019, 5:34:28.496 PM | powershell.exe ran Powershell command: 'Format-Table' | | lily.jarvis | powershell.exe > powershell.exe > PowerShell command |
| Aug 15, 2019, 5:34:27.002 PM | powershell.exe successfully established connection with 10.0.20.10:443 (apt29-ex... | | lily.jarvis | powershell.exe > powershell.exe > 10.0.20.10:443 (apt29-exchange.ap |
| Aug 15, 2019, 5:34:26.987 PM | Event of type [LdapSearch] observed on machine | | | |
| Aug 15, 2019, 5:28:22.111 PM | Suspicious Powershell commandline | Execution | | |
| Aug 15, 2019, 5:28:22.111 PM | powershell.exe created process cmd.exe | Execution | lily.jarvis | powershell.exe > powershell.exe > cmd.exe |
| Aug 15, 2 | | | lily.jarvis | powershell.exe > powershell.exe > PowerShell command |
| Aug 15, 2 | powershell.exe created file starter.bat | | lily.jarvis | powershell.exe > powershell.exe > starter.bat |
| Aug 15, 2 | | | lily.jarvis | powershell.exe > powershell.exe > PowerShell command |
| Aug 15, 2019, 5:28:21.643 PM | powershell.exe ran Powershell command: 'format-default' | | lily.jarvis | powershell.exe > powershell.exe > PowerShell command |
| Aug 15, 2019, 5:28:21.643 PM | powershell.exe ran Powershell command: 'out-lineoutput' | | lily.jarvis | powershell.exe > powershell.exe > PowerShell command |
| Aug 15, 2019, 5:28:21.268 PM | powershell.exe ran Powershell command: 'New-Object' | | lily.jarvis | powershell.exe > powershell.exe > PowerShell command |
| Aug 15, 2019, 5:28:21.199 PM | powershell.exe ran Powershell command: 'Import-Module' | | lily.jarvis | powershell.exe > powershell.exe > PowerShell command |
| Aug 15, 2019, 5:28:21.158 PM | powershell.exe ran Powershell command: 'Out-Default' | | lily.jarvis | powershell.exe > powershell.exe > PowerShell command |
| Aug 15, 2019, 5:28:21.157 PM | powershell.exe ran Powershell command: 'ZynCxlHzDa.ps1' | | lily.jarvis | powershell.exe > powershell.exe > PowerShell command |
| Aug 15, 2019, 5:28:20.878 PM | powershell.exe created file __PSScriptPolicyTest_pdhkevxm.5bm.ps1 | | lily.jarvis | powershell.exe > powershell.exe > __PSScriptPolicyTest_pdhkevxm.5b |
| Aug 15, 2019, 5:28:20.716 PM | powershell.exe created process powershell.exe | SuspiciousActivity | lily.jarvis | explorer.exe > powershell.exe > powershell.exe |
| Aug 15, 2 | | | | explorer.exe > powershell.exe > PowerShell command |
| Aug 15, 2 | powershell.exe created file ZynCxlHzDa.ps1 | | | explorer.exe > powershell.exe > ZynCxlHzDa.ps1 |
| Aug 15, 2 | | | | explorer.exe > powershell.exe > PowerShell command |

**explorer.exe**

**powershell.exe**

**powershell.exe**

| | |
|---|---|
| Process name | powershell.exe |
| Execution time | Aug 15, 2019, 5:28:20.715 PM |
| Path | c:\windows\system32\windowspowershell\v1.0\powershell.exe |
| Integrity level | Medium |
| Access privileges (UAC) | Restricted |
| Process ID | 1288 |
| Command line | "powershell.exe" -ExecutionPolicy Bypass -Window Hidden -File C:\Users\lily.jarvis\AppData\Local\Temp\ZynCxlHzDa.ps1 |
| SHA1 | 36c5d12033b2eaf251bae61c00690ffb17fddc87 |
| Signer | Microsoft Windows |
| Issuer | Microsoft Windows Production PCA 2011 |

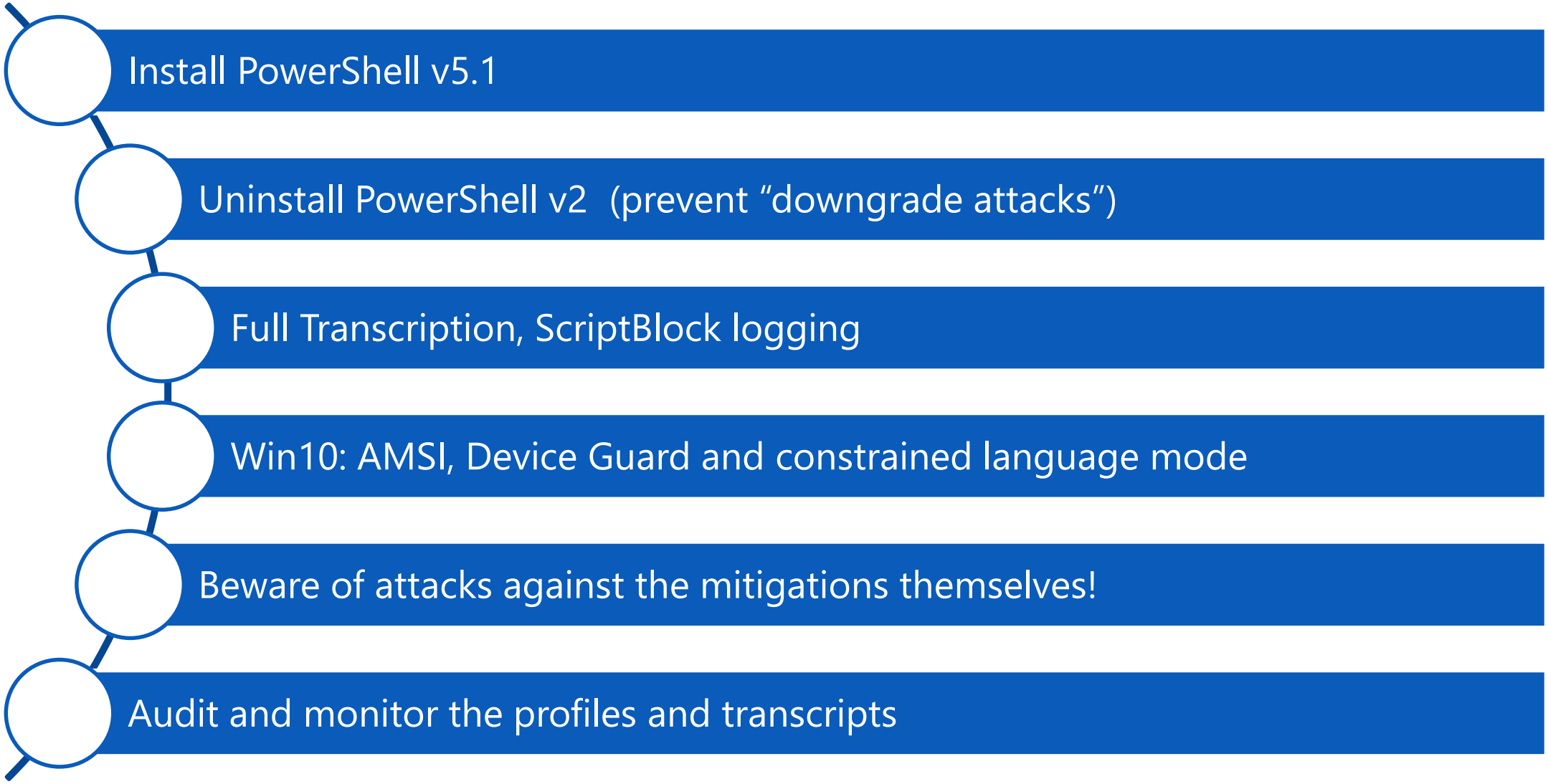
**cmd.exe**

| | |
|---|---|
| Process name | cmd.exe |
| Path | C:\Windows\System32 |
| Integrity level | Medium |
| Access privileges (UAC) | Restricted |
| Process ID | 4704 |
| Command line | cmd.exe /c ""C:\Users\lily.jarvis\AppData\Local\Temp\starter.bat" " |
| SHA1 | a1dbd4949df9e892e52201b06a2d24aa5082b3d5 |
| SHA256 | d0ceb18272966ab62b8edff100e9b4a6a3cb5dc0f |
| Signer | Microsoft Windows |
| Issuer | Microsoft Windows Production PCA 2011 |

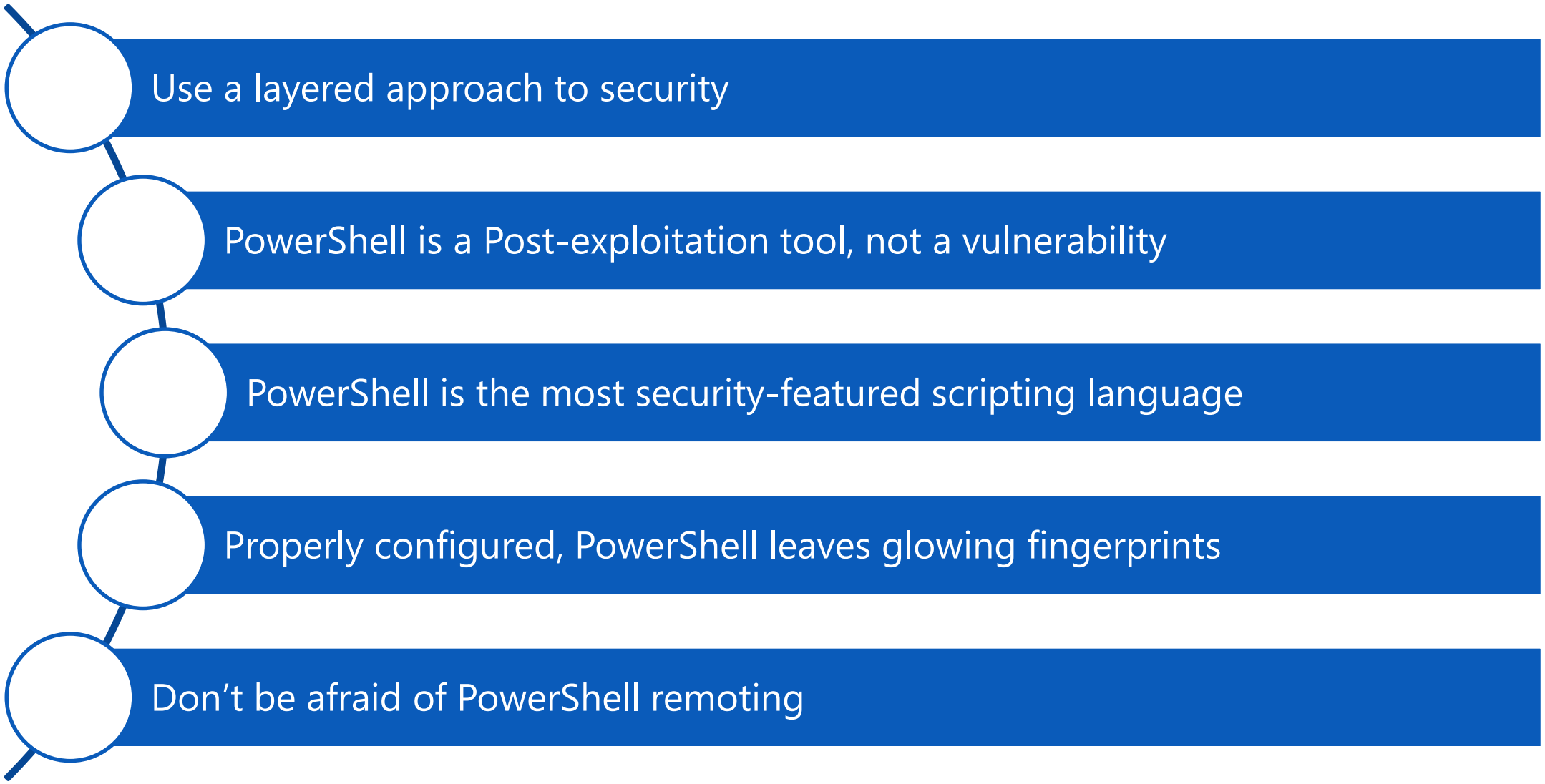
# Security in PowerShell

## Windows Defender ATP

# PowerShell Security Best Practices



# Key Takeaways




Microsoft