



# PowerShell

SECURESET ACADEMY  
HACKING 101 SERIES

## What is PowerShell?



SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

### PowerShell:

Built on **.NET framework**. It is a task-based command-line shell and scripting language.

Designed for system administrators and power-users, to automate the administration of multiple operating systems (Linux, macOS, Unix, and Windows) and the processes related to the applications that run on those operating systems.

## PowerShell History



SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

### PowerShell:

The first version of PowerShell was released in November 2006 for different Windows Versions and Distributions:

- XP
- Server 2003
- Vista

PowerShell versions come associated with .NET versions or revisions.

With Windows 10 is packaged delivered with .NET version 5, so it would be PowerShell version5 .

PowerShell is also installed natively on:

- Server 2008 R2
- Server 2012
- Server 2012 R2
- server 2016
- 7 Service Pack 1

- 8.1 (But we won't talk about that disaster)

In addition to the standard command-line shell, you can also find the Windows PowerShell ISE .

ISE stands for Integrated Scripting Environment , and it is a graphical user interface that allows you to easily create different scripts without having to type all the commands in the command line. Think of it like a IDE for Python.

IDE stands for Integrated Developer Environment. It is a GUI (Graphical User Interface) application that assists developers in writing code.

## Help is not far away

```
PS C:\Users\yenri> get-help get-process

NAME
    Get-Process

SYNTAX
    Get-Process [[-Name <string[]>] [-ComputerName <string[]>] [-Module] [-FileVersionInfo] [<CommonParameters>]
    Get-Process [[-Name <string[]>] -IncludeUserName [<CommonParameters>]
    Get-Process -Id <int[]> -IncludeUserName [<CommonParameters>]
    Get-Process -Id <int[]> [-ComputerName <string[]>] [-Module] [-FileVersionInfo] [<CommonParameters>]
    Get-Process -InputObject <Process[]> -IncludeUserName [<CommonParameters>]
    Get-Process -InputObject <Process[]> [-ComputerName <string[]>] [-Module] [-FileVersionInfo] [<CommonParameters>]

ALIASES
    gps
    ps

REMARKS
    Get-Help cannot find the Help files for this cmdlet on this computer. It is displaying only partial help.
    -- To download and install Help files for the module that includes this cmdlet, use Update-Help.
    -- To view the Help topic for this cmdlet online, type: "Get-Help Get-Process -Online" or
       go to https://go.microsoft.com/fwlink/?LinkID=113324.

PS C:\Users\yenri>
```

SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

Windows PowerShell includes detailed Help topics that explain Windows PowerShell concepts and the Windows PowerShell language, very similar to Linux **man** pages.

There are also Help topics for each cmdlet and provider and Help topics for many functions and scripts.

```
Get-Help <command>
Get-help get-process
```

To get help for a Windows **PowerShell** command, type Get-Help followed by the command name, such as: Get-Help Get-Process. To get a list of all help topics on your system, type Get-Help \*. You can display the whole help topic or use the parameters of the Get-Help cmdlet to get selected parts of the topic, such as the syntax, parameters, or examples.

What can PowerShell do?

# Possibilities



Automate Administration



Automate Defenses



Automate Attacks

**SECURESET.COM**

©2018 SecureSet Academy, Inc. | All Rights Reserved

PowerShell as a tool that helps you automate and quickly solve a lot of tedious administration tasks.

PowerShell's capabilities allow you to simplify and automate tedious and repetitive tasks by creating scripts and combining multiple commands together.

Is it beatific or diabolical? Its just a tool, it can be used for ill as well as for good.

## Words

5 PowerShell Essentials		
Concept	What's it Do?	A Handy Alias
PS C:\> <b>Get-Help</b> [cmdlet] -examples	Shows help & examples	PS C:\> help [cmdlet] -examples
PS C:\> <b>Get-Command</b>	Shows a list of commands	PS C:\> gcm *[string]*
PS C:\> <b>Get-Member</b>	Shows properties & methods	PS C:\> [cmdlet]   gm
PS C:\> <b>ForEach-Object</b> { \$_ }	Takes each item on the pipeline and handles it as \$_	PS C:\> [cmdlet]   % { [cmdlet] \$_ }
PS C:\> <b>Select-String</b>	Searches for strings in files or output, like grep	PS C:\> sls -path [file] -pattern [string]

SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

Some fun this to do with **PowerShell**:

- Report all of the USB devices installed
- Perform your favorite CMD tasks in PowerShell
- Kill a process in PowerShell instead of Task Manager
- Export NTFS folder permissions — recursive or not
- Export AD users
- Export AD users failing password policies
- Create/terminate AD accounts
- Add/remove security groups
- Machine management - wallpaper, home drive mapping,
- Internet explorer configurations



## What in the world is a CMDLET?



SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

In PowerShell, administrative tasks are generally performed by **cmdlets**.

**PowerShell** cmdlets use the Verb-Noun format as in Get-Service, Stop-Service, or Import-Csv. The verb portion of the cmdlet name indicates the action to be performed on the noun. Typically cmdlets used to request information use the Get verb, as is the case with Get-Process or Get-Content. Commands used to modify something will usually begin with the verb Set, while those adding a new entity to something often begin with Add or New. In many cases these verb-noun combinations can be guessed or predicted because of the standard naming convention.

To retrieve the basic help for a PowerShell cmdlet you can use **Get-Help <cmdlet name>**. This displays the information such as the cmdlet's description, syntax, and related links. Alternatively you can view the help for a specific cmdlet by using the **-?** switch, as in **Get-Process -?**. Additional information such as parameter usage and examples can be requested from Get-Help using the **-Detailed** switch. The **-Full** switch adds alias information as well as details on the variable types used for input and output.

## Application

```
PS C:\> help get-aduser -parameter identity

-Identity <ADUser>
  Specifies an Active Directory user object by providing one of the following property values. The identifier in
  parentheses is the LDAP display name for the attribute. The acceptable values for this parameter are:

  -- A Distinguished Name
  -- A GUID (objectGUID)
  -- A Security Identifier (objectsid)
  -- A SAM Account Name (sAMAccountName)

  The cmdlet searches the default naming context or partition to find the object. If two or more objects are
  found, the cmdlet returns a non-terminating error.

  This parameter can also get this object through the pipeline or you can set this parameter to an object
  instance.

  Required?                true
  Position?                1
  Default value
  Accept pipeline input?    True (ByValue)
  Accept wildcard characters? false
```

SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

This applied PowerShell task focuses on a very common IT task, which is grabbing members of an Active Directory group.

You will need to have the Windows [Remote Server Administration Tools](#) (RSAT) installed and **configured to use the Active Directory PowerShell module**.

## Active Directory

```
PS C:\> get-adgroupmember "chicago It"

distinguishedName : CN=Paul Newby,OU=IT,OU=Departments,OU=Employees,DC=GLOBOMANTICS,DC=local
name              : Paul Newby
objectClass       : user
objectGUID        : 791917e5-8593-4db2-ae4c-c00b2c3c49cc
SamAccountName    : pnewby
SID               : S-1-5-21-2552845031-2197025230-307725880-14309

distinguishedName : CN=Stephanie Quick,OU=IT,OU=Departments,OU=Employees,DC=GLOBOMANTICS,DC=local
name              : Stephanie Quick
objectClass       : user
objectGUID        : 34b63a5b-3510-45a5-8903-9561921b60ba
SamAccountName    : squick
SID               : S-1-5-21-2552845031-2197025230-307725880-15284

distinguishedName : CN=Chris Graham,OU=Marketing,OU=Sales and
                  : Marketing,OU=Departments,OU=Employees,DC=GLOBOMANTICS,DC=local
name              : Chris Graham
objectClass       : user
objectGUID        : 06570516-caa0-44d5-bf02-4140c6c19498
SamAccountName    : cgraham
SID               : S-1-5-21-2552845031-2197025230-307725880-1140
```

SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

Active Directory works well with PowerShell, it is well automated using these scripts.

Adding new users, removing individuals that have left the company are really good examples of how a **SA** or **Server Administrator** might automate their job function and improve their productivity.

## Application

```
PS C:\Users\yenri> Get-FileHash -Algorithm md5 .\openstego.ini

Algorithm      Hash                                     Path
-----
MD5            5AEB2E4308494C336CF5BD1BAD18B182      C:\Users\yenri\openstego.ini

PS C:\Users\yenri> Get-FileHash -Algorithm sha1 .\openstego.ini

Algorithm      Hash                                     Path
-----
SHA1          EA9AC8A898A181E51FF4378E410C91C64F1ACC67 C:\Users\yenri\openstego.ini
```

SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

Hashing – a specific applied cryptographic method.

A [cryptographic hash function](#) is a function which everybody can compute efficiently (there is nothing secret about it), and which offers some interesting characteristics:

- Its output has a fixed, small size (128 bits for MD5, 160 bits for SHA-1).
- It is deterministic (hash the same input twice, possibly with different machines or implementations, and you will still get the same output twice).

**DOWNLOAD VALIDATION:** The **small output size** is the reason we use them for downloads. If you obtain the **MD5 hash** from a reputable source (the *SSL-enabled* Web site of the publisher), then you can download the actual data from anywhere, e.g. a P2P network like [BitTorrent](#).

Once you have the data, you hash it (on *your* machine) and then check that the MD5 or SHA-1 hash value matches the one you got from the reputable source. On match, you *know* that you have the right file, down to the last bit, because, in order to feed you an altered file while keeping the correct hash value.

**File Hashing** is used for **FILE INTEGRITY MONITORING** – Ensuring a file stays the same.

**File Hashing** is also used as the basics for **signature based Antivirus and Malware Detection**.

## Threat Intelligence Automation - SAO

```
### Bad Evil Naughty Nasty Awful Mean Hurtful Recalcitrant Bad IP List ###
```

```
101.200.81.187
103.19.89.118
103.230.84.239
103.4.52.150
103.7.59.135
104.238.158.106
104.247.219.41
109.127.8.242
109.229.210.250
109.229.36.65
113.29.230.24
120.31.134.133
```

```
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$true}

# Uses the .Net Object Net.WebClient to scrape the listed website and store the contents in a text file.
$blocklist = New-Object Net.WebClient
$blocklist.DownloadString("https://threatlab.com/assets/Powercat/Powercat_Hacking_Tools.txt") > .\tempTicIP.txt

#checks for blank text file and exits the program if the file is blank
Get-Content .\tempTicIP.txt | Measure-Object -word
if ($word -eq 0){
    Break
}

#Get-Content will put each individual line in the text file as an individual object which sets up the "if" loop below.
$blocklist = Get-Content .\tempTicIP.txt

# removes temp blocklist text file
Remove-Item .\tempTicIP.txt
```

SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

PowerShell can be used in a emerging area of Cybersecurity. **SAO – Security Automation and Orchestration**

Scripts can be used to automate the collection of CTI – Cybersecurity Threat Intelligence from open source or paid or vendor threat feeds.

These lists can be imported into a number of cybersecurity tools or appliances such as:

Stateful and Next Gen Firewalls

IDS / IPS systems

Corporate Web Filters

Corporate Network Proxies

SIEMs

DLP Data Loss Prevention Systems

# Threat Intelligence Automation - SAO

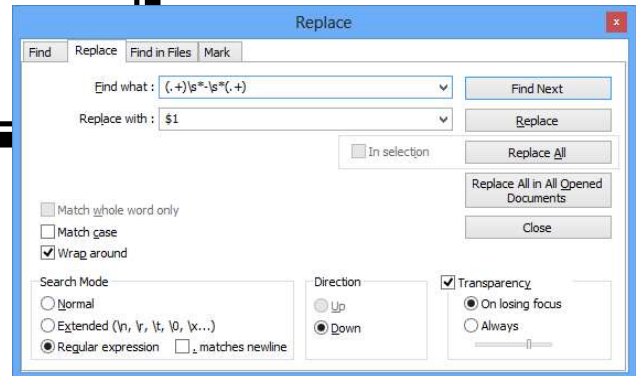
```
# if a line in $blocklist is not a header column AND the max number of objects has not been
# reached ($ItemMax) then strip of the extraneous content and append to text file.
$blocklist | ForEach-Object{
    if( $_ -match "^[^#]" -and $ItemMax -gt 0 ){

        #decrement count to limit the amount of objects in final text file
        $ItemMax = $ItemMax - 1

        #increase counter to count number of items on webpage
        $Count = $Count +1

        ## remove trailing text
        Foreach-Object {$_ -replace " # .*\\b", ""} |

        Sort-Object | out-file $FilePath -append } # End of the if statement
```



SECURESET.COM

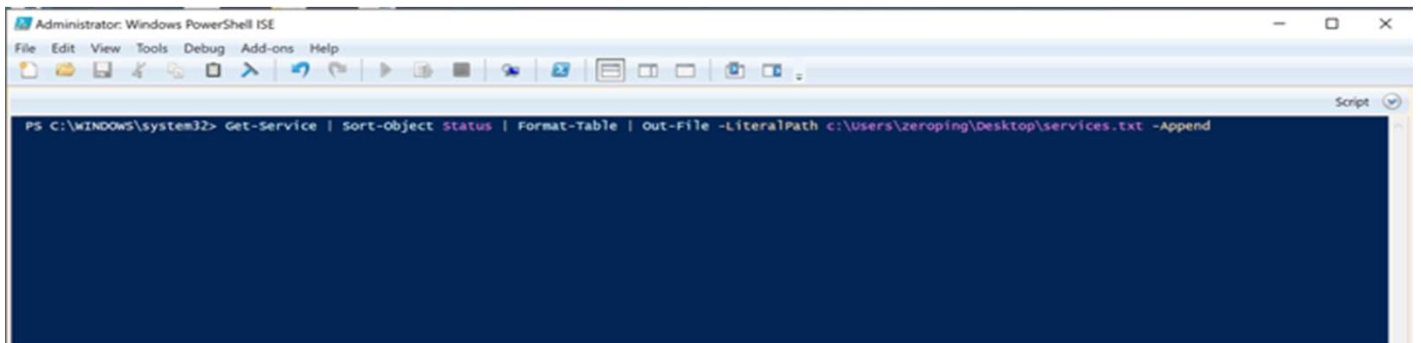
©2018 SecureSet Academy, Inc. | All Rights Reserved

PowerShell can be used manipulating files to suit your needs.

PowerShell can be used to conduct what is known as **Regular Expression** or **REGEX**.

A regular expression is a special text string for describing a search pattern. You can think of regular expressions as wildcards on steroids. You are probably familiar with wildcard notations such as \*.txt to find all text files in a file manager.

# The PIPELINE

A screenshot of the Windows PowerShell ISE (Integrated Scripting Environment) window. The title bar reads "Administrator: Windows PowerShell ISE". The menu bar includes "File", "Edit", "View", "Tools", "Debug", "Add-ons", and "Help". The command bar shows the command: `PS C:\WINDOWS\system32> Get-Service | Sort-Object Status | Format-Table | Out-File -LiteralPath c:\users\zeroping\Desktop\services.txt -Append`. The main console area is dark blue and currently empty, indicating the command has not yet been executed.

```
Administrator: Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
PS C:\WINDOWS\system32> Get-Service | Sort-Object Status | Format-Table | Out-File -LiteralPath c:\users\zeroping\Desktop\services.txt -Append
```

SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

In **PowerShell**, piping refers to the process of passing the results of one cmdlet as an argument into a second cmdlet.

In Unix/Linux systems a pipe is a form of redirection that is used to send the output of one command to another (command/program/process) for further processing.

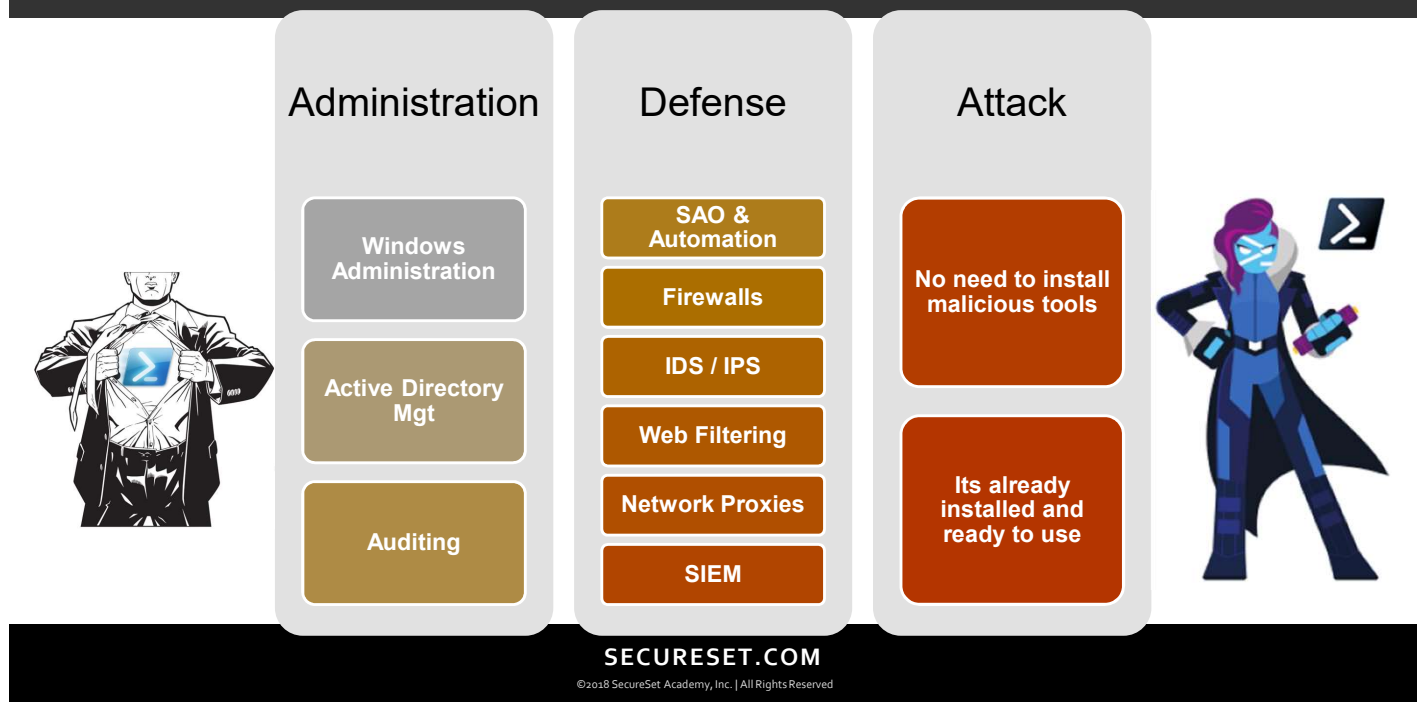
It's the same thing within Windows PowerShell, this pipeline enables us to create compound cmdlet sequences that perform multiple tasks in a single operation.

In the example, we “chain” together three PowerShell cmdlets by using the pipe (|) character. The net result in this case is that we receive a nifty, formatted list of our Windows services, organized by their run status.

We could add another pipe and use output the file to a location... This is really useful when gathering information and putting it into a .txt or .csv format..



## Sometimes what can help can hurt



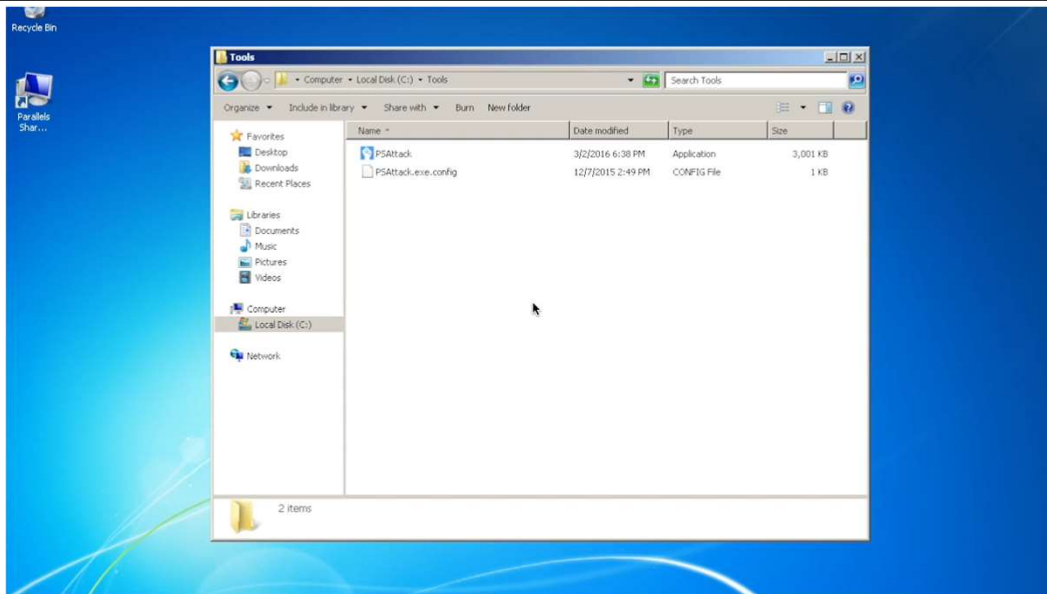
**PowerShell** has been used heavily for cyber attacks, especially recently during the **Petya/NotPetya** campaigns.

The most important aspect for attackers is its native integration with the .NET Framework, which offers multiple options for infecting or manipulating the target.

**PowerShell's** most attractive attributes to adversaries are:

- Simple access to network sockets
- Ability to assemble malicious binaries dynamically in memory
- Direct access to the Win32 Application Programming Interface (API)
- Simple interface with Windows Management Instrumentation (WMI)
- Powerful scripting environment
- Dynamic, runtime method calls
- Easy access to crypto libraries, e.g. IPsec, hashing algorithms
- Ability to hook managed code
- Simple bindings to Component Object Model (COM)  
(<https://msdn.microsoft.com/en-us/library/windows/desktop/ms694363%28v=vs.85%29.aspx>)
- All the above render PowerShell an extremely effective attack vector.

## Sometimes what can help can hurt



SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

Here is a image of the **PSAttack** that is being used to escalate privileges and dump the Active Directory database from a domain.

It demonstrates how combining the "**get-attack**" command with **PowerShell's** help system allows us to figure how what tools to use and how.

The demo was used during **CarolinaCon 12** where **PSAttack** was introduced.

Sometimes what can help can hurt



SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

**Cybersecurity is a game of finesse.**

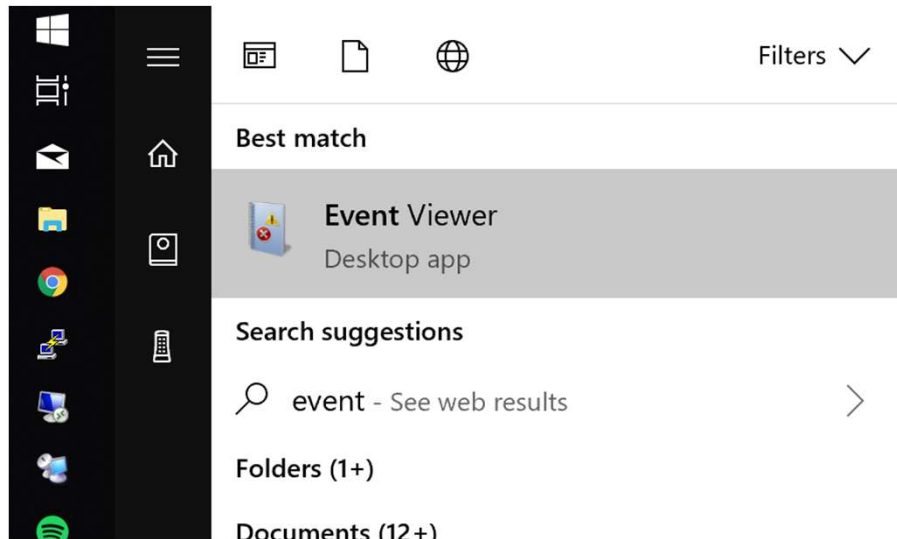
So is PowerShell, is especially so. Organizations need to do Active Directory right. The consequences of doing it wrong are dire.

Lets talk about the things one needs to do:

- **PSLockDownPolicy ? NEED MORE INFO**
- **PowerShell V.5 With Applocker And Device Guard**
- **PowerShell v.5** comes with significant embedded security features that make its use more secure for enterprise environments.
- **Script block logging.** Script block logging provides the ability to log de-obfuscated PowerShell code to the event log.
- **System-wide transcripts.** System-wide transcription can be enabled via Group Policy and provides an “over the shoulder” transcript file of every PowerShell command and code block executed on a system by every user on that system.

- **Constrained PowerShell.** Constrained Language mode. [Constrained language mode](#) limits the capability of PowerShell to base functionality, removing advanced feature support, such as .NET and Windows API calls and COM access. This lack of advanced functionality stops most PowerShell attack tools, because they rely on these methods. However, in enterprise environments it can negatively affect legitimate scripts; thus it is highly recommended to schedule a testing period before activating this option, to filter out the legitimately used code.
- **Antimalware integration (Windows 10).** The new [Windows 10 Antimalware Scan Interface](#) (AMSI) enables all the scripting engines (PowerShell, VBScript, and JScript) to request analysis of dynamic content: from a script file, typed commands at the command line, and even code downloaded and executed in memory. This enables scanning of PowerShell code before it is executed on the computer.
- **PowerShell logging** can be enabled via Group Policy for PowerShell modules. If you are not logging and auditing those logs you are basically blind.
- **Remove older versions of PowerShell** and ensure your systems are patched and up to date.
- **Just Enough Administration – JEA** - This is included with the latest update of Windows Management Framework 5.0 and 5.1, and is a security technology that helps organizations enforce information security by restricting IT administrative rights. **JEA** provides a practical, role-based approach to set up and automate restrictions for IT personnel, and reduces the risks associated with providing users with full administrative rights following the principle of least privilege.
- Scripts Code Signing
- If **PowerShell scripts** are used in an enterprise environment, **code signing** is another control that improves security posture, by **ensuring authenticity and integrity**. This feature, along with a defined **Execution Policy** or **Group Policy** as “AllSigned” or “RemoteSigned”, **will permit only digitally signed scripts to run**. However we have to consider that several attacks in the past used malicious files digitally signed so this control just adds another security layer since it can be bypassed.

## Before we start the lab



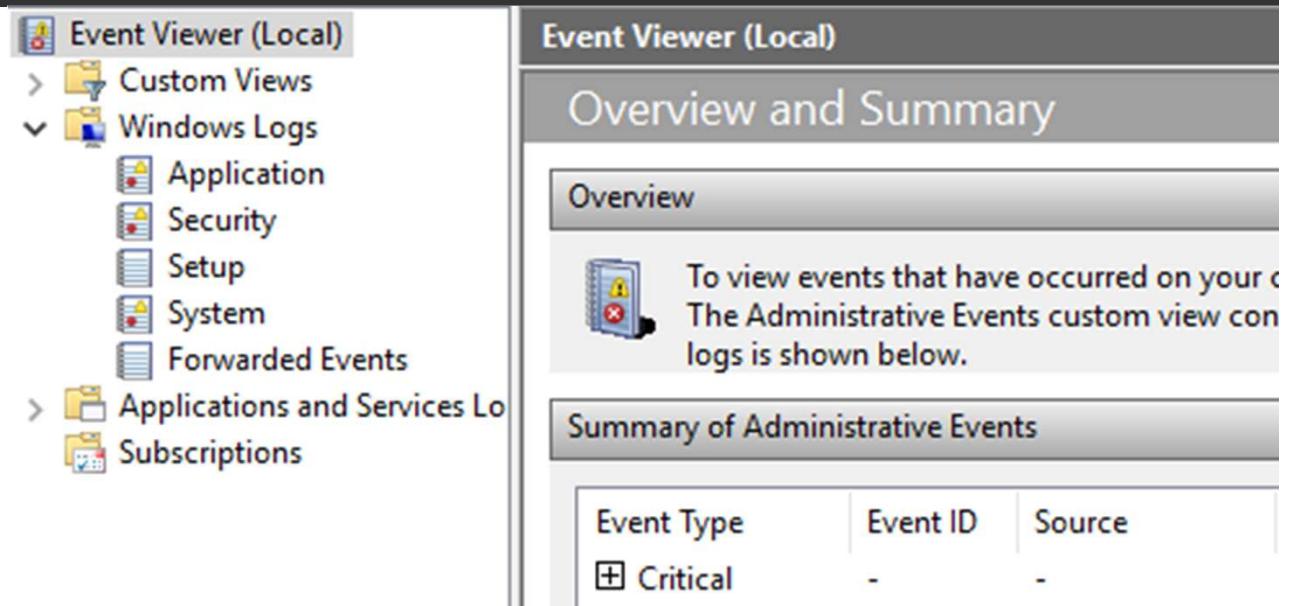
**SECURESET.COM**

©2018 SecureSet Academy, Inc. | All Rights Reserved

Before we start the lab, I want to point out that every windows computer has logs, there are 3 default logs in Windows. (More on that in a minute)

But to get to those logs, you just click on start, type in event and you will find the EVENT VIEWER

## Before we start the lab



SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

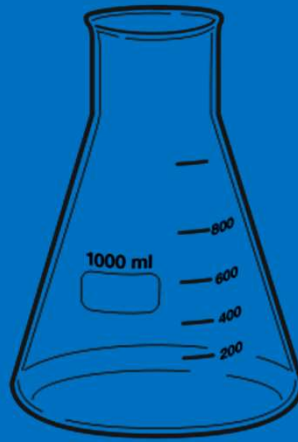
Once you enter the EVENT VIEWER application, and you expand the “Windows Logs” you will see the 3 default LOGS for Windows

- Application
- Security
- System

There actually a ton more depending if it's a server and if you have other applications such as DHCP or DNS installed, but we are just focusing on the basics today. =)

In the lab we will be collecting them via PowerShell. Automation makes your life much easier and better.

{LAB}  
Time



SECURESET.COM

©2018 SecureSet Academy, Inc. | All Rights Reserved

OK, LETS START THE LAB!



## Thanks for coming

- We provide career-launching security education
- Go from security novice to professional in 12 or 20 weeks
- The world is looking for 1,000,000 security pros
- Wondering if you're a fit? Try your hand at "Challenges for the Cyber-Curious"
  - Available at: <http://bit.ly/CoreReady>

**SECURESET.COM**

©2018 SecureSet Academy, Inc. | All Rights Reserved



