

# PowerShell Functions for Security Analysis

---

PowerShell Uses for Security Analysis



**Liam Cleary**

Microsoft MVP and Microsoft Certified Trainer

@helloitsliam [www.helloitsliam.com](http://www.helloitsliam.com)



# Overview



## Goal: Understand Why PowerShell Is a Core Security Tool

- Why should you use PowerShell for security tasks?
- What security tasks can PowerShell handle?
- Using PowerShell for network analysis, incident response, forensics, and malware analysis

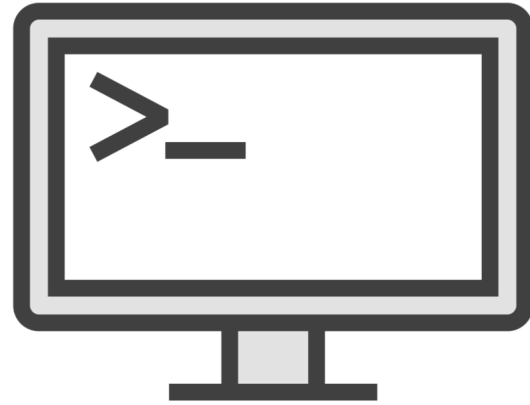


# Why Should You Use PowerShell for Security Tasks?

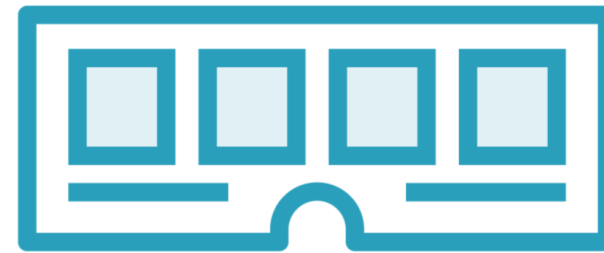
---



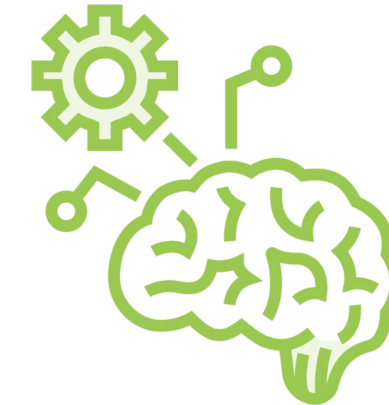
# Why Attackers Choose PowerShell?



**Default**



**Memory**



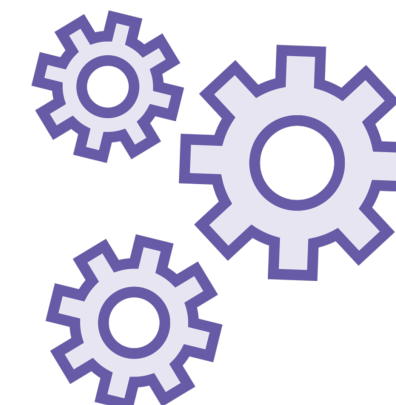
**Analysis**



**Remote**



**Whitelisting**



**Common**



# Online Repositories

**Many locations online with readily available scripts. Multiple repositories with sample code, including malicious. Multiple security testing and hacking frameworks are available to download for PowerShell.**



# Download Files from the Internet



**Using out-of-the-box commands, you can stream files from the internet**

- Invoke-WebRequest
- Invoke-RestMethod
- Start-BitsTransfer
- .NET Library (System.Net.WebClient)



# Downloading Files

```
$file = "file.zip"
$url = http://files.domain.com/\$file
$destination = "C:\Temp\Files"

# Download using "Invoke-WebRequest"
Invoke-WebRequest -Uri $url -OutFile "$destination\$file"

# Download using "Start-BitsTransfer"
Start-BitsTransfer -Source $url -OutFile $destination

# Download using "Invoke-RestMethod"
Invoke-RestMethod -Uri $url -OutFile "$destination\$file"
```



# Downloading Files

```
$file = "file.zip"
$url = http://files.domain.com/\$file
$destination = "C:\Temp\Files"

# Download using ".NET Class"
$client = [System.Net.WebClient]::new()
$client.DownloadFile($url, "$destination\$file")
```





# Why Use PowerShell for Security Tasks?

1

## **Built-in Command Line Tool**

No need to even check if a command line tool exists

2

## **Runs in Memory not on Disk**

PowerShell executes within memory, allowing potential bypass of anti-virus providers

3

## **Included with every version of Windows**

If the target machine is windows, it will contain a version of PowerShell

4

## **Ability to leverage .NET Libraries**

Existing .NET libraries can be imported and utilized within scripts

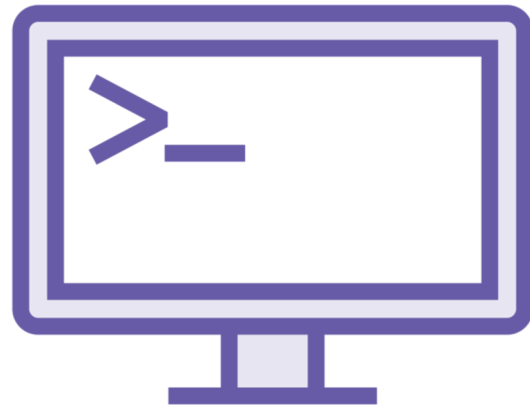
5

## **Built-in Commands for Remote Execution**

Native commands exist to allow remote execution across windows clients and servers, as well as other devices that support PowerShell



# Why Use PowerShell for Security Tasks?



**Command Line**



**Downloader**



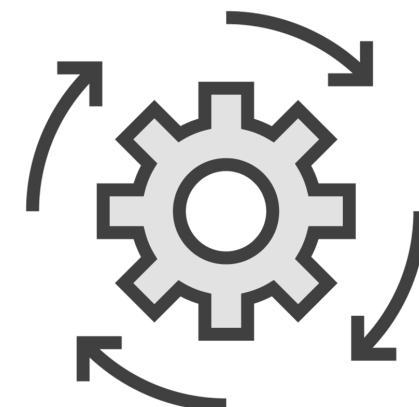
**Monitoring**



**Parameters**



**Hidden**



**Operating System**

# What Security Tasks Can PowerShell Handle?

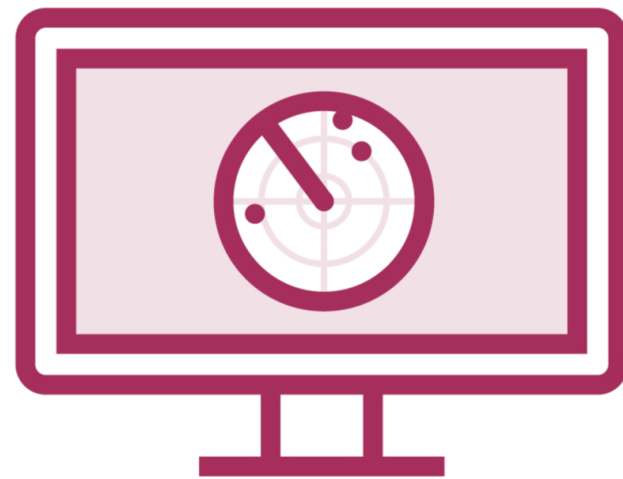
---



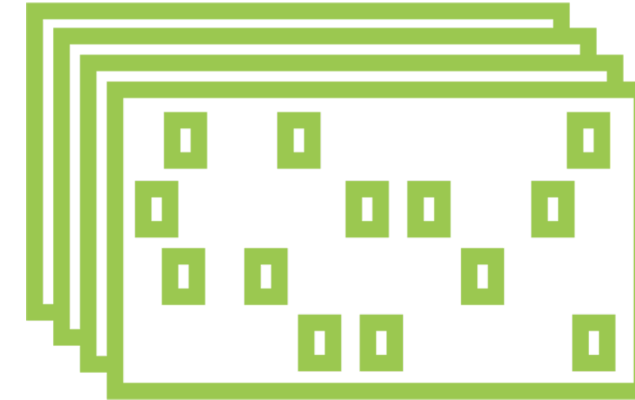
# What Security Tasks Can PowerShell Handle?



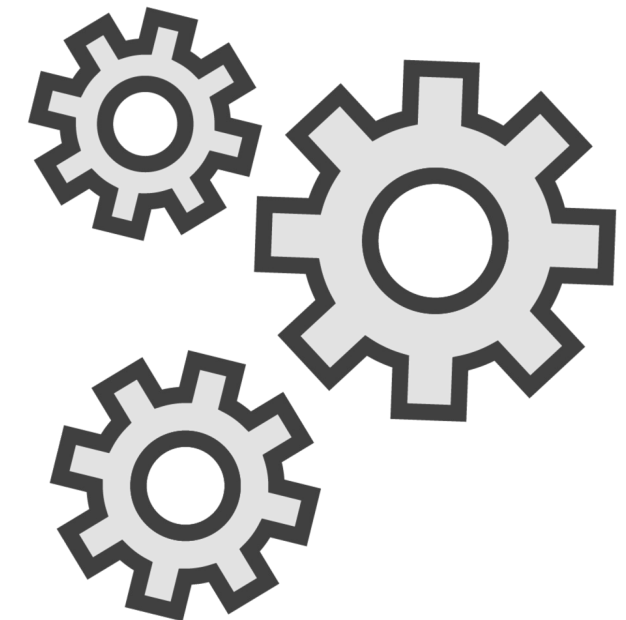
**Log and File  
Analysis**



**Information  
Gathering  
(Operating System  
and Network)**

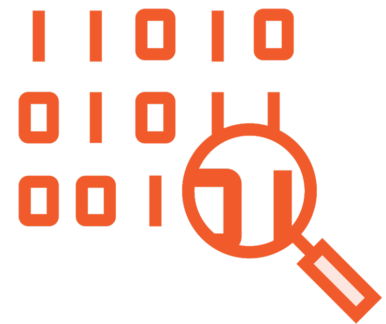


**Task Automation  
and Scheduling**



**Security  
Configuration and  
Management**

# Log and File Analysis



**Parse event logs from Windows operating systems**



**Retrieve file contents and associated metadata**



**Perform advanced queries within files, logs, and other types of contents**



# Information Gathering



**Computer name**



**IP address**



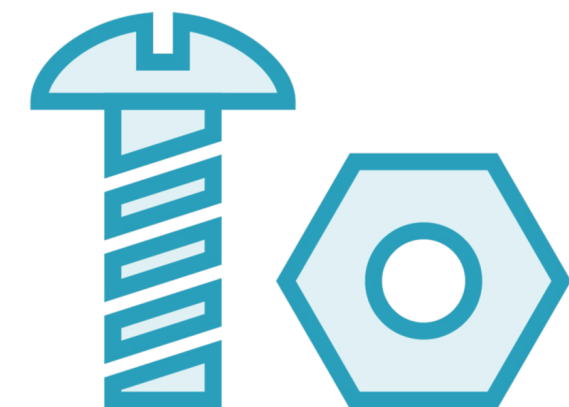
**Operating system**



**Domain membership**

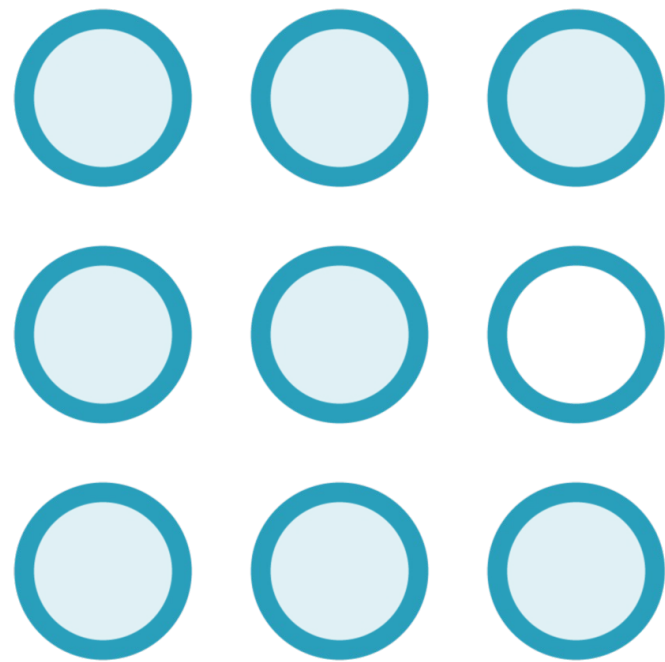


**Physical location**



**Hardware type**

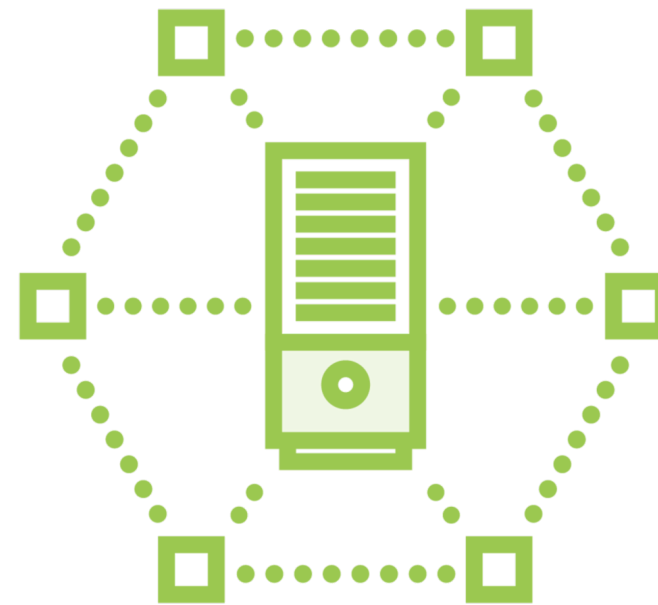
# Task Automation



**Single tasks**



**Multiple tasks**

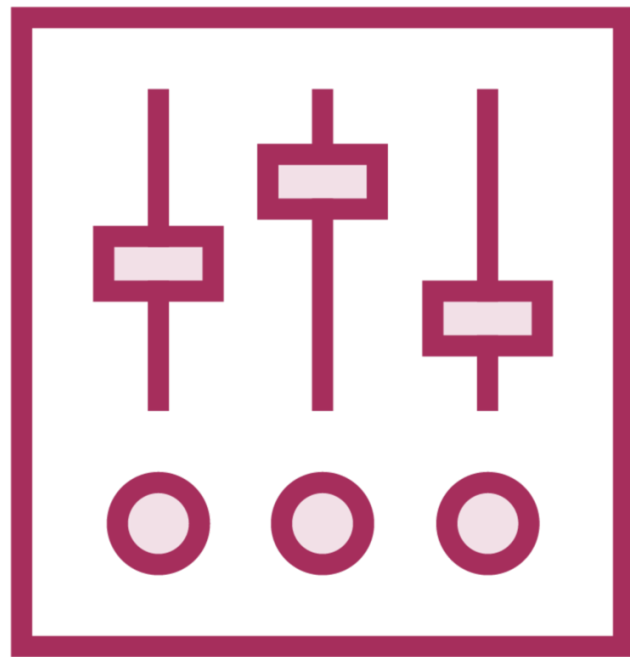


**Remote tasks**

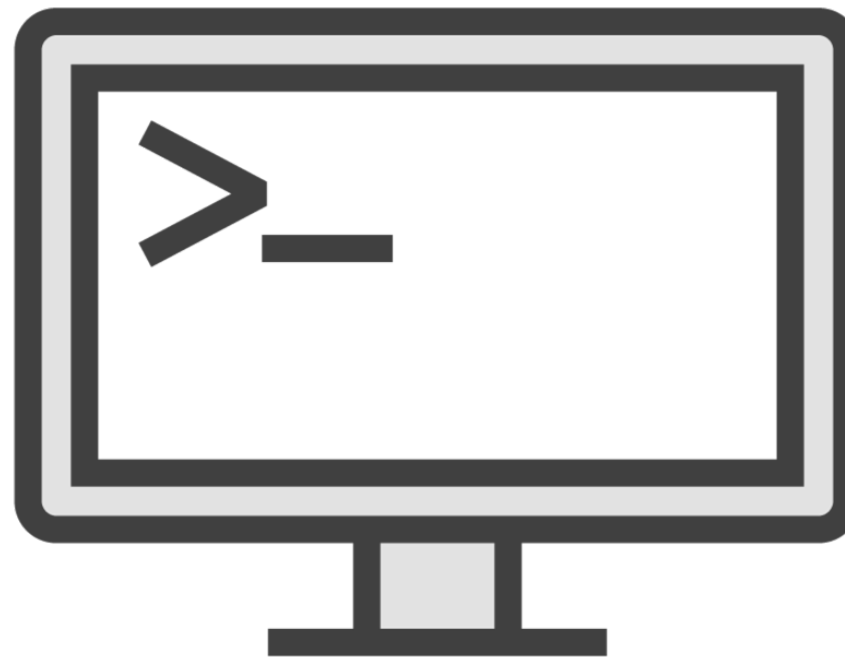


**Scheduled tasks**

# Security Configuration



**Multiple Security Frameworks are available that provide PowerShell configuration options**



**You can use direct PowerShell commands to implement security best practices**



**PowerShell DSC can be used to harden Windows operating systems**



# Using PowerShell for Security Analysis

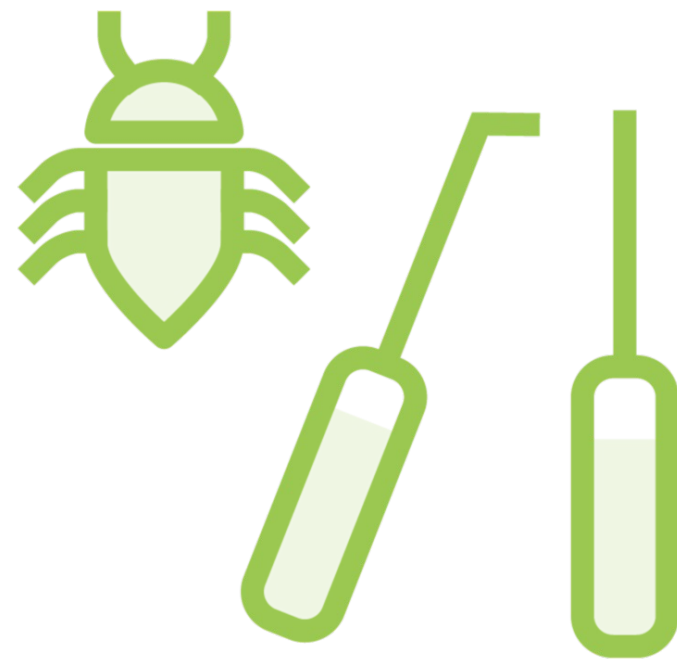
---



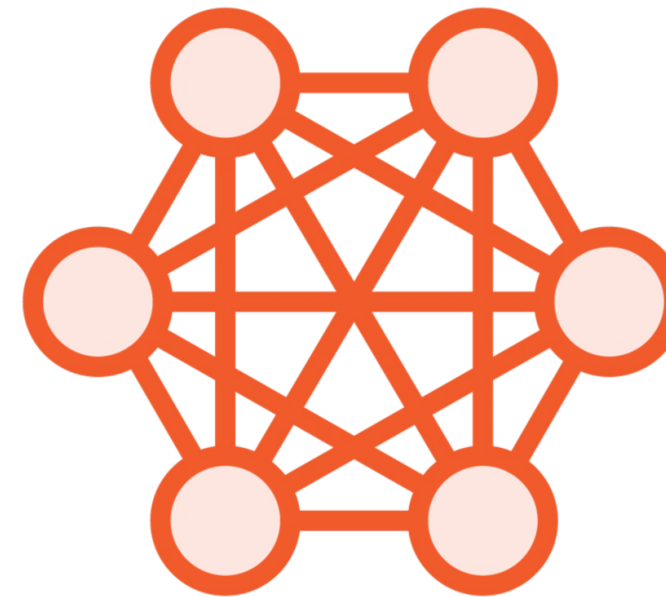
# Using PowerShell for Security Analysis



**Log Analysis**



**Malware  
Inspection**



**Network  
Analysis**



**Forensics**

# Example: Retrieving Log Entries

```
# Retrieve Log Entries using the Event ID 102 from Saved Event Log
$Id = "102"
$events = Get-WinEvent -FilterHashtable @{
    Path = "C:\Users\Administrator\Downloads\pwsh.evtx";
    Id = $Id
}
$events | Select ID, Message
```



# Example: Capture Network Traffic

## # Capture Network Traffic

```
$cim = New-CimSession -ComputerName "WIN10"
```

```
New-NetEventSession `
    -Name "Net-Session-001" `
    -CimSession $cim `
    -LocalFilePath "C:\Temp\Files\Traffic.etl" `
    -CaptureMode SaveToFile
```

```
Add-NetEventProvider `
    -CimSession $cim `
    -Name 'Microsoft-Windows-SMBClient' `
    -SessionName "Net-Session-001"
```

```
Start-NetEventSession -Name "Net-Session-001" -CimSession $cim
```



# Summary



## **Goal: Understand Why PowerShell Is a Core Security Tool**

- Why you can use PowerShell for Security analysis
- Tasks you can perform using PowerShell for Security analysis
- Discussed PowerShell commands



Up Next:

Installing and Remotely Connecting Using  
PowerShell

---

