

Enumerating Services and Processes



Liam Cleary

Microsoft MVP and Microsoft Certified Trainer at SharePlicity

@helloitsliam | www.helloitsliam.com



Overview

Goal: Use PowerShell for Enumeration

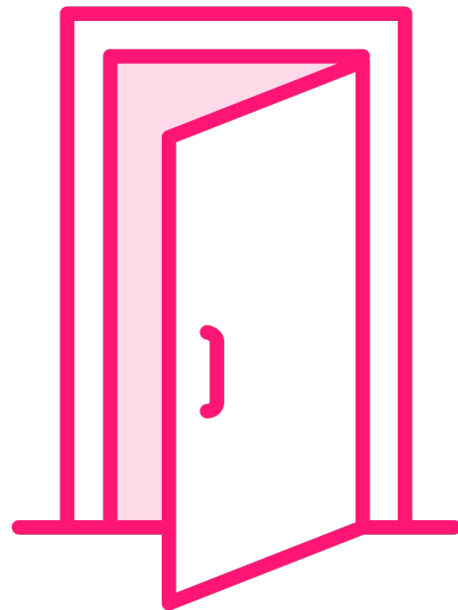
- Identify open ports on individual and multiple networked devices
- Identifying running processes and services on remote devices



Identify Open Ports on Individual and Multiple Networked Devices

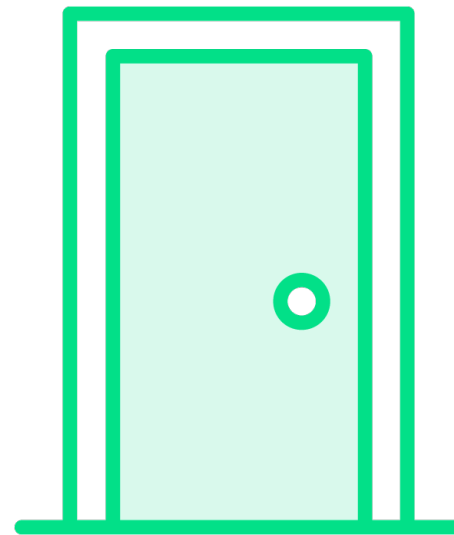


Port Status



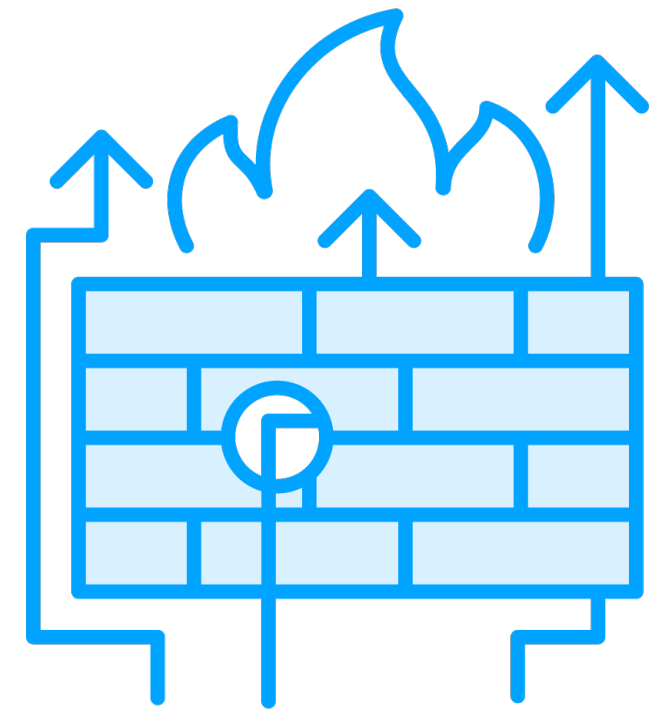
Open

Target machine is listening for connections and accepting packets to the port



Closed

Rejects connections and ignores all packets to the port



Filtered

A firewall, filter, or other network device is blocking the port



Port Scanning Tools

1

.NET API Namespace

2

Test-Connection Cmdlet

3

3rd Party PowerShell Modules

4

Nmap Tools



.NET API Namespace

Provides client connections for TCP network services using the "System.Net.Sockets.TcpClient" class

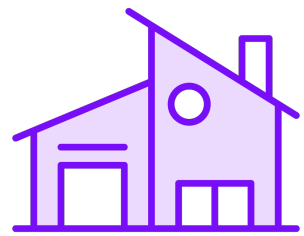


.NET API Namespace Constructors



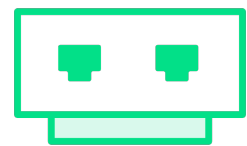
TcpClient(AddressFamily)

Initializes a new instance of the TcpClient class with the specified family such as "InterNetwork" or "InterNetworkIPv6"



TcpClient(IPEndPoint)

Initializes a new instance of the TcpClient class and binds it to the specified local endpoint



TcpClient(String, Int32)

Initializes a new instance of the TcpClient class and connects to the specified port on the specified host

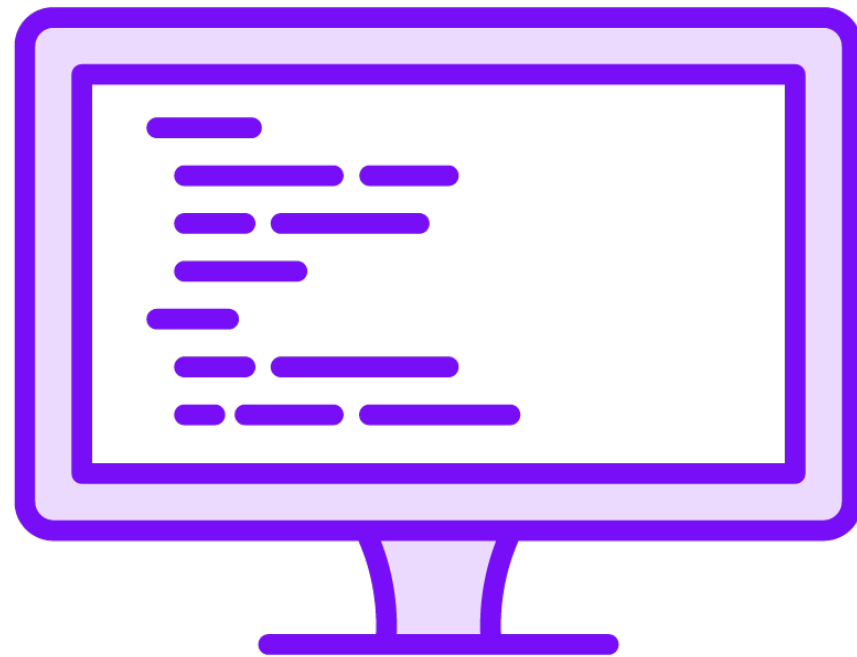


Use the .NET API Namespace to Port Scan

```
# Connect to Device and Check Port 80
$ip = "192.168.1.72"
$port = "80"
$client = New-Object System.Net.Sockets.TCPClient
$client.Connect($ip, $port)
```



Test-Connection Cmdlet



Sends ICMP echo request packets, or pings, to one or more computers

Tests specific ports for one or more computers



Port Scan using Test-Connection Cmdlet

Connect to Device and Check Port 80

```
$ip = "192.168.1.72"
```

```
$port = "80"
```

```
Test-Connection -TargetName $ip -TcpPort $port
```

Check Multiple Ports on Single Device

```
$ip = "192.168.1.72"
```

```
1..1024 | ForEach-Object { Test-Connection -TargetName $ip -TcpPort $_ }
```



Using Nmap to Scan Ports

1

Use comma separated list of ports e.g., "80, 443, 3389"

2

Use a port range e.g., "22-389"

3

Set the port protocol to either TCP or UDP e.g., "T:25, U:53"



Use Nmap with PowerShell for Port Scanning

Use a Comma Separated Port List

```
$ip = "192.168.1.72"
```

```
nmap -p80,443 $ip
```

Use a Port range

```
$ip = "192.168.1.72"
```

```
nmap -p80-100 $ip
```

Set the Port Protocol to Scan

```
$ip = "192.168.1.72"
```

```
nmap -pT:25,U:53 $ip
```



Example 3rd Party PowerShell Modules



PSnmap

Asynchronous Linux Nmap tool for PowerShell. Performs ping sweeps, and scans networks for specified open ports



PortScan

Simple PowerShell script for performing external port scans



PowerSploit

Collection of PowerShell modules to aid penetration testers during all phases of an assessment



Use a 3rd Party PowerShell Module

Scan Multiple Ports on a Single Device

```
$ip = "192.168.1.72"
```

```
Invoke-Portscan -hosts $ip -ports "80, 443, 3389"
```

Scan Multiple Ports on Multiple Devices

```
$ips = "192.168.1.72, 192.168.1.73, 192.168.1.74"
```

```
Invoke-Portscan -hosts $ips -ports "80, 443, 3389"
```



Demo

Check for Open Ports on Devices

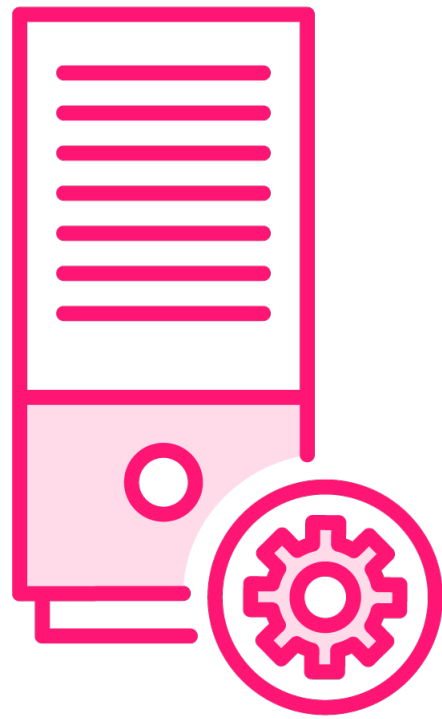
- Create a Port "Echo" Server
- Use the .NET API to check for an open port
- Use Nmap to check for an open port
- Use a 3rd Party Module to check for an open port



Identifying Running Processes and Services on Devices

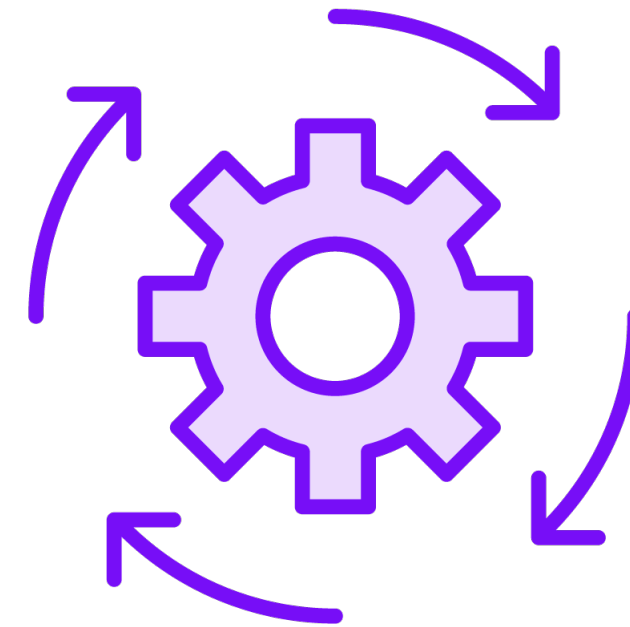


Services Versus Processes



Services

Operating Systems provide services to both end-users and programs.
Services perform program execution



Processes

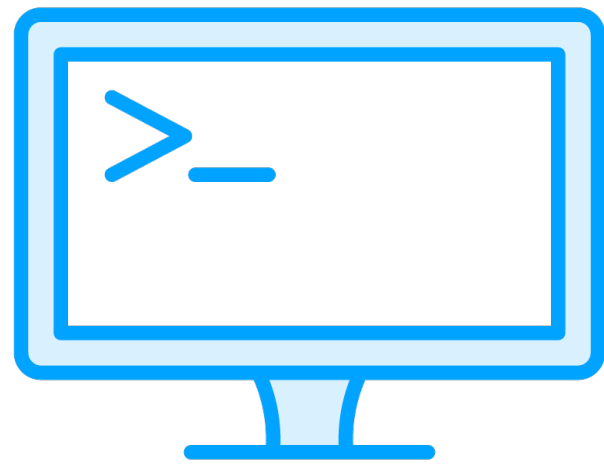
A process is a running program. It is code running in the "execution phase"

Prerequisites

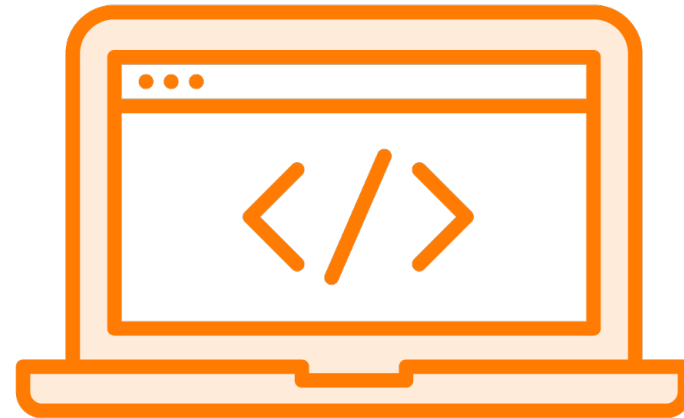
- 1** Know the credentials of the remote device
- 2** Enabled PowerShell remote management
- 3** Know the IP address or name of the device



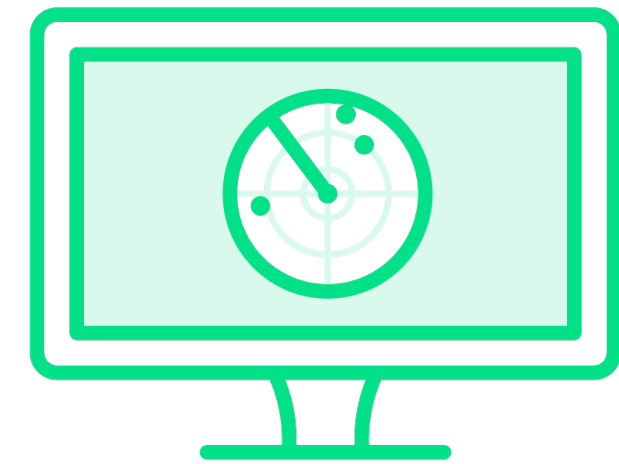
Identifying Services



**Get-Service
(Local) Cmdlet**



**Get-Service
(Remote) Cmdlet**



Nmap



Identifying Services

Get Running Services

```
Get-Service
```

```
Get-Service | Where-Object {$_.Status -eq "Running"}
```

Get Running Services on Remote Windows Server

```
Get-Service -ComputerName WIN10
```

```
Get-Service -ComputerName WIN10 | Where-Object {$_.Status -eq "Running"}
```

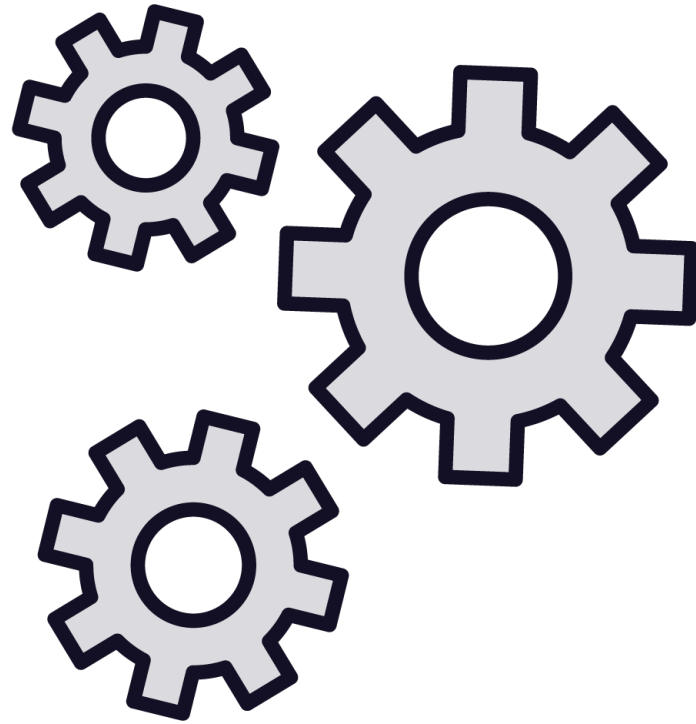
Scan Windows Server for Running Services using Nmap

```
$ip = "192.168.1.72"
```

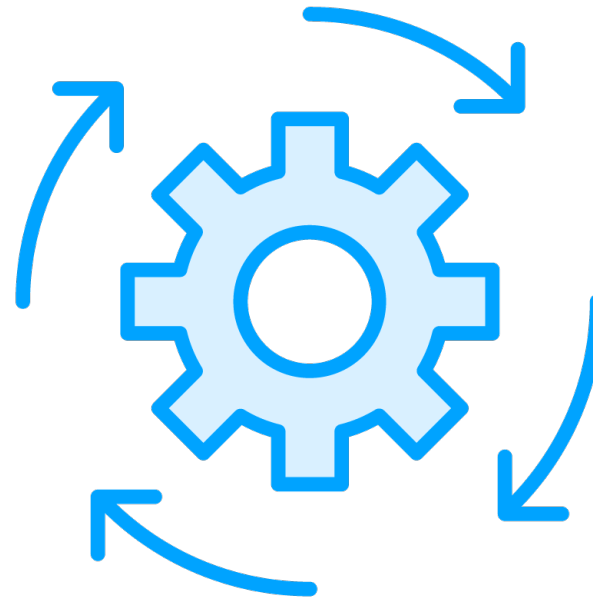
```
$results = nmap -sV $ip --version-all
```



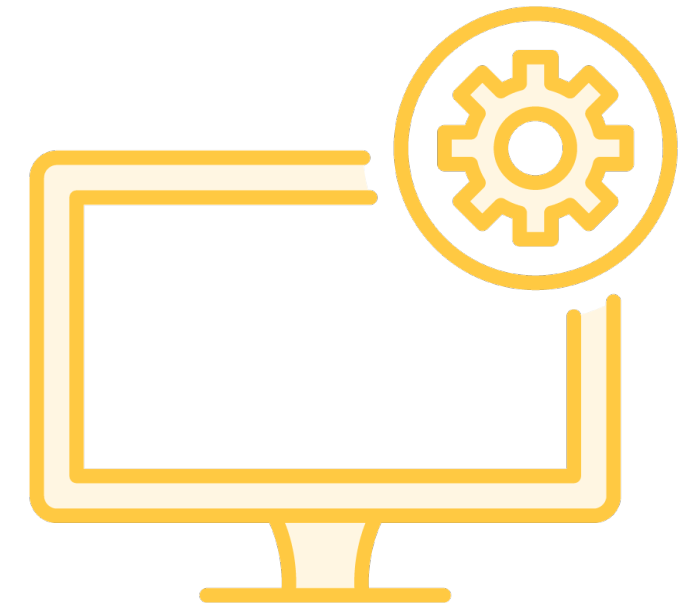
Identifying Processes



Get-Process



Get-WmiObject



Get-CimInstance



Identifying Processes

List Processes using Get-Process

```
Get-Process
```

```
Get-Process -ComputerName WIN10
```

List Processes using Get-WmiObject

```
Get-WmiObject Win32_Process
```

```
Get-WmiObject Win32_Process -ComputerName WIN10
```

List Processes using Win32_Process

```
Get-CimInstance Win32_Process
```

```
Get-CimInstance Win32_Process -ComputerName WIN10
```



Demo

Check for Running Services and Processes

- Check for running services
- Use Nmap to identify services for common ports
- Check for running processes



Summary

Goal: Use PowerShell for Enumeration

- Used port, service, and process scanning tools to test network devices
- Interrogated individual and multiple networked devices for open ports, services, and processes



Up Next:

Using the Common Information Model (CIM) Cmdlets to Inspect the Windows Operating System

