

Continuous Monitoring with PowerShell

Performing a Network Discovery



Liam Cleary

Microsoft MVP and Microsoft Certified Trainer at SharePlicity

@helloitsliam | www.helloitsliam.com



Overview

Goal: Use PowerShell for Network Discovery

- Pinging individual networked devices
- Pinging multiple networked devices
- Identifying network devices
- Creating an asset list of networked devices



Pinging Individual Networked Devices



Ping Capabilities

1

PING, PATHPING, and PSPING Tools

2

.NET API "System.Net.NetworkInformation" Namespace

3

WMI Class "Win32_PingStatus"

4

Test-NetConnection and Test-Connection PowerShell Cmdlets



Using PING and PATHPING

Ping IP Address / Ping with 10 Requests

```
ping 192.168.1.72
```

```
ping /n 10 192.168.1.72
```

Ping IP Address / Ping with 10 Requests

```
pathping /n 192.168.1.72
```

```
pathping /q 10 /n 10 192.168.1.72
```

Using Ping and PathPing with PowerShell

```
$ip = "192.168.1.73"
```

```
echo (ping $ip)
```

```
echo (pathping $ip)
```



Using PSPING

Ping IP Address 10 Times

```
psping -n 10 192.168.1.73
```

Ping IP Address 10 Times, Then Test Port Connection

```
psping -n 10 -i 0 -q 192.168.1.73:80
```

Using PsPing with PowerShell

```
$ip = "192.168.1.73"
```

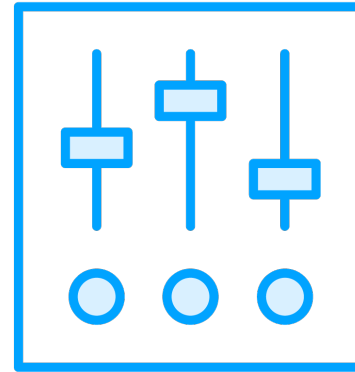
```
echo (.\psping $ip)
```



Using the .NET API Namespace



IP Status



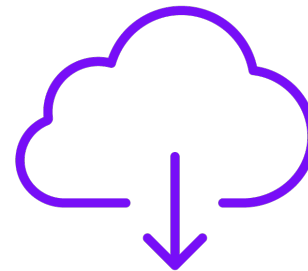
Ping Options



Ping Reply



Send



Send Async



Send Ping Async

Ping Using the .NET API

Ping Computer By Name

```
$computer = "TRAINER"  
$ping = New-Object System.Net.NetworkInformation.Ping  
$ping.Send($computer)
```

Ping Computer By IP Address

```
$ip = "192.168.1.73"  
$ping = New-Object System.Net.NetworkInformation.Ping  
$ping.Send($ip)
```



Using the .NET API with Ping Options

```
# Set Ping Options and then Ping Computer By IP Address
```

```
$ip = "192.168.1.73"
```

```
$bytes = 64
```

```
$ttl = 57
```

```
$timeout = 120
```

```
[byte[]]$buffer = [byte[]]::new($bytes);
```

```
$options = New-Object System.Net.NetworkInformation.PingOptions $ttl
```

```
$pinger = [System.Net.NetworkInformation.Ping]::new()
```

```
$pinger.send($ip, $timeout, $buffer, $options)
```

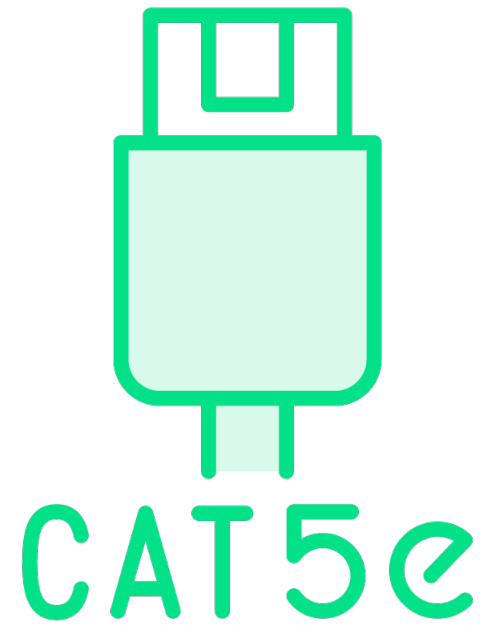


Win32_PingStatus

The Win32_PingStatus WMI class represents the values returned by the standard ping command

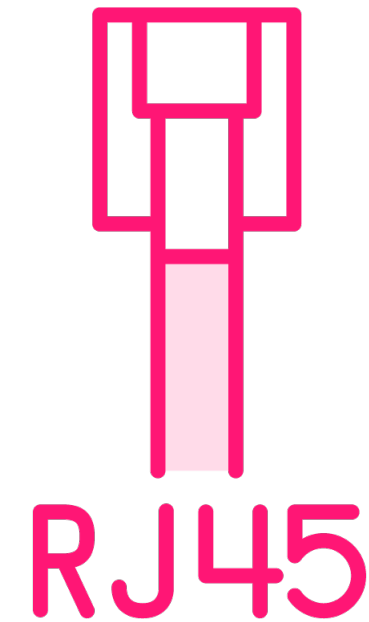


PowerShell Cmdlets



Test-Connection

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



Test-NetConnection

Supports ping test, TCP test, route tracing, and route selection diagnostics



WMI and PowerShell Cmdlets

Ping Using WMI

```
$ip = "Address='192.168.1.73' "`  
Get-WmiObject Win32_PingStatus `"  
    -filter $ip  
$ping = Get-WmiObject Win32_PingStatus `"  
    -filter $ip  
$ping.StatusCode
```

Ping Using PowerShell

```
$ip = "192.168.1.73"  
Test-Connection $ip  
Test-NetConnection $ip
```



Pinging Multiple Networked Devices



Ping Multiple Devices Using PING Tools

Ping Multiple IP Address

```
$ips = @("192.168.1.71", "192.168.1.72", "192.168.1.73")  
$ips | ForEach-Object { ping $_ }  
$ips | ForEach-Object { pathping $_ }  
$ips | ForEach-Object { .\psping $_ }
```

Ping IP Range

```
1..254 | ForEach-Object { ping "192.168.1.$_" }  
1..254 | ForEach-Object { pathping "192.168.1.$_" }  
1..254 | ForEach-Object { .\psping "192.168.1.$_" }
```



Ping Multiple Devices Using the .NET API

Ping Range of IP Addresses

```
$ping = New-Object System.Net.NetworkInformation.Ping;  
1..254 | ForEach-Object { $ping.Send("192.168.1.$_") }
```

Ping Range of IP Addresses, Then Add Ping Completed Event

```
(  
    (1..254) | ForEach-Object {  
        $ping = New-Object System.Net.NetworkInformation.Ping;  
        [Void](Register-ObjectEvent $ping PingCompleted -Action {  
            param($s, $e);  
            Write-Host $e.Reply.Address, ($e.Reply.RoundtripTime.ToString() + "ms")  
        })  
        $ping.SendPingAsync("192.168.1.$_")  
    }  
) .Wait()
```



Demo

Ping Individual and Multiple Networked Devices

- Help to Identify Live Network Hosts
- Ping Specific IP Addresses
- Ping IP Address Ranges
- Create Reusable Ping Function





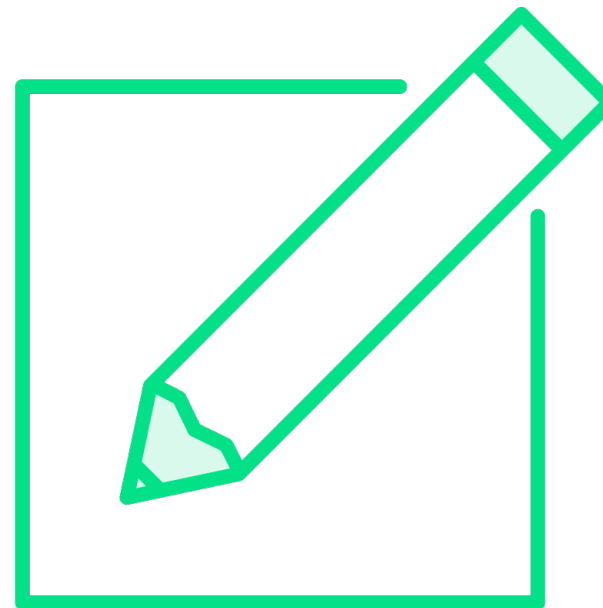
Identifying Network Devices



Identifying Networked Devices



IP Address



Name Resolution



Vendor Identification

Performing Name Resolution

Resolve IP Address to Name Using "System.NET" Library

```
$ip = "192.168.1.73"
```

```
[System.Net.DNS]::GetHostEntry($ip)
```

Resolve Using PowerShell Command

```
Resolve-DNSname $ip
```



Identifying Device Vendor

```
# Use Nmap and PowerShell to Get MAC Address
```

```
$ip = "192.168.1.73"
```

```
$mac = nmap -sn $ip | Select-String "MAC Address"
```

```
Write-Host $mac
```

```
# Identify Vendor from MAC Address
```

```
$mac = ($mac -replace "MAC Address:", "").Substring(1,17)
```

```
$vendor = (Invoke-WebRequest -Uri "https://api.macvendors.com/$mac").Content
```

```
Write-Host $vendor
```



Demo

Retrieve Further Details of Networked Devices

- Perform Domain Name Service Name Resolution
- Use an API to Lookup MAC Address to Identify Manufacturer





Creating an Asset List of Networked Devices



Asset List Creation Tasks



Query IP Range for Online Devices

Resolve Device Name

Retrieve Device MAC Address

Query Vendor Information using MAC Address

Generate CSV file of Retrieved Devices



Demo

Create an Asset List of Networked Devices



Summary

Goal: Use PowerShell for Network Discovery

- Utilized various approaches to pinging individual and multiple networked devices
- Retrieved further details from identified networked devices
- Created an asset list of networked devices



Up Next:

Enumerating Services and Processes

