

# Querying System Information and Logs Using PowerShell

---



**Liam Cleary**

Microsoft MVP and Microsoft Certified Trainer

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



# Overview



## **Goal: Use PowerShell to Query Common Operating System Information**

- Query running processes and services
- Create scripts to collect log data
- Query networking information
- Create a script to combine retrieved data into a readable document



# Querying Running Processes and Services

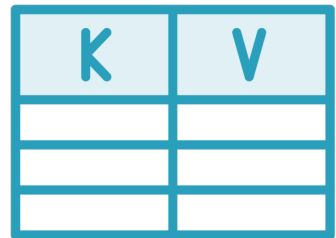
---



# Retrieving Processes



**Retrieve a list of running processes**



**Query and return specific process attributes**



**Retrieve process memory usage**



**Querying processes on remote computers**



# Retrieving Processes

**# Retrieve running processes**

```
Get-Process
```

```
Get-Process -Name 'CalculatorApp'
```

```
Get-Process -Name CalculatorApp, Notepad | Format-List *
```

**# Retrieve running process attributes**

```
(Get-Process -Name 'CalculatorApp').Path
```

```
(Get-Process -Name 'mate-calc').Path
```

```
Get-Process -Name 'CalculatorApp' -IncludeUserName
```

```
Get-Process -Name 'CalculatorApp' -FileVersionInfo
```

```
Get-Process -IncludeUserName | where { $_.Username -like "*System*" }
```

**# Retrieve process memory and CPU usage**

```
[math]::Round((Get-Process -Name 'CalculatorApp').CPU, 2)
```

```
[math]::Round((Get-Process -Name 'CalculatorApp').VM / 1GB)
```



# Querying Processes on Remote Computers

**# Retrieve running processes from a remote computer**

```
Get-Process -ComputerName 'REMOTE'
```

```
Get-Process -ComputerName 'REMOTE' -Name 'CalculatorApp'
```

**# Retrieve running process attributes from a remote computer**

```
(Get-Process -ComputerName 'REMOTE' -Name 'CalculatorApp').Path
```

```
(Get-Process -ComputerName 'REMOTE' -Name 'mate-calc').Path
```

```
Get-Process -ComputerName 'REMOTE' -Name 'CalculatorApp' -IncludeUserName
```

```
Get-Process -ComputerName 'REMOTE' -Name 'CalculatorApp' -FileVersionInfo
```

**# Retrieve process memory and CPU usage from a remote computer**

```
[math]::Round((Get-Process -ComputerName 'REMOTE' -Name 'CalculatorApp').CPU, 2)
```

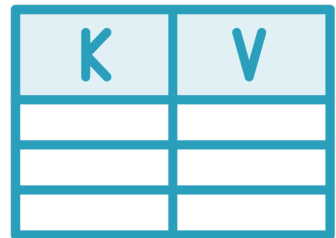
```
[math]::Round((Get-Process -ComputerName 'REMOTE' -Name 'CalculatorApp').VM / 1GB)
```



# Retrieving Services



**Retrieve a list of running services**



**Query and return specific service attributes**



**Start, stop, suspend, and restart services**



**Querying services on remote computers**



# Retrieving Processes

## # Retrieve running services

```
Get-Service
```

```
Get-Service -Name 'wscsvc'
```

```
Get-Service -Name wscsvc, bthserv | Format-List *
```

```
Get-Service -ComputerName 'REMOTE' -Name wscsvc, bthserv | Format-List *
```

## # Retrieve running service attributes

```
(Get-Service -Name 'wscsvc').DisplayName
```

```
(Get-Service -Name 'wscsvc').RequiredServices
```

```
Get-Service -Name * | Where-Object {$_.RequiredServices -or $_.DependentServices} |  
    Format-Table -Property Status, Name, RequiredServices, DependentServices -auto
```

## # Controlling services

```
Start-Service -Name 'wscsvc'
```

```
Stop-Service -Name 'wscsvc'
```

```
Restart-Service -Name 'wscsvc'
```





# Demo



**Retrieve running process from a local and remote computer**

**Retrieve running servicesd from a local and remote computer**



# Create Scripts To Collect Log Data

---



# Collect Windows Event Log Entries



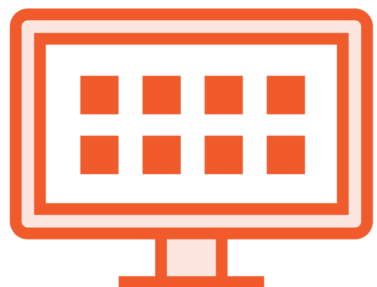
## **Get-EventLog (Microsoft.PowerShell.Management)**

Gets the events in an event log, or a list of the event logs, on the local computer or remote computers. Legacy command



## **Get-WinEvent (Microsoft.PowerShell.Diagnostics)**

Gets events from event logs and event tracing log files on local and remote computers



## **Event Viewer**

GUI interface for all event logs



# Collect Windows Event Log Entries

**# Retrieve error and warnings using Get-EventLog**

```
Get-EventLog system -After ((Get-Date).AddHours(-12)) |  
    Where {($_.EntryType -Match "Error") -or ($_.EntryType -Match "Warning")} |  
    Format-Table -Wrap
```

**# Retrieve error and warnings using Get-WinEvent**

```
Get-WinEvent -FilterHashTable @{  
    LogName = "Application", "System";  
    Level = 2,3;  
    StartTime = ((Get-Date).AddHours(-12))  
}
```

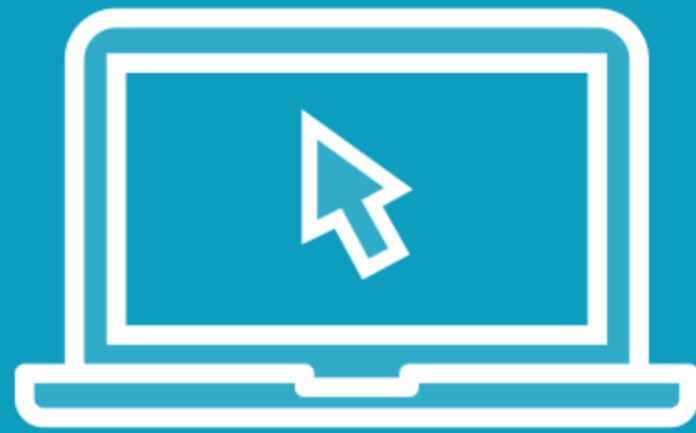


# Collect Windows Event Log Entries

```
# Combine error and warnings from multiple servers
$file = "C:\Temp\Servers.txt"
$servers = Get-Content -Path $file
foreach($server in $servers) {
    Get-WinEvent
        -ComputerName $server
        -MaxEvents 25
        -FilterHashtable @{
            LogName = "Application", "System";
            Level = 2,3;
        }
}
```



# Demo



**Retrieve and query event logs**

**Export the current event log**



# Querying Networking Information

---



# Query Network Components

1

**Viewing IP addresses of a computer**

2

**Retrieve detailed IP configuration information**

3

**Pinging computers**

4

**Retrieving network adapter properties**





# Query Computer Network Properties

**# Display specific network adapter statistics**

```
Get-NetAdapterStatistics -Name "Wi-Fi"
```

```
Get-NetAdapterStatistics -Name "Wi-Fi" | Format-List -Property "*"
```

**# Retrieve network adapter properties**

```
Get-NetAdapter -Name *
```

```
Get-NetAdapter -Name * -IncludeHidden
```

```
Get-NetAdapter -Name "Wi-Fi" | Format-List -Property *
```

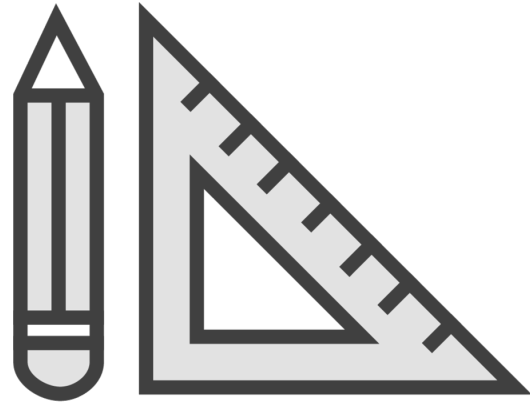
**# Retrieve network adapter properties from a remote server**

```
$session = New-CimSession -ComputerName 'REMOTE'
```

```
Get-NetAdapter -Name * -CimSession $session
```



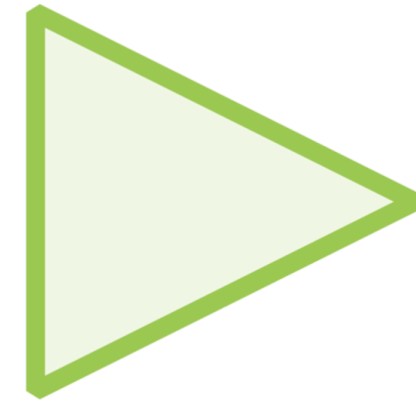
# Perform a Network Trace



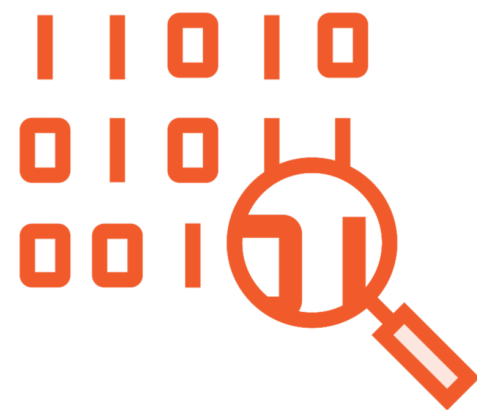
**Create network  
event session**



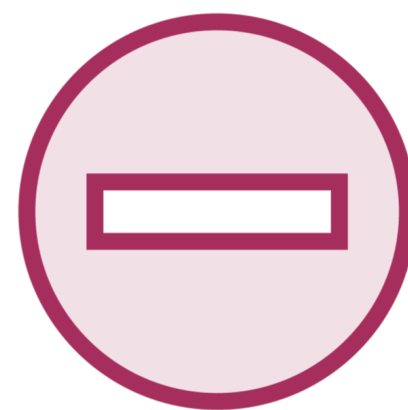
**Add event  
provider**



**Start the session**



**Retrieve session  
details**



**Stop the event  
session**



**Remove the  
created session**



# Perform a Network Trace

**# Create network session**

```
New-NetEventSession -Name "Session-001"
```

**# Add TCP/IP provider to session**

```
Add-NetEventProvider -Name "Microsoft-Windows-TCP/IP" -SessionName "Session-001"
```

**# Start the network session**

```
Start-NetEventSession -Name "Session-001"
```

**# Get the session details**

```
Get-NetEventSession
```

**# End and remove the session**

```
Stop-NetEventSession -Name "Session-001"
```

```
Remove-NetEventSession
```



# Perform a Network Trace

```
# Use netsh networking tracing
```

```
$log = "C:\Temp\Trace\"
```

```
$netsh = Start-Process "$($env:windir)\System32\netsh.exe" `
    -ArgumentList "trace start persistent=yes capture=yes tracefile=\"$log\Trace.etl" `
    -RedirectStandardOutput "$log\Trace.txt" `
    -Wait `
    -NoNewWindow `
    -PassThru
```



# Query a Network Trace

**# Retrieve network session entries**

```
$log = Get-WinEvent -Path "C:\AppData\Local\Trace.etl" -Oldest
```

**# Query log entries**

```
$log.Where({$_ .id -eq 1001})
```

```
$log.Where({$_ .id -eq 1001})[0].Message
```

```
$log.Where({$_ .id -eq 1001}).Message
```

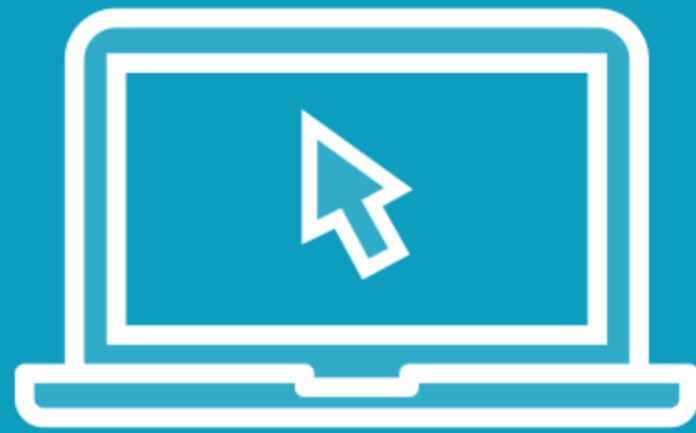
```
$log | Where-Object {$_ .ProviderName -eq 'Microsoft-Windows-TCPIP' }
```

**# Export to CSV file**

```
$log | Export-Csv "C:\AppData\Local\Trace.csv" -Delimiter ';' 
```



# Demo



**Query local computer networking  
information**

**Query and retrieve network traffic**



# Create a Script To Combine Retrieved Data Into a Readable Document Format

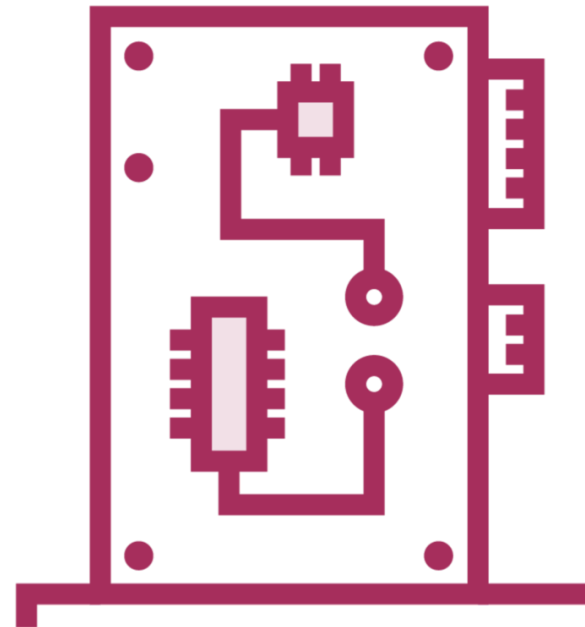
---



# PowerShell Script Requirements



**Retrieve local  
computer  
information**



**Query  
networking  
information**



**Export current  
event log entries**



**Create readable  
document of  
information**





# Demo



## Create PowerShell script

- Retrieve local computer information
- Query networking information
- Export current event log entries
- Create readable document of information



# Summary



## **Goal: Use PowerShell to Query Common Operating System Information**

- Queried running processes and services
- Created scripts to collect log data
- Queried networking information
- Created a script that combined retrieved data into a readable document

