CYBER RANGERS

ARMORING YOUR BUSINESS

POWERSHELL BASIC SCRIPTING
MODULE 2: POWERSHELL PIPELINE

Jan Marek | Cyber Rangers

CEH | CHFI | CompTIA Pentest+ | CEI | MVP | MCT | MCC | MCSA | MCSE

Ethical Hacker, Forensic Investigator & Security Engineer

jan@cyber-rangers.com | www.cyber-rangers.com

AGENDA

- How does pipelining in PowerShell work
- How to use pipeline to sort, select, group, filter and enumerate objects
- Various options of cmdlet outputs

HOW DOES PIPELINING IN POWERSHELL WORK (1/2)

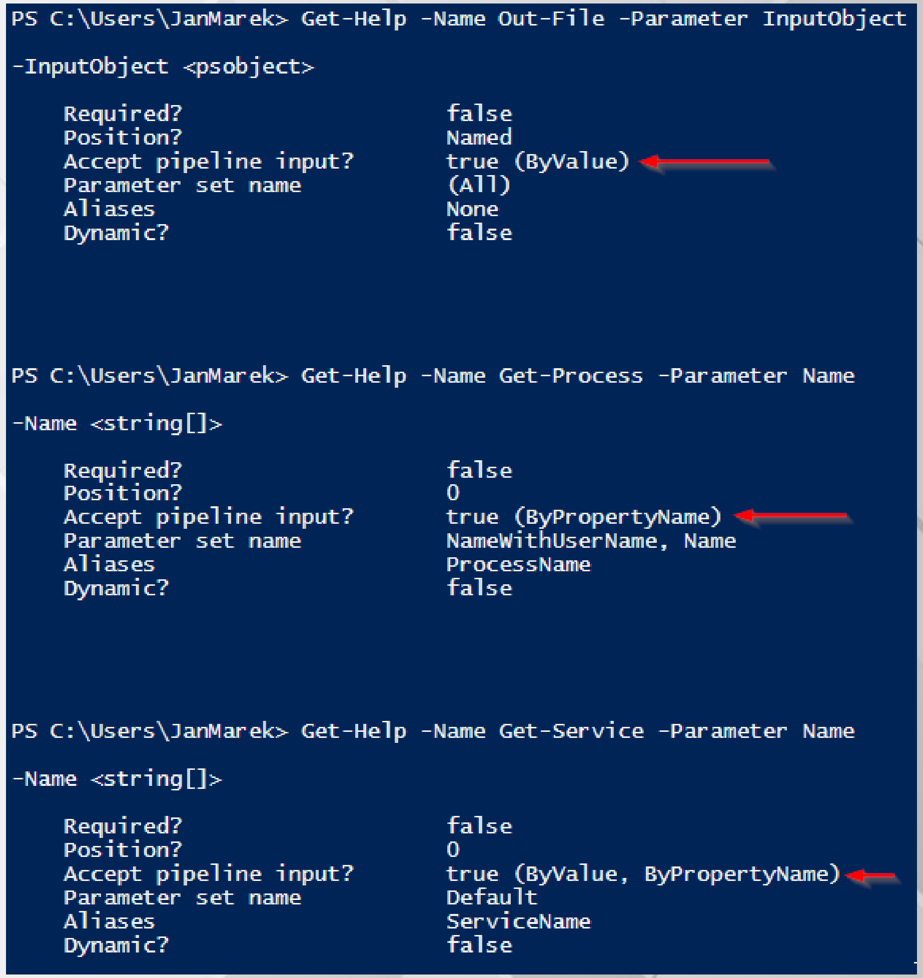
- Commands run in pipeline
- Commands separated by the pipe (|)
- Left to right execution
- First command output piped to next command
- Output produced as object(s) => `Get-Process | Get-Member`
  - Object
  - Property
  - Collection
- Pipeline output "stored" in variables

  `$_`

  `$PSItem`

HOW DOES PIPELINING IN POWERSHELL WORK (2/2)

- Commands accept input only from one parameter
- Two parameter binding types
  - **ByValue** (always tried first)
  - **ByPropertyName** (if ByValue fails)



```
PS C:\Users\JanMarek> Get-Help -Name Out-File -Parameter InputObject

-InputObject <psobject>

    Required?                    false
    Position?                    Named
    Accept pipeline input?       true (ByValue)
    Parameter set name           (All)
    Aliases                      None
    Dynamic?                     false



PS C:\Users\JanMarek> Get-Help -Name Get-Process -Parameter Name

-Name <string[]>

    Required?                    false
    Position?                    0
    Accept pipeline input?       true (ByPropertyName)
    Parameter set name           NameWithUserName, Name
    Aliases                      ProcessName
    Dynamic?                     false



PS C:\Users\JanMarek> Get-Help -Name Get-Service -Parameter Name

-Name <string[]>

    Required?                    false
    Position?                    0
    Accept pipeline input?       true (ByValue, ByPropertyName)
    Parameter set name           Default
    Aliases                      ServiceName
    Dynamic?                     false
```

PASS DATA BYVALUE

```
Get-Service -Name 'spooler'
```

```
'spooler' | Get-Service
```

PASS DATA BYPROPERTYNAME

```
Get-Process -Name 'spoolsv' | Stop-Process
```

```
Get-Volume -DriveLetter 'C' | Optimize-Volume -Analyze
```

PARENTHETICAL PROCESSING

```
'notepad' | Out-File 'C:\Temp\mod002-t1.txt'
Get-Process -Name (Get-Content 'C:\temp\mod002-t1.txt’)



Add-ADGroupMember -Identity 'Czech Users' -Members `
    (Get-ADUser -Filter {City -eq 'Prague'})
```

CALCULATED PROPERTY

```
Get-Process | Select-Object -Property Name,@{n='ComputerName';e={$env:COMPUTERNAME}}
```

```
Get-Process | Select-Object -Property Name,`
    @{n='Virtual Memory [MB]';e={$_.virtualmemorysize/1MB}}
```

```
Get-History | Select-Object commandline, `
    @{n='ExecTime';e={$_.EndExecutionTime - $_.StartExecutionTime}} | `
    Sort-Object -Property ExecTime
```

```
@{name=‘Name of custom property’;expression={scriptblock for calculation}}
@{name=‘’;expression={}}
@{n=‘’;e={}}
```

OBJECT PROCESSING

- Sort

```
Get-Process | Sort-Object -Property Name -Descending
```

- Select

```
Get-Process | Select-Object -Property Name,ID
Get-Process | Select-Object -Property Name -Unique
Get-Process | Select-Object -ExpandProperty Name
Get-Process | Select-Object -First 4
Get-Process | Select-Object -Property * -ExcludeProperty Id
```

- Group

```
Get-Process | Group-Object -Property Name
```

OBJECT PROCESSING

- Filter

```
Get-Process -Name 'conhost'
Get-Process | Where-Object -Property Name -EQ -Value 'conhost'
Get-Process | Where-Object {$_.Name -eq 'conhost’}
```

- Enumerate

```
Get-Process | ForEach-Object {Write-Host "Process name is $($_.name)"}
Get-ChildItem -Path 'C:\Temp\ToEncrypt' | ForEach-Object {$_.Encrypt()}
Get-ChildItem -Path 'C:\Temp\ToEncrypt' | ForEach-Object -MemberName Encrypt
```

OUTPUT FORMATTING

```
Get-Process | Format-Table -Property Name,Id
Get-Service | Format-Table -Property Name,DisplayName,Status -AutoSize -Wrap

Get-Process | Format-List -Property *

Get-Process | Format-Wide -Property id
```

OUTPUT PROCESSING

```
Get-Process | Out-File 'C:\Temp\processes1.txt'
Get-Process | Out-Host -Paging
Get-Process | Out-Printer -Name 'MyPrinter1'
Get-Process | Out-GridView
Get-Process | Out-GridView -Title 'Processes'  -PassThru | Out-GridView

Get-Process | ConvertTo-Csv | Out-File 'C:\temp\processes2.csv'
Get-Process | Export-CSV 'C:\temp\processes2.csv' -Delimiter "`t"
Get-Process | Select-Object -f 1 -Property name,id | ConvertTo-Json

Get-Process | Select-Object -f 3 -Property name,id | `
    ConvertTo-Html -Title 'Processes' -PreContent (Get-Date) | `
    Out-File 'C:\temp\processes.htm'
```

LAB

Q & A

POWERSHELL BASIC SCRIPTING
MODULE 2: POWERSHELL PIPELINE

Jan Marek | Cyber Rangers

CEH | CHFI | CompTIA Pentest+ | CEI | MVP | MCT | MCC | MCSA | MCSE

Ethical Hacker, Forensic Investigator & Security Engineer

jan@cyber-rangers.com | www.cyber-rangers.com

CYBER RANGERS

ARMORING YOUR BUSINESS