Windows PowerShell

powershell can be launched in cmd.exe using powershell command

Basics

Commands:

Get-Command - lists all available cmdlets

Get-Command -CommandType "Function" - lists all commands with the command type function

Get-Help - Provides detailed information about cmdlets

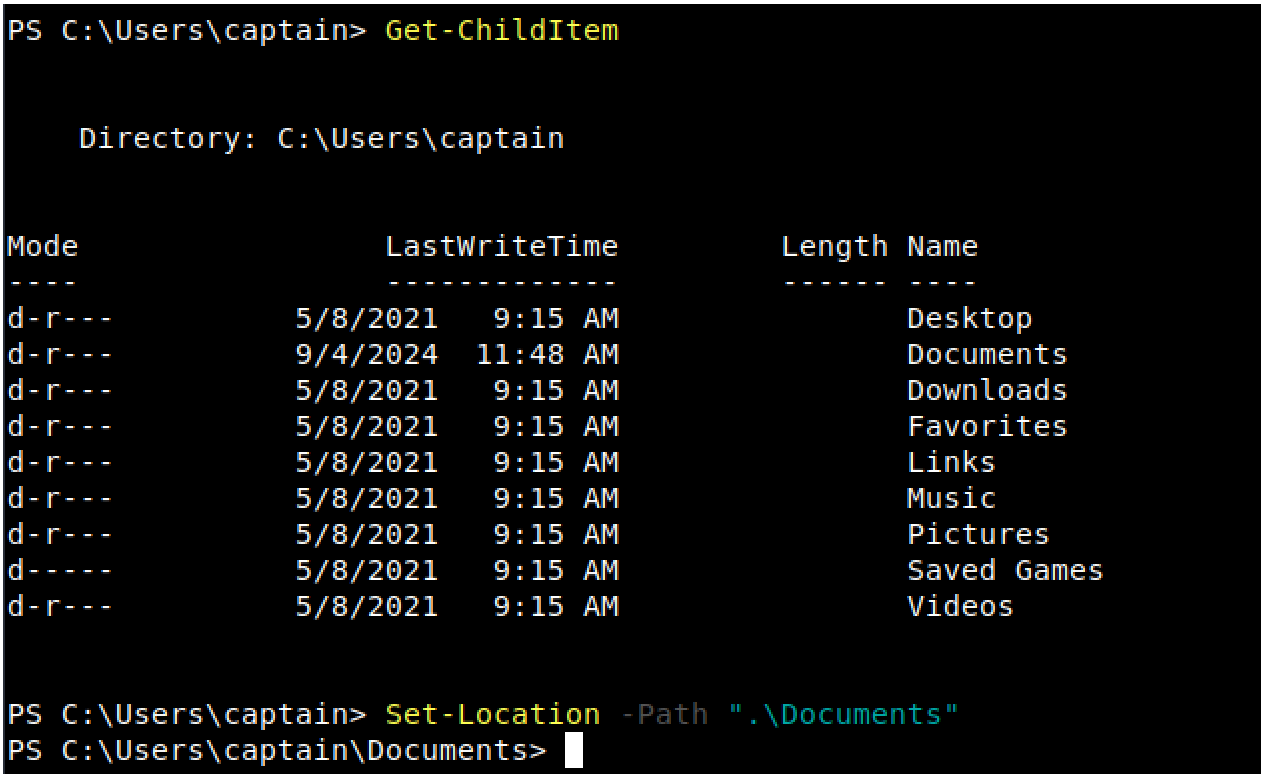
Get-Help-examples - shows list of common ways in which chosen cmdlet can be used

Get-Alias - lists all aliases available

Navigating the File System and Working with Files

Get-ChildItem - lists files and directories in a location specified with the path parameter like linux ls

Set-Location -Path [path name] - sets the path like linux cd



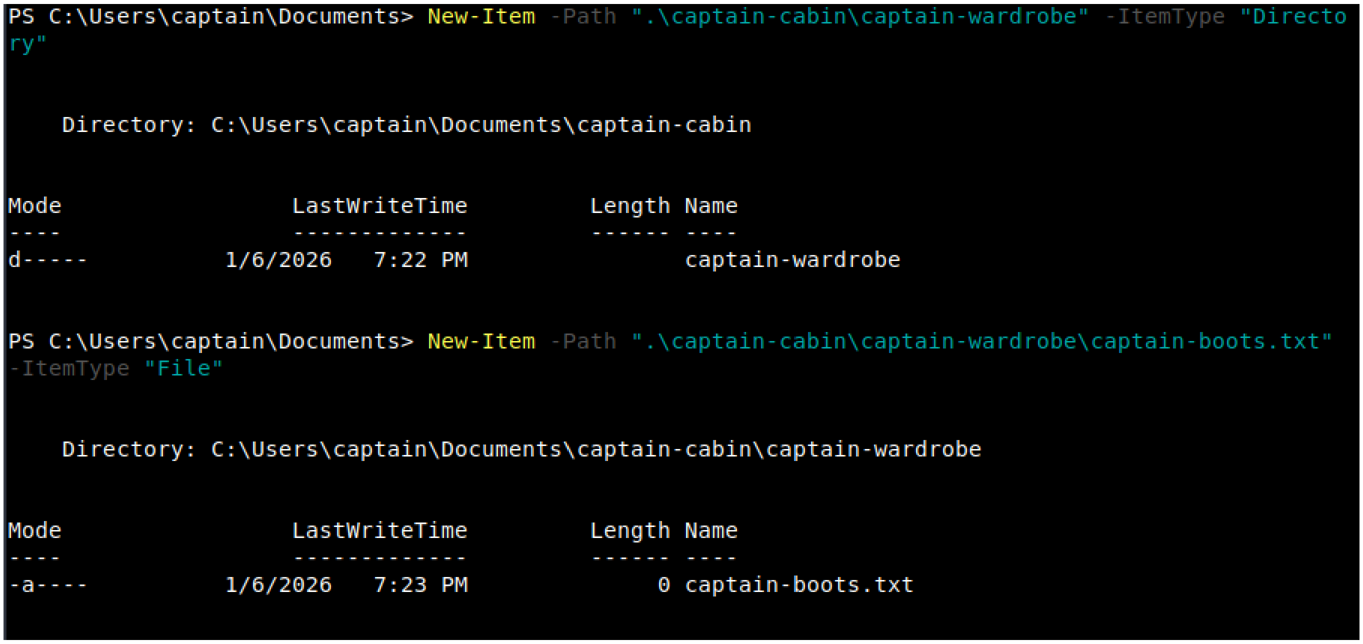
```
PS C:\Users\captain> Get-ChildItem


    Directory: C:\Users\captain


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-r---          5/8/2021   9:15 AM                Desktop
d-r---          9/4/2024  11:48 AM                Documents
d-r---          5/8/2021   9:15 AM                Downloads
d-r---          5/8/2021   9:15 AM                Favorites
d-r---          5/8/2021   9:15 AM                Links
d-r---          5/8/2021   9:15 AM                Music
d-r---          5/8/2021   9:15 AM                Pictures
d-----          5/8/2021   9:15 AM                Saved Games
d-r---          5/8/2021   9:15 AM                Videos


PS C:\Users\captain> Set-Location -Path ".\Documents"
PS C:\Users\captain\Documents>
```

New-Item - creates an item essentially can create new directories/files



```
PS C:\Users\captain\Documents> New-Item -Path ".\captain-cabin\captain-wardrobe" -ItemType "Directo
ry"


    Directory: C:\Users\captain\Documents\captain-cabin


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          1/6/2026   7:22 PM                captain-wardrobe


PS C:\Users\captain\Documents> New-Item -Path ".\captain-cabin\captain-wardrobe\captain-boots.txt"
-ItemType "File"


    Directory: C:\Users\captain\Documents\captain-cabin\captain-wardrobe


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          1/6/2026   7:23 PM              0 captain-boots.txt
```

Remove-Item - removes files and directories
e.g. Remove-Item -Path "[path of file or directory]"

Copy-Item - (equivalent to `copy`)
Move-Item - (equivalent to `move`)

Get-Content works like type in command promt and cat in linux so reads file contents

Piping, Filtering, and Sorting Data

Piping is a technique used in command-line environments that allows the output of one command to be used as the input for another.

In PowerShell, piping is even more powerful because it passes **objects** rather than just text. These objects carry not only the data but also the properties and methods that describe and interact with the data.

Where-Object - filters files by extension property

The operator `-eq` (i.e. "**equal to**") is part of a set of **comparison operators** that are shared with other scripting languages (e.g. Bash, Python). To show the potentiality of the PowerShell's filtering, some of the most useful operators from that list:

- `-ne` : "**not equal**". This operator can be used to exclude objects from the results based on specified criteria.
- `-gt` : "**greater than**". This operator will filter only objects which exceed a specified value. It is important to note that this is a strict comparison, meaning that objects that are equal to the specified value will be excluded from the results.

- `-ge` : "**greater than or equal to**". This is the non-strict version of the previous operator. A combination of `-gt` and `-eq` .
- `-lt` : "**less than**". Like its counterpart, "greater than", this is a strict operator. It will include only objects which are strictly below a certain value.
- `-le` : "**less than or equal to**". Just like its counterpart `-ge` , this is the non-strict version of the previous operator. A combination of `-lt` and `-eq` .

System and Network Information

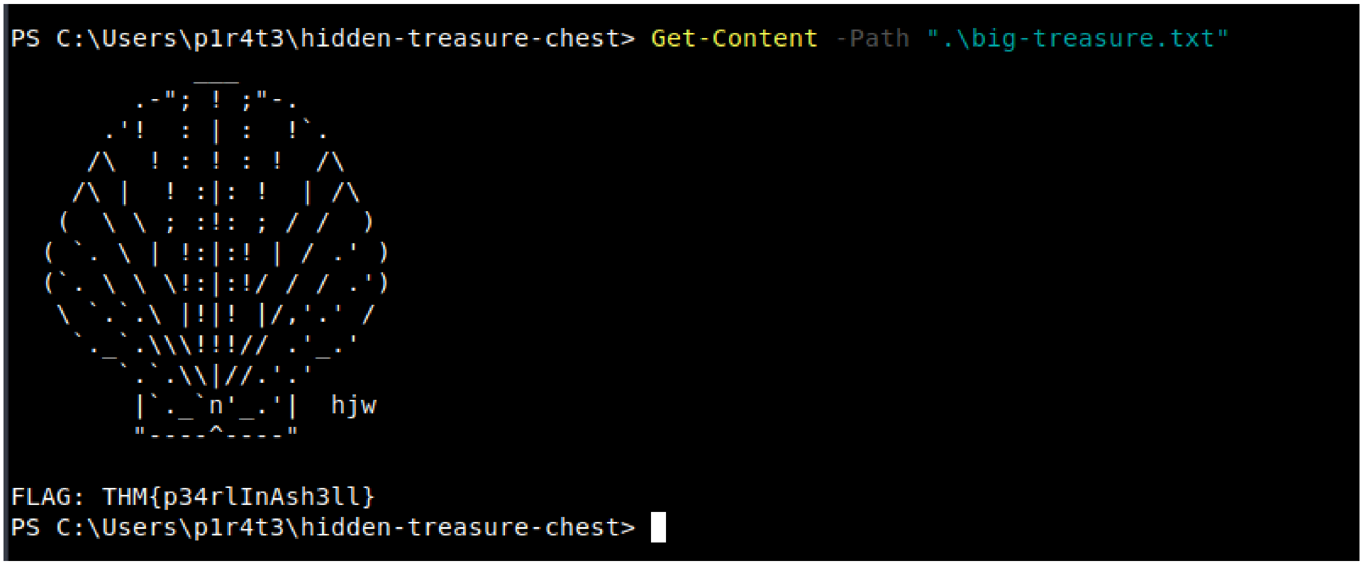
Get-ComputerInfo - is like systeminfo

Get-LocalUser - lists all local user accounts on a system

Get-NetIPConfiguration - provides detailed info about network interfaces on the sytem, including IP addresses, DNS servers and gateway configurations

Get-NetIPAddress - show details for all IP addresses configured on the system, including those that are not currently active

used a combination of the Get-Content, Get-ChildItem and Set-Location commands to find the flag



```
PS C:\Users\p1r4t3\hidden-treasure-chest> Get-Content -Path ".\big-treasure.txt"
            ___
        .-"; ! ;"-.
      .'!  : | :  !`.
     /\  ! : ! : !  /\
    /\ |  ! :|: !  | /\
   (  \ \ ; :!: ; / /  )
  ( `. \ | !:|:! | / .' )
  (`. \ \ \!:|:!/ / / .')
   \ `.`.\ |!|! |/,'.' /
    `._`.\\\!!!// .'_.'
       `.`.\\|//.'.'
        |`._`n'_.'|  hjw
        "----^----"

FLAG: THM{p34rlInAsh3ll}
PS C:\Users\p1r4t3\hidden-treasure-chest>
```

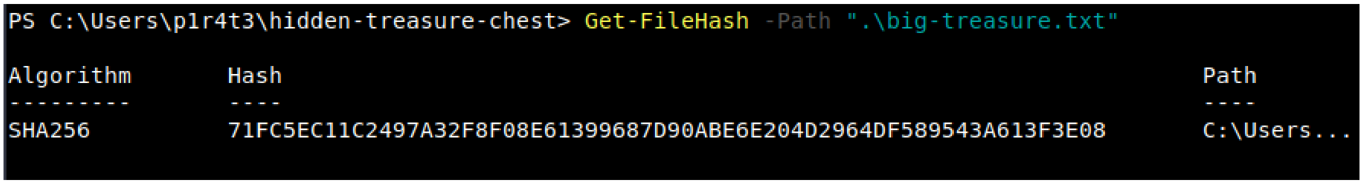
Real-Time System Analysis

`Get-Process` provides a detailed view of all currently running processes, including CPU and memory usage, making it a powerful tool for monitoring and troubleshooting.

`Get-Service` allows the retrieval of information about the status of services on the machine, such as which services are running, stopped, or paused.

`Get-NetTCPConnection` displays current TCP connections, giving insights into both local and remote endpoints

To monitor active network connections, `Get-NetTCPConnection` displays current TCP connections, giving insights into both local and remote endpoints



```
PS C:\Users\p1r4t3\hidden-treasure-chest> Get-FileHash -Path ".\big-treasure.txt"

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
SHA256          71FC5EC11C2497A32F8F08E61399687D90ABE6E204D2964DF589543A613F3E08       C:\Users...
```

here is the hash of the flag file

Scripting

Learning scripting with PowerShell goes beyond the scope of this room. Nonetheless, we must understand that its power makes it a crucial skill across all cyber security roles.

- For **blue team** professionals such as incident responders, malware analysts, and threat hunters, PowerShell scripts can automate many different tasks, including log analysis, detecting anomalies, and extracting indicators of compromise (IOCs). These scripts can also be used to reverse-engineer malicious code (malware) or automate the scanning of systems for signs of intrusion.
- For the **red team**, including penetration testers and ethical hackers, PowerShell scripts can automate tasks like system enumeration, executing remote commands, and crafting obfuscated scripts to bypass defences. Its deep integration with all types of systems makes it a powerful tool for simulating attacks and testing systems’ resilience against real-world threats.
- Staying in the context of cyber security, **system administrators** benefit from PowerShell scripting for automating integrity checks, managing system configurations, and securing networks, especially in remote or large-scale environments. PowerShell scripts can be designed to enforce security policies, monitor systems health, and respond automatically to security incidents, thus enhancing the overall security posture.

`Invoke-Command` is essential for executing commands on remote systems, making it fundamental for system administrators, security engineers and penetration testers.