POWERSHELL-BASED ETHICAL HACKING TOOLS

Jordan Patterson

PS>Attack

PS>Attack is a portable console which combines several PowerShell pentesting tools into one convenient package.

Some notable features of PS>Attack are that it doesn't rely on powershell.exe. Instead, it calls PowerShell directly through the .NET framework. This makes it harder for enterprises to block.

The modules that are bundled with the PS>Attack are encrypted. When PS>Attack starts, they are decrypted into memory. The unencrypted payloads never touch the systems disk, making it difficult for most antivirus engines to catch them.

PS>Attack contains over 100 commands for Privilege Escalation, Recon and Data Exfiltration. It includes the following modules and commands:

- Powersploit
  - Invoke-Mimikatz
  - Get-GPPPassword
  - Invoke-NinjaCopy
  - Invoke-Shellcode
  - Invoke-WMICommand
  - VolumeShadowCopyTools

- PowerTools
  - PowerUp
  - PowerView

- Nishang
  - Gupt-Backdoor
  - Do-Exfiltration
  - DNS-TXT-Pwnage
  - Get-Infromation
  - Get-WLAN-Keys
  - Invoke-PsUACme

- Powercat

- Inveigh

Our 3 favourite features of this tool are:

The infrastructure of the tool: It organizes and tags attacks in a way that makes them easy to find, execute and understand. It also offers great built-in help and examples. This is especially useful for newcomers to pentesting or PowerShell.

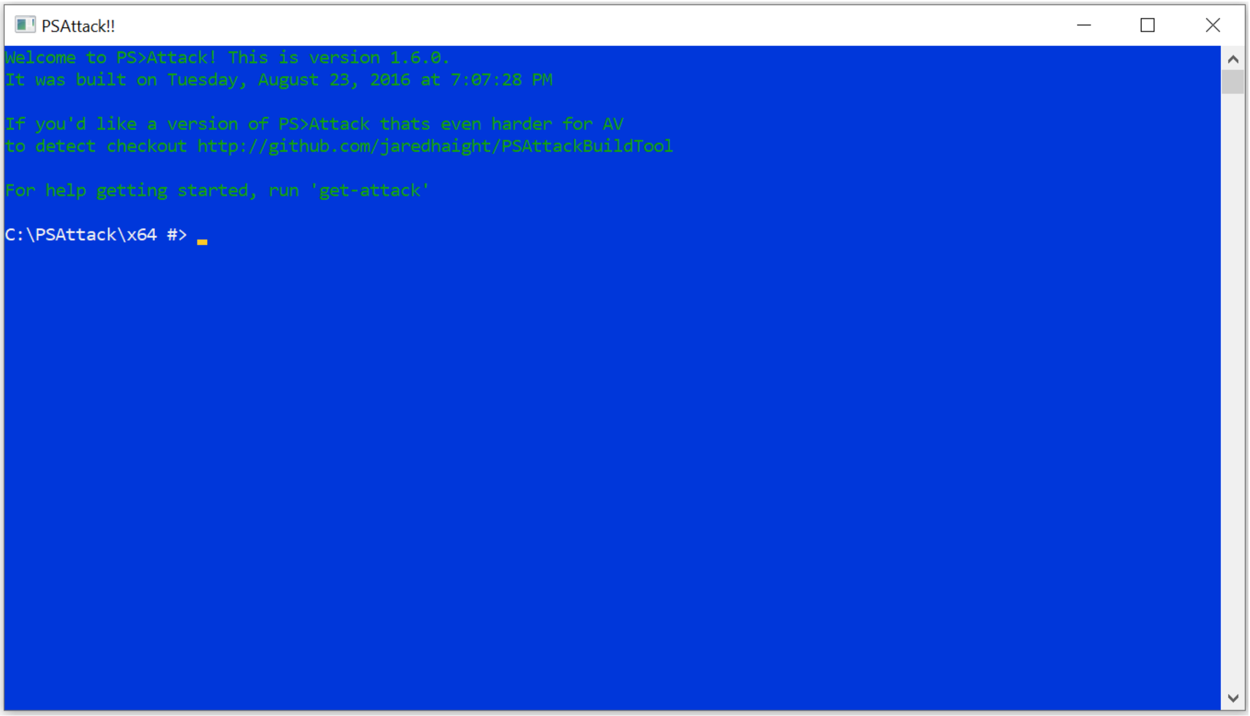
The tool can be run to get information from the local computer: Most attacks in this tool can be run locally, so if you have physical access to the computer you wish to compromise, you don't need a secondary system to run the attacks, just this tool on a USB stick.

The tool can be run to get information from a remote computer: If your target computer is not physically accessible, you can run the attacks remotely against it.

We will now perform a complete demonstration of PS>Attack which should show all 3 of these features in action.

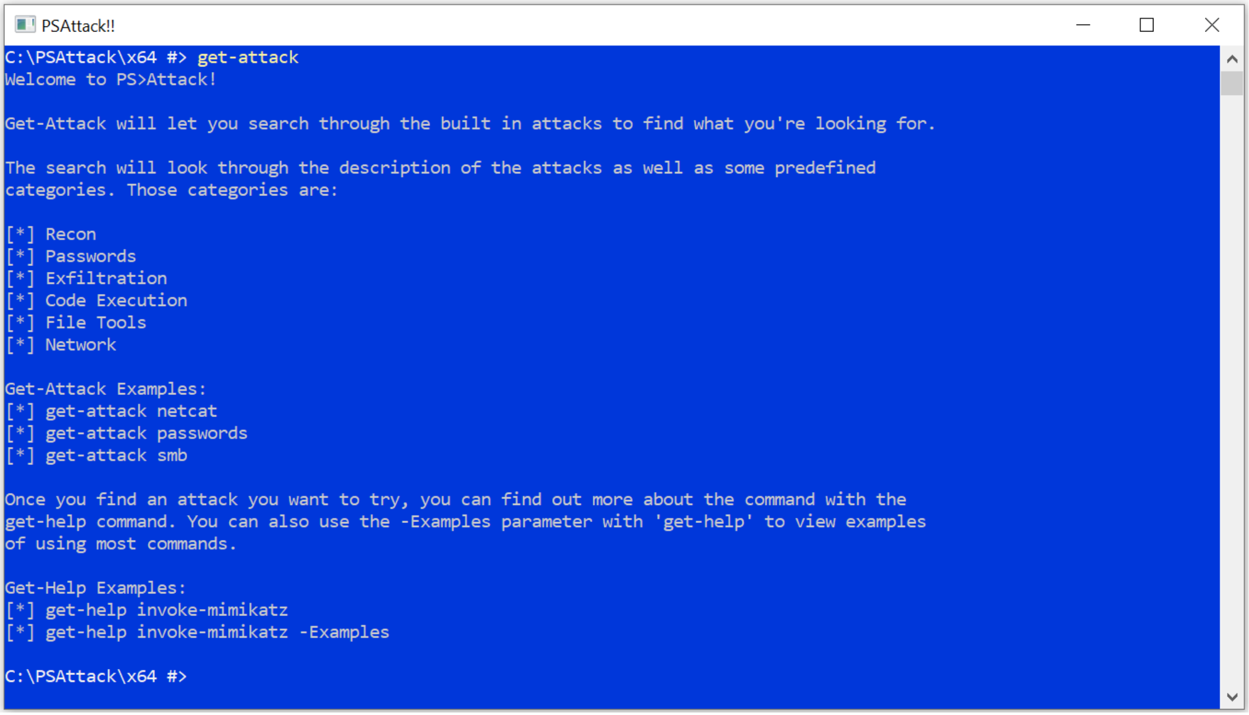
1. PS>Attack can be downloaded from GitHub at the following link: GitHub - GDSSecurity/PSAttack: A portable console aimed at making pentesting with PowerShell a little easier.

2. Once downloaded, you must extract the folder and run the executable PSAttack.exe. This tool is portable and does not need to be installed. It can be run from removable storage.

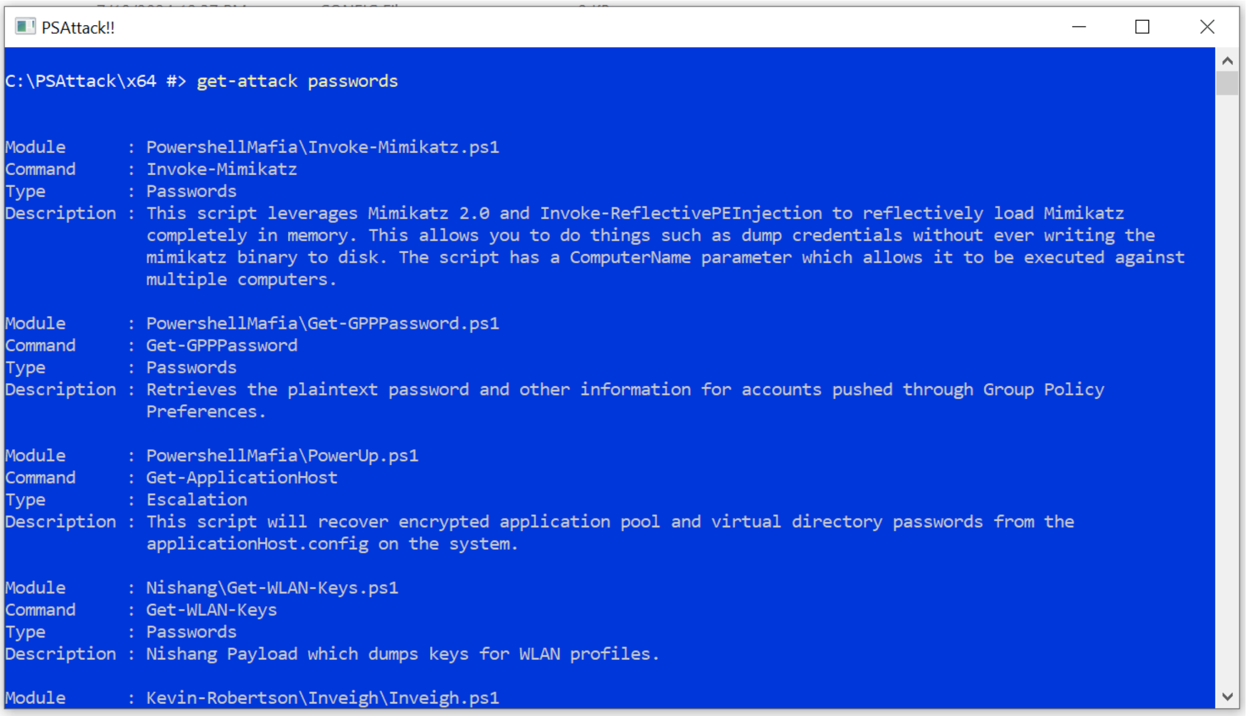


```
Welcome to PS>Attack! This is version 1.6.0.
It was built on Tuesday, August 23, 2016 at 7:07:28 PM

If you'd like a version of PS>Attack thats even harder for AV
to detect checkout http://github.com/jaredhaight/PSAttackBuildTool

For help getting started, run 'get-attack'

C:\PSAttack\x64 #>
```

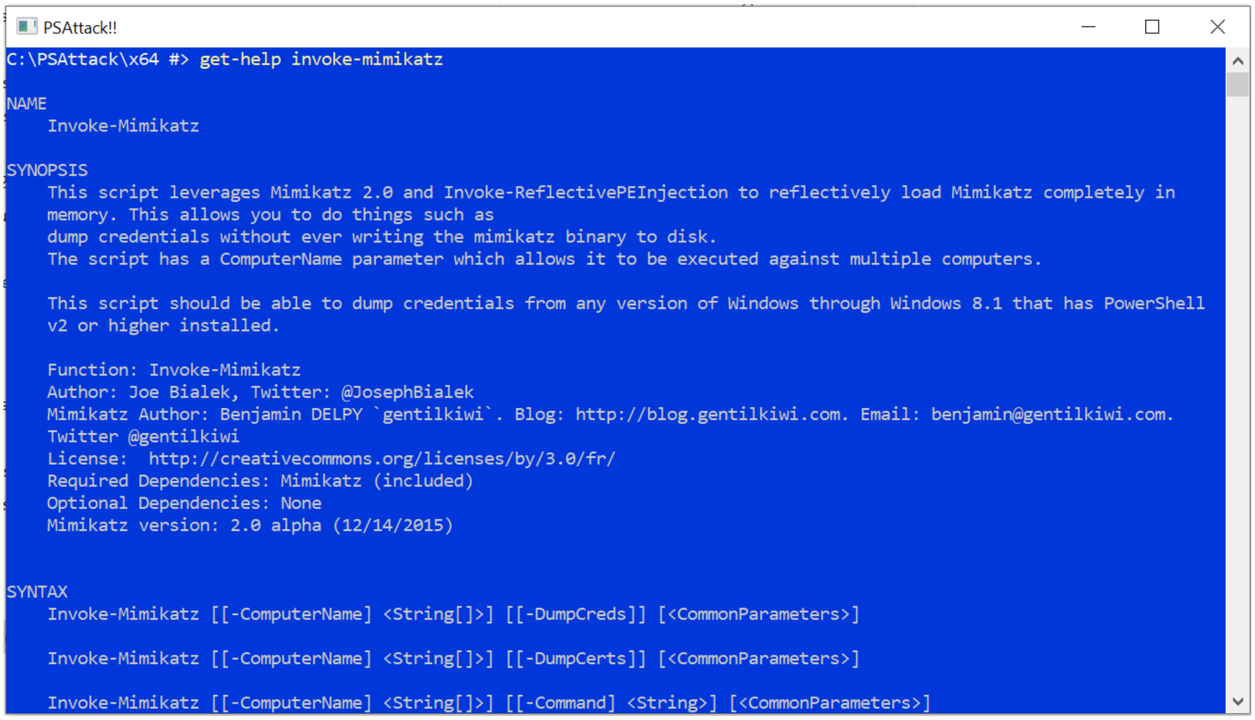
3. Once opened, you can use the "get-attack" command to search for your desired attack. For example, if I use the command "get-attack passwords", PS>Attack will show me all its available tools for password related attacks along with a description of the tool. If I want to see examples of the tool and find more information, I can use the "get-help" command to find out more. For example, "get-help invoke-mimikatz" displays examples and information about the tool.



```
C:\PSAttack\x64 #> get-attack
Welcome to PS>Attack!

Get-Attack will let you search through the built in attacks to find what you're looking for.

The search will look through the description of the attacks as well as some predefined
categories. Those categories are:

[*] Recon
[*] Passwords
[*] Exfiltration
[*] Code Execution
[*] File Tools
[*] Network

Get-Attack Examples:
[*] get-attack netcat
[*] get-attack passwords
[*] get-attack smb

Once you find an attack you want to try, you can find out more about the command with the
get-help command. You can also use the -Examples parameter with 'get-help' to view examples
of using most commands.

Get-Help Examples:
[*] get-help invoke-mimikatz
[*] get-help invoke-mimikatz -Examples

C:\PSAttack\x64 #>
```

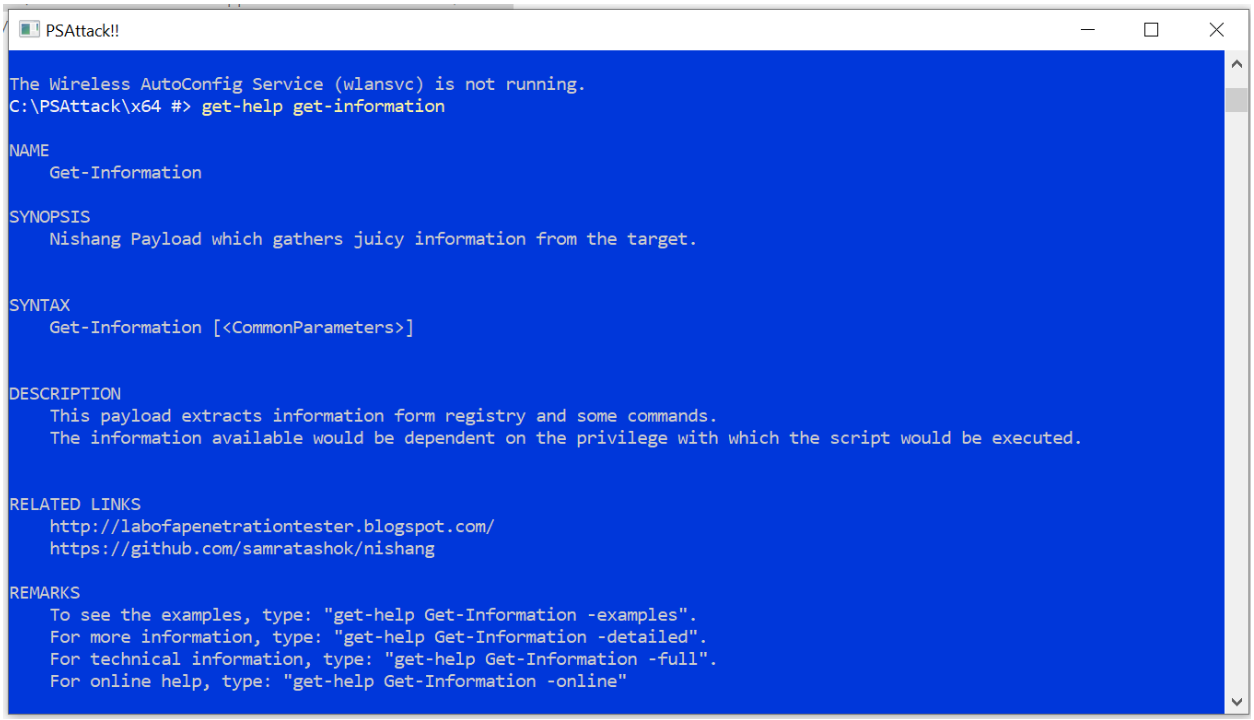


```
C:\PSAttack\x64 #> get-attack passwords


Module      : PowershellMafia\Invoke-Mimikatz.ps1
Command     : Invoke-Mimikatz
Type        : Passwords
Description : This script leverages Mimikatz 2.0 and Invoke-ReflectivePEInjection to reflectively load Mimikatz
              completely in memory. This allows you to do things such as dump credentials without ever writing the
              mimikatz binary to disk. The script has a ComputerName parameter which allows it to be executed against
              multiple computers.

Module      : PowershellMafia\Get-GPPPassword.ps1
Command     : Get-GPPPassword
Type        : Passwords
Description : Retrieves the plaintext password and other information for accounts pushed through Group Policy
              Preferences.

Module      : PowershellMafia\PowerUp.ps1
Command     : Get-ApplicationHost
Type        : Escalation
Description : This script will recover encrypted application pool and virtual directory passwords from the
              applicationHost.config on the system.

Module      : Nishang\Get-WLAN-Keys.ps1
Command     : Get-WLAN-Keys
Type        : Passwords
Description : Nishang Payload which dumps keys for WLAN profiles.

Module      : Kevin-Robertson\Inveigh\Inveigh.ps1
```

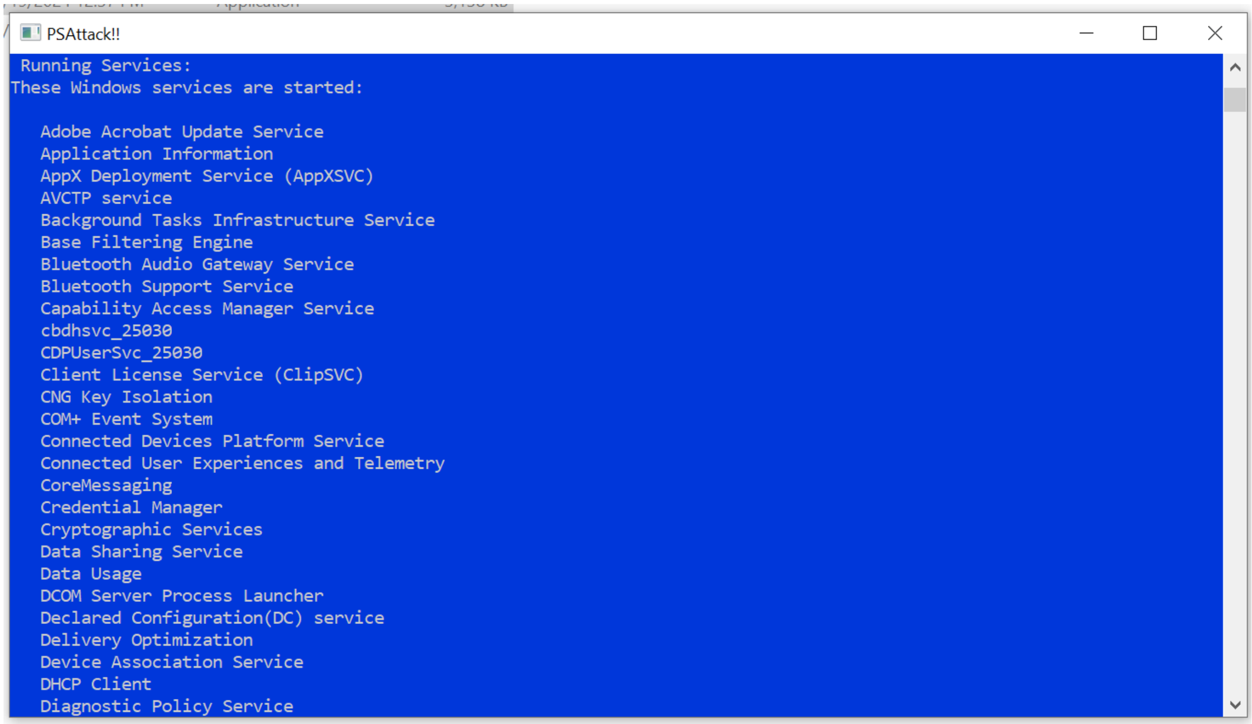


```
C:\PSAttack\x64 #> get-help invoke-mimikatz

NAME
    Invoke-Mimikatz

SYNOPSIS
    This script leverages Mimikatz 2.0 and Invoke-ReflectivePEInjection to reflectively load Mimikatz completely in
    memory. This allows you to do things such as
    dump credentials without ever writing the mimikatz binary to disk.
    The script has a ComputerName parameter which allows it to be executed against multiple computers.

    This script should be able to dump credentials from any version of Windows through Windows 8.1 that has PowerShell
    v2 or higher installed.

    Function: Invoke-Mimikatz
    Author: Joe Bialek, Twitter: @JosephBialek
    Mimikatz Author: Benjamin DELPY `gentilkiwi`. Blog: http://blog.gentilkiwi.com. Email: benjamin@gentilkiwi.com.
    Twitter @gentilkiwi
    License:  http://creativecommons.org/licenses/by/3.0/fr/
    Required Dependencies: Mimikatz (included)
    Optional Dependencies: None
    Mimikatz version: 2.0 alpha (12/14/2015)


SYNTAX
    Invoke-Mimikatz [[-ComputerName] <String[]>] [[-DumpCreds]] [<CommonParameters>]

    Invoke-Mimikatz [[-ComputerName] <String[]>] [[-DumpCerts]] [<CommonParameters>]

    Invoke-Mimikatz [[-ComputerName] <String[]>] [[-Command] <String>] [<CommonParameters>]
```

4. We will now execute an attack locally. Using PS>Attack we will run “get-information” which will use Nishang to gather and display information about the target such as running processes, installed applications, and logged in users. This information could be invaluable in finding a process, application or user account with an exploitable vulnerability.

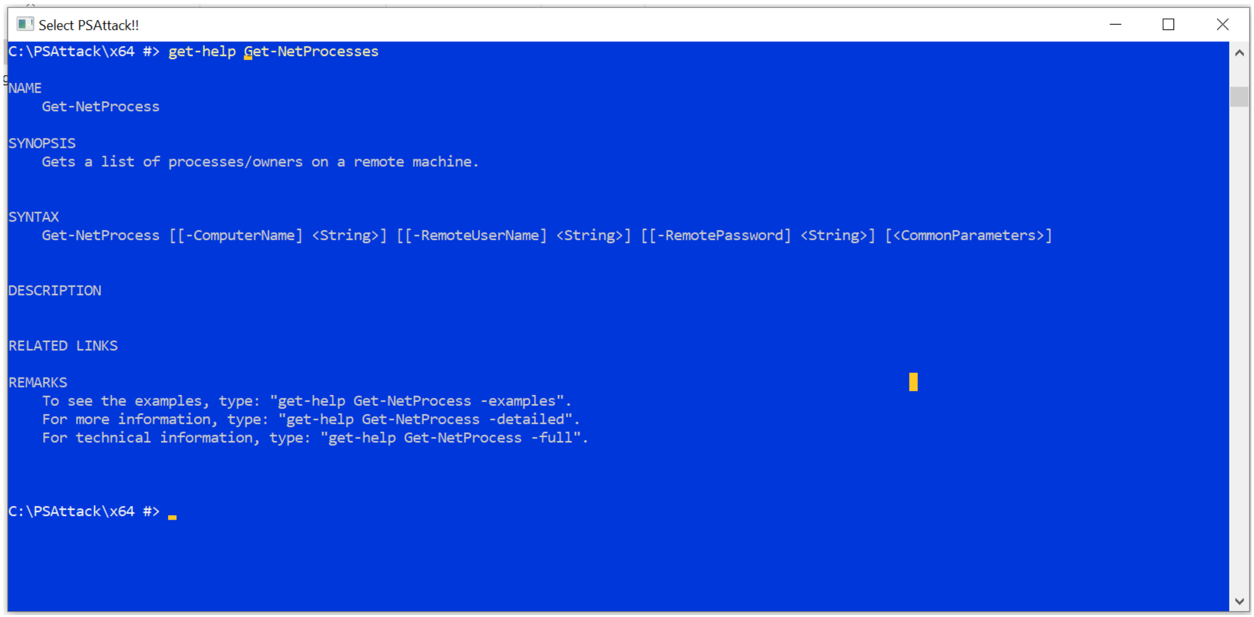


```
The Wireless AutoConfig Service (wlansvc) is not running.
C:\PSAttack\x64 #> get-help get-information

NAME
    Get-Information

SYNOPSIS
    Nishang Payload which gathers juicy information from the target.


SYNTAX
    Get-Information [<CommonParameters>]


DESCRIPTION
    This payload extracts information form registry and some commands.
    The information available would be dependent on the privilege with which the script would be executed.


RELATED LINKS
    http://labofapenetrationtester.blogspot.com/
    https://github.com/samratashok/nishang

REMARKS
    To see the examples, type: "get-help Get-Information -examples".
    For more information, type: "get-help Get-Information -detailed".
    For technical information, type: "get-help Get-Information -full".
    For online help, type: "get-help Get-Information -online"
```

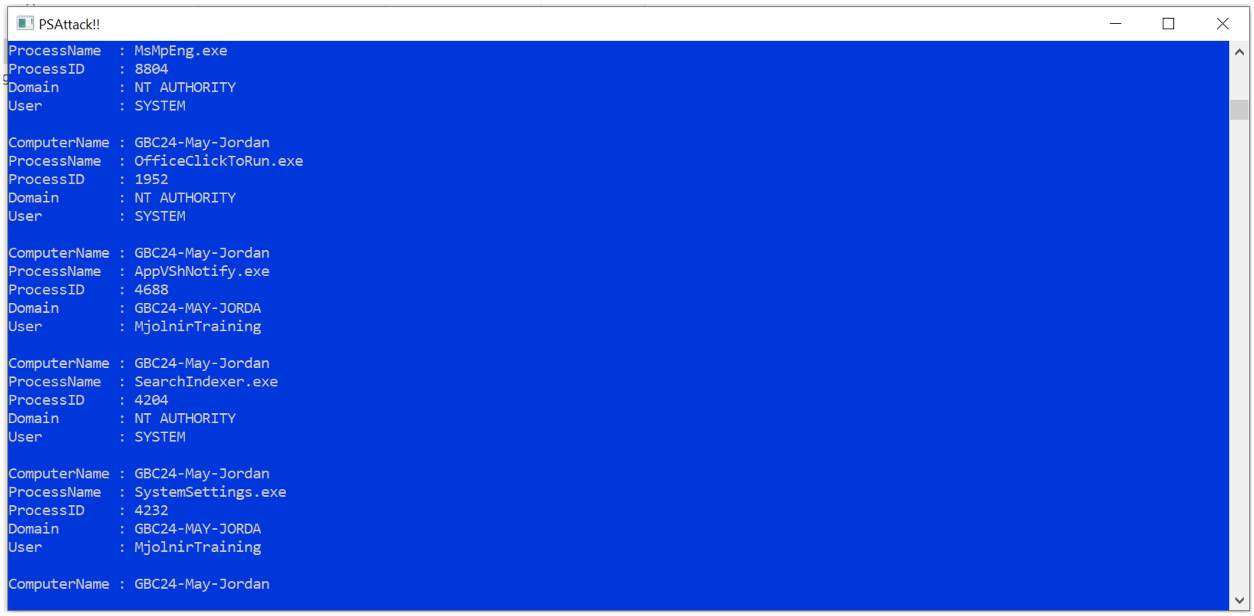


```
 Running Services:
These Windows services are started:

   Adobe Acrobat Update Service
   Application Information
   AppX Deployment Service (AppXSVC)
   AVCTP service
   Background Tasks Infrastructure Service
   Base Filtering Engine
   Bluetooth Audio Gateway Service
   Bluetooth Support Service
   Capability Access Manager Service
   cbdhsvc_25030
   CDPUserSvc_25030
   Client License Service (ClipSVC)
   CNG Key Isolation
   COM+ Event System
   Connected Devices Platform Service
   Connected User Experiences and Telemetry
   CoreMessaging
   Credential Manager
   Cryptographic Services
   Data Sharing Service
   Data Usage
   DCOM Server Process Launcher
   Declared Configuration(DC) service
   Delivery Optimization
   Device Association Service
   DHCP Client
   Diagnostic Policy Service
```

5. We will now execute an attack on a remote computer. We will run the command “Get-NetProcesses -Computername GBC-May-Jordan” which will give us all the processes and their owners running on that remote computer.



```
C:\PSAttack\x64 #> get-help Get-NetProcesses

NAME
    Get-NetProcess

SYNOPSIS
    Gets a list of processes/owners on a remote machine.


SYNTAX
    Get-NetProcess [[-ComputerName] <String>] [[-RemoteUserName] <String>] [[-RemotePassword] <String>] [<CommonParameters>]


DESCRIPTION


RELATED LINKS

REMARKS
    To see the examples, type: "get-help Get-NetProcess -examples".
    For more information, type: "get-help Get-NetProcess -detailed".
    For technical information, type: "get-help Get-NetProcess -full".


C:\PSAttack\x64 #> _
```



```
ProcessName  : MsMpEng.exe
ProcessID    : 8804
Domain       : NT AUTHORITY
User         : SYSTEM

ComputerName : GBC24-May-Jordan
ProcessName  : OfficeClickToRun.exe
ProcessID    : 1952
Domain       : NT AUTHORITY
User         : SYSTEM

ComputerName : GBC24-May-Jordan
ProcessName  : AppVShNotify.exe
ProcessID    : 4688
Domain       : GBC24-MAY-JORDA
User         : MjolnirTraining

ComputerName : GBC24-May-Jordan
ProcessName  : SearchIndexer.exe
ProcessID    : 4204
Domain       : NT AUTHORITY
User         : SYSTEM

ComputerName : GBC24-May-Jordan
ProcessName  : SystemSettings.exe
ProcessID    : 4232
Domain       : GBC24-MAY-JORDA
User         : MjolnirTraining

ComputerName : GBC24-May-Jordan
```