CYB631 Automating Information Security with Python and Shell Scripting

Lab 6: Connecting Python with Security Tools

Your Name: Vaibhav mayekar

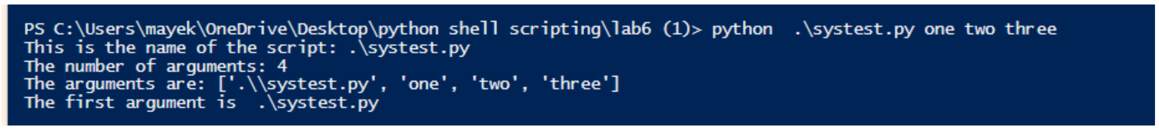
Exercises:

[Exercise I: Passing arguments from command line]

1. We will learn how to pass arguments in python programs.
2. Using Python IDLE, open **systest.py**. Review the codes to understand what it does. The program reads 3 arguments and then print them out. To run it, under PowerShell command line

   python .\systest.py one two three

3. Paste your results here:



```
PS C:\Users\mayek\OneDrive\Desktop\python shell scripting\lab6 (1)> python  .\systest.py one two three
This is the name of the script: .\systest.py
The number of arguments: 4
The arguments are: ['.\\systest.py', 'one', 'two', 'three']
The first argument is  .\systest.py
```

4. What would be the value of **sys.argv[3]** in your command above? **3**
5. Now, let revise log frequency analysis program in the previous lab so that it can read the filename from command line to analyze log files.
6. Please use Python IDLE to review the program **logfreqbyname.py**. The program has additional codes to read and process the first argument and pass it to the program as filename to read into the dataframe.
7. Make sure that you have the application log file (appevent.csv) under your work folder from previous lab. If you do not, use the following PowerShell command to generate it.

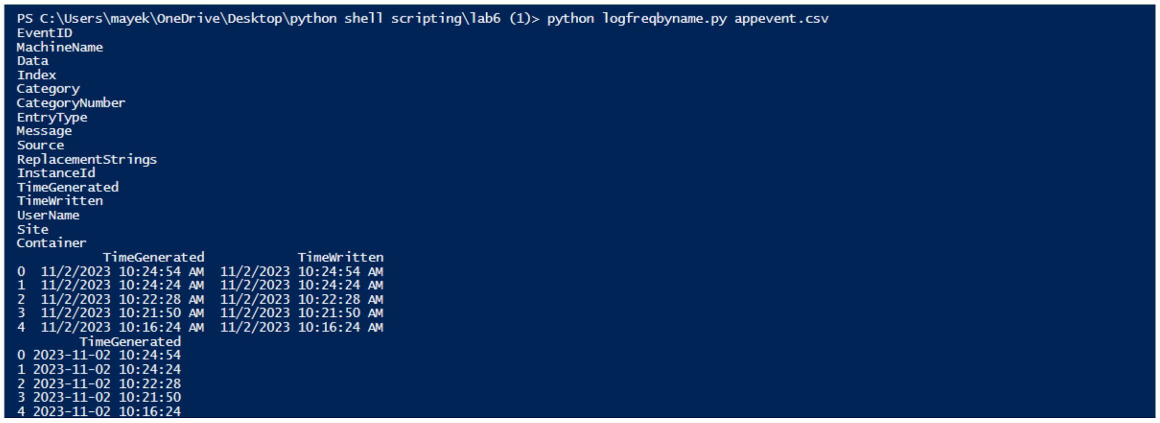
   Get-EventLog -logname Application -Newest 10000| Export-Csv -Path .\appevent.csv

   cat appevent.csv -head 10

8. Run the program under PowerShell command line

   python logfreqbyname.py appevent.csv

9. Paste your results here



```
PS C:\Users\mayek\OneDrive\Desktop\python shell scripting\lab6 (1)> python logfreqbyname.py appevent.csv
EventID
MachineName
Data
Index
Category
CategoryNumber
EntryType
Message
Source
ReplacementStrings
InstanceId
TimeGenerated
TimeWritten
UserName
Site
Container
           TimeGenerated            TimeWritten
0  11/2/2023 10:24:54 AM  11/2/2023 10:24:54 AM
1  11/2/2023 10:24:24 AM  11/2/2023 10:24:24 AM
2  11/2/2023 10:22:28 AM  11/2/2023 10:22:28 AM
3  11/2/2023 10:21:50 AM  11/2/2023 10:21:50 AM
4  11/2/2023 10:16:24 AM  11/2/2023 10:16:24 AM
        TimeGenerated
0 2023-11-02 10:24:54
1 2023-11-02 10:24:24
2 2023-11-02 10:22:28
3 2023-11-02 10:21:50
4 2023-11-02 10:16:24
```

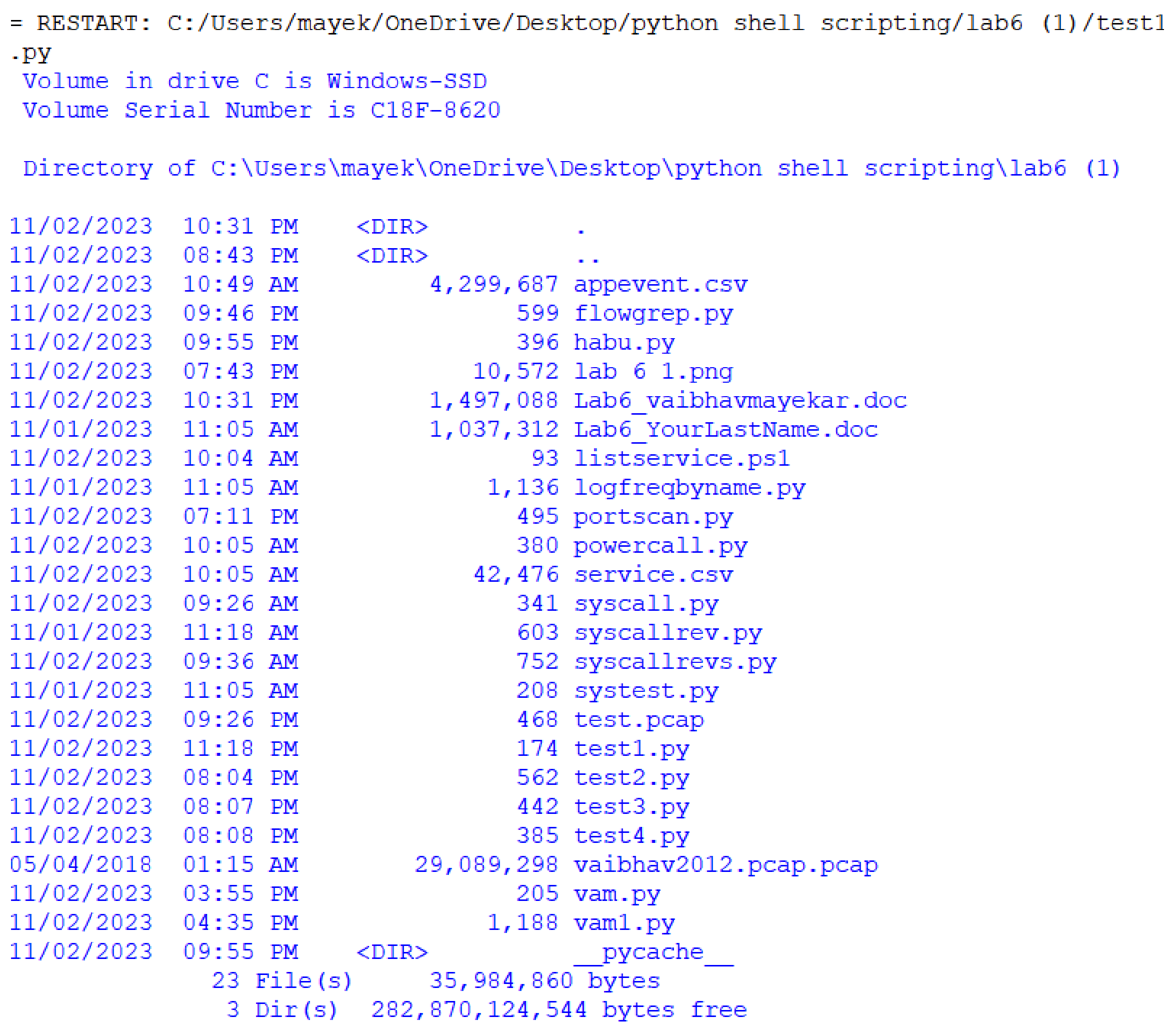
.

[Exercise II: Run Terminal Command in Python]

10. It is convenient to be able to run terminal command in Python. This will reply on two native Python modules: **os** and **subprocess**. Use Python IDLE, open and review **syscall.py**.

    - Show an example of output from **subprocess.check_output('dir',shell=True)** and explain what it is.

```
import subprocess

# Use subprocess to run the "dir" command
output = subprocess.check_output('dir', shell=True, encoding='utf-8')

# Print the output
print(output)
```



```
= RESTART: C:/Users/mayek/OneDrive/Desktop/python shell scripting/lab6 (1)/test1
.py
 Volume in drive C is Windows-SSD
 Volume Serial Number is C18F-8620

 Directory of C:\Users\mayek\OneDrive\Desktop\python shell scripting\lab6 (1)

11/02/2023  10:31 PM    <DIR>          .
11/02/2023  08:43 PM    <DIR>          ..
11/02/2023  10:49 AM         4,299,687 appevent.csv
11/02/2023  09:46 PM               599 flowgrep.py
11/02/2023  09:55 PM               396 habu.py
11/02/2023  07:43 PM            10,572 lab 6 1.png
11/02/2023  10:31 PM         1,497,088 Lab6_vaibhavmayekar.doc
11/01/2023  11:05 AM         1,037,312 Lab6_YourLastName.doc
11/02/2023  10:04 AM                93 listservice.ps1
11/01/2023  11:05 AM             1,136 logfreqbyname.py
11/02/2023  07:11 PM               495 portscan.py
11/02/2023  10:05 AM               380 powercall.py
11/02/2023  10:05 AM            42,476 service.csv
11/02/2023  09:26 AM               341 syscall.py
11/01/2023  11:18 AM               603 syscallrev.py
11/02/2023  09:36 AM               752 syscallrevs.py
11/01/2023  11:05 AM               208 systest.py
11/02/2023  09:26 PM               468 test.pcap
11/02/2023  11:18 PM               174 test1.py
11/02/2023  08:04 PM               562 test2.py
11/02/2023  08:07 PM               442 test3.py
11/02/2023  08:08 PM               385 test4.py
05/04/2018  01:15 AM        29,089,298 vaibhav2012.pcap.pcap
11/02/2023  03:55 PM               205 vam.py
11/02/2023  04:35 PM             1,188 vam1.py
11/02/2023  09:55 PM    <DIR>          __pycache__
              23 File(s)     35,984,860 bytes
               3 Dir(s)  282,870,124,544 bytes free
```

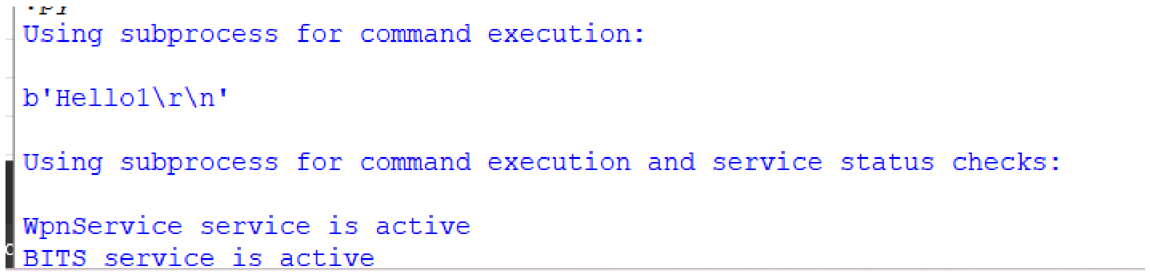
11. Add codes in the program to show the status of services.

12. Show your revised program here

```
import subprocess

print('Using subprocess for command execution:\n')
res = subprocess.check_output('echo Hello1\n', shell=True)
print(res)

print('\nUsing subprocess for command execution and service status checks:\n')
```

```
def check_service_status(service_name):
    try:
        status_output = subprocess.check_output(['sc', 'query', service_name], text=True)
        if 'STATE' in status_output and 'RUNNING' in status_output:
            print(f'{service_name} service is active')
        else:
            print(f'{service_name} service is not active')
    except subprocess.CalledProcessError as e:
        print(f'Failed to check {service_name} service status. Error: {e}')

# Check the status of the WpnService service
check_service_status('WpnService')

# Check the status of the BITS service
    check_service_status('BITS')
```

13. Show your results:



```
Using subprocess for command execution:

b'Hello1\r\n'

Using subprocess for command execution and service status checks:

WpnService service is active
BITS service is active
```

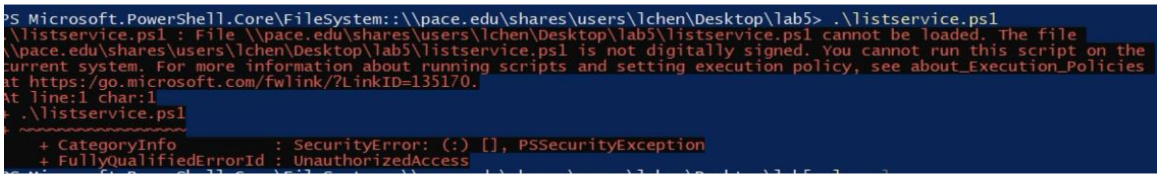
14. Create a PowerShell program **listservice.ps1** to list all of the running service in the system and this file outputs the results to **service.csv**.

    Get-Service | Where-Object {$_.Status -EQ "Running"} | Export-Csv -Path .\service.csv

15. Try running this program under PowerShell command line:

    .\listservice.ps1

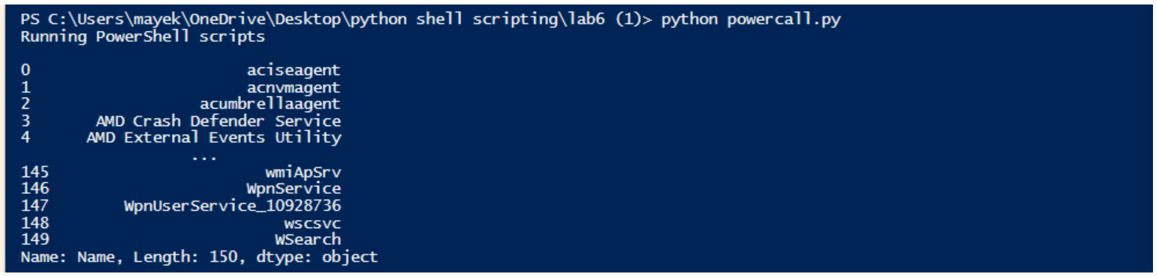
16. [Debugging] You may run into problem when executing listservice.ps1 due to execution policy restriction. In this case, adjust the execution policy using **Set-ExecutionPolicy** cmdlet to either RemoteSigned or ByPass so that you can run it. Remember to set it back later so that your computer will not execute random scripts downloaded from the Internet. If none of them work, you will have to digitally sign listservice.ps1.



```
PS Microsoft.PowerShell.Core\FileSystem::\\pace.edu\shares\users\lchen\Desktop\lab5> .\listservice.ps1
.\listservice.ps1 : File \\pace.edu\shares\users\lchen\Desktop\lab5\listservice.ps1 cannot be loaded. The file
\\pace.edu\shares\users\lchen\Desktop\lab5\listservice.ps1 is not digitally signed. You cannot run this script on the
current system. For more information about running scripts and setting execution policy, see about_Execution_Policies
at https:/go.microsoft.com/fwlink/?LinkID=135170.
At line:1 char:1
+ .\listservice.ps1
+ ~~~~~~~~~~~~~~~~~
    + CategoryInfo          : SecurityError: (:) [], PSSecurityException
    + FullyQualifiedErrorId : UnauthorizedAccess
```

17. Let us try to run this program from a Python program. Use Python IDLE, review **powercall.py**. This program will call **listservice.ps1** inside the Python program and read the results into a dataframe. Run the program.

18. Show the results here.



```
PS C:\Users\mayek\OneDrive\Desktop\python shell scripting\lab6 (1)> python powercall.py
Running PowerShell scripts

0                        aciseagent
1                        acnvmagent
2                      acumbrellaagent
3         AMD Crash Defender Service
4        AMD External Events Utility
                          ...
145                        wmiApSrv
146                      WpnService
147         WpnUserService_10928736
148                          wscsvc
149                         WSearch
Name: Name, Length: 150, dtype: object
```

[Exercise III: Setup a OpenSSH server]

19. To run our later exercises, we will need a OpenSSH server and a OpenSSH client on the Windows host. Instructions in this exercise follow Microsoft official documentation, https://learn.microsoft.com/en-us/windows-server/administration/openssh/openssh_install_firstuse?tabs=powershell#install-openssh-for-windows.

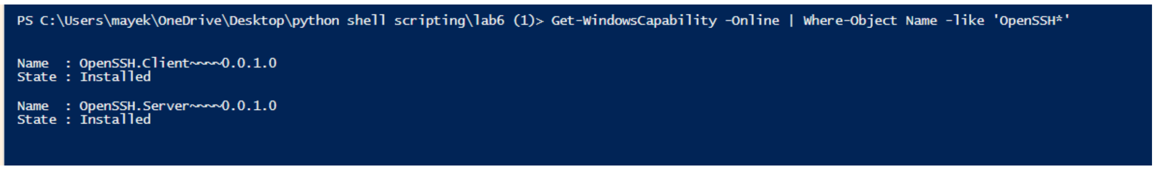
20. Before starting, make sure that you have the administrator privilege to install software.

21. First, check to see if the Windows machine has already installed OpenSSH. In PowerShell,

    Get-WindowsCapability -Online | Where-Object Name -like 'OpenSSH*'

22. The result from above will show if the OpenSSH server and client are either "Installed" or "NotPresent". If either of them is "NotPresent", you will need to install the server, the client, or both.

23. Paste a screenshot of your results above to show OpenSSH status.



```
PS C:\Users\mayek\OneDrive\Desktop\python shell scripting\lab6 (1)> Get-WindowsCapability -Online | Where-Object Name -like 'OpenSSH*'

Name  : OpenSSH.Client~~~~0.0.1.0
State : Installed

Name  : OpenSSH.Server~~~~0.0.1.0
State : Installed
```

24. Install OpenSSH on Windows. You can use whatever way that works for you. I would recommend either one of the following two ways.

<u>Method 1- Use PowerShell to install on Windows server:</u>

- Use the command below to install either the client, the server, or both.

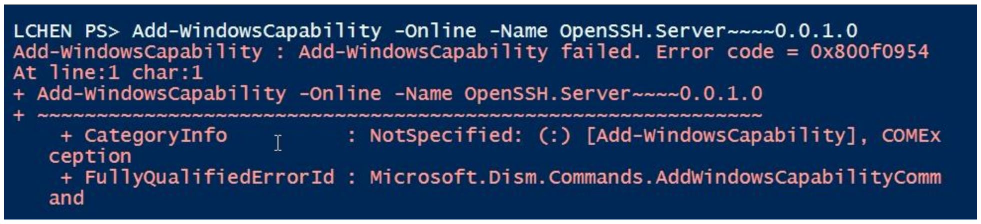
# Install the OpenSSH Client

Add-WindowsCapability -Online -Name OpenSSH.Client~~~~0.0.1.0

# Install the OpenSSH Server

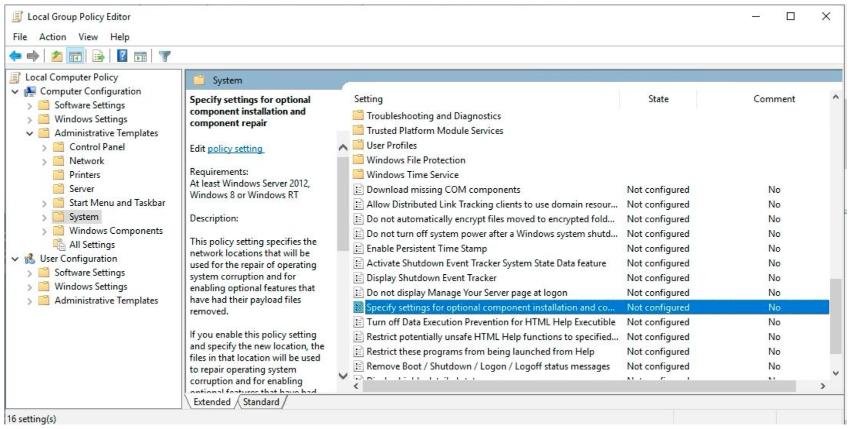
Add-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0

- However, you might encounter error code below because the group policy does not allow you to add software.



```
LCHEN PS> Add-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0
Add-WindowsCapability : Add-WindowsCapability failed. Error code = 0x800f0954
At line:1 char:1
+ Add-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [Add-WindowsCapability], COMEx
   ception
    + FullyQualifiedErrorId : Microsoft.Dism.Commands.AddWindowsCapabilityComm
   and
```

- In this case, you can solve the problem by changing that group policy as below.
  - Open local group policy editor, gpedit.msc
  - As the figure below, click on Computer Configuration, Administrative Templates, System.



Specify settings for optional component installation and component repair

Edit policy setting

Requirements:
At least Windows Server 2012, Windows 8 or Windows RT

Description:

This policy setting specifies the network locations that will be used for the repair of operating system corruption and for enabling optional features that have had their payload files removed.

If you enable this policy setting and specify the new location, the files in that location will be used to repair operating system corruption and for enabling

Setting	State	Comment
Troubleshooting and Diagnostics		
Trusted Platform Module Services		
User Profiles		
Windows File Protection		
Windows Time Service		
Download missing COM components	Not configured	No
Allow Distributed Link Tracking clients to use domain resour...	Not configured	No
Do not automatically encrypt files moved to encrypted fold...	Not configured	No
Do not turn off system power after a Windows system shutd...	Not configured	No
Enable Persistent Time Stamp	Not configured	No
Activate Shutdown Event Tracker System State Data feature	Not configured	No
Display Shutdown Event Tracker	Not configured	No
Do not display Manage Your Server page at logon	Not configured	No
Specify settings for optional component installation and co...	Not configured	No
Turn off Data Execution Prevention for HTML Help Executible	Not configured	No
Restrict potentially unsafe HTML Help functions to specified...	Not configured	No
Restrict these programs from being launched from Help	Not configured	No
Remove Boot / Shutdown / Logon / Logoff status messages	Not configured	No

  - Open Specify settings for optional component installation and component repair. In this setting, select **Enabled** and check **Download repair content and optional features directly from Windows Updates instead of Windows Server Updates Services (WSUS).**
  - Close local group policy editor. Restart the machine.
  - Then, try install the client or the server again using **Add-WindowsCapability.**
- You can later remove OpenSSH by the following when you do not need them on your machine.

  # Uninstall the OpenSSH Client

Remove-WindowsCapability -Online -Name OpenSSH.Client~~~~0.0.1.0

# Uninstall the OpenSSH Server

Remove-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0

<u>Method 2- Directly install OpenSSH from external sources:</u>

- In Microsoft's official document, the Github below is recommended and use as your own liability.

  https://github.com/PowerShell/Win32-OpenSSH/releases

- You can download OpenSSH for Windows from the link below.

https://github.com/PowerShell/Win32-OpenSSH/releases/download/v9.4.0.0p1-Beta/OpenSSH-Win64.zip

- Extract the file, change the directory name to OpenSSH and then move the entire directory under **C:\Program Files\ .**
- Under PowerShell ISE, navigate to the OpenSSH-Win64 directory under OpenSSH. If you are using the Windows VM on Horizon Desktop, navigate to the directory using the command below:

  Set-Location 'C:\Program Files\OpenSSH\OpenSSH-Win64'

- You should find the installation PS script (install-sshd.ps1) under this directory. Let us run the PS script to install the server.

  powershell.exe -ExecutionPolicy Bypass -File install-sshd.ps1

- You can later remove OpenSSH by the following when you do not need them.

  powershell.exe -ExecutionPolicy Bypass -File uninstall-sshd.ps1

25. Once both OpenSSH server and client are installed, check to see if OpenSSH server and client are running. They are most likely not running at this point.

    Get-Service 'ssh*'

26. You should status of sshd (SSH server) and ssh-agent (SSH authentication agent). If none of them are running. Start openssh server.

    Start-Service sshd

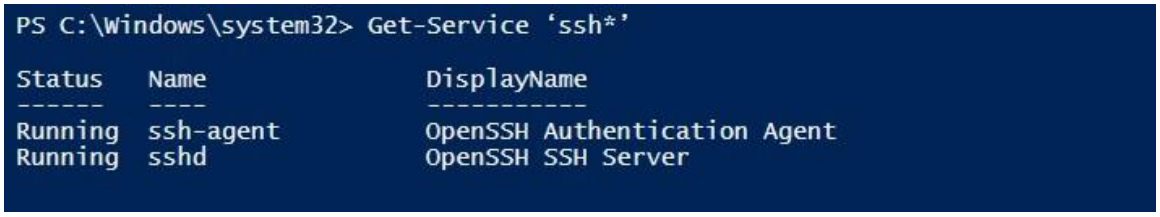
If you would like OpenSSH to run automatically every time you started the host, try

Set-Service -Name sshd -StartupType 'Automatic'

27. Check to see if the server is running.

    Get-Service 'ssh*'

28. It should show that the OpenSSH server is running. Paste your results from above.



```
PS C:\Windows\system32> Get-Service 'ssh*'

Status   Name               DisplayName
------   ----               -----------
Running  ssh-agent          OpenSSH Authentication Agent
Running  sshd               OpenSSH SSH Server
```

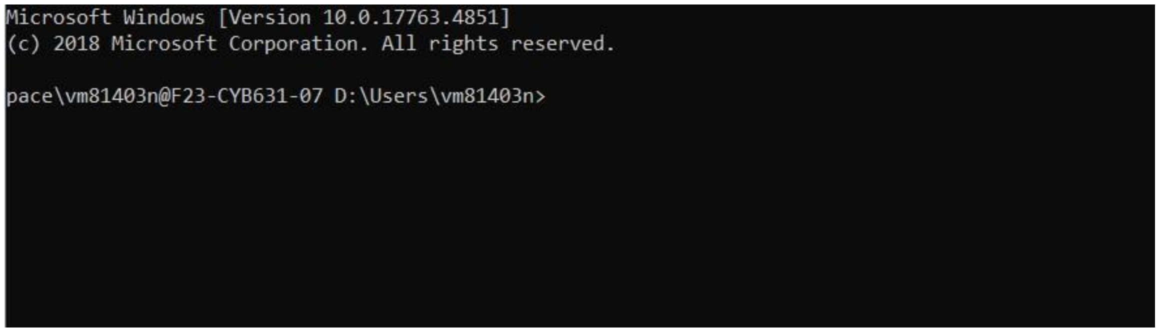
29. Now, review firewall rules about SSH and add rules to allow SSH.

```
if ( !(Get-NetFirewallRule -Name "OpenSSH-Server-In-TCP" -ErrorAction SilentlyContinue |
Select-Object Name, Enabled))
{
        Write-Output "Firewall Rule 'OpenSSH-Server-In-TCP' does not exist, creating it..."
        New-NetFirewallRule -Name 'OpenSSH-Server-In-TCP' -DisplayName 'OpenSSH Server
(sshd)' -Enabled True -Direction Inbound -Protocol TCP -Action Allow -LocalPort 22
} else {
         Write-Output "Firewall rule 'OpenSSH-Server-In-TCP' has been created and exists."
}
```

30. Try connecting to the SSH server either from outside or from inside the host. Please be aware that we did not configure the authentication agent for this exercise. The service does not have secure communications. From inside the host and under command prompt,

    ssh username@localhost

31. Paste a screenshot here to show that you can successfully login the OpenSSH server.



```
Microsoft Windows [Version 10.0.17763.4851]
(c) 2018 Microsoft Corporation. All rights reserved.

pace\vm81403n@F23-CYB631-07 D:\Users\vm81403n>
```

[Exercise IV: Host Scanning using Python-nMap]

32. In this exercise, we will use Python to communicate with nMap.
33. Download and install nMap for Windows.

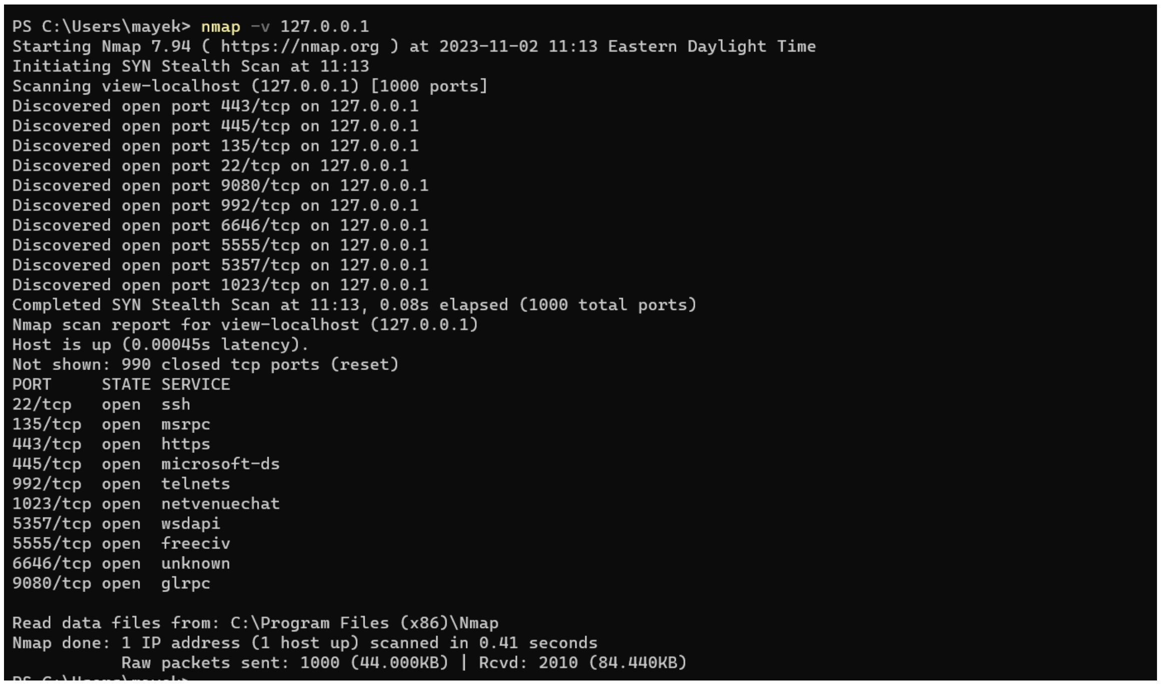
    https://nmap.org/dist/nmap-7.94-setup.exe

34. Try nMap under PS command line.

    nmap -v 127.0.0.1

35. This will show you results of a regular nMap scan. You should see SSH is running on TCP port 22 along with other Windows services.

    Paste your results here.



```
PS C:\Users\mayek> nmap -v 127.0.0.1
Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-02 11:13 Eastern Daylight Time
Initiating SYN Stealth Scan at 11:13
Scanning view-localhost (127.0.0.1) [1000 ports]
Discovered open port 443/tcp on 127.0.0.1
Discovered open port 445/tcp on 127.0.0.1
Discovered open port 135/tcp on 127.0.0.1
Discovered open port 22/tcp on 127.0.0.1
Discovered open port 9080/tcp on 127.0.0.1
Discovered open port 992/tcp on 127.0.0.1
Discovered open port 6646/tcp on 127.0.0.1
Discovered open port 5555/tcp on 127.0.0.1
Discovered open port 5357/tcp on 127.0.0.1
Discovered open port 1023/tcp on 127.0.0.1
Completed SYN Stealth Scan at 11:13, 0.08s elapsed (1000 total ports)
Nmap scan report for view-localhost (127.0.0.1)
Host is up (0.00045s latency).
Not shown: 990 closed tcp ports (reset)
PORT     STATE SERVICE
22/tcp   open  ssh
135/tcp  open  msrpc
443/tcp  open  https
445/tcp  open  microsoft-ds
992/tcp  open  telnets
1023/tcp open  netvenuechat
5357/tcp open  wsdapi
5555/tcp open  freeciv
6646/tcp open  unknown
9080/tcp open  glrpc

Read data files from: C:\Program Files (x86)\Nmap
Nmap done: 1 IP address (1 host up) scanned in 0.41 seconds
           Raw packets sent: 1000 (44.000KB) | Rcvd: 2010 (84.440KB)
```

Based on your screenshot above, explain the meaning of each service discovered by nmap on the host. For example, 22/tcp ssh means OpenSSH, etc.

22/tcp - Secure Shell (SSH) - Used for secure logins, file transfers (scp, sftp) and port forwarding.

135/tcp - Microsoft Remote Procedure Call (RPC) - Used for various Windows services, including file sharing, printing, and remote administration.

443/tcp - Hypertext Transfer Protocol Secure (HTTPS) - Used for secure web browsing and other encrypted web traffic.

445/tcp - Microsoft-DS - Used for Samba file sharing and other Windows networking protocols.

992/tcp - Telnet over SSL (TLS) - A secure version of the Telnet protocol for remote administration.

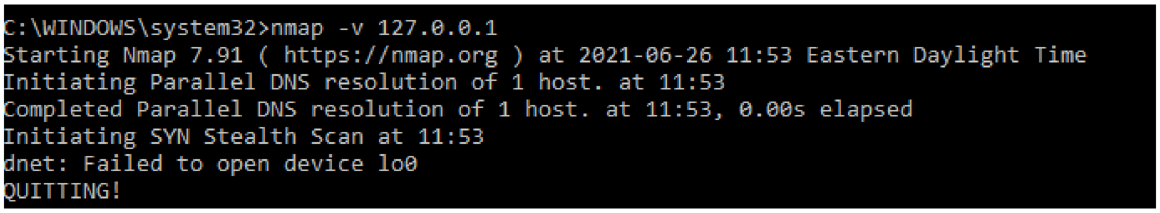
1023/tcp - Reserved - This port is reserved for use by system services.

5357/tcp - OpenStack Compute (Nova) - Used for managing virtual machines in an OpenStack cloud environment.

6646/tcp - Internet Relay Chat (IRC) - Used for real-time text communication over the internet.

9080/tcp - Squid Proxy Server - A popular web proxy server that can be used for caching web content and improving performance and security

36. **[Debugging]** Sometime, you might encounter problems with nMap scan due to Windows system update or installation of other software. For example, if you see the error below. It is often a result of lacking Npcap library which is needed for nMap. The problem can be solved by reinstalling nMap and replacing the existing Npcap library.



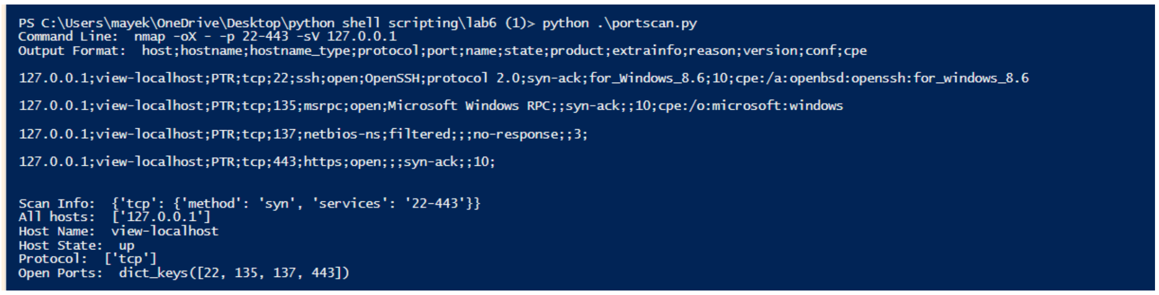
```
C:\WINDOWS\system32>nmap -v 127.0.0.1
Starting Nmap 7.91 ( https://nmap.org ) at 2021-06-26 11:53 Eastern Daylight Time
Initiating Parallel DNS resolution of 1 host. at 11:53
Completed Parallel DNS resolution of 1 host. at 11:53, 0.00s elapsed
Initiating SYN Stealth Scan at 11:53
dnet: Failed to open device lo0
QUITTING!
```

37. Install python-nmap API module.

    pip install python-nmap

38. Using Python IDLE, open **portscan.py**. Review the codes and run the file to understand what it does. It will take a few minutes to run the program since it is scanning multiple TCP ports. Your results should show the open ports on this machine which should include the OpenSSH server (port 22).

39. Paste your results here:



```
PS C:\Users\mayek\OneDrive\Desktop\python shell scripting\lab6 (1)> python .\portscan.py
Command Line:  nmap -oX - -p 22-443 -sV 127.0.0.1
Output Format:  host;hostname;hostname_type;protocol;port;name;state;product;extrainfo;reason;version;conf;cpe

127.0.0.1;view-localhost;PTR;tcp;22;ssh;open;OpenSSH;protocol 2.0;syn-ack;for_Windows_8.6;10;cpe:/a:openbsd:openssh:for_windows_8.6

127.0.0.1;view-localhost;PTR;tcp;135;msrpc;open;Microsoft Windows RPC;;syn-ack;;10;cpe:/o:microsoft:windows

127.0.0.1;view-localhost;PTR;tcp;137;netbios-ns;filtered;;;no-response;;3;

127.0.0.1;view-localhost;PTR;tcp;443;https;open;;;syn-ack;;10;


Scan Info:  {'tcp': {'method': 'syn', 'services': '22-443'}}
All hosts:  ['127.0.0.1']
Host Name:  view-localhost
Host State:  up
Protocol:  ['tcp']
Open Ports:  dict_keys([22, 135, 137, 443])
```

40. You are welcome to review the python-nmap project site at http://xael.org/pages/python-nmap-en.html

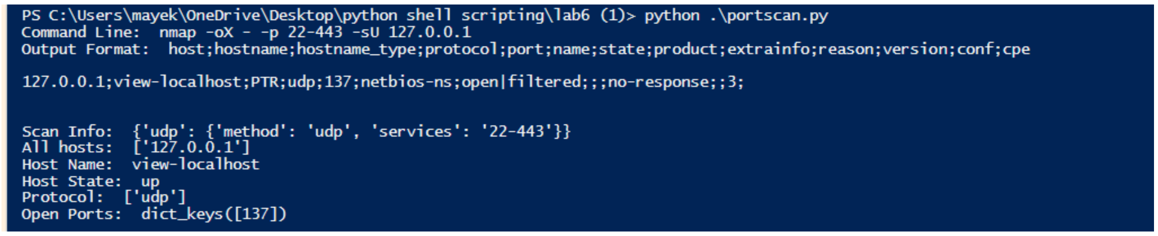
41. Here is a cheat sheet for nMap options that you can specify in the arguments in the nmap.scan function. https://github.com/jasonniebauer/Nmap-Cheatsheet

42. Revised **portscan.py** to conduct a UDP port scan via nMap.

43. Show your revised program here.

```
import nmap

nmScan = nmap.PortScanner()

nmScan.scan('127.0.0.1', '22-443','-sU')

print('Command Line: ',nmScan.command_line())
print('Output Format: ',nmScan.csv())

print('Scan Info: ',nmScan.scaninfo())

print('All hosts: ',nmScan.all_hosts())

print('Host Name: ',nmScan['127.0.0.1'].hostname())

print('Host State: ',nmScan['127.0.0.1'].state())

print('Protocol: ',nmScan['127.0.0.1'].all_protocols())


print('Open Ports: ',nmScan['127.0.0.1']['udp'].keys())
```

44. Show the results from your revised program.



```
PS C:\Users\mayek\OneDrive\Desktop\python shell scripting\lab6 (1)> python .\portscan.py
Command Line:  nmap -oX - -p 22-443 -sU 127.0.0.1
Output Format:  host;hostname;hostname_type;protocol;port;name;state;product;extrainfo;reason;version;conf;cpe

127.0.0.1;view-localhost;PTR;udp;137;netbios-ns;open|filtered;;;no-response;;3;


Scan Info:  {'udp': {'method': 'udp', 'services': '22-443'}}
All hosts:  ['127.0.0.1']
Host Name:  view-localhost
Host State:  up
Protocol:  ['udp']
Open Ports:  dict_keys([137])
```

[Exercise V: Python Security Tools]

45. Many developers have used Python to create security tools, such as https://github.com/dloss/python-pentest-tools). It is important that you develop the ability to follow instructions and install/test some of the tools that might be useful for you in conducting security tasks.

46. Pick one tool (excluding Scapy since we did it in the previous lab) from the list (https://github.com/dloss/python-pentest-tools) to try. Some of the tools are more complicated than the other. The simplest one is probably **dpkt** which is like Scapy). What is the tool that you pick? **Habu**

47. Show a Python program (either you wrote, or you use the one from the Github) or attach it as a separate submission to this lab.

from habu.lib.nmap import Nmap

```
from habu.lib.iface import Iface
import json

iface = Iface().get_working_iface()

nmap = Nmap()
result = nmap.ping('-n', '-v', '-v', '-v', 'scanme.nmap.org')  # Replace with your target

print(json.dumps(result, indent=4))
```

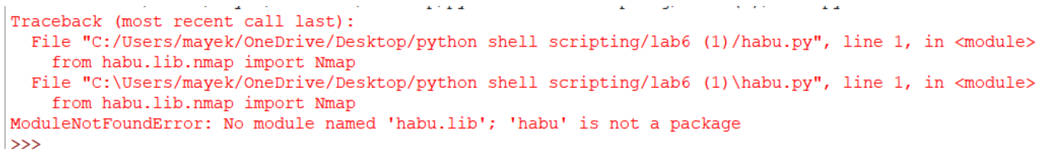
Did you write the program above or the program was included in Github? No

Explain what the problem does briefly:

The program solves the problem of performing a non-intrusive ping scan of a network host. This can be useful for a variety of purposes, such as checking if a host is up, identifying active hosts on a network, or troubleshooting network connectivity issues.

48. Open-source tools are typically not guaranteed for its quality. Some of experiments might not work as the project claims to be. Describe the experiments you tried.

I attempted to install and run a program, but Habu, a software package required to run the program, failed to install on my computer or in virtual machines. I received the same error message in both cases, indicating that there was a problem with the Habu installation packages.Paste screenshots of your results and explain them (either successfully or not successful results).



```
Traceback (most recent call last):
  File "C:/Users/mayek/OneDrive/Desktop/python shell scripting/lab6 (1)/habu.py", line 1, in <module>
    from habu.lib.nmap import Nmap
  File "C:\Users/mayek/OneDrive/Desktop/python shell scripting/lab6 (1)\habu.py", line 1, in <module>
    from habu.lib.nmap import Nmap
ModuleNotFoundError: No module named 'habu.lib'; 'habu' is not a package
>>>
```