Synopsis

Name: Lim Yu Liang

Admin No.: 2227452

Module: Programming in Security (PSEC)

Year: AY22/23 Semester 2

Assignment: CA1-2

Files

Legend:
Scripts - Blue
Others/Text Files - Red
Folders - Purple

Folders:
- ftpServerData
- ftpClientData

Client Files:
- ftpClientData-file.txt
- ftpClientData-file2.txt
- hello.txt

Server Files:
- ftpServerData-file.txt
- ftpServerData-file2.txt
- hi.txt

main.py
ftp_client.py
ftp_server.py
nmap_scanner.py
send_custom_packet.py

Preparation/Installation

A ‘requirements.txt’ file has already been created under the ‘scripts’ folder.

Installation of modules:

Use the ‘requirements.txt’ file to install required modules:
(Enter this in the terminal)

```
pip install -r requirements.txt
```

Once you have done the steps above, you are ready to run the scripts.
python <filename.py>

Preparation:

Open 2 terminals (1 for ftp_server script to run, 1 for main.py to run):

To start the FTP Server:
(Enter this in the terminal)

```
python ./ftp_server.py
```

To run the main program:
(Enter this in the other terminal)

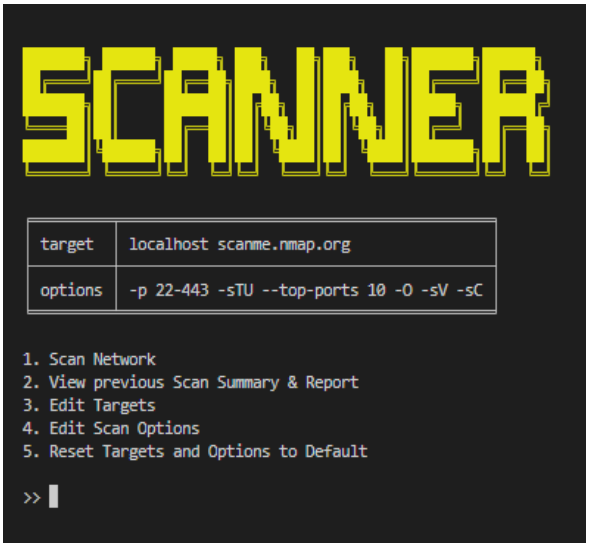
```
python ./main.py
```

Assignment Requirements

The following section will show that the program meets the basic requirements for the assignment as stated inside the brief with Screenshots

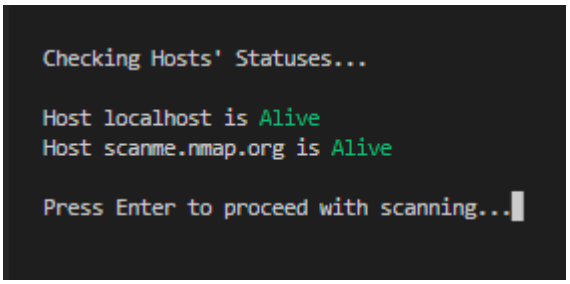
NMAP Port-Scanner

main.py -> nmap_scanner.py

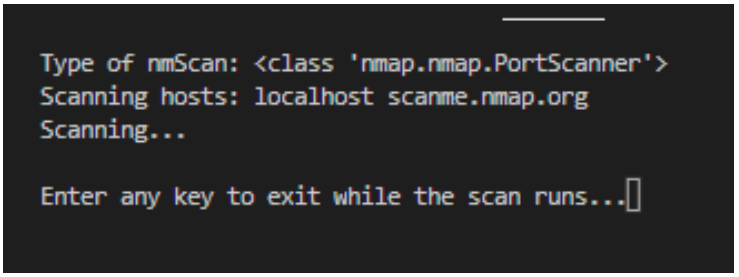


When Option 1 (Scan Network) is selected:

Checks if hosts are alive before Scanning



Scanning is done **Asynchronously**
(Press enter to exit and continue with other features while scan runs)



If user **exits** and re-chooses Option 1 (Scan Network)

```
Scan is still running...

Enter any key to exit while the scan runs...
```

Once the scan is complete, the Table will display as such:

5 Local host is shown only. Click here for Full-Scan Results.

Host	Hostname	Protocol	Port ID	State	Product	Extra Info	Reason	CPE
127.0.0.1	localhost	tcp	22	filtered	ssh		no-response	
127.0.0.1	localhost	tcp	23	filtered	telnet		no-response	
127.0.0.1	localhost	tcp	25	filtered	smtp		no-response	
127.0.0.1	localhost	tcp	53	filtered	domain		no-response	
127.0.0.1	localhost	tcp	80	filtered	http		no-response	

5 scanme.nmap.org is shown only. Click here for the Full-Scan Results.

45.33.32.156	scanme.nmap.org	tcp	143	filtered	imap		no-response	
45.33.32.156	scanme.nmap.org	tcp	443	filtered	https		no-response	
45.33.32.156	scanme.nmap.org	udp	53	closed	domain		port-unreach	
45.33.32.156	scanme.nmap.org	udp	67	closed	dhcps		port-unreach	
45.33.32.156	scanme.nmap.org	udp	68	open|filtered	dhcpc		no-response	

If hosts are dead, user will not be allowed to scan
(Pressing enter will bring user back to the Scan Menu)

```
Checking Hosts' Statuses...

Host 'localhost.com' is Dead

Hosts are either Dead or have blocked off incoming NMAP Scans.

Press Enter to continue...
```

FTP Server and Client

Start the server on another terminal

```
PS C:\PSEC-CA2\PSEC-CA2> python .\ftp_server.py
C:\Users\Yu Liang\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\LocalCache\local-packages\Python
310\site-packages\pyftpdlib\authorizers.py:243: RuntimeWarning: write permissions assigned to anonymous user.
  warnings.warn("write permissions assigned to anonymous user.",
[I 2023-02-05 23:08:58] concurrency model: async
[I 2023-02-05 23:08:58] masquerade (NAT) address: None
[I 2023-02-05 23:08:58] passive ports: None
[I 2023-02-05 23:08:58] >>> starting FTP server on 127.0.0.1:2121, pid=7380 <<<
```

FTP Client Menu
main.py -> **ftp_client.py**

```
Welcome to the FTP Client Menu!

1. Download File
2. Upload File
3. Quit

>>
```

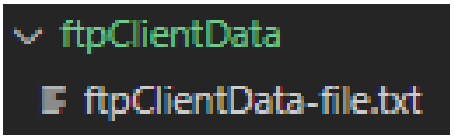
Option 1 (Download File) is selected:

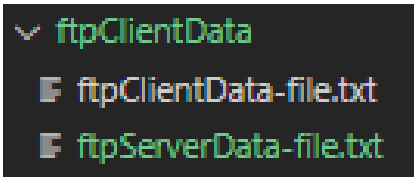
File to be **downloaded** is entered *(ftpServerData-file)*:

```
List of Files present in the FTP Server's Directory available for download:

1. ftpServerData-file.txt
2. hi.txt
3. hi2.txt

Enter the file name that you want to download (Allowed to Omit file extensions e.g. '.txt')

>> ftpServerData-file
```

File has been downloaded **./ftpServerData -> ./ftpClientData**

```
Successfully Downloaded file: ftpServerData-file.txt

Press Enter to continue...
```





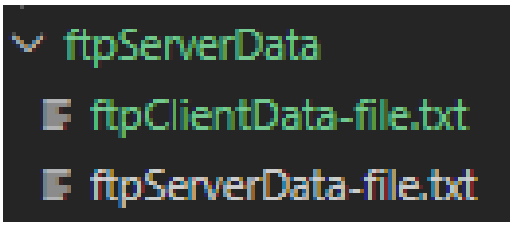
Option 2 (Upload File) is selected:

File to be **uploaded** is entered *(ftpClientData-file)*:

```
List of Files present in the FTP Client's Directory available for upload:

1. ftpClientData-file.txt
2. hello.txt
3. hello2.txt

Enter the file name that you want to upload (Allowed to Omit file extensions e.g. '.txt')

>> ftpClientData-file
```

File has been uploaded **./ftpClientData -> ./ftpServerData**

```
Successfully Uploaded file: ftpClientData-file.txt

Press Enter to continue...
```



Custom Packet Sender

main.py -> custom_packet_sender.py

```
************************
* Custom Packet        *
************************

1. Source Address:

2. Source Port:

3. Destination Address:

4. Destination Port:

5. Packet Type:

6. Packet Data:

Enter Source address of Packet (Accepts IP-Addr/www/http/https)
>> www.sp.com
```

Everything has been filled out:

5 packets sent

```
************************
* Custom Packet        *
************************

1. Source Address: www.sp.com

2. Source Port: 3000

3. Destination Address: www.pp.com

4. Destination Port: 5666

5. Packet Type: ICMP

6. Packet Data: DISM-DISM-DISM-DISM

How many Packets to send? (1-65535)
>> 5

Are you sure you want to send 5 packets to www.pp.com? [Y/N]
>> y

 5 packet(s) sent!

Press any key to continue...
```

Screenshot for Packets Sent:

ICMP:

No.	Time	Source	Destination	Protocol	Length	Info
49	7.064002	165.160.15.20	76.223.65.111	ICMP	61	Echo (ping) request id=0x0000, seq=0/0, ttl=64 (no response found!)
50	7.066083	165.160.15.20	76.223.65.111	ICMP	61	Echo (ping) request id=0x0000, seq=0/0, ttl=64 (no response found!)
51	7.073986	165.160.15.20	76.223.65.111	ICMP	61	Echo (ping) request id=0x0000, seq=0/0, ttl=64 (no response found!)
53	7.082200	165.160.15.20	76.223.65.111	ICMP	61	Echo (ping) request id=0x0000, seq=0/0, ttl=64 (no response found!)
54	7.087920	165.160.15.20	76.223.65.111	ICMP	61	Echo (ping) request id=0x0000, seq=0/0, ttl=64 (no response found!)

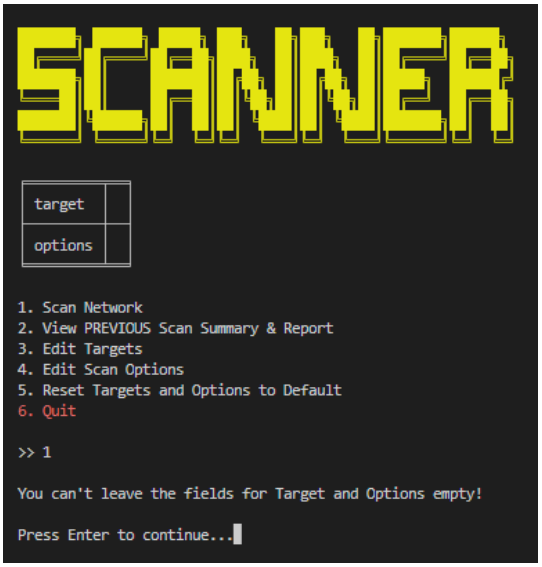
TCP:

No.	Time	Source	Destination	Protocol	Length	Info
17	2.964296	165.160.13.20	76.223.65.111	TCP	73	3000 → 5666 [SYN] Seq=0 Win=8192 Len=19
18	2.978399	165.160.13.20	76.223.65.111	TCP	73	[TCP Retransmission] [TCP Port numbers reused] 3000 → 5666 [SYN] Seq=0 Win=8192 Len=19
19	2.980281	165.160.13.20	76.223.65.111	TCP	73	[TCP Retransmission] [TCP Port numbers reused] 3000 → 5666 [SYN] Seq=0 Win=8192 Len=19
20	2.982881	165.160.13.20	76.223.65.111	TCP	73	[TCP Retransmission] [TCP Port numbers reused] 3000 → 5666 [SYN] Seq=0 Win=8192 Len=19
21	2.984652	165.160.13.20	76.223.65.111	TCP	73	[TCP Retransmission] [TCP Port numbers reused] 3000 → 5666 [SYN] Seq=0 Win=8192 Len=19

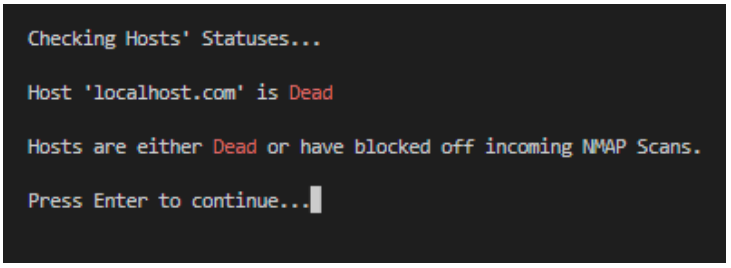
Comprehensive Tests

NMAP Scanner

Validates either empty Target/Options or Both

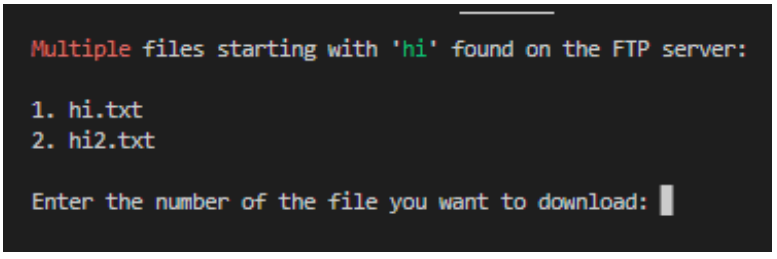


If target hosts are dead, unable to start scanning:

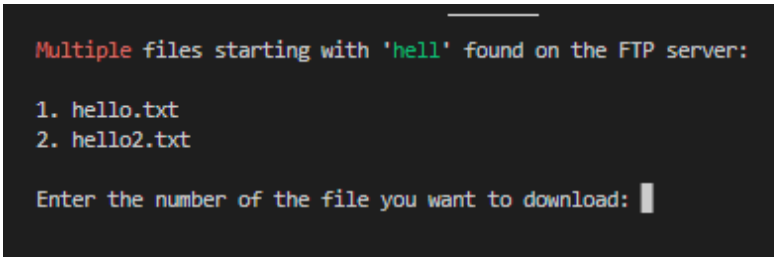


FTP Server/Client

If there are more than 1 files starting with the same name:



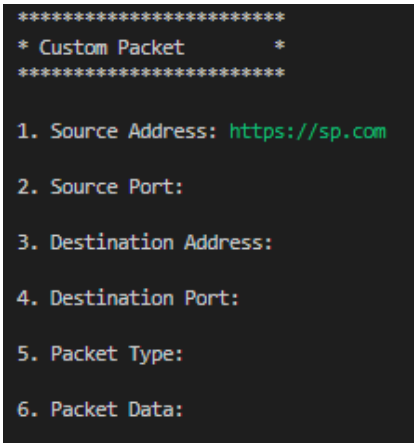
Even if user-input is 1 letter off or a few letters off, files with similar starting names will be picked and listed:



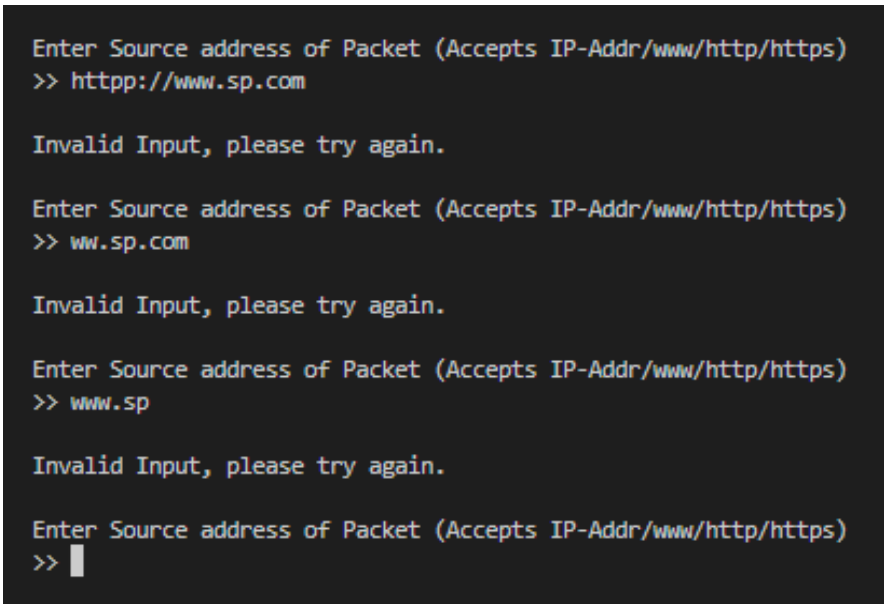
Custom Packet Sender

Addresses:

Accepts a Variety of URL types:



Anything that returns False from **Regex Validation** will not work:



(Destination Address uses the same Regex Validation)

Ports:

Only ports 1-65535 will be accepted as valid inputs:

```
2. Source Port: 3000

3. Destination Address: www.pp.com

4. Destination Port: 5666
```

Anything above the range of ports 1-65535 will not work:

```
Enter Source Port of Packet (1-65535)
>> 0

Please enter only ports from 1-65535!
```

```
Enter Source Port of Packet (1-65535)
>> -1

Please enter only ports from 1-65535!
```

```
Enter Source Port of Packet (1-65535)
>> 65536

Please enter only ports from 1-65535!
```

(Destination Port uses the same Regex Validation)

Packets:

Anything that is not T/U/I will not be accepted as valid inputs:

```
Enter Type (T) TCP, (U) UDP, (I) ICMP echo request (T/U/I)
>> z

Please enter only Types (T) TCP, (U) UDP, (I) ICMP!
```

Anything that is T/U/I will be accepted:
(Capitalisation does not matter. Program will Auto-caps and validate)

```
Enter Type (T) TCP, (U) UDP, (I) ICMP echo request (T/U/I)
>> t
```

Program checks if it is T/U/I and fills it in with their corresponding protocol names:
(TCP / UDP / ICMP)

```
5. Packet Type: TCP
```