

COMP 7003

Assignment 3

Report

Towa Quimbayo
A01086002
February 10th, 2025

Purpose	3
Requirements	3
Platforms	3
Language	3
Command Line Arguments	4
Findings	4
Testing Criteria	6
Tests	7
Test 1 (-h)	7
Test 2 (-x)	7
Test 3 (No target IP)	8
Test 4 (No port)	9
Test 5 (No filter for results)	9
Test 6 (Too many arguments)	10
Test 7 (Single target IP)	10
Test 8 (Range of target IPs)	11
Test 9 (Single port)	11
Test 10 (Range of ports)	12
Test 11 (Single filter)	12
Test 12 (Multiple filters)	13
Manual Service Identification	13
Identifying Host Types from Open Ports	14
Analysis of Each IP:Port	14
Determining the Type of Host for Each IP:Port	17

Purpose

The purpose of this document is to provide a comprehensive guide for setting up and running the Assignment 3 program which is a custom SYN scanner shell. To further elaborate, the program creates and sends custom TCP SYN packets using Python Scapy to a target IP(s) and then the program analyzes the response. If the response received a SYN-ACK then this indicates that the port is open, if an RST was received then the port is closed for the following target IP(s), and finally, if no response is received then it would indicate that the port is filtered which most likely means that a firewall blocks it. The document will explain the requirements, and configurations, describe the command line arguments, include sample commands and screenshots, and provide test cases.

Requirements

Task	Status
Target Host Identification: The scanner can handle single IPs, IP ranges, and subnets, automatically detecting the local subnet if no target is provided.	Fully implemented
Host Availability Check: Uses ARP requests to determine if a target host is online before attempting a scan. If a host is unreachable, it is skipped to optimize scanning.	Fully implemented
SYN Packet Crafting and Transmission: Constructs and sends TCP SYN packets to the specified target and ports using Scapy.	Fully implemented
Response Handling and Port Classification: Correctly interprets TCP SYN packet responses as Open, Closed or Filtered.	Fully implemented
Result Filtering: Allows users to filter output based on port status (open, closed, filtered) using the --show argument.	Fully implemented

Platforms

This program has been tested on:

- Ubuntu 2024.04 LTS

Language

- Python 3.12.3
- Compiles with GCC 13.3.0 on Linux

Command Line Arguments

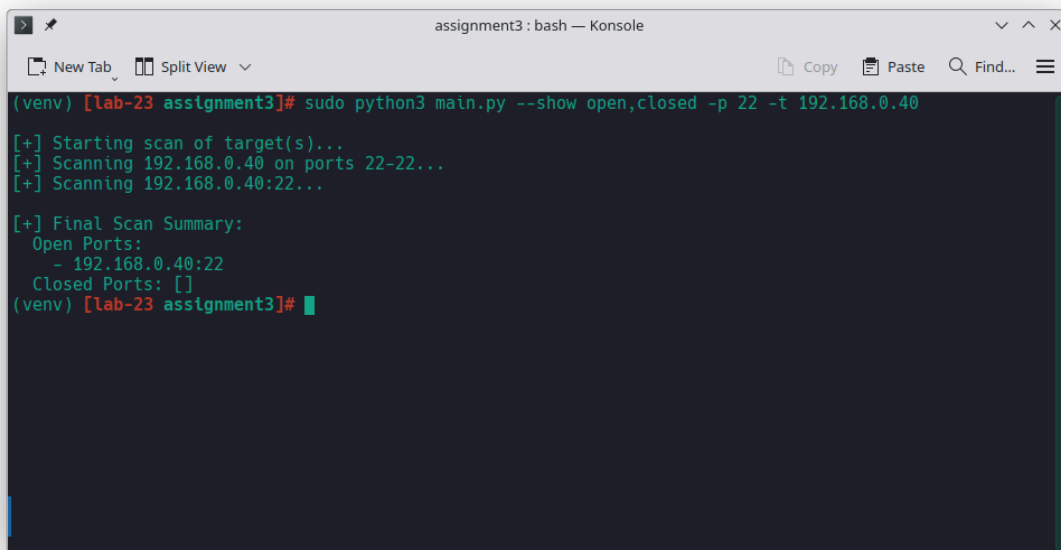
The program supports the following command line arguments:

Variable	Purpose
-t or --target	Specifies the target IP address, IP range, or subnet to scan and send the SYN packet to.
-p or --ports	Specifies the IP port(s) to scan on the target(s) and supports a single or a range of ports.
--show	Filters scan results based on port status and accept Open, Closed, and Filtered values.

Findings

1) The screenshot below is an example output after running the program for sending a SYN packet to a single target host IP and a single port, filtering results to display open and closed ports.

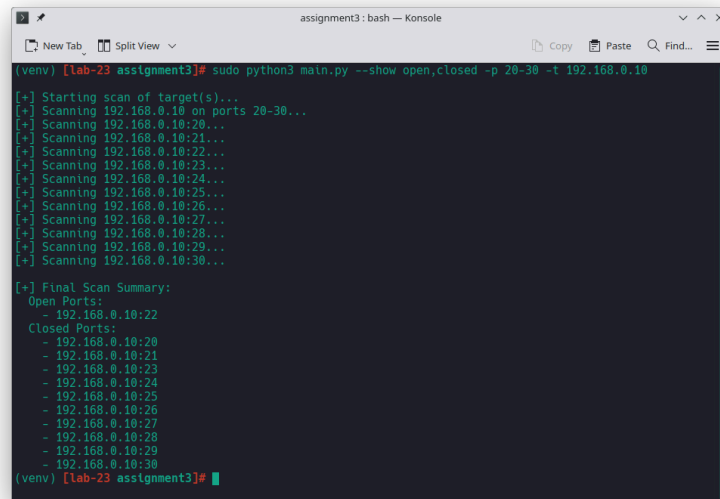
Command: `sudo python3 main.py --show open,closed -p 22 -t 192.168.0.40`

A screenshot of a terminal window titled "assignment3: bash — Konsole". The terminal shows the execution of a Python script. The prompt is "(venv) [lab-23 assignment3]#". The command entered is "sudo python3 main.py --show open,closed -p 22 -t 192.168.0.40". The output is as follows:

```
[+] Starting scan of target(s)...  
[+] Scanning 192.168.0.40 on ports 22-22...  
[+] Scanning 192.168.0.40:22...  
  
[+] Final Scan Summary:  
  Open Ports:  
    - 192.168.0.40:22  
  Closed Ports: []  
(venv) [lab-23 assignment3]#
```

2) The screenshot below is an example output after running the program for sending a SYN packet to a single target host IP and many ports, filtering results to display open and closed ports.

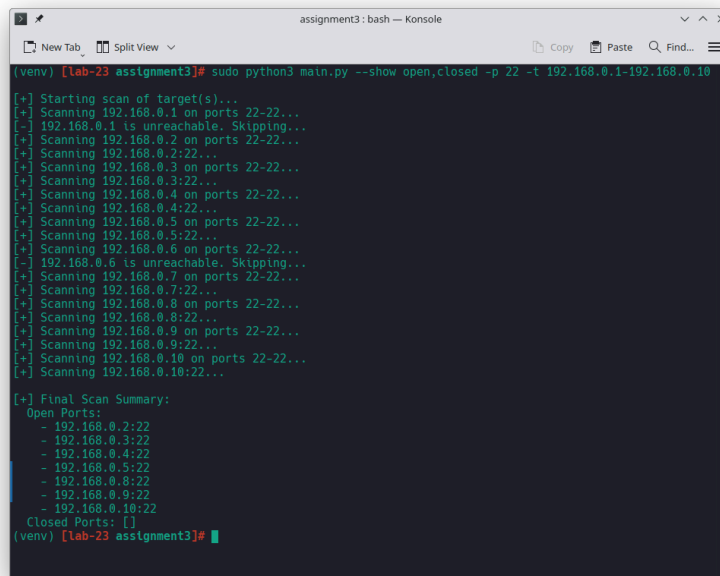
Command: `sudo python3 main.py --show open,closed -p 20-30 -t 192.168.0.10`

A terminal window titled 'assignment3: bash - Konsole' showing the output of a port scan command. The command is: `(venv) [Lab-23 assignment3]# sudo python3 main.py --show open,closed -p 20-30 -t 192.168.0.10`. The output shows a scan of target 192.168.0.10 on ports 20-30. It lists open ports (192.168.0.10:22) and closed ports (192.168.0.10:20, 192.168.0.10:21, 192.168.0.10:23, 192.168.0.10:24, 192.168.0.10:25, 192.168.0.10:26, 192.168.0.10:27, 192.168.0.10:28, 192.168.0.10:29, 192.168.0.10:30).

```
assignment3: bash - Konsole
[Lab-23 assignment3]# sudo python3 main.py --show open,closed -p 20-30 -t 192.168.0.10
[+] Starting scan of target(s)...
[+] Scanning 192.168.0.10 on ports 20-30...
[+] Scanning 192.168.0.10:20...
[+] Scanning 192.168.0.10:21...
[+] Scanning 192.168.0.10:22...
[+] Scanning 192.168.0.10:23...
[+] Scanning 192.168.0.10:24...
[+] Scanning 192.168.0.10:25...
[+] Scanning 192.168.0.10:26...
[+] Scanning 192.168.0.10:27...
[+] Scanning 192.168.0.10:28...
[+] Scanning 192.168.0.10:29...
[+] Scanning 192.168.0.10:30...
[+] Final Scan Summary:
Open Ports:
- 192.168.0.10:22
Closed Ports:
- 192.168.0.10:20
- 192.168.0.10:21
- 192.168.0.10:23
- 192.168.0.10:24
- 192.168.0.10:25
- 192.168.0.10:26
- 192.168.0.10:27
- 192.168.0.10:28
- 192.168.0.10:29
- 192.168.0.10:30
(venv) [Lab-23 assignment3]#
```

3) The screenshot below is an example output after running the program for sending a SYN packet to many target host IPs and a single port, filtering results to display open and closed ports.

Command: `sudo python3 main.py --show open,closed -p 22 -t 192.168.0.1-192.168.0.10`

A terminal window titled 'assignment3: bash - Konsole' showing the output of a port scan command. The command is: `(venv) [Lab-23 assignment3]# sudo python3 main.py --show open,closed -p 22 -t 192.168.0.1-192.168.0.10`. The output shows a scan of target 192.168.0.1 on ports 22-22. It lists open ports (192.168.0.2:22, 192.168.0.3:22, 192.168.0.4:22, 192.168.0.5:22, 192.168.0.8:22, 192.168.0.9:22, 192.168.0.10:22) and closed ports (192.168.0.1:22, 192.168.0.6:22, 192.168.0.7:22, 192.168.0.10:22).

```
assignment3: bash - Konsole
[Lab-23 assignment3]# sudo python3 main.py --show open,closed -p 22 -t 192.168.0.1-192.168.0.10
[+] Starting scan of target(s)...
[+] Scanning 192.168.0.1 on ports 22-22...
[+] Scanning 192.168.0.1 on ports 22-22...
[+] Scanning 192.168.0.2 on ports 22-22...
[+] Scanning 192.168.0.2:22...
[+] Scanning 192.168.0.3 on ports 22-22...
[+] Scanning 192.168.0.3:22...
[+] Scanning 192.168.0.4 on ports 22-22...
[+] Scanning 192.168.0.4:22...
[+] Scanning 192.168.0.5 on ports 22-22...
[+] Scanning 192.168.0.5:22...
[+] Scanning 192.168.0.6 on ports 22-22...
[+] Scanning 192.168.0.6 is unreachable. Skipping...
[+] Scanning 192.168.0.7 on ports 22-22...
[+] Scanning 192.168.0.7:22...
[+] Scanning 192.168.0.8 on ports 22-22...
[+] Scanning 192.168.0.8:22...
[+] Scanning 192.168.0.9 on ports 22-22...
[+] Scanning 192.168.0.9:22...
[+] Scanning 192.168.0.10 on ports 22-22...
[+] Scanning 192.168.0.10:22...
[+] Final Scan Summary:
Open Ports:
- 192.168.0.2:22
- 192.168.0.3:22
- 192.168.0.4:22
- 192.168.0.5:22
- 192.168.0.8:22
- 192.168.0.9:22
- 192.168.0.10:22
Closed Ports: []
(venv) [Lab-23 assignment3]#
```

4) The screenshot below is an example output after running the program for sending a SYN packet to many target host IPs and many ports, filtering results to display all open, closed, and filtered ports.

Command: `sudo python3 main.py -p 20-30 -t 192.168.0.1,192.168.0.10`

```
assignment3: bash — Konsole
New Tab Split View
Copy Paste Find...
(venv) [Lab-23 assignment3]# sudo python3 main.py -p 20-30 -t 192.168.0.1,192.168.0.10
[+] Starting scan of target(s)...
[+] Scanning 192.168.0.1 on ports 20-30...
[-] 192.168.0.1 is unreachable. Skipping...
[+] Scanning 192.168.0.10 on ports 20-30...
[+] Scanning 192.168.0.10:20...
[+] Scanning 192.168.0.10:21...
[+] Scanning 192.168.0.10:22...
[+] Scanning 192.168.0.10:23...
[+] Scanning 192.168.0.10:24...
[+] Scanning 192.168.0.10:25...
[+] Scanning 192.168.0.10:26...
[+] Scanning 192.168.0.10:27...
[+] Scanning 192.168.0.10:28...
[+] Scanning 192.168.0.10:29...
[+] Scanning 192.168.0.10:30...

[+] Final Scan Summary:
Open Ports:
- 192.168.0.10:22
Closed Ports:
- 192.168.0.10:20
- 192.168.0.10:21
- 192.168.0.10:23
- 192.168.0.10:24
- 192.168.0.10:25
- 192.168.0.10:26
- 192.168.0.10:27
- 192.168.0.10:28
- 192.168.0.10:29
- 192.168.0.10:30
Filtered Ports: []
(venv) [Lab-23 assignment3]#
```

Testing Criteria

Test	Expected	Actual	Screenshot
-h	pass	pass	Test 1
-x	fail	fail	Test 2
No target IP	pass	pass	Test 3
No port	pass	pass	Test 4
No filter for results	pass	pass	Test 5
Too many arguments	fail	fail	Test 6
Single target IP	pass	pass	Test 7
Range of target IPs	pass	pass	Test 8
Single port	pass	pass	Test 9
Range of ports	pass	pass	Test 10
Single filter	pass	pass	Test 11
Multiple filters	pass	pass	Test 12

Tests

Test 1 (-h)

```
towa@Towa-Laptop: ~/comp' X + v
towa@Towa-Laptop:~/comp7003/assignment3$ sudo python3 main.py -h
usage: main.py [-h] [-t TARGET] [-p PORTS] [--show SHOW]

SYN Scanner Shell for Students

options:
  -h, --help            show this help message and exit
  -t TARGET, --target TARGET
                        Target IP, range, or subnet
  -p PORTS, --ports PORTS
                        Port(s) to scan (e.g., 80,443,1-100)
  --show SHOW           Filter results: open, closed, filtered
towa@Towa-Laptop:~/comp7003/assignment3$ |
```

Test 2 (-x)

```
towa@Towa-Laptop: ~/comp' X + v
towa@Towa-Laptop:~/comp7003/assignment3$ sudo python3 main.py -x
usage: main.py [-h] [-t TARGET] [-p PORTS] [--show SHOW]
main.py: error: unrecognized arguments: -x
towa@Towa-Laptop:~/comp7003/assignment3$ |
```

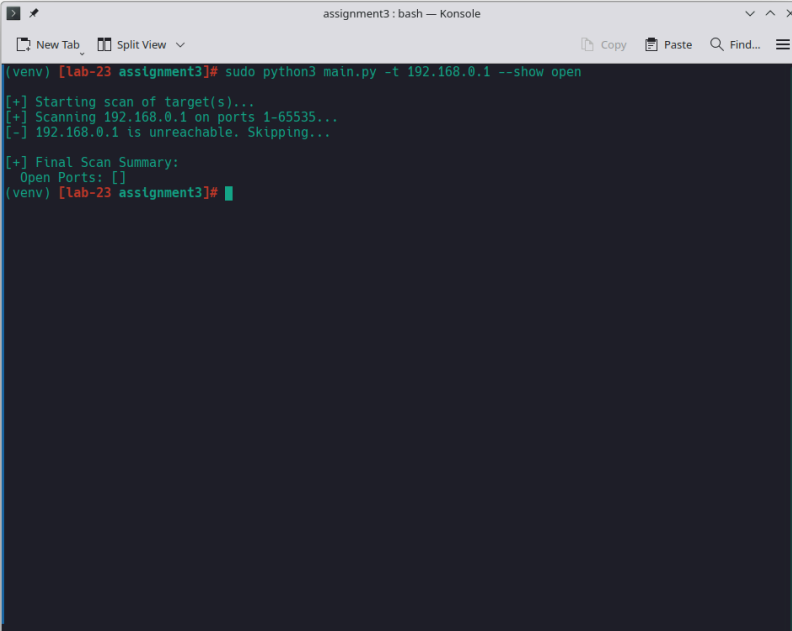
Test 3 (No target IP)

```
towa@Towa-Laptop: ~/comp' x + v - □ x
towa@Towa-Laptop:~/comp7003/assignment3$ sudo python3 main.py -p 20-30 --show open
[*] No target specified. Scanning local subnet: 172.28.221.0/24

[+] Starting scan of target(s)...
[+] Scanning 172.28.221.0 on ports 20-30...
[-] 172.28.221.0 is unreachable. Skipping...
[+] Scanning 172.28.221.1 on ports 20-30...
[-] 172.28.221.1 is unreachable. Skipping...
[+] Scanning 172.28.221.2 on ports 20-30...
[-] 172.28.221.2 is unreachable. Skipping...
[+] Scanning 172.28.221.3 on ports 20-30...
[-] 172.28.221.3 is unreachable. Skipping...
[+] Scanning 172.28.221.4 on ports 20-30...
[-] 172.28.221.4 is unreachable. Skipping...
[+] Scanning 172.28.221.5 on ports 20-30...
[+] Scanning 172.28.221.251 on ports 20-30...
[-] 172.28.221.251 is unreachable. Skipping...
[+] Scanning 172.28.221.252 on ports 20-30...
[-] 172.28.221.252 is unreachable. Skipping...
[+] Scanning 172.28.221.253 on ports 20-30...
[-] 172.28.221.253 is unreachable. Skipping...
[+] Scanning 172.28.221.254 on ports 20-30...
[-] 172.28.221.254 is unreachable. Skipping...
[+] Scanning 172.28.221.255 on ports 20-30...
[-] 172.28.221.255 is unreachable. Skipping...

[+] Final Scan Summary:
    Open Ports: []
    Closed Ports: []
towa@Towa-Laptop:~/comp7003/assignment3$
```

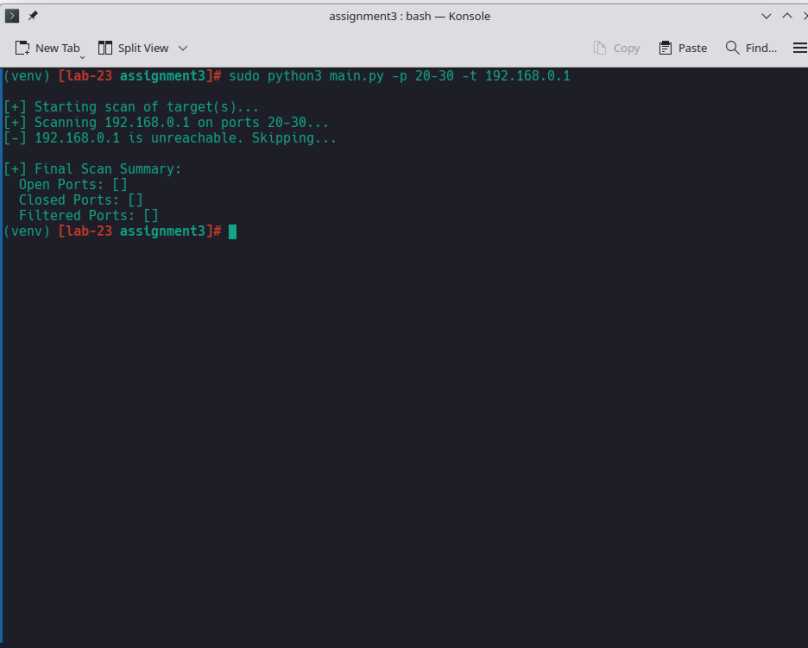

Test 4 (No port)



```
assignment3: bash — Konsole
New Tab Split View Copy Paste Find...
(venv) [lab-23 assignment3]# sudo python3 main.py -t 192.168.0.1 --show open
[+] Starting scan of target(s)...
[+] Scanning 192.168.0.1 on ports 1-65535...
[-] 192.168.0.1 is unreachable. Skipping...

[+] Final Scan Summary:
Open Ports: []
(venv) [lab-23 assignment3]#
```

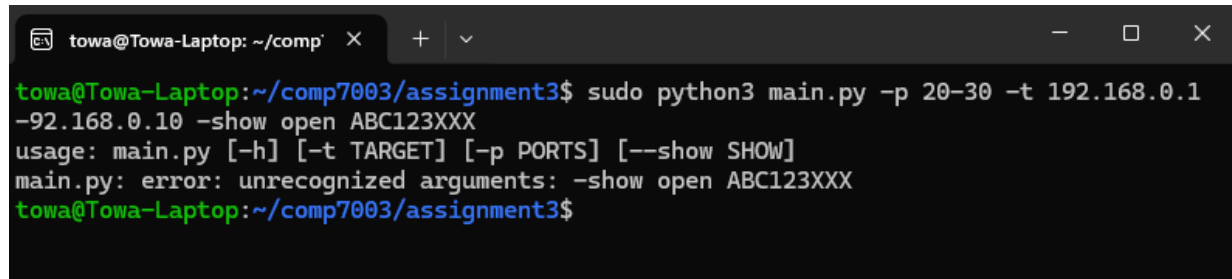
Test 5 (No filter for results)



```
assignment3: bash — Konsole
New Tab Split View Copy Paste Find...
(venv) [lab-23 assignment3]# sudo python3 main.py -p 20-30 -t 192.168.0.1
[+] Starting scan of target(s)...
[+] Scanning 192.168.0.1 on ports 20-30...
[-] 192.168.0.1 is unreachable. Skipping...

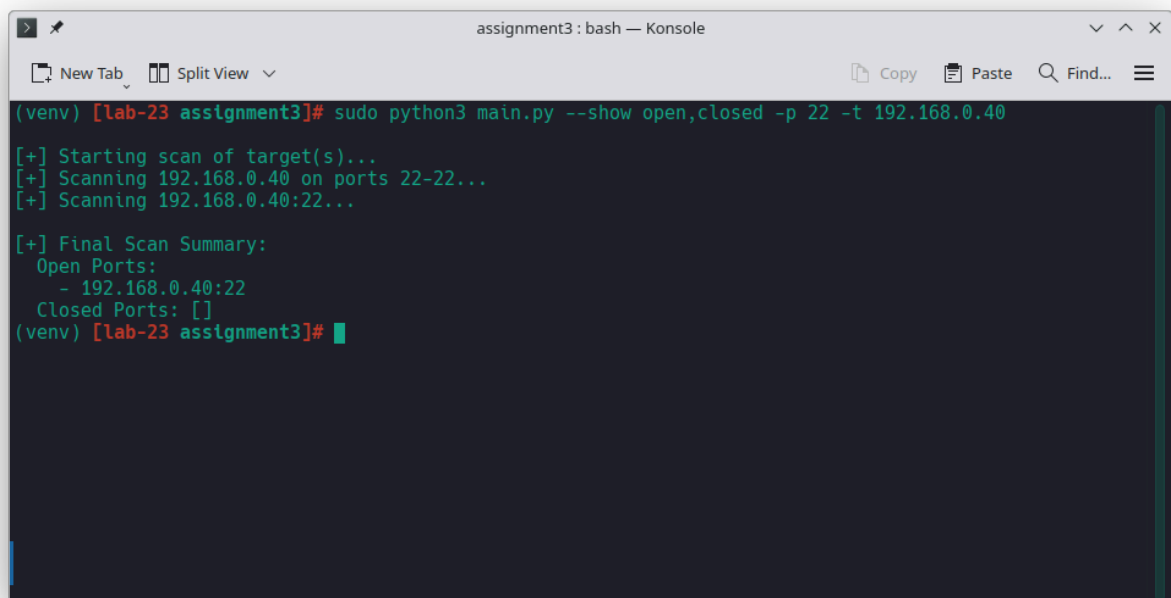
[+] Final Scan Summary:
Open Ports: []
Closed Ports: []
Filtered Ports: []
(venv) [lab-23 assignment3]#
```

Test 6 (Too many arguments)



```
towa@Towa-Laptop: ~/comp' x + v
towa@Towa-Laptop:~/comp7003/assignment3$ sudo python3 main.py -p 20-30 -t 192.168.0.1
-92.168.0.10 -show open ABC123XXX
usage: main.py [-h] [-t TARGET] [-p PORTS] [--show SHOW]
main.py: error: unrecognized arguments: -show open ABC123XXX
towa@Towa-Laptop:~/comp7003/assignment3$
```

Test 7 (Single target IP)



```
assignment3: bash — Konsole
New Tab Split View Copy Paste Find...
(venv) [lab-23 assignment3]# sudo python3 main.py --show open,closed -p 22 -t 192.168.0.40
[+] Starting scan of target(s)...
[+] Scanning 192.168.0.40 on ports 22-22...
[+] Scanning 192.168.0.40:22...

[+] Final Scan Summary:
  Open Ports:
    - 192.168.0.40:22
  Closed Ports: []
(venv) [lab-23 assignment3]#
```

Test 8 (Range of target IPs)

```
assignment3 : bash — Konsole
New Tab Split View Copy Paste Find...
(venv) [lab-23 assignment3]# sudo python3 main.py -p 22 -t 192.168.1.1-192.168.1.10
[+] Starting scan of target(s)...
[+] Scanning 192.168.1.1 on ports 22-22...
[-] 192.168.1.1 is unreachable. Skipping...
[+] Scanning 192.168.1.2 on ports 22-22...
[-] 192.168.1.2 is unreachable. Skipping...
[+] Scanning 192.168.1.3 on ports 22-22...
[-] 192.168.1.3 is unreachable. Skipping...
[+] Scanning 192.168.1.4 on ports 22-22...
[-] 192.168.1.4 is unreachable. Skipping...
[+] Scanning 192.168.1.5 on ports 22-22...
[-] 192.168.1.5 is unreachable. Skipping...
[+] Scanning 192.168.1.6 on ports 22-22...
[-] 192.168.1.6 is unreachable. Skipping...
[+] Scanning 192.168.1.7 on ports 22-22...
[-] 192.168.1.7 is unreachable. Skipping...
[+] Scanning 192.168.1.8 on ports 22-22...
[-] 192.168.1.8 is unreachable. Skipping...
[+] Scanning 192.168.1.9 on ports 22-22...
[-] 192.168.1.9 is unreachable. Skipping...
[+] Scanning 192.168.1.10 on ports 22-22...
[-] 192.168.1.10 is unreachable. Skipping...

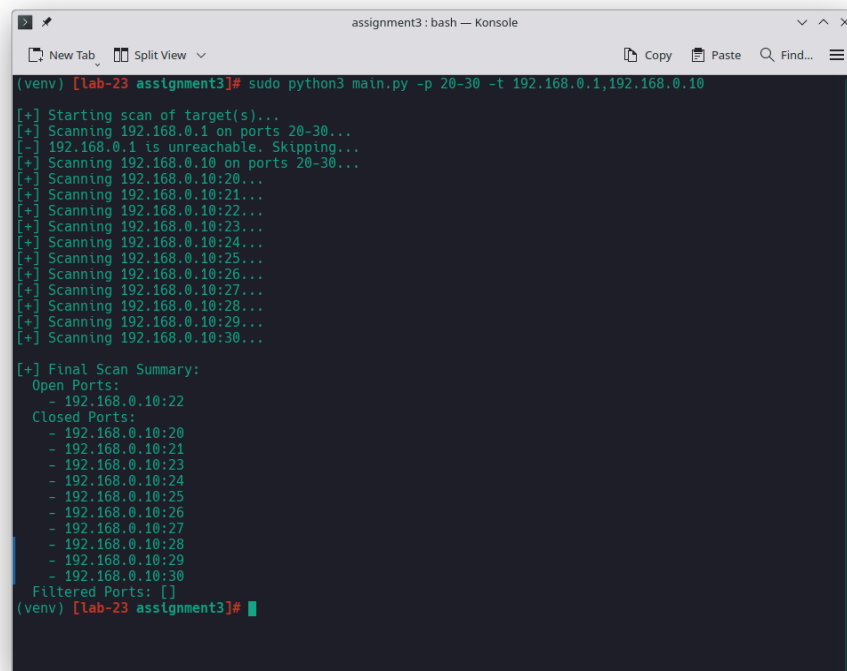
[+] Final Scan Summary:
Open Ports: []
Closed Ports: []
Filtered Ports: []
(venv) [lab-23 assignment3]#
```

Test 9 (Single port)

```
assignment3 : bash — Konsole
New Tab Split View Copy Paste Find...
(venv) [lab-23 assignment3]# sudo python3 main.py --show open,closed -p 22 -t 192.168.0.1-192.168.0.10
[+] Starting scan of target(s)...
[+] Scanning 192.168.0.1 on ports 22-22...
[-] 192.168.0.1 is unreachable. Skipping...
[+] Scanning 192.168.0.2 on ports 22-22...
[+] Scanning 192.168.0.2:22...
[+] Scanning 192.168.0.3 on ports 22-22...
[+] Scanning 192.168.0.3:22...
[+] Scanning 192.168.0.4 on ports 22-22...
[+] Scanning 192.168.0.4:22...
[+] Scanning 192.168.0.5 on ports 22-22...
[+] Scanning 192.168.0.5:22...
[+] Scanning 192.168.0.6 on ports 22-22...
[-] 192.168.0.6 is unreachable. Skipping...
[+] Scanning 192.168.0.7 on ports 22-22...
[+] Scanning 192.168.0.7:22...
[+] Scanning 192.168.0.8 on ports 22-22...
[+] Scanning 192.168.0.8:22...
[+] Scanning 192.168.0.9 on ports 22-22...
[+] Scanning 192.168.0.9:22...
[+] Scanning 192.168.0.10 on ports 22-22...
[+] Scanning 192.168.0.10:22...

[+] Final Scan Summary:
Open Ports:
- 192.168.0.2:22
- 192.168.0.3:22
- 192.168.0.4:22
- 192.168.0.5:22
- 192.168.0.8:22
- 192.168.0.9:22
- 192.168.0.10:22
Closed Ports: []
(venv) [lab-23 assignment3]#
```

Test 10 (Range of ports)



```
assignment3 : bash — Konsole
New Tab Split View Copy Paste Find...
(venv) [tab-23 assignment3]# sudo python3 main.py -p 20-30 -t 192.168.0.1,192.168.0.10
[+] Starting scan of target(s)...
[+] Scanning 192.168.0.1 on ports 20-30...
[-] 192.168.0.1 is unreachable. Skipping...
[+] Scanning 192.168.0.10 on ports 20-30...
[+] Scanning 192.168.0.10:20...
[+] Scanning 192.168.0.10:21...
[+] Scanning 192.168.0.10:22...
[+] Scanning 192.168.0.10:23...
[+] Scanning 192.168.0.10:24...
[+] Scanning 192.168.0.10:25...
[+] Scanning 192.168.0.10:26...
[+] Scanning 192.168.0.10:27...
[+] Scanning 192.168.0.10:28...
[+] Scanning 192.168.0.10:29...
[+] Scanning 192.168.0.10:30...

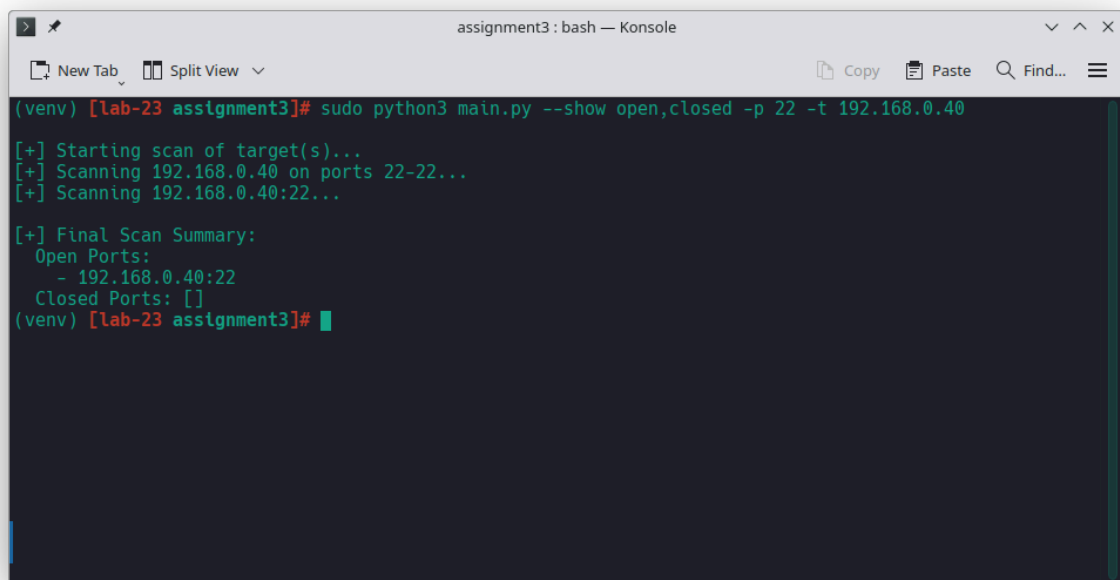
[+] Final Scan Summary:
Open Ports:
- 192.168.0.10:22
Closed Ports:
- 192.168.0.10:20
- 192.168.0.10:21
- 192.168.0.10:23
- 192.168.0.10:24
- 192.168.0.10:25
- 192.168.0.10:26
- 192.168.0.10:27
- 192.168.0.10:28
- 192.168.0.10:29
- 192.168.0.10:30
Filtered Ports: []
(venv) [tab-23 assignment3]#
```

Test 11 (Single filter)

```
towa@Towa-Laptop:~/comp7003/assignment3$ sudo python3 main.py -t 172.28.208.1 -p 80 --show open
[+] Starting scan of target(s)...
[+] Scanning 172.28.208.1 on ports 80-80...
[+] Scanning 172.28.208.1:80...

[+] Final Scan Summary:
Open Ports:
- 172.28.208.1:80
towa@Towa-Laptop:~/comp7003/assignment3$
```

Test 12 (Multiple filters)



```
assignment3: bash — Konsole
New Tab Split View
Copy Paste Find...
(venv) [lab-23 assignment3]# sudo python3 main.py --show open,closed -p 22 -t 192.168.0.40
[+] Starting scan of target(s)...
[+] Scanning 192.168.0.40 on ports 22-22...
[+] Scanning 192.168.0.40:22...

[+] Final Scan Summary:
  Open Ports:
    - 192.168.0.40:22
  Closed Ports: []
(venv) [lab-23 assignment3]#
```

Manual Service Identification

Application 1: 192.168.0.10:22

Host	192.168.0.10
Port	22
Process Name	sshd
PID	252
What does the program do?	SSHD, the Solid-State Hybrid Drive (OpenSSH Daemon) is a service that listens for incoming SSH connections on a server, handles user authentication, and encryption, and facilitates secure remote access to the system using SSH.
Why might it be running?	The device at this IP is an Arch Linux server from the lab computers where SSH is enabled for remote administration.
Potential security risks	Brute-force or MITM attacks if a user has weak passwords by credential sniffing through accessing the SSH over an insecure network. Unauthorized access vulnerability if an attacker gains access through the root login credentials.

Application 2: 172.28.208.1:80

Host	172.28.208.1
Port	80
Process Name	nginx
PID	481
What does the program do?	This IP is associated with a web server via nginx that hosts web content and serves as the communication gateway for HTTP requests over the network.
Why might it be running?	The IP is likely associated with a router device or network gateway / internal web service that provides a configuration panel or a hosted application.
Potential security risks	Any data that is transmitted over HTTP is unencrypted which makes it highly vulnerable to MITM attacks and may also allow unauthorized access if this is a router or web interface.

Identifying Host Types from Open Ports

Analysis of Each IP:Port

IP:Port	Common Service	Typical Device Type	Potential Security Risks
192.168.0.1:21	FTP (File Transfer Protocol)	Router, NAS, FTP Server	Unencrypted, vulnerable to sniffing and brute force attacks.
192.168.0.1:53	DNS (Domain Name System)	Router, DNS Server	Can be exploited for DNS poisoning and DDoS attacks.
192.168.0.1:1900	SSDP (Simple Service Discovery Protocol) / UPnP (Universal Plug and Play)	Smart Devices, IoT, Home Routers	Vulnerable to DDoS amplification and UPnP exploits which allows unauthorized access.
192.168.0.1:8200	DLNA Media Server	Smart TV, Media Server, NAS	May leak media files or expose unauthorized access if publicly accessible.
192.168.0.1:20001	Custom Service (not associated with a widely recognized standard service)	Proprietary IoT, Router Service	Possible backdoor or remote management service which may reveal unknown vulnerabilities.
192.168.0.2:23	Telnet	Old Network Devices,	Unencrypted and it sends

		Routers, Embedded Systems, Switch, IoT	data in cleartext, making it highly vulnerable to MITM attacks and credential sniffing.
192.168.0.2:80	HTTP (Web Server)	Router, Web Server	It transmits data in plain text, making it highly vulnerable to eavesdropping by malicious actors.
192.168.0.2:443	HTTPS (Secure Web Server)	Router, Web Server	Considered more secure as it encrypts data, but can still be susceptible to attacks like cross-site scripting, SQL injections, and DDoS attacks if not properly secured.
192.168.0.2:40001	Custom Service (not associated with a widely recognized standard service)	Proprietary IoT, Router Service	Likely an undocumented or proprietary service. Could be used for proprietary remote management.
192.168.0.2:40002	Custom Service (not associated with a widely recognized standard service)	Proprietary IoT, Router Service	Likely an undocumented or proprietary service. Could be used for proprietary remote management.
192.168.0.3:23	Telnet	Old Network Devices, Routers, Embedded Systems, Switch, IoT	Unencrypted and it sends data in cleartext, making it highly vulnerable to MITM attacks and credential sniffing.
192.168.0.3:80	HTTP (Web Server)	Router, Web Server	It transmits data in plain text, making it highly vulnerable to eavesdropping by malicious actors.
192.168.0.3:443	HTTPS (Secure Web Server)	Router, Web Server	Considered more secure as it encrypts data, but can still be susceptible to attacks like cross-site scripting, SQL injections, and DDoS attacks if not properly secured.
192.168.0.3:40001	Custom Service (not associated with a widely recognized standard service)	Proprietary IoT, Router Service	Likely an undocumented or proprietary service. Could be used for proprietary remote

			management.
192.168.0.3:40002	Custom Service (not associated with a widely recognized standard service)	Proprietary IoT, Router Service	Likely an undocumented or proprietary service. Could be used for proprietary remote management.
192.168.0.40:22	SSH (Secure Shell)	Linux/Unix Server, Raspberry Pi	Secure by design if key-based authentication is used. Otherwise, risk of brute force attacks.
192.168.0.200:853	DNS over TLS (DoT)	Router, DNS Server	Secure DNS, but could be misconfigured which can expose DNS logs and potentially leak data.
192.168.0.200:49152	Dynamic / Ephemeral Port	Router, Smart Device (any device or application needing a temporary connection to communicate with a server)	Can be vulnerable to port scanning, port exhaustion, and other risks.
192.168.0.200:62078	Sync Port for Apple Lockdown Protocol	iPhone, iPad, Mac	Used for device pairing with iTunes. Could allow unauthorized device access.
192.168.0.203:853	DNS over TLS (DoT)	Router, DNS Server	Secure DNS, but could be misconfigured which can expose DNS logs and potentially leak data.
192.168.0.203:5000	UPnP / Web Server	NAS, Synology, Media Server	Often used for web control panels or APIs. Could expose admin login credentials.
192.168.0.203:7000	Web Services	NAS, IoT, Home Assistant	Could be used for remote management or streaming services.
192.168.0.203:7100	Streaming Services	NAS, DLNA, Home Server	Used for media streaming, possibly exposing media files.
192.168.0.203:49152	UPnP (Universal Plug and Play)	Smart Home, IoT	UPnP services can be exploited for remote attacks.
192.168.0.203:49159	UPnP (Universal Plug and Play)	Smart Home, IoT	Similar to 49152, could be part of home automation or security

			systems.
192.168.0.203:61029	Dynamic / Ephemeral Port:	IoT (not assigned to a specific device)	Likely used by an application for temporary or outbound connection and communication.
192.168.0.203:62078	Sync Port for Apple Lockdown Protocol	iPhone, iPad, Mac	Used for device pairing with iTunes. Could allow unauthorized device access.

Determining the Type of Host for Each IP:Port

IP:Port	Likely Device Type	Reasoning
192.168.0.1	Router / Gateway	This IP is associated with FTP (21), DNS (53), SSDP (1900), and UPnP (8200) which are typical for home routers.
192.168.0.2	IoT Device / Web Server	This IP is associated with Telnet (23), HTTP (80, 443), and custom ports (40001, 40002) which suggests an embedded device or smart home product.
192.168.0.3	Router or IoT Device	Similar to 192.168.0.2, but more open ports suggest it might be a managed network device.
192.168.0.40	Linux Server / Raspberry Pi	SSH (22) is a common indicator of a Linux-based system.
192.168.0.200	Apple Device (Mac, iPhone, or iPad)	Port 62078 suggests Apple's Lockdown Protocol which is used for device pairing.
192.168.0.203	NAS / Media Server	This IP is associated with Port 5000 (Synology, web control), 7000-7100 (streaming), and UPnP (49152, 49159) which suggests a networked storage or smart home hub.