COMP 7003
Assignment 2
Testing

Parth Chaturvedi
A01256537
09/29/2025

Test Cases

Test #	Command	Description	Expected Result	Traffic Generation
1	`python3 main.py --help`	Display help message	Shows all arguments and usage	N/A
2	`python3 main.py -i any -c 1 -f arp`	Capture ARP packet	Displays parsed ARP header	`sudo arping -c 1 192.168.1.1`
3	`python3 main.py -i any -c 1 -f icmp`	Capture ICMP packet	Displays Ethernet→IPv4→ICMP	`ping -c 1 8.8.8.8`
4	`python3 main.py -i any -c 1 -f udp`	Capture UDP packet	Displays Ethernet→IPv4→UDP	`nslookup google.com`
5	`python3 main.py -i any -c 1 -f tcp`	Capture TCP packet	Displays Ethernet→IPv4→TCP	`curl http://google.com`
6	`python3 main.py -i any -c 1 -f "udp and port 53"`	Capture DNS traffic	Displays UDP→DNS headers	`nslookup google.com 8.8.8.8`
7	`python3 main.py -i any -c 1 -f ip`	Capture IPv4 packet	Displays Ethernet→ IPv4 header	`curl http://google.com`

8	`python3 main.py -i any -c 1 -f ip6`	Capture IPv6 packet	Displays Ethernet→IPv6 header	Automatic (mDNS traffic)
9	`python3 main.py -i any -c 1 -f icmp6`	Capture ICMPv6 packet	Displays IPv6→ICMPv6	`ping6 -c 1 ::1`
10	`python3 main.py -i any -c 3`	Capture 3 packets (no filter)	Captures any 3 packets	Browse web/generate traffic
11	`python3 main.py -i any`	Missing required argument	Error: -c required	N/A

Detailed Test Results

Test 1: Help Message

Command: `python3 main.py --help`
Expected: Display usage and all arguments
Result: Pass - Shows all options correctly

```
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- % python3 main.py -h
usage: main.py [-h] -i INTERFACE [-f FILTER] -c COUNT

Packet sniffer using Scapy with manual HEX parsing

optional arguments:
  -h, --help            show this help message and exit
  -i INTERFACE, --interface INTERFACE
                        The interface to capture packets on (e.g., eth0, wlan0, any)
  -f FILTER, --filter FILTER
                        BPF filter to apply (e.g., 'tcp and port 80'). If not provided, captures all packets.
  -c COUNT, --count COUNT
                        Number of packets to capture (default: 1)
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- %
```

Test 2: ARP Capture

Command: `python3 main.py -i any -c 1 -f arp`
Expected: Ethernet header with EtherType 0806, complete ARP header with sender/target

MAC and IP
Result: Pass - All ARP fields parsed correctly

```
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- % python3 main.py -c 1 -i any -f arp
Available interfaces: ['lo0', 'gif0', 'stf0', 'en4', 'ap1', 'en0', 'awdl0', 'llw0', 'en2', 'en1', 'bridge0', 'utun0', 'utun1', 'utun2', 'utun3']
Starting packet capture on en0
Starting packet capture on en0 with filter: arp

Captured Packet 1:
Ethernet Header:
  Destination MAC:          a483e75dc46c          | a4:83:e7:5d:c4:6c
  Source MAC:               8c19b53ae74e          | 8c:19:b5:3a:e7:4e
  EtherType:                0806                  | 2054
ARP Header:
  Hardware Type:            0001                  | 1
  Protocol Type:            0800                  | 2048
  Hardware Size:            06                    | 6
  Protocol Size:            04                    | 4
  Operation:                0001                  | 1
  Sender MAC:               8c19b53ae74e          | 8c:19:b5:3a:e7:4e
  Sender IP:                c0a801fe              | 192.168.1.254
  Target MAC:               000000000000          | 00:00:00:00:00:00
  Target IP:                c0a80152              | 192.168.1.82

Packet capture completed on en0.
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- %
```

Test 3: ICMP Capture

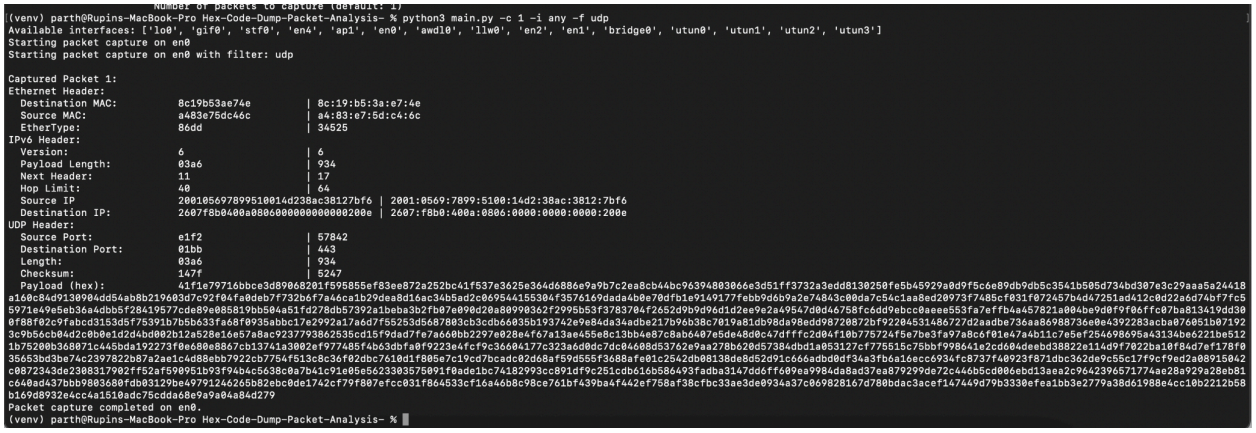
Command: `python3 main.py -i any -c 1 -f icmp`
Expected: IPv4 protocol field = 1, ICMP type 8 (request) or 0 (reply)
Result: Pass - Complete protocol chain parsed

```
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- % python3 main.py -c 1 -i any -f icmp
Available interfaces: ['lo0', 'gif0', 'stf0', 'en4', 'ap1', 'en0', 'awdl0', 'llw0', 'en2', 'en1', 'bridge0', 'utun0', 'utun1', 'utun2', 'utun3']
Starting packet capture on en0
Starting packet capture on en0 with filter: icmp

Captured Packet 1:
Ethernet Header:
  Destination MAC:          8c19b53ae74e          | 8c:19:b5:3a:e7:4e
  Source MAC:               a483e75dc46c          | a4:83:e7:5d:c4:6c
  EtherType:                0800                  | 2048
IPv4 Header:
  Version:                  4                     | 4
  Header Length:            5                     | 20
  Total Length:             0054                  | 84
  Flags & Frag Offset:      0000                  | 0
    Reserved:  0
    DF (Do not Fragment): 0
    MF (More Fragments): 0
    Fragment Offset: 0x0 | 0
  Protocol:                 01                    | 1
  Source IP:                c0a80152              | 192.168.1.82
  Destination IP:           08080808              | 8.8.8.8
ICMP Header:
  Type:                     08                    | 8
  Code:                     00                    | 0
  Checksum:                 032a                  | 810
  Payload (hex):            db08000068db165f000eaf8108090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f202122232425262728292a2b2c2d2e2f3031323334353637
Packet capture completed on en0.
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- %
```

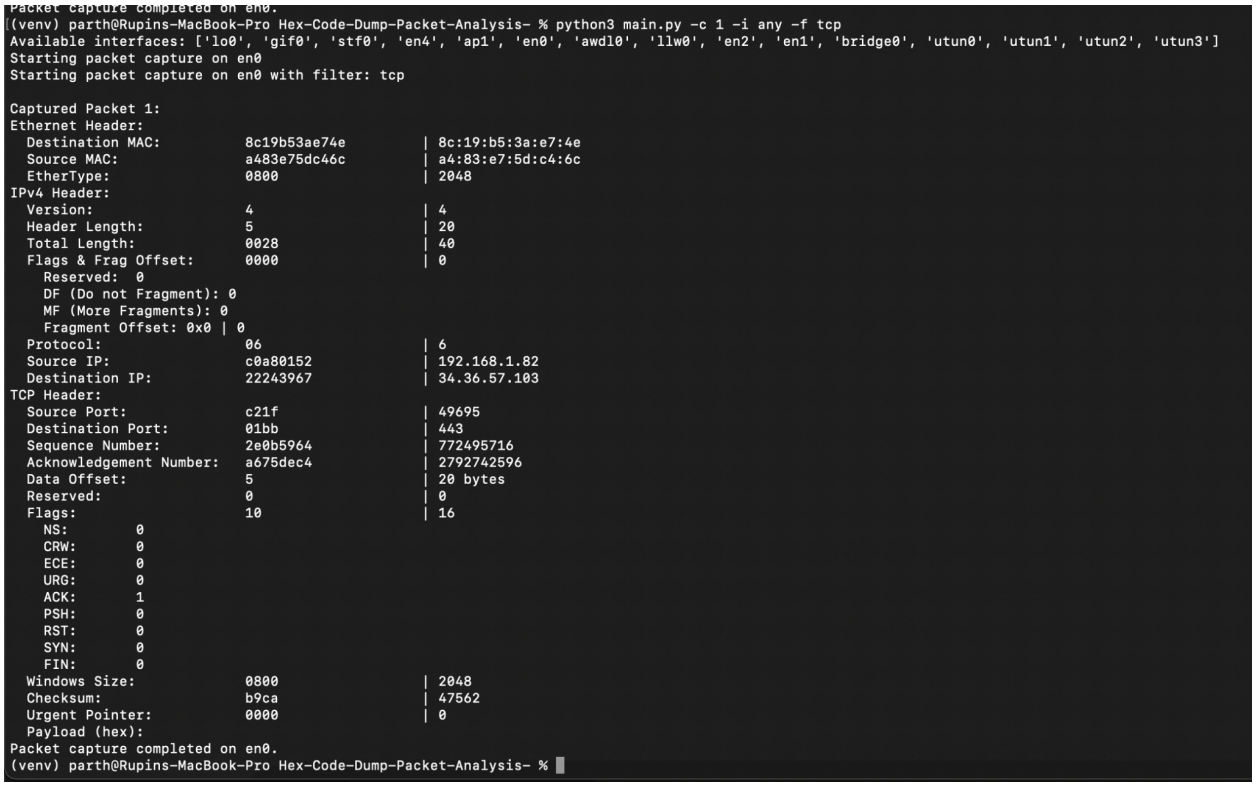
Test 4: UDP Capture

Command: `python3 main.py -i any -c 1 -f udp`
Expected: IPv4 protocol = 17, UDP ports, payload
Result: Pass - UDP header displayed correctly



Test 5: TCP Capture

Command: `python3 main.py -i any -c 1 -f tcp`
Expected: IPv4 protocol = 6, TCP flags (SYN/ACK), ports
Result: Pass - All TCP fields and flags correct



Test 6: DNS Capture

Command: `python3 main.py -i any -c 1 -f "udp and port 53"`
Expected: UDP port 53, DNS header with transaction ID, flags, question count
Result: Pass - DNS query and response both parsed

```
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- % python3 main.py -c 1 -i any -f "udp and port 53"
Available interfaces: ['lo0', 'gif0', 'stf0', 'en4', 'ap1', 'en0', 'awdl0', 'llw0', 'en2', 'en1', 'bridge0', 'utun0', 'utun1', 'utun2', 'utun3']
Starting packet capture on en0
Starting packet capture on en0 with filter: udp and port 53

Captured Packet 1:
Ethernet Header:
  Destination MAC:          8c19b53ae74e          | 8c:19:b5:3a:e7:4e
  Source MAC:               a483e75dc46c          | a4:83:e7:5d:c4:6c
  EtherType:                86dd                  | 34525
IPv6 Header:
  Version:                  6                     | 6
  Payload Length:           0034                  | 52
  Next Header:              11                    | 17
  Hop Limit:                40                    | 64
  Source IP                 20010569789951001 4d238ac38127bf6 | 2001:0569:7899:5100:14d2:38ac:3812:7bf6
  Destination IP:           20010568ff09010c0000000000000068 | 2001:0568:ff09:010c:0000:0000:0000:0068
UDP Header:
  Source Port:              cb59                  | 52057
  Destination Port:         0035                  | 53
  Length:                   0034                  | 52
  Checksum:                 8be4                  | 35812
  Payload (hex):            72b30100000100000000000d636f6e6669677572617469 6e026c73056170706c6503636f6d0000410001
DNS Header:
  Transaction ID:           72b3                  | 29363
  Flags:                    0100                  | 256
    QR (Query/Response): 0
    Opcode:    0
    AA (Auth Answer): 0
    TC (Truncated): 0
    RD (Recursion Des): 1
    RA (Recursion Avail): 0
    Response Code: 0
  Questions:                0001                  | 1
  Answer RRs:               0000                  | 0
  Authority RRs:            0000                  | 0
  Additional RRs:           0000                  | 0
  Payload (hex):            0d636f6e6669677572617469 6e026c73056170706c6503636f6d0000410001
Packet capture completed on en0.
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- %
```

Test 7: IPv4 Capture

Command: `python3 main.py -i any -c 1 -f ip`
Result: Pass - IPv4 header fields correct

```
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- % python3 main.py -c 1 -i any -f ip
Available interfaces: ['lo0', 'gif0', 'stf0', 'en4', 'ap1', 'en0', 'awdl0', 'llw0', 'en2', 'en1', 'bridge0', 'utun0', 'utun1', 'utun2', 'utun3']
Starting packet capture on en0
Starting packet capture on en0 with filter: ip

Captured Packet 1:
Ethernet Header:
  Destination MAC:          01005e0000fb          | 01:00:5e:00:00:fb
  Source MAC:               ccf411aa274a          | cc:f4:11:aa:27:4a
  EtherType:                0800                  | 2048
IPv4 Header:
  Version:                  4                     | 4
  Header Length:            5                     | 20
  Total Length:             0044                  | 68
  Flags & Frag Offset:      4000                  | 16384
    Reserved:  0
    DF (Do not Fragment): 1
    MF (More Fragments): 0
    Fragment Offset: 0x0 | 0
  Protocol:                 11                    | 17
  Source IP:                c0a80145              | 192.168.1.69
  Destination IP:           e00000fb              | 224.0.0.251
UDP Header:
  Source Port:              14e9                  | 5353
  Destination Port:         14e9                  | 5353
  Length:                   0030                  | 48
  Checksum:                 e8ba                  | 59578
  Payload (hex):            0000000000010000000000000b5f676f6f676c65636173740 45f746370056c6f63616c00000c0001
Packet capture completed on en0.
```

Test 8: IPv6 Capture

Command: `python3 main.py -i any -c 1 -f ip6`
Expected: EtherType 86dd, IPv6 addresses in colon notation
Result: Pass - IPv6 header fields correct

```
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- % python3 main.py -c 1 -i any -f ip6
Available interfaces: ['lo0', 'gif0', 'stf0', 'en4', 'ap1', 'en0', 'awdl0', 'llw0', 'en2', 'en1', 'bridge0', 'utun0', 'utun1', 'utun2', 'utun3']
Starting packet capture on en0
Starting packet capture on en0 with filter: ip6

Captured Packet 1:
Ethernet Header:
  Destination MAC:          8c19b53ae74e          | 8c:19:b5:3a:e7:4e
  Source MAC:               a483e75dc46c          | a4:83:e7:5d:c4:6c
  EtherType:                86dd                  | 34525
IPv6 Header:
  Version:                  6                     | 6
  Payload Length:           0025                  | 37
  Next Header:              11                    | 17
  Hop Limit:                40                    | 64
  Source IP                 200105697899510014d238ac38127bf6 | 2001:0569:7899:5100:14d2:38ac:3812:7bf6
  Destination IP:           2607f8b0400a080a000000000000200e | 2607:f8b0:400a:080a:0000:0000:0000:200e
UDP Header:
  Source Port:              c39e                  | 50078
  Destination Port:         01bb                  | 443
  Length:                   0025                  | 37
  Checksum:                 f7a8                  | 63400
  Payload (hex):            48fcc0896196e2d36ad0fa49431d62b4b5a67eb21ab9642f677ab2a3a5
Packet capture completed on en0.
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- %
```

Test 9: ICMPv6 Capture

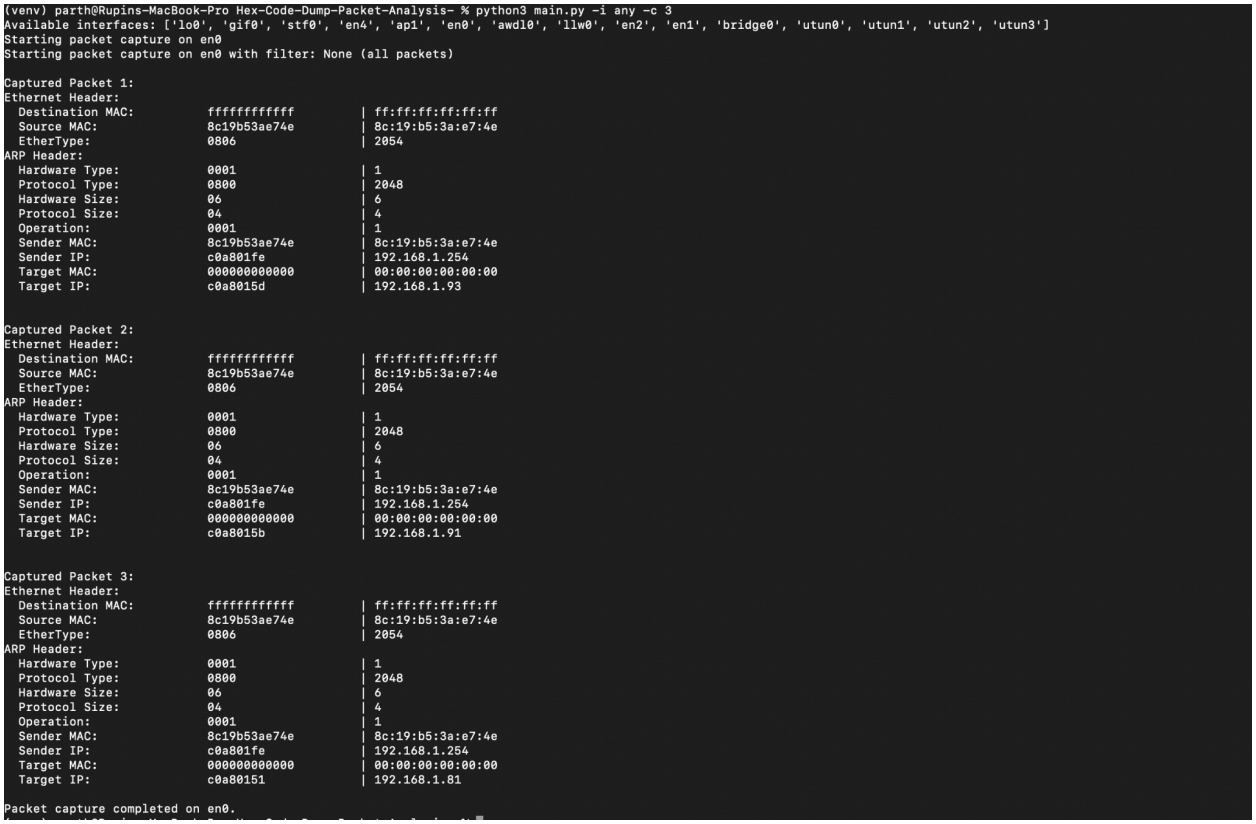
Command: `python3 main.py -i any -c 1 -f icmp6`
Expected: IPv6 next header = 58, ICMPv6 type/code
Result: Pass/Partial - Works on systems with IPv6

```
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- % python3 main.py -c 1 -i any -f icmp6
Available interfaces: ['lo0', 'gif0', 'stf0', 'en4', 'ap1', 'en0', 'awdl0', 'llw0', 'en2', 'en1', 'bridge0', 'utun0', 'utun1', 'utun2', 'utun3']
Starting packet capture on en0
Starting packet capture on en0 with filter: icmp6

Captured Packet 1:
Ethernet Header:
  Destination MAC:          8c19b53ae74e          | 8c:19:b5:3a:e7:4e
  Source MAC:               a483e75dc46c          | a4:83:e7:5d:c4:6c
  EtherType:                86dd                  | 34525
IPv6 Header:
  Version:                  6                     | 6
  Payload Length:           0020                  | 32
  Next Header:              3a                    | 58
  Hop Limit:                ff                    | 255
  Source IP                 fe800000000000001c9545fcb587425f | fe80:0000:0000:0000:1c95:45fc:b587:425f
  Destination IP:           fe800000000000008e19b5fffe3ae74e | fe80:0000:0000:0000:8e19:b5ff:fe3a:e74e
ICMP6 Header:
  Type:                     87                    | 135
  Code:                     00                    | 0
  Checksum:                 7e14                  | 32276
  Payload (hex):            00000000fe800000000000008e19b5fffe3ae74e0101a483e75dc46c
Packet capture completed on en0.
(venv) parth@Rupins-MacBook-Pro Hex-Code-Dump-Packet-Analysis- %
```

Test 10: Capture three packets

Command: `python3 main.py -i any -c 3`
Expected: Three different packets captured
Result: Shows three different packets captured with all the correct details



Test 11: Capture three packets

Command: `python3 main.py -i any`
Expected: Error as the count flags was not specified
Result: Shows that the script does not execute without the count flag