



# NSE Coursework: Network attack and defence

Project Report

Authors: Ahmed Bedair, Yukesh Shrestha, Varishdan Kumariah

April 23, 2024

# Contents

<b>1</b>	<b>Level 1: Build a network and test its connectivity</b>	<b>1</b>
1.1	Network Topology . . . . .	1
1.2	Connectivity Tests . . . . .	2
<b>2</b>	<b>Level 2: Generate and analyse traffic on your network</b>	<b>3</b>
2.1	Protocol Analysis . . . . .	3
2.2	Packet Analysis . . . . .	4
2.3	Flow Analysis . . . . .	5
2.4	Performance Analysis . . . . .	5
<b>3</b>	<b>Level 3: Network attacks</b>	<b>7</b>
<b>4</b>	<b>Level 4: Network defence</b>	<b>8</b>
<b>5</b>	<b>Level 5: Critical evaluation and reflection</b>	<b>9</b>

# Chapter 1

## Level 1: Build a network and test its connectivity

### 1.1 Network Topology

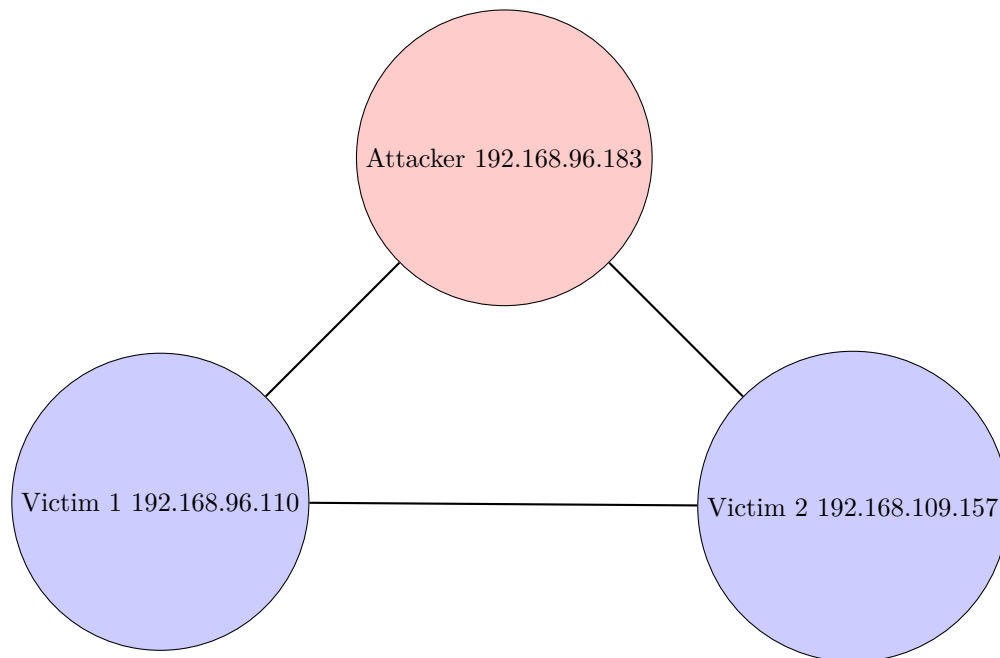
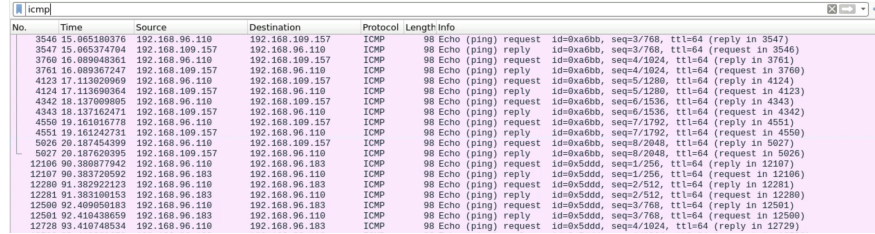


Figure 1.1: Network Topology

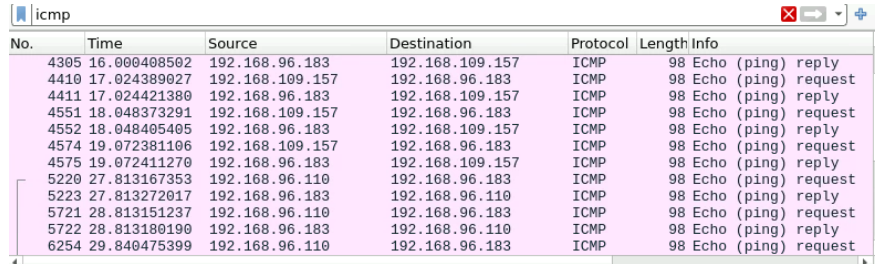
## 1.2 Connectivity Tests

The following screenshots demonstrate the successful ICMP ping tests between each machine, confirming that the machines are fully connected, as shown in Figure 1.1. The machines are also fully connected to the internet. We used the `iperf` tool instead of just browsing, as it reliably generates similar traffic on demand, further enhancing our ability to test the network's performance.



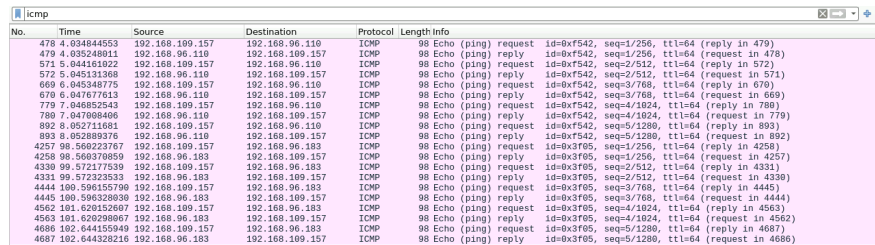
No.	Time	Source	Destination	Protocol	Length	Info
3546	15.065180376	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) request id=0xa6bb, seq=3/768, ttl=64 (reply in 3547)
3547	15.065374784	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) reply id=0xa6bb, seq=3/768, ttl=64 (request in 3546)
3760	16.080948361	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) request id=0xa6bb, seq=4/1024, ttl=64 (reply in 3761)
3761	16.0809387247	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) reply id=0xa6bb, seq=4/1024, ttl=64 (request in 3760)
4123	17.113020969	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) request id=0xa6bb, seq=5/1280, ttl=64 (reply in 4124)
4124	17.113090364	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) reply id=0xa6bb, seq=5/1280, ttl=64 (request in 4123)
4342	18.137090985	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) request id=0xa6bb, seq=6/1536, ttl=64 (reply in 4343)
4343	18.137162471	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) reply id=0xa6bb, seq=6/1536, ttl=64 (request in 4342)
4550	19.161016778	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) request id=0xa6bb, seq=7/1792, ttl=64 (reply in 4551)
4551	19.161242731	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) reply id=0xa6bb, seq=7/1792, ttl=64 (request in 4550)
5026	20.187454399	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) request id=0xa6bb, seq=8/2048, ttl=64 (reply in 5027)
5027	20.187620395	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) reply id=0xa6bb, seq=8/2048, ttl=64 (request in 5026)
12186	96.388877842	192.168.96.110	192.168.96.183	ICMP	98	Echo (ping) request id=0x5ddd, seq=1/256, ttl=64 (reply in 12187)
12187	96.393720592	192.168.96.183	192.168.96.110	ICMP	98	Echo (ping) reply id=0x5ddd, seq=1/256, ttl=64 (request in 12186)
12280	91.382922123	192.168.96.110	192.168.96.183	ICMP	98	Echo (ping) request id=0x5ddd, seq=2/512, ttl=64 (reply in 12281)
12281	91.383100153	192.168.96.183	192.168.96.110	ICMP	98	Echo (ping) reply id=0x5ddd, seq=2/512, ttl=64 (request in 12280)
12500	92.409050183	192.168.96.110	192.168.96.183	ICMP	98	Echo (ping) request id=0x5ddd, seq=3/768, ttl=64 (reply in 12501)
12501	92.410438659	192.168.96.183	192.168.96.110	ICMP	98	Echo (ping) reply id=0x5ddd, seq=3/768, ttl=64 (request in 12500)
12728	93.410748534	192.168.96.110	192.168.96.183	ICMP	98	Echo (ping) request id=0x5ddd, seq=4/1024, ttl=64 (reply in 12729)

Figure 1.2: Pings from Attacker to Other Machines



No.	Time	Source	Destination	Protocol	Length	Info
4305	16.000408502	192.168.96.183	192.168.109.157	ICMP	98	Echo (ping) reply
4410	17.024389027	192.168.109.157	192.168.96.183	ICMP	98	Echo (ping) request
4411	17.024421380	192.168.96.183	192.168.109.157	ICMP	98	Echo (ping) reply
4551	18.048373291	192.168.109.157	192.168.96.183	ICMP	98	Echo (ping) request
4552	18.048405405	192.168.96.183	192.168.109.157	ICMP	98	Echo (ping) reply
4574	19.072381106	192.168.109.157	192.168.96.183	ICMP	98	Echo (ping) request
4575	19.072411270	192.168.96.183	192.168.109.157	ICMP	98	Echo (ping) reply
5220	27.813167353	192.168.96.110	192.168.96.183	ICMP	98	Echo (ping) request
5223	27.813272047	192.168.96.183	192.168.96.110	ICMP	98	Echo (ping) reply
5721	28.813151237	192.168.96.110	192.168.96.183	ICMP	98	Echo (ping) request
5722	28.813180190	192.168.96.183	192.168.96.110	ICMP	98	Echo (ping) reply
6254	29.840475399	192.168.96.110	192.168.96.183	ICMP	98	Echo (ping) request

Figure 1.3: Pings from Victim 1 to Other Machines



No.	Time	Source	Destination	Protocol	Length	Info
478	4.038444053	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) request id=0xf542, seq=1/256, ttl=64 (reply in 479)
479	4.038248011	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) reply id=0xf542, seq=1/256, ttl=64 (request in 478)
571	5.044161022	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) request id=0xf542, seq=2/512, ttl=64 (reply in 572)
572	5.045131368	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) reply id=0xf542, seq=2/512, ttl=64 (request in 571)
609	6.045348775	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) request id=0xf542, seq=3/768, ttl=64 (reply in 610)
610	6.047677613	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) reply id=0xf542, seq=3/768, ttl=64 (request in 609)
779	7.040602543	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) request id=0xf542, seq=4/1024, ttl=64 (reply in 780)
780	7.047008406	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) reply id=0xf542, seq=4/1024, ttl=64 (request in 779)
892	8.052711681	192.168.109.157	192.168.96.110	ICMP	98	Echo (ping) request id=0xf542, seq=5/1280, ttl=64 (reply in 893)
893	8.052809376	192.168.96.110	192.168.109.157	ICMP	98	Echo (ping) reply id=0xf542, seq=5/1280, ttl=64 (request in 892)
4257	98.560223767	192.168.109.157	192.168.96.183	ICMP	98	Echo (ping) request id=0x3f05, seq=1/256, ttl=64 (reply in 4258)
4258	98.560370859	192.168.96.183	192.168.109.157	ICMP	98	Echo (ping) reply id=0x3f05, seq=1/256, ttl=64 (request in 4257)
4330	99.572177539	192.168.109.157	192.168.96.183	ICMP	98	Echo (ping) request id=0x3f05, seq=2/512, ttl=64 (reply in 4331)
4331	99.572323533	192.168.96.183	192.168.109.157	ICMP	98	Echo (ping) reply id=0x3f05, seq=2/512, ttl=64 (request in 4330)
4444	100.590155790	192.168.109.157	192.168.96.183	ICMP	98	Echo (ping) request id=0x3f05, seq=3/768, ttl=64 (reply in 4445)
4445	100.590320830	192.168.96.183	192.168.109.157	ICMP	98	Echo (ping) reply id=0x3f05, seq=3/768, ttl=64 (request in 4444)
4562	101.620152607	192.168.109.157	192.168.96.183	ICMP	98	Echo (ping) request id=0x3f05, seq=4/1024, ttl=64 (reply in 4563)
4563	101.620296067	192.168.96.183	192.168.109.157	ICMP	98	Echo (ping) reply id=0x3f05, seq=4/1024, ttl=64 (request in 4562)
4686	102.644155940	192.168.109.157	192.168.96.183	ICMP	98	Echo (ping) request id=0x3f05, seq=5/1280, ttl=64 (reply in 4687)
4687	102.644328216	192.168.96.183	192.168.109.157	ICMP	98	Echo (ping) reply id=0x3f05, seq=5/1280, ttl=64 (request in 4686)

Figure 1.4: Pings from Victim 2 to Other Machines

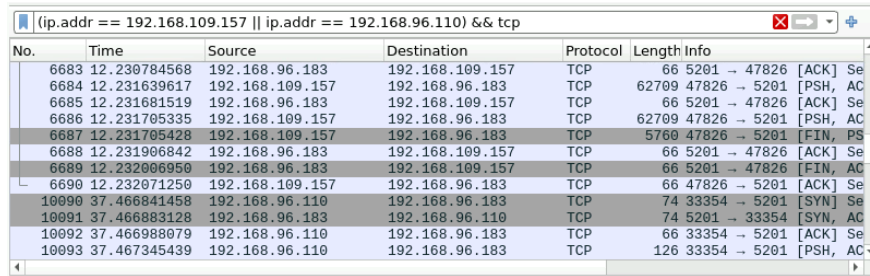
## Chapter 2

# Level 2: Generate and analyse traffic on your network

### 2.1 Protocol Analysis

To generate network traffic, we utilised the `iperf` tool to create UDP traffic between the machines in our network. We chose UDP traffic because the `iperf` tool automatically displays performance metrics such as jitter and packet loss when sending UDP traffic. We then analysed the captured traffic using Wireshark, a powerful network protocol analyser, to gain insights into the network's behaviour and performance.

Figure 2.1 provides a comprehensive view of the protocol hierarchy of the captured traffic. We used Wireshark's built-in filters to focus specifically on the UDP traffic generated by the `iperf` tool. The protocol analysis revealed that the TCP protocol dominated the network communication when including TCP traffic, likely due to the background connections the VM was making to the rest of the KCL network. However, the UDP traffic generated by the `iperf` tool was visible as a separate protocol in the Wireshark analysis, allowing us to isolate and analyse the UDP traffic independently.



No.	Time	Source	Destination	Protocol	Length Info
6683	12.230784568	192.168.96.183	192.168.109.157	TCP	66 5201 → 47826 [ACK] Se
6684	12.231639617	192.168.109.157	192.168.96.183	TCP	62709 47826 → 5201 [PSH, AC
6685	12.231681519	192.168.96.183	192.168.109.157	TCP	66 5201 → 47826 [ACK] Se
6686	12.231705335	192.168.109.157	192.168.96.183	TCP	62709 47826 → 5201 [PSH, AC
6687	12.231705428	192.168.109.157	192.168.96.183	TCP	5760 47826 → 5201 [FIN, PS
6688	12.231906842	192.168.96.183	192.168.109.157	TCP	66 5201 → 47826 [ACK] Se
6689	12.232006950	192.168.96.183	192.168.109.157	TCP	66 5201 → 47826 [FIN, AC
6690	12.232071250	192.168.109.157	192.168.96.183	TCP	66 47826 → 5201 [ACK] Se
10090	37.466841458	192.168.96.110	192.168.96.183	TCP	74 33354 → 5201 [SYN] Se
10091	37.466883128	192.168.96.183	192.168.96.110	TCP	74 5201 → 33354 [SYN, AC
10092	37.466988079	192.168.96.110	192.168.96.183	TCP	66 33354 → 5201 [ACK] Se
10093	37.467345439	192.168.96.110	192.168.96.183	TCP	126 33354 → 5201 [PSH, AC

Figure 2.1: Wireshark Protocol Analysis

## 2.2 Packet Analysis

We can examine individual packets at the packet level to understand their structure and contents. Figure 2.2 displays a UDP packet captured during the traffic generation. The packet details show the source and destination IP addresses, ports, packet length, and checksum. This information is crucial for troubleshooting and identifying any anomalies in the network communication. While these packets are not malicious in this case, we can use this information to identify malicious packets at future levels.

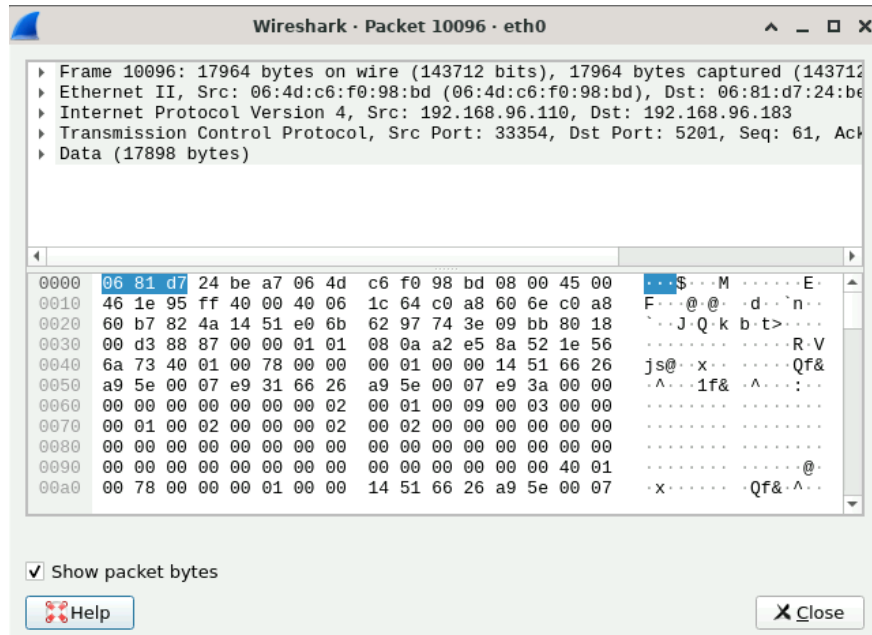


Figure 2.2: Wireshark Packet Analysis

## 2.3 Flow Analysis

Our analysis extended to the flow level, where we used Wireshark's Conversations feature to group packets into logical connections between endpoints. Figure 2.3 presents the UDP conversations in the captured traffic. Each row represents a unique UDP connection, showing the source and destination addresses, the number of packets exchanged, and the total amount of data transferred. This flow-level analysis demonstrates how we used **iperf** to generate UDP traffic between the machines in our network, setting up the attacker as an **iperf** server and the victims as **iperf** clients, with a separate connection between the attacker and each victim.

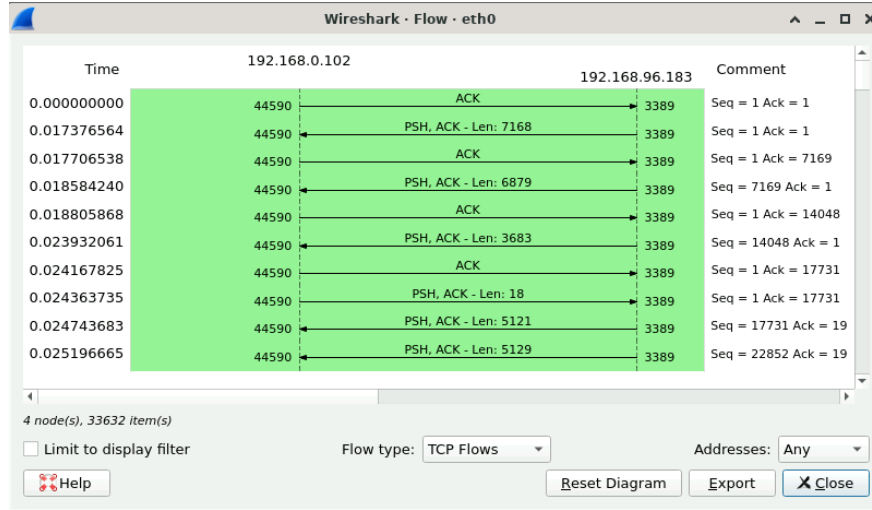


Figure 2.3: Wireshark Flow Analysis

## 2.4 Performance Analysis

Finally, we assessed the network performance by measuring key metrics such as throughput, delay, and packet loss. Figure 2.4 summarises the performance analysis results obtained from **iperf**. The UDP traffic achieved an average bitrate of 3.81 Gbits/sec, which, as we will observe in future levels, is exceptionally high and will drop when we introduce malicious traffic. This baseline measurement provides a reference point for evaluating the impact of network attacks on the overall performance.

```

[ 6] local 192.168.96.183 port 5201 connected with 192.168.96.110 port 53250 (icwnd
/mss/irrt=87/8949/170)
[ ID] Interval      Transfer      Bandwidth
[ 6] 0.0000-1.0059 sec  434 MBytes  3.62 Gbits/sec
[ 7] local 192.168.96.183 port 5201 connected with 192.168.109.157 port 33474 (icwn
d/mss/irrt=87/8949/146)
[ ID] Interval      Transfer      Bandwidth
[ 7] 0.0000-1.0046 sec  456 MBytes  3.81 Gbits/sec

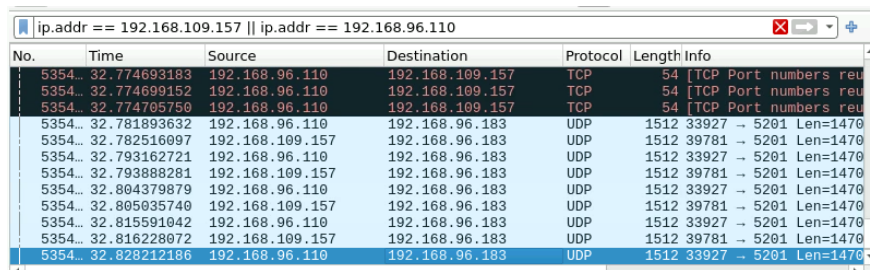
```

Figure 2.4: Performance Analysis



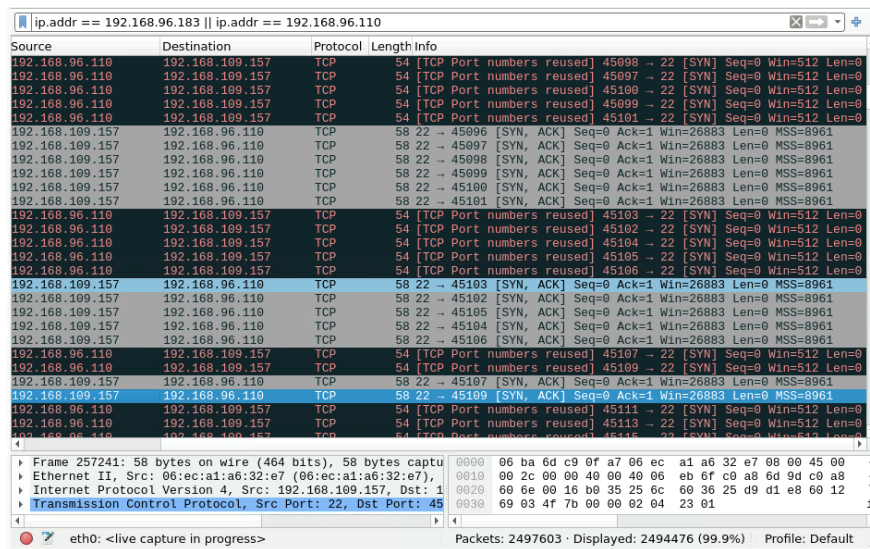
# Chapter 3

## Level 3: Network attacks



No.	Time	Source	Destination	Protocol	Length	Info
5354...	32.774693183	192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reu
5354...	32.774699152	192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reu
5354...	32.774705750	192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reu
5354...	32.781893632	192.168.96.110	192.168.96.183	UDP	1512	33927 → 5201 Len=1470
5354...	32.782516097	192.168.109.157	192.168.96.183	UDP	1512	39781 → 5201 Len=1470
5354...	32.793162721	192.168.96.110	192.168.96.183	UDP	1512	33927 → 5201 Len=1470
5354...	32.793888281	192.168.109.157	192.168.96.183	UDP	1512	39781 → 5201 Len=1470
5354...	32.804379879	192.168.96.110	192.168.96.183	UDP	1512	33927 → 5201 Len=1470
5354...	32.805035740	192.168.109.157	192.168.96.183	UDP	1512	39781 → 5201 Len=1470
5354...	32.815591042	192.168.96.110	192.168.96.183	UDP	1512	33927 → 5201 Len=1470
5354...	32.816228072	192.168.109.157	192.168.96.183	UDP	1512	39781 → 5201 Len=1470
5354...	32.828212186	192.168.96.110	192.168.96.183	UDP	1512	33927 → 5201 Len=1470

Figure 3.1: TCP SYN Flood *Attacker*



Source	Destination	Protocol	Length	Info
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45098 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45097 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45100 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45099 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45101 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.109.157	192.168.96.110	TCP	58	22 → 45096 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.109.157	192.168.96.110	TCP	58	22 → 45097 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.109.157	192.168.96.110	TCP	58	22 → 45098 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.109.157	192.168.96.110	TCP	58	22 → 45099 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.109.157	192.168.96.110	TCP	58	22 → 45100 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.109.157	192.168.96.110	TCP	58	22 → 45101 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45103 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45102 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45104 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45105 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45106 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.109.157	192.168.96.110	TCP	58	22 → 45103 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.109.157	192.168.96.110	TCP	58	22 → 45102 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.109.157	192.168.96.110	TCP	58	22 → 45105 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.109.157	192.168.96.110	TCP	58	22 → 45104 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.109.157	192.168.96.110	TCP	58	22 → 45106 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45107 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45109 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.109.157	192.168.96.110	TCP	58	22 → 45107 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.109.157	192.168.96.110	TCP	58	22 → 45109 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=8961
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45111 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45113 → 22 [SYN] Seq=0 Win=512 Len=0
192.168.96.110	192.168.109.157	TCP	54	[TCP Port numbers reused] 45114 → 22 [SYN] Seq=0 Win=512 Len=0

Frame 257241: 58 bytes on wire (464 bits), 58 bytes captured on interface eth0, Src: 06:ec:a1:a6:32:e7 (06:ec:a1:a6:32:e7), Internet Protocol Version 4, Src: 192.168.109.157, Dst: 192.168.96.110, Transmission Control Protocol, Src Port: 22, Dst Port: 45103

eth0: <live capture in progress> Packets: 2497603 · Displayed: 2494476 (99.9%) Profile: Default

Figure 3.2: TCP SYN Flood *Victim*

## Chapter 4

### Level 4: Network defence

## Chapter 5

# Level 5: Critical evaluation and reflection