COMP 7003
Assignment 4

Manraj Bhullar
A01018465
Tas
A0101
November 26, 2024

Purpose & Objectives

The purpose of this assignment is to deepen understanding of network security mechanisms, focusing on the roles of firewalls and intrusion detection systems (IDS). This assignment involves conducting a series of network attacks from Host A (attacker) to Host B (victim) to evaluate how various security configurations affect the network's ability to detect, block, or mitigate these threats. This assignment utilizes nftables as the firewall tool and Snort3 as the IDS. The assignment is divided into four parts:

- *Baseline Analysis Without Defense:* Analyze the impact of attacks on an unprotected network to identify attack patterns and vulnerabilities.
- *Defensive Measures on Host B:* Implement and test firewall (nftables) and IDS (Snort3) configurations on Host B to observe how well these mechanisms defend against specific attacks.
- *Attacker-Side Defense:* Configure security measures on Host A to block outgoing attacks and analyze their effectiveness in preventing threats from originating.
- *Full Defense on Both Hosts:* Combine defensive configurations on both Host A and Host B to assess the comprehensive impact of deploying layered security.

Attack Scenarios		
Attack Name	**Command**	**Description**
SYN Flood	sudo hping3 -S -p 80 --flood 10.0.0.253	Floods the victim with SYN packets on port 80.
UDP Flood	sudo hping3 --udp -p 53 --flood 10.0.0.253	Floods the victim with UDP packets on port 53.
XMAS Tree Scan	sudo nmap -sX 10.0.0.253	Sends packets with unusual TCP flags (FIN, PSH, URG).
Ping of Death	sudo ping -s 65500 10.0.0.253	Sends oversized ICMP packets to cause a potential crash.
Buffer Overflow	sudo python3 -c 'print("A" * 1000)' | nc 10.0.0.253 1234	Sends an oversized payload of 1,000 'A' characters to the victim's port, potentially causing a buffer overflow if the receiving service does not correctly handle the input size.

Packet Capture Strategy

Network traffic will be captured using Wireshark on both hosts during the execution of all attacks. The attacks will be conducted consecutively and all traffic will be captured within a single .pcap file on each host to simplify analysis. Wireshark filters will be applied to the capture files to isolate and analyze individual attacks. This process will group all packets related to a specific attack, allowing targeted analysis. Two separate filters will be used for each attack, one for the outgoing packets sent from Host A (attacker) and one for outgoing response packets from Host B. This makes it easier to visualize behaviour, whereas combining all packets together makes it more difficult to trace.

Host Machines		
Name	**IP Address**	**Interface**
Host A: Attacker	10.0.0.129	enp0s31f6
Host B: Victim	10.0.0.253	wlp0s20f3

PCAP Wireshark Filters		
Attack Name	**Attack Packets from Host A**	**Response Packets from Host B**
SYN Flood	ip.src == 10.0.0.129 && ip.dst == 10.0.0.253 && tcp.dstport == 80	ip.src == 10.0.0.253 && ip.dst == 10.0.0.129 && tcp.srcport == 80
UDP Flood	ip.src == 10.0.0.129 && ip.dst == 10.0.0.253 && udp.dstport == 53	ip.src == 10.0.0.253 && ip.dst == 10.0.0.129 && udp
XMAS Tree Scan	ip.src == 10.0.0.129 && ip.dst == 10.0.0.253 && tcp.flags == 0x29	ip.src == 10.0.0.253 && ip.dst == 10.0.0.129 && tcp.dstport == 45208
Ping of Death	ip.src == 10.0.0.129 && ip.dst == 10.0.0.253 && icmp.type == 8	ip.src == 10.0.0.253 && ip.dst == 10.0.0.129 && icmp.type == 0
Buffer Overflow	ip.src == 10.0.0.129 && ip.dst == 10.0.0.253 && tcp.dstport == 1234	ip.src == 10.0.0.129 && ip.dst == 10.0.0.253 && tcp.dstport == 1234 && !(tcp.flags == 0x29)

Logging Configuration

Logging for this assignment is configured so that the outputs from both nftables and Snort3 are written to single respective .txt files for each of the four parts in the assignment. Both hosts will have separate log files for each part, capturing all relevant blocked packets and IDS alerts. Specific attack information can be isolated using grep commands. These commands will filter the logs to identify the relevant attack patterns or blocked packets for each scenario. The two commands below are ran in separate terminals interactively to log nftables and Snort3 activity for each part of the assignment.

Run/Log Commands	
Tool	**Command**
nftables	journalctl -f | grep -E 'SYN-FLOOD-ATTEMPT|PING-OF-DEATH' > /home/manraj/logs/nftables_log.txt
Snort3	sudo /usr/local/snort/bin/snort -i wlp0s20f3 -c /usr/local/snort/etc/snort/snort.lua -R /usr/local/snort/etc/snort/local.rules -A full > /home/manraj/logs/snort_log.txt

Host B Victim Defense Rules		
Tool	**Rule**	**Description**
nftables	ct state established,related accept	Accepts established and related connections.
nftables	ct state invalid drop	Drops invalid packets.
nftables	tcp dport 1234 accept	Open port for nc connection.
nftables	tcp flags syn limit rate 20/second log prefix "SYN-FLOOD-ATTEMPT: "	Logs SYN flood packets that are received above 20 per second.
nftables	tcp flags syn limit rate 20/second drop	Drops any SYN flood packets that are received above 20 per second.
nftables	tcp flags syn log prefix "SYN-FLOOD-ATTEMPT: "	Logs any SYN flood packets that bypass the rate limit.
nftables	tcp flags syn drop	Drops any SYN flood packets that bypass the rate limit.
nftables	ip protocol icmp icmp type echo-request meta length > 1500 log prefix "PING-OF-DEATH: " drop	Prevents Ping of Death by dropping packets that exceed a certain size.
snort3	alert udp any any -> any any (msg:"UDP-FLOOD-DETECTED"; flow:stateless; detection_filter:track by_src, count 200, seconds 1; sid:1000001; rev:1;)	Detects UDP flood by keeping track of a total UDP count from a specific source within one second. Alerts packets that exceed the threshold.
snort3	alert tcp any any -> any any (msg:"XMAS-TREE-SCAN-DETECTED"; flags:FPU; sid:1000002; rev:1;)	Detects XMAS Tree Scan by checking to see if incoming packet has the corresponding TCP flags for the attack.
snort3	alert tcp any any -> any any (msg:"BUFFER-OVERFLOW-DETECTED"; dsize:>1000; sid:1000003; rev:1;)	Detects Buffer Overflow by alerting if the payload size is above a certain threshold.

Host A Attacker Defense Rules		
Tool	**Rule**	**Description**
nftables	tcp flags syn limit rate 100/second burst 200 packets log prefix "SYN-FLOOD-ATTEMPT: " drop	Blocks outgoing SYN flood packets
nftables	ip protocol icmp meta length gt 1500 log prefix "PING-OF-DEATH: " counter drop	Blocks outgoing Ping of Death Packets
nftables	accept	Accepts legitimate traffic
snort3	alert udp any any -> any any (msg:"UDP-FLOOD-DETECTED"; threshold:type both, track by_dst, count 200, seconds 1; sid:1000001; rev:1;)	Detects outgoing UDP flood.
snort3	alert tcp any any -> any any (flags:FPU; msg:"XMAS-TREE-SCAN-DETECTED"; sid:1000002; rev:1;)	Detects outgoing XMAS Tree Scan.
snort3	alert tcp any any -> any any (msg:"BUFFER-OVERFLOW-DETECTED"; content:"|41 41 41 41|"; depth:4; sid:1000003; rev:1;)	Detects outgoing Buffer Overflow.

Part 1: Baseline Analysis Without Defense

Objective: In this scenario, nftables and Snort3 have been disabled on both hosts. The purpose of this task is to analyze the effects of the attacks with no defense mechanisms in place and how it affects the victim.

PCAP Files	
Name	**Description**
part1_hosta.pcap	Holds packet capture information from all attacks on the attacker side.
part1_hostb.pcap	Holds packet capture information from all attacks on the victim side.

SYN Flood

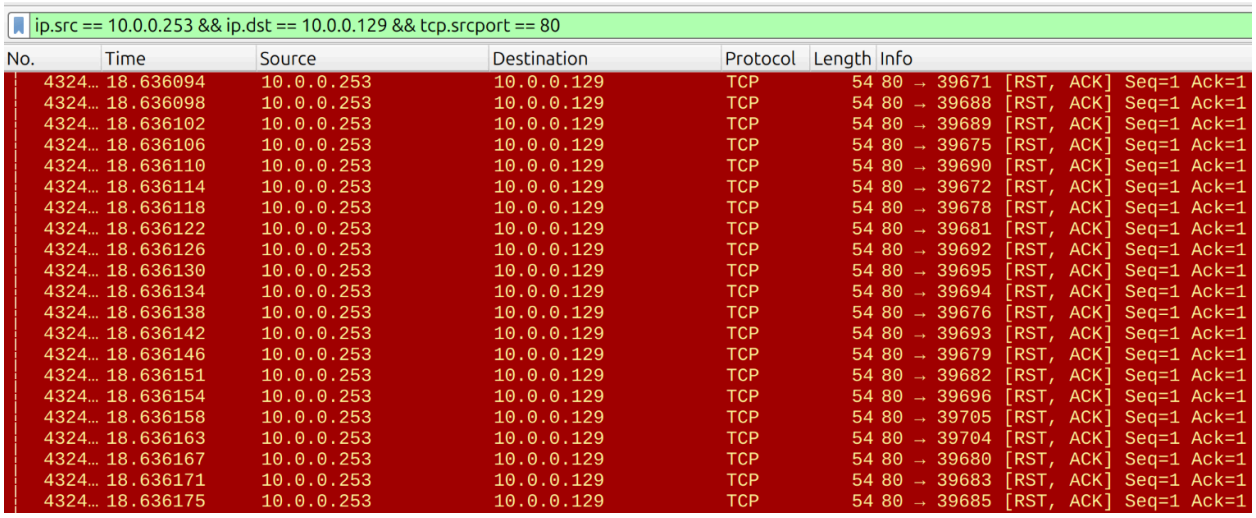
Screenshot 1: Outgoing Packets from Host A

ip.src == 10.0.0.129 && ip.dst == 10.0.0.253 && tcp.dstport == 80

No.	Time	Source	Destination	Protocol	Length	Info
206	10.445612302	10.0.0.129	10.0.0.253	TCP	60	2048 → 80 [SYN] Seq=0 Win=512 Len=0
207	10.445613714	10.0.0.129	10.0.0.253	TCP	60	2042 → 80 [SYN] Seq=0 Win=512 Len=0
208	10.445612458	10.0.0.129	10.0.0.253	TCP	60	2055 → 80 [SYN] Seq=0 Win=512 Len=0
209	10.445613841	10.0.0.129	10.0.0.253	TCP	60	2045 → 80 [SYN] Seq=0 Win=512 Len=0
210	10.445612571	10.0.0.129	10.0.0.253	TCP	60	2065 → 80 [SYN] Seq=0 Win=512 Len=0
211	10.445613919	10.0.0.129	10.0.0.253	TCP	60	2057 → 80 [SYN] Seq=0 Win=512 Len=0
212	10.445612657	10.0.0.129	10.0.0.253	TCP	60	2070 → 80 [SYN] Seq=0 Win=512 Len=0
213	10.445613985	10.0.0.129	10.0.0.253	TCP	60	2062 → 80 [SYN] Seq=0 Win=512 Len=0
214	10.445612740	10.0.0.129	10.0.0.253	TCP	60	2082 → 80 [SYN] Seq=0 Win=512 Len=0
215	10.445614092	10.0.0.129	10.0.0.253	TCP	60	2072 → 80 [SYN] Seq=0 Win=512 Len=0
216	10.445612823	10.0.0.129	10.0.0.253	TCP	60	2085 → 80 [SYN] Seq=0 Win=512 Len=0
217	10.445614154	10.0.0.129	10.0.0.253	TCP	60	2079 → 80 [SYN] Seq=0 Win=512 Len=0
218	10.445612901	10.0.0.129	10.0.0.253	TCP	60	2099 → 80 [SYN] Seq=0 Win=512 Len=0
219	10.445614211	10.0.0.129	10.0.0.253	TCP	60	2091 → 80 [SYN] Seq=0 Win=512 Len=0
220	10.445612983	10.0.0.129	10.0.0.253	TCP	60	2100 → 80 [SYN] Seq=0 Win=512 Len=0
221	10.445614275	10.0.0.129	10.0.0.253	TCP	60	2092 → 80 [SYN] Seq=0 Win=512 Len=0
223	10.445658596	10.0.0.129	10.0.0.253	TCP	60	2049 → 80 [SYN] Seq=0 Win=512 Len=0
224	10.445658812	10.0.0.129	10.0.0.253	TCP	60	2043 → 80 [SYN] Seq=0 Win=512 Len=0
225	10.445658842	10.0.0.129	10.0.0.253	TCP	60	2054 → 80 [SYN] Seq=0 Win=512 Len=0
226	10.445658935	10.0.0.129	10.0.0.253	TCP	60	2044 → 80 [SYN] Seq=0 Win=512 Len=0
227	10.445658923	10.0.0.129	10.0.0.253	TCP	60	2064 → 80 [SYN] Seq=0 Win=512 Len=0

- This shows a consistent stream of SYN packets sent from Host A to Host B on port 80.
- These SYN packets attempt to initiate TCP connections rapidly without completing the connection handshake.
- The attack aims to cripple Host B by leaving connections half-open, overwhelming its connection table.
- Since no defense mechanisms are configured on the attacker side, Host A can freely send out these malicious packets.

Screenshot 2: Response Packets from Host B

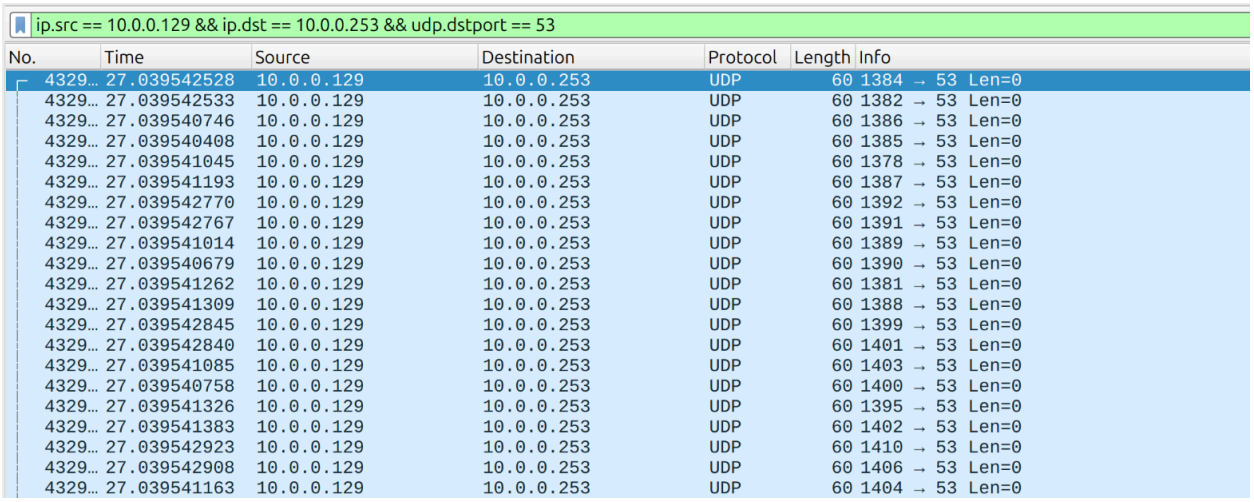


- This shows RST+ACK packets sent from Host B back to Host A in response to the incoming SYN packets.
- Host B is sending RST packets because it is not actively listening for connections on port 80. If the port was open, it would return SYN+ACK responses and have many active half-open connections.
- Without a service listening on port 80, Host B rejects the SYN packets, attempting to reset the connection with RST+ACK responses.
- Processing these connection attempts still consumes a lot of resources on Host B, leaving it overwhelmed during the attack and causing disruption.

Anomalies: Continuous stream of SYN packets from a single source without completing the handshake.

UDP Flood

Screenshot 1: Outgoing Packets from Host A

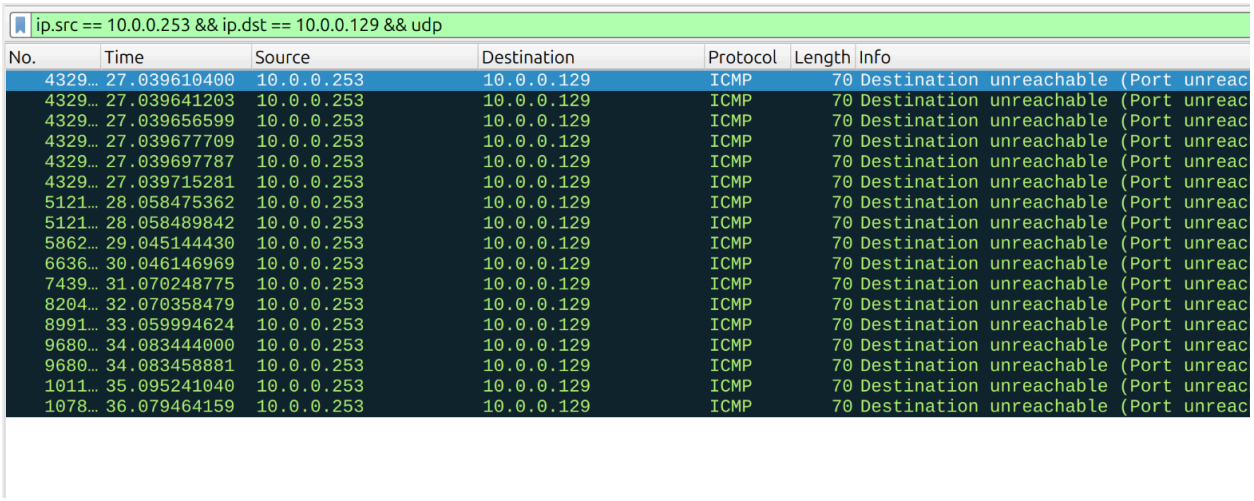


ip.src == 10.0.0.129 && ip.dst == 10.0.0.253 && udp.dstport == 53

No.	Time	Source	Destination	Protocol	Length	Info
4329…	27.039542528	10.0.0.129	10.0.0.253	UDP	60	1384 → 53 Len=0
4329…	27.039542533	10.0.0.129	10.0.0.253	UDP	60	1382 → 53 Len=0
4329…	27.039540746	10.0.0.129	10.0.0.253	UDP	60	1386 → 53 Len=0
4329…	27.039540408	10.0.0.129	10.0.0.253	UDP	60	1385 → 53 Len=0
4329…	27.039541045	10.0.0.129	10.0.0.253	UDP	60	1378 → 53 Len=0
4329…	27.039541193	10.0.0.129	10.0.0.253	UDP	60	1387 → 53 Len=0
4329…	27.039542770	10.0.0.129	10.0.0.253	UDP	60	1392 → 53 Len=0
4329…	27.039542767	10.0.0.129	10.0.0.253	UDP	60	1391 → 53 Len=0
4329…	27.039541014	10.0.0.129	10.0.0.253	UDP	60	1389 → 53 Len=0
4329…	27.039540679	10.0.0.129	10.0.0.253	UDP	60	1390 → 53 Len=0
4329…	27.039541262	10.0.0.129	10.0.0.253	UDP	60	1381 → 53 Len=0
4329…	27.039541309	10.0.0.129	10.0.0.253	UDP	60	1388 → 53 Len=0
4329…	27.039542845	10.0.0.129	10.0.0.253	UDP	60	1399 → 53 Len=0
4329…	27.039542840	10.0.0.129	10.0.0.253	UDP	60	1401 → 53 Len=0
4329…	27.039541085	10.0.0.129	10.0.0.253	UDP	60	1403 → 53 Len=0
4329…	27.039540758	10.0.0.129	10.0.0.253	UDP	60	1400 → 53 Len=0
4329…	27.039541326	10.0.0.129	10.0.0.253	UDP	60	1395 → 53 Len=0
4329…	27.039541383	10.0.0.129	10.0.0.253	UDP	60	1402 → 53 Len=0
4329…	27.039542923	10.0.0.129	10.0.0.253	UDP	60	1410 → 53 Len=0
4329…	27.039542908	10.0.0.129	10.0.0.253	UDP	60	1406 → 53 Len=0
4329…	27.039541163	10.0.0.129	10.0.0.253	UDP	60	1404 → 53 Len=0

- Host A sends a continuous stream of UDP packets to Host B on port 53, overwhelming the victim with high traffic volume.
- Since UDP is connectionless, no handshake or acknowledgment is required, making this type of attack lightweight for the attacker but resource-intensive for the victim.
- With no defenses configured on Host A, it can send packets freely and rapidly.

Screenshot 2: Response Packets from Host B



ip.src == 10.0.0.253 && ip.dst == 10.0.0.129 && udp

No.	Time	Source	Destination	Protocol	Length	Info
4329…	27.039610400	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
4329…	27.039641203	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
4329…	27.039656599	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
4329…	27.039677709	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
4329…	27.039697787	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
4329…	27.039715281	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
5121…	28.058475362	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
5121…	28.058489842	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
5862…	29.045144430	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
6636…	30.046146969	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
7439…	31.070248775	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
8204…	32.070358479	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
8991…	33.059994624	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
9680…	34.083444000	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
9680…	34.083458881	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
1011…	35.095241040	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac
1078…	36.079464159	10.0.0.253	10.0.0.129	ICMP	70	Destination unreachable (Port unreac

- Host B sends ICMP Destination Unreachable responses back to Host A to indicate that the port is unreachable.
- Fewer ICMP responses are sent compared to the number of incoming UDP packets. This is due to ICMP rate limiting on Host B's operating system, which restricts the frequency of these responses to preserve resources.
- While Host B does not accept the UDP packets, it still overwhelms system resources due to the high amount of incoming packets it must process, this affects Host B's performance and causes disruption.

Anomalies: Continuous stream of UDP packets from a single source with intermittent ICMP Destination Unreachable responses from the victim.

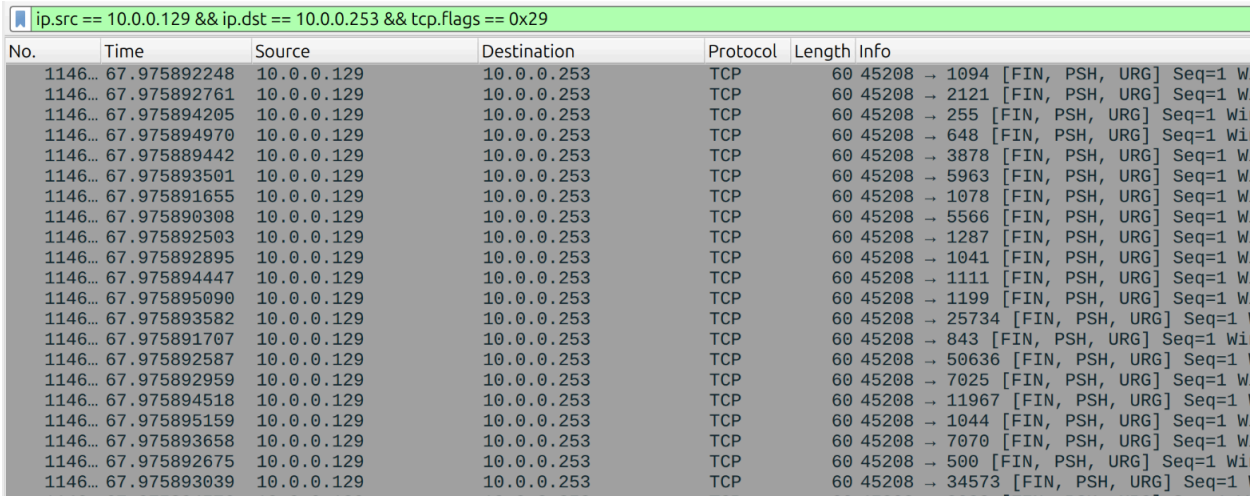
XMAS Tree Scan

Screenshot 1: Console Output of Xmas Tree Scan

```
manraj@manraj-MS-7998:~$ sudo nmap -sX 10.0.0.253
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-27 00:15 PST
Nmap scan report for 10.0.0.253
Host is up (0.0061s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE         SERVICE
22/tcp open|filtered ssh
MAC Address: C0:3C:59:6A:EC:8A (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 2.56 seconds
manraj@manraj-MS-7998:~$
```

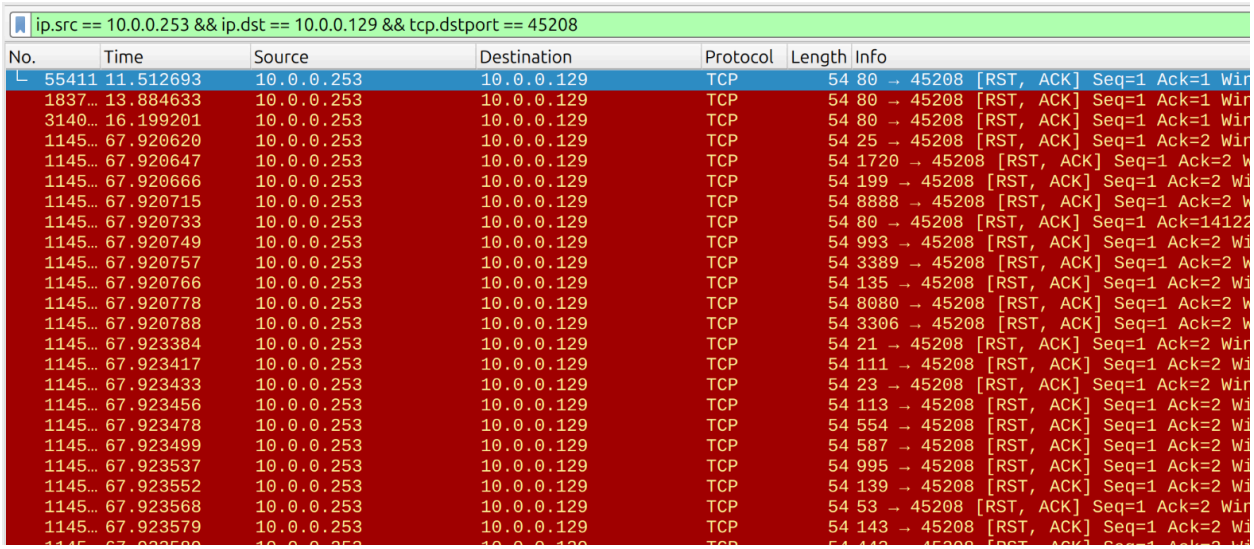
- The attack identified port 22 (SSH) is open on Host B.
- All other ports on Host B are closed, giving insight into Host B's network setup.
- This information helps the attacker map potential vulnerabilities or entry points on Host B.

Screenshot 2: Outgoing Packets from Host A



- Host A sends TCP packets with unusual flags (FIN, PSH, URG) designed to provoke specific responses from Host B.
- The attack takes advantage of how the system handles malformed packets to identify open, closed, or filtered ports.
- Without defenses on Host A, the attack proceeds uninterrupted, sending packets rapidly and effectively scanning Host B revealing its open ports.

Screenshot 3: Response Packets from Host B

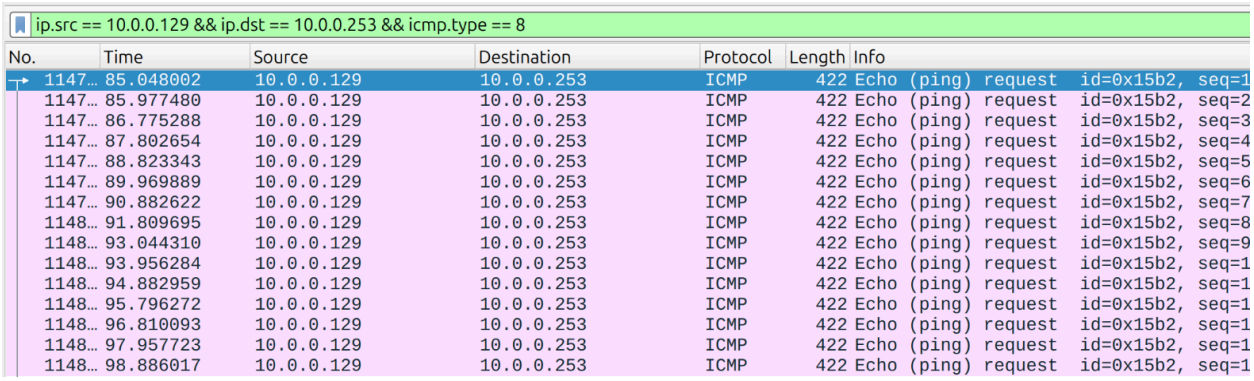


- Host B responds with RST+ACK packets for closed ports, confirming their state, while open ports stay silent.
- With no defenses active on Host B, it processes all incoming packets, providing detailed responses to the attack machine.

Anomalies: TCP packets with unusual flag combinations (**FIN, PSH, URG**) sent to multiple ports.

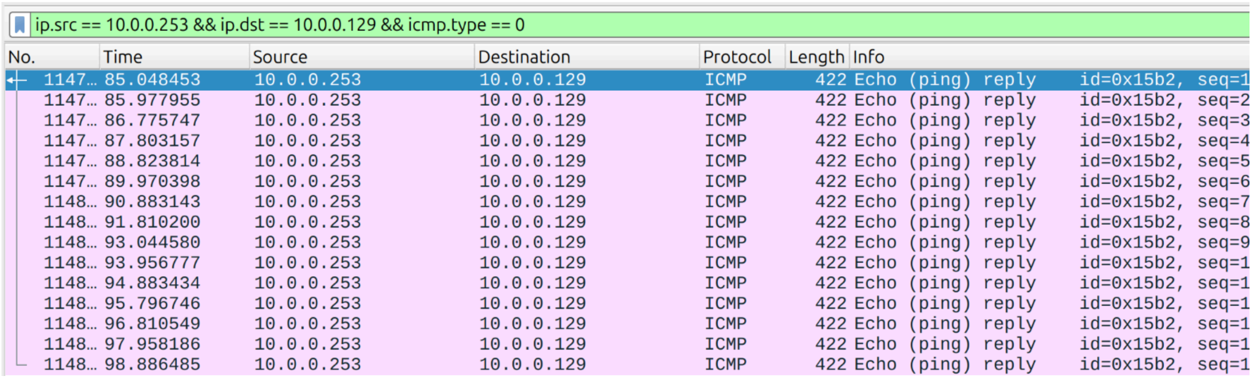
Ping of Death

Screenshot 1: Outgoing Packets from Host A



- Host A sends a continuous stream of oversized ping requests to Host B.
- With no defenses active on Host A, it freely generates and sends these oversized packets without restriction.
- The goal of this attack is denial of service.

Screenshot 2: Response Packets from Host B



ip.src == 10.0.0.253 && ip.dst == 10.0.0.129 && icmp.type == 0

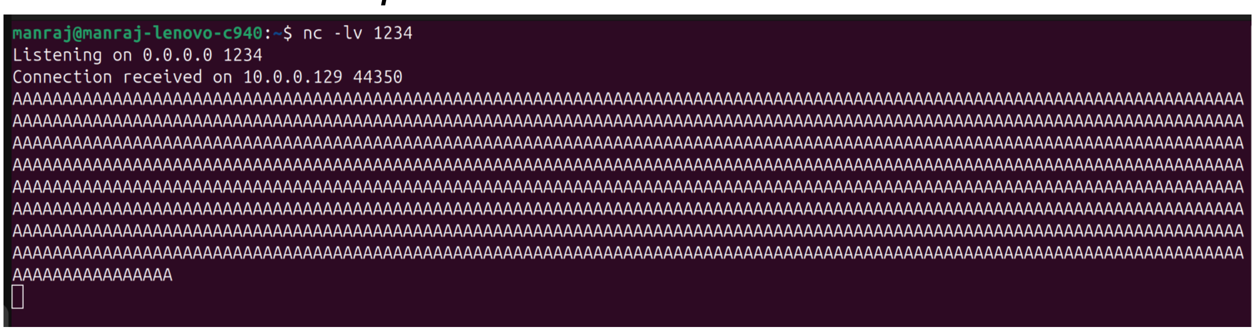
No.	Time	Source	Destination	Protocol	Length	Info
1147…	85.048453	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=1
1147…	85.977955	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=2
1147…	86.775747	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=3
1147…	87.803157	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=4
1147…	88.823814	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=5
1147…	89.970398	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=6
1148…	90.883143	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=7
1148…	91.810200	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=8
1148…	93.044580	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=9
1148…	93.956777	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=1
1148…	94.883434	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=1
1148…	95.796746	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=1
1148…	96.810549	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=1
1148…	97.958186	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=1
1148…	98.886485	10.0.0.253	10.0.0.129	ICMP	422	Echo (ping) reply id=0x15b2, seq=1

- Host B responds to each oversized ping request with a reply, indicating it processed the packets.
- Having no defenses on Host B means it attempts to handle the malformed packets instead of dropping or rejecting them.
- Although modern systems are not typically vulnerable to this attack, Host B still expends resources to respond which degrades performance and can affect the network stack of the machine.

Anomalies: Oversized ICMP Echo Request packets exceeding the normal size limit

Buffer Overflow

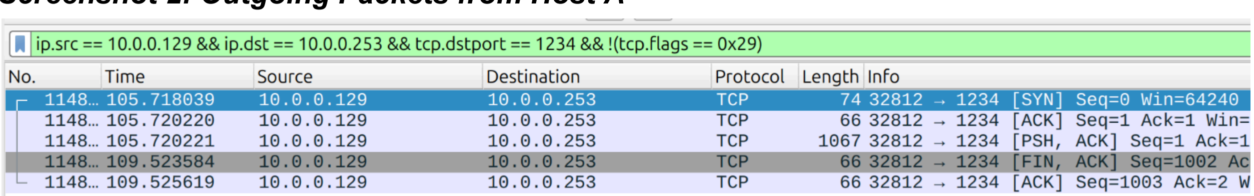
Screenshot 1: Terminal Output on Victim Host B



```
manraj@manraj-lenovo-c940:~$ nc -lv 1234
Listening on 0.0.0.0 1234
Connection received on 10.0.0.129 44350
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAA
```

- Host B is set to listen on port 1234 using netcat and receives the oversized payload.
- Host B successfully processes the payload.

Screenshot 2: Outgoing Packets from Host A



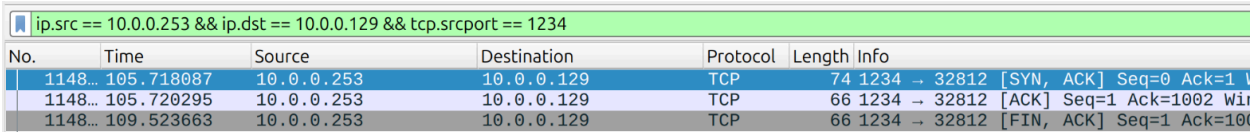
ip.src == 10.0.0.129 && ip.dst == 10.0.0.253 && tcp.dstport == 1234 && !(tcp.flags == 0x29)

No.	Time	Source	Destination	Protocol	Length	Info
1148…	105.718039	10.0.0.129	10.0.0.253	TCP	74	32812 → 1234 [SYN] Seq=0 Win=64240
1148…	105.720220	10.0.0.129	10.0.0.253	TCP	66	32812 → 1234 [ACK] Seq=1 Ack=1 Win=
1148…	105.720221	10.0.0.129	10.0.0.253	TCP	1067	32812 → 1234 [PSH, ACK] Seq=1 Ack=1
1148…	109.523584	10.0.0.129	10.0.0.253	TCP	66	32812 → 1234 [FIN, ACK] Seq=1002 Ac
1148…	109.525619	10.0.0.129	10.0.0.253	TCP	66	32812 → 1234 [ACK] Seq=1003 Ack=2 W

- Host A sends the oversized payload to port 1234 on Host B.
- The attack attempts to cause a buffer overflow on Host B by exceeding the memory allocated for input.

- With no defenses on Host A, it can freely send the oversized payload without restriction, allowing full execution of the attack.

Screenshot 3: Response Packets from Host B



- Host B completes the TCP handshake and acknowledges the oversized payload, responding with ACK and FIN packets.
- With no defenses active on Host B, it handles all incoming packets and payloads without restriction.
- Host B successfully handles the payload, indicating the system has built-in resilience against this type of attack even when nftables and Snort3 are disabled.

Anomalies: Oversized payload of data sent to a single port.

Part 2: Defending with nftables (Firewall) and Snort3 (IDS)

Objective: In this scenario, nftables and Snort3 will be configured on Host B to defend against incoming attacks. The purpose of this task is to analyze how effectively these tools block attacks and detect malicious traffic.

Host B Defense Strategy: nftables will block straightforward attacks like SYN Flood and Ping of Death, which can be easily be handled by the firewall. However, nftables relies on static rules, therefore it can struggle with other types of attacks. An attack that uses slightly modified packet patterns can bypass simple firewall rules. Additionally, blocking traffic too aggressively with nftables can affect legitimate traffic. Thus, Snort3 is used to detect the stealthier attacks like UDP Flood, Xmas Tree Scan, and Buffer Overflow, which require deeper packet analysis. Snort3 will be used as an IDS to alert and detect these attacks, allowing them to be mitigated after detection.

PCAP Files	
Name	**Description**
part2_hosta.pcap	Holds packet capture information from all attacks on the attacker side.
part2_hostb.pcap	Holds packet capture information from all attacks on the victim side.

Log Files		
Attack	**File Name**	**Grep Command**
SYN Flood	part2_hostb_nftables_log.txt	grep 'SYN-FLOOD-ATTEMPT: ' part2_hostb_nftables_log.txt
Ping of Death	part2_hostb_nftables_log.txt	grep 'PING-OF-DEATH: ' part2_hostb_nftables_log.txt
UDP Flood	part2_hostb_snort_log.txt	grep -A 5 'UDP-FLOOD-DETECTED' part2_hostb_snort_log.txt
XMAS Tree Scan	part2_hostb_snort_log.txt	grep -A 5 'XMAS-TREE-SCAN-DETECTED' part2_hostb_snort_log.txt
Buffer Overflow	part2_hostb_snort_log.txt	grep -A 5 'BUFFER-OVERFLOW-DETECTED' part2_hostb_snort_log.txt

Screenshot 1: Host B nftables Configuration File and Rules

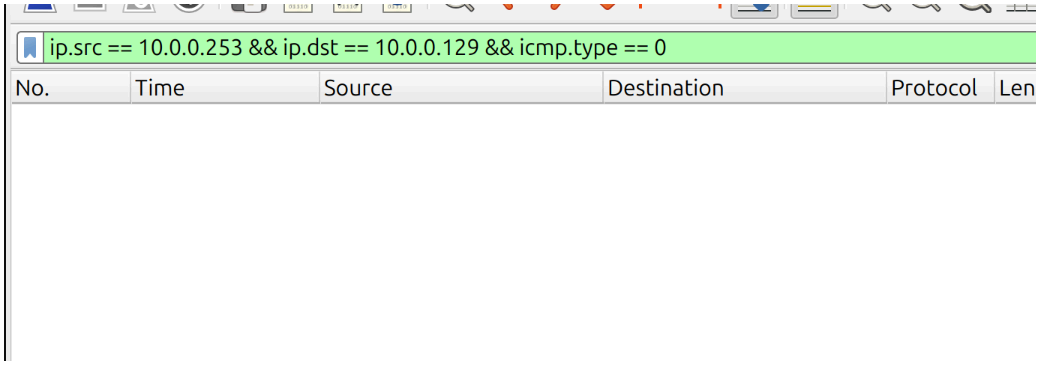
```
#!/usr/sbin/nft -f

flush ruleset

table inet filter {
        chain input {
                type filter hook input priority 0; policy accept;

                # Allow established and related traffic
                ct state established,related accept

                # Drop invalid packets
                ct state invalid drop

                # Open port for nc
                tcp dport 1234 accept

                # Prevent SYN Flood
                tcp flags syn limit rate 20/second log prefix "SYN-FLOOD-ATTEMPT: "
                tcp flags syn limit rate 20/second drop
                tcp flags syn log prefix "SYN-FLOOD-ATTEMPT: "
                tcp flags syn drop

                # Prevent Ping of Death
                ip protocol icmp icmp type echo-request meta length > 1500 log prefix "PING-OF-DEATH: " drop

        }
        chain forward {
                type filter hook forward priority filter;
        }
        chain output {
                type filter hook output priority filter;
        }
}
```

Screenshot 2: Host B Snort3 Rules

```
alert udp any any -> any any (msg:"UDP-FLOOD-DETECTED"; flow:stateless; detection_filter:track by_src, count 200, seconds 1; sid:1000001; rev:1;)
alert tcp any any -> any any (msg:"XMAS-TREE-SCAN-DETECTED"; flags:FPU; sid:1000002; rev:1;)
alert tcp any any -> any any (msg:"BUFFER-OVERFLOW-DETECTED"; dsize:>1000; sid:1000003; rev:1;)
```

SYN Flood

Screenshot 1: Response Packets from Host B



- Host B is no longer sending response packets to the SYN Flood attack indicating the firewall rules are working and the attack is being blocked.

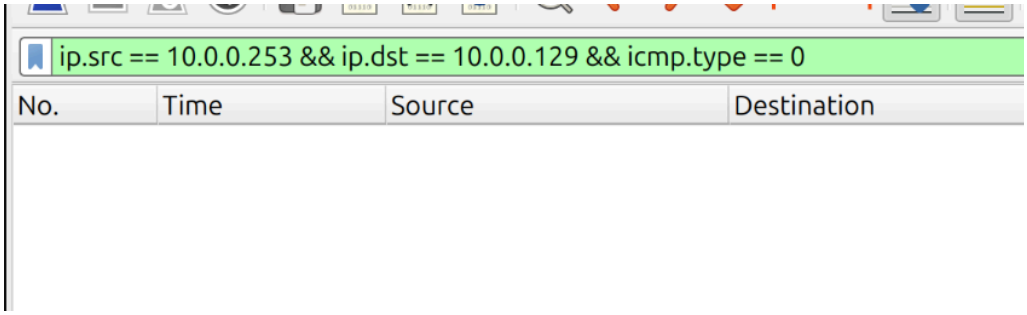
Screenshot 2: Host B nftables Log

```
manraj@manraj-lenovo-c940:~/logs$ grep 'SYN-FLOOD-ATTEMPT' part2_hostb_nftables_log.txt
Nov 27 18:02:07 manraj-lenovo-c940 kernel: SYN-FLOOD-ATTEMPT: IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 SRC=
127.0.0.1 DST=127.0.0.1 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=24278 DF PROTO=TCP SPT=60178 DPT=2321 WINDOW=65495 RES=0x00 SYN
 URGP=0
Nov 27 18:02:15 manraj-lenovo-c940 kernel: SYN-FLOOD-ATTEMPT: IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 SRC=
127.0.0.1 DST=127.0.0.1 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=24279 DF PROTO=TCP SPT=60178 DPT=2321 WINDOW=65495 RES=0x00 SYN
 URGP=0
Nov 27 18:02:32 manraj-lenovo-c940 kernel: SYN-FLOOD-ATTEMPT: IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 SRC=
127.0.0.1 DST=127.0.0.1 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=24280 DF PROTO=TCP SPT=60178 DPT=2321 WINDOW=65495 RES=0x00 SYN
 URGP=0
Nov 27 18:03:04 manraj-lenovo-c940 kernel: SYN-FLOOD-ATTEMPT: IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 SRC=
127.0.0.1 DST=127.0.0.1 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=24281 DF PROTO=TCP SPT=60178 DPT=2321 WINDOW=65495 RES=0x00 SYN
 URGP=0
Nov 27 18:03:26 manraj-lenovo-c940 kernel: SYN-FLOOD-ATTEMPT: IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 SRC=
127.0.0.1 DST=127.0.0.1 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=10581 DF PROTO=TCP SPT=45916 DPT=2321 WINDOW=65495 RES=0x00 SYN
 URGP=0
Nov 27 18:03:27 manraj-lenovo-c940 kernel: SYN-FLOOD-ATTEMPT: IN=wlp0s20f3 OUT= MAC=c0:3c:59:6a:ec:8a:4c:cc:6a:0d:c5:5c:08:
00 SRC=10.0.0.129 DST=10.0.0.253 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=21140 PROTO=TCP SPT=1204 DPT=80 WINDOW=512 RES=0x00 SY
N URGP=0
```

- All SYN flood packets that get dropped are being logged.

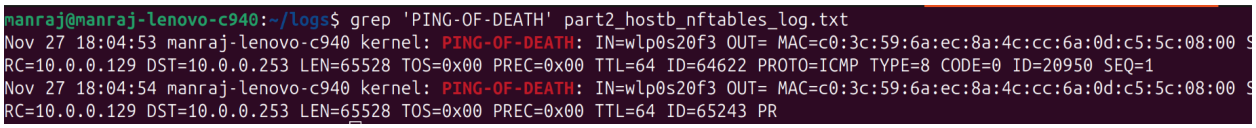
Ping of Death

Screenshot 1: Response Packets from Host B



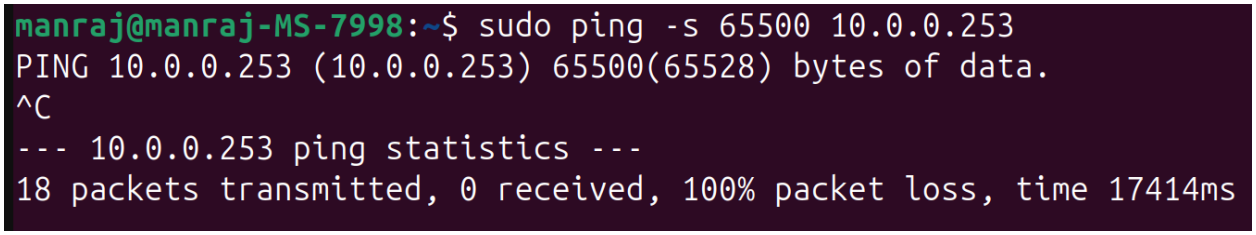
- Host B is no longer sending ICMP reply packets in response to the Ping of Death attack indicating the firewall rules are working and the attack has been blocked.

Screenshot 2: Host B nftables Log



- All Ping of Death packets that are dropped are being logged.

Screenshot 3: Terminal Output of Host A



- The Host A terminal shows that the attack is unable to successfully send the oversized packets as they are not being received.

UDP Flood

```
[**] [1:1000001:1] "UDP-FLOOD-DETECTED" [**]
[Priority: 0]
11/27-18:03:50.140945 10.0.0.129:43007 -> 10.0.0.253:53
UDP TTL:64 TOS:0x0 ID:38036 IpLen:20 DgmLen:28
Len: 0

[**] [1:1000001:1] "UDP-FLOOD-DETECTED" [**]
[Priority: 0]
11/27-18:03:50.140945 10.0.0.129:42997 -> 10.0.0.253:53
UDP TTL:64 TOS:0x0 ID:26579 IpLen:20 DgmLen:28
Len: 0

[**] [1:1000001:1] "UDP-FLOOD-DETECTED" [**]
[Priority: 0]
11/27-18:03:50.140945 10.0.0.129:43022 -> 10.0.0.253:53
UDP TTL:64 TOS:0x0 ID:19945 IpLen:20 DgmLen:28
Len: 0
```

- Snort3 rules to detect UDP Flood packets are working. They are being logged.

XMAS Tree Scan

```
manraj@manraj-lenovo-c940:~/logs$ grep -A 5 'XMAS-TREE-SCAN-DETECTED' part2_hostb_snort_log.txt
[**] [1:1000002:1] "XMAS-TREE-SCAN-DETECTED" [**]
[Priority: 0]
11/27-18:04:00.622564 10.0.0.129:56753 -> 10.0.0.253:1723
TCP TTL:45 TOS:0x0 ID:6777 IpLen:20 DgmLen:40
**U*P**F Seq: 0xC98F39FF  Ack: 0x0  Win: 0x400  TcpLen: 20  UrgPtr: 0x0

[**] [1:1000002:1] "XMAS-TREE-SCAN-DETECTED" [**]
[Priority: 0]
11/27-18:04:00.622578 10.0.0.129:56753 -> 10.0.0.253:587
TCP TTL:50 TOS:0x0 ID:761 IpLen:20 DgmLen:40
**U*P**F Seq: 0xC98F39FF  Ack: 0x0  Win: 0x400  TcpLen: 20  UrgPtr: 0x0

[**] [1:1000002:1] "XMAS-TREE-SCAN-DETECTED" [**]
[Priority: 0]
11/27-18:04:00.622578 10.0.0.129:56753 -> 10.0.0.253:53
TCP TTL:59 TOS:0x0 ID:58159 IpLen:20 DgmLen:40
**U*P**F Seq: 0xC98F39FF  Ack: 0x0  Win: 0x400  TcpLen: 20  UrgPtr: 0x0
```

- Snort3 rules to detect XMAS Tree Scan packets are working. They are being logged.

Buffer Overflow

```
manraj@manraj-lenovo-c940:~/logs$ grep -A 5 'BUFFER-OVERFLOW-DETECTED' part2_hostb_snort_log.txt
[**] [1:1000003:1] "BUFFER-OVERFLOW-DETECTED" [**]
[Priority: 0]
11/27-18:03:13.275333 2607:f8b0:400a:804::200a:443 -> 2604:3d08:7774:cc00:915:56df:5d8a:f8fa:58432
TCP TTL:60 TOS:0x0 ID:0 IpLen:40 DgmLen:1280
***AP*** Seq: 0x22F7212F  Ack: 0x3BB6A45F  Win: 0x417  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1799901587 4153481300
--
[**] [1:1000003:1] "BUFFER-OVERFLOW-DETECTED" [**]
[Priority: 0]
11/27-18:03:13.275949 2607:f8b0:400a:804::200a:443 -> 2604:3d08:7774:cc00:915:56df:5d8a:f8fa:58432
TCP TTL:60 TOS:0x0 ID:0 IpLen:40 DgmLen:1280
***AP*** Seq: 0x22F725E7  Ack: 0x3BB6A45F  Win: 0x417  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1799901587 4153481300
--
[**] [1:1000003:1] "BUFFER-OVERFLOW-DETECTED" [**]
[Priority: 0]
11/27-18:03:47.213754 2607:f8b0:400e:c01::54:443 -> 2604:3d08:7774:cc00:915:56df:5d8a:f8fa:54894
TCP TTL:124 TOS:0x0 ID:0 IpLen:40 DgmLen:1280
***AP*** Seq: 0xD63313B7  Ack: 0xDACF642D  Win: 0x41C  TcpLen: 32
TCP Options (3) => NOP NOP TS: 3367192615 4226112792
```

- Snort3 rules to detect Buffer Overflow packets are working. They are being logged.

Part 3: Attacker-Side Defense

PCAP Files	
Name	**Description**
part3_hosta.pcap	Holds packet capture information from all attacks on the attacker side.
part3_hostb.pcap	Holds packet capture information from all attacks on the victim side.

Log Files		
Attack	**File Name**	**Grep Command**
SYN Flood	part3_hosta_nftables_log.txt	grep 'SYN-FLOOD-ATTEMPT' part3_hosta_nftables_log.txt
Ping of Death	part3_hosta_nftables_log.txt	grep 'PING-OF-DEATH: ' part2_hosta_nftables_log.txt
UDP Flood	part3_hosta_snort_log.txt	grep -A 5 'UDP-FLOOD-DETECTED' part2_hosta_snort_log.txt

XMAS Tree Scan	part3_hosta_snort_log.txt	grep -A 5 'XMAS-TREE-SCAN-DETECTED' part3_hosta_snort_log.txt
Buffer Overflow	part3_hosta_snort_log.txt	grep -A 5 'BUFFER-OVERFLOW-DETECTED' part3_hosta_snort_log.txt

Part 4: Full Defense on Both Hosts

PCAP Files	
Name	**Description**
part4_hosta.pcap	Holds packet capture information from all attacks on the attacker side.
part4_hostb.pcap	Holds packet capture information from all attacks on the victim side.

Log Files	
Attack	**File Name**
SYN Flood	part4_hosta_nftables.txt part4_hostb_nftables.txt
Ping of Death	part4_hosta_nftables.txt part4_hostb_nftables.txt
UDP Flood	part4_hosta_snort_log.txt part4_hostb_snort_log.txt
XMAS Tree Scan	part4_hosta_snort_log.txt part4_hostb_snort_log.txt
Buffer Overflow	part4_hosta_snort_log.txt part4_hostb_snort_log.txt