**Title**: Network Traffic Analysis for Identifying Anomalies, Optimizing Traffic Flow, and Enhancing Security

Internship Company: NTH PIXEL FACTORY

Intern: Grandhi Srinivas Yeswanth

Date: [

**Abstract**

Network security is a critical aspect of modern computing. This project aims to analyze network traffic, detect anomalies, and uncover potential security threats by monitoring real-time packet transmissions. Using Wireshark, Python scripts, and other network analysis tools, different types of traffic were captured, including normal and malicious activities.

Simulated cyber threats such as brute force attacks, DoS attacks, and malicious packet injections were introduced to understand attack patterns and security loopholes. This report details the methodologies used, findings from the analysis, and recommendations to improve network security. The implementation of firewall rules, intrusion detection techniques, and monitoring strategies are also discussed.

Keywords: Network Traffic Analysis, Cybersecurity, Wireshark, Anomaly Detection, Intrusion Detection

**1. Introduction**

1.1 Purpose of the Study

The goal of this project is to:

* Capture and analyse network traffic to understand normal and abnormal patterns.
* Identify security risks such as malware infections, suspicious connections, and performance bottlenecks.
* Simulate cybersecurity threats to study attack patterns and validate detection techniques.
* Provide recommendations to improve network security and prevent cyberattacks.

1.2 Scope of Work

This project focuses on analysing network traffic on a laptop connected via a mobile hotspot. The following activities were performed:

* Network traffic monitoring using Wireshark
* Simulating cyber threats to generate attack traffic
* Analysing captured packets for anomalies
* Implementing security measures such as firewalls and intrusion detection

**2. Methodology**

2.1 Tools Used

|  |  |
| --- | --- |
| **TOOL** | **PURPOSE** |
| Wireshark | Packet capture and analysis |
| Tcpdump | Command-line packet analysis |
| Scapy (Python) | Creating and sending custom network packets |
| Fail2ban | Brute force attack prevention |
| UFW/IPTables | Firewall and traffic filtering |

2.2 Step-by-Step Implementation

2.2.1 Capturing Normal Network Traffic

Procedure

1. Start Wireshark and select the active network interface (mobile hotspot).

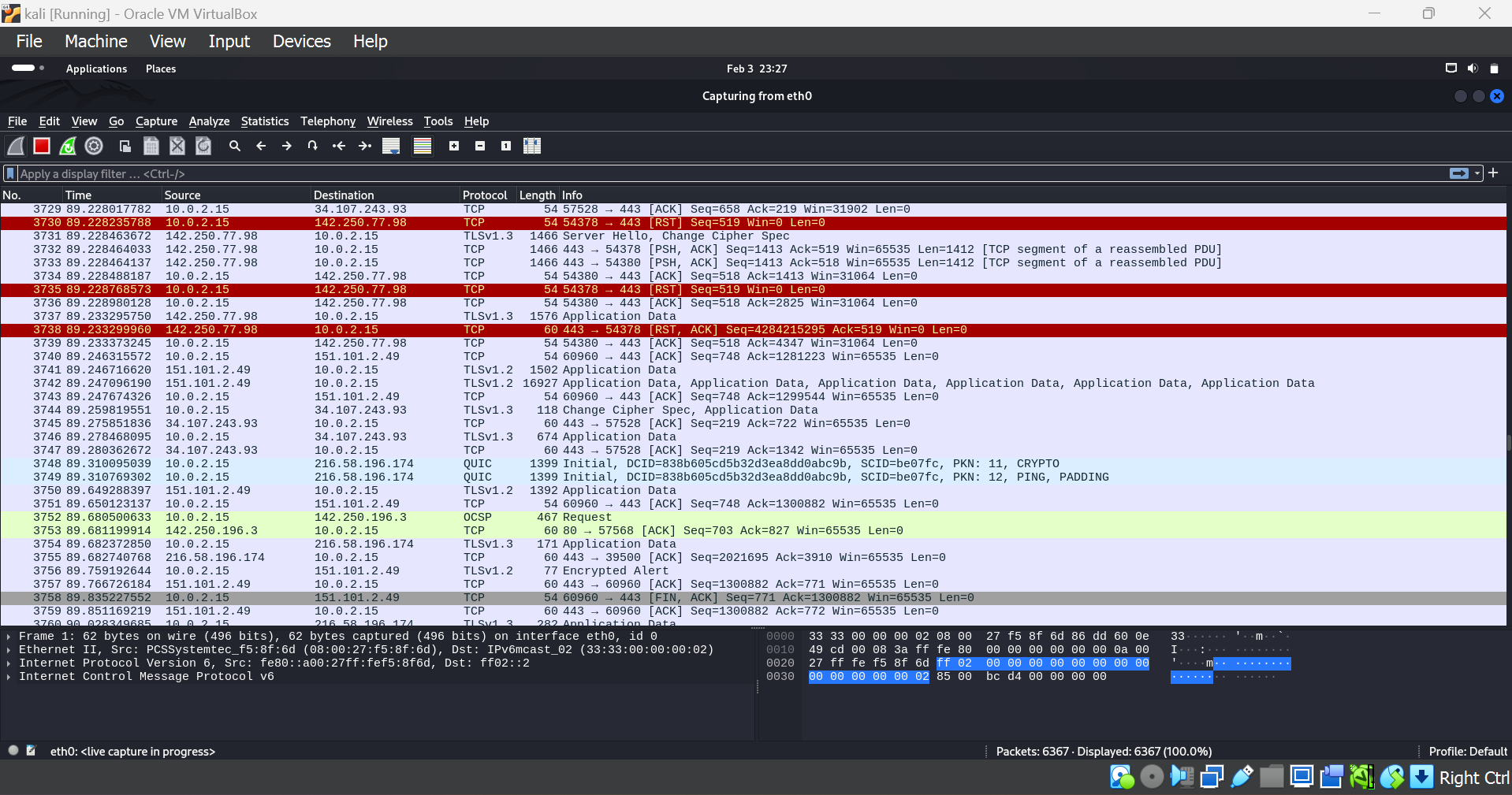
2. Apply filters to monitor essential traffic:

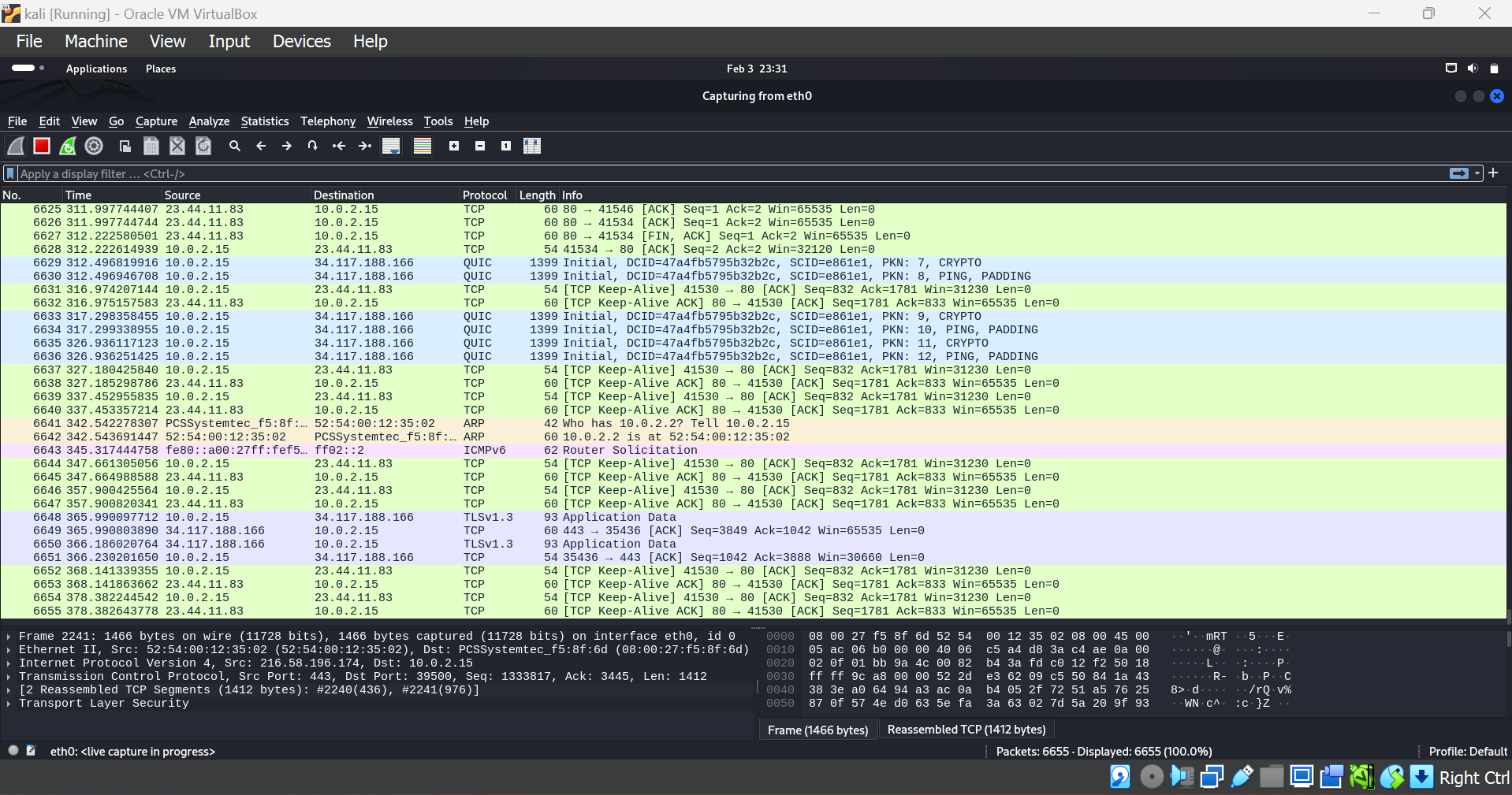
* ip.addr == <Your Device IP>

3. Perform regular network activities:

* Browsing websites (Google, YouTube, Wikipedia)
* Uploading/downloading files
* Using applications (WhatsApp, Instagram)

4. Stop capturing and save the normal traffic PCAP file (normal\_traffic.pcap).

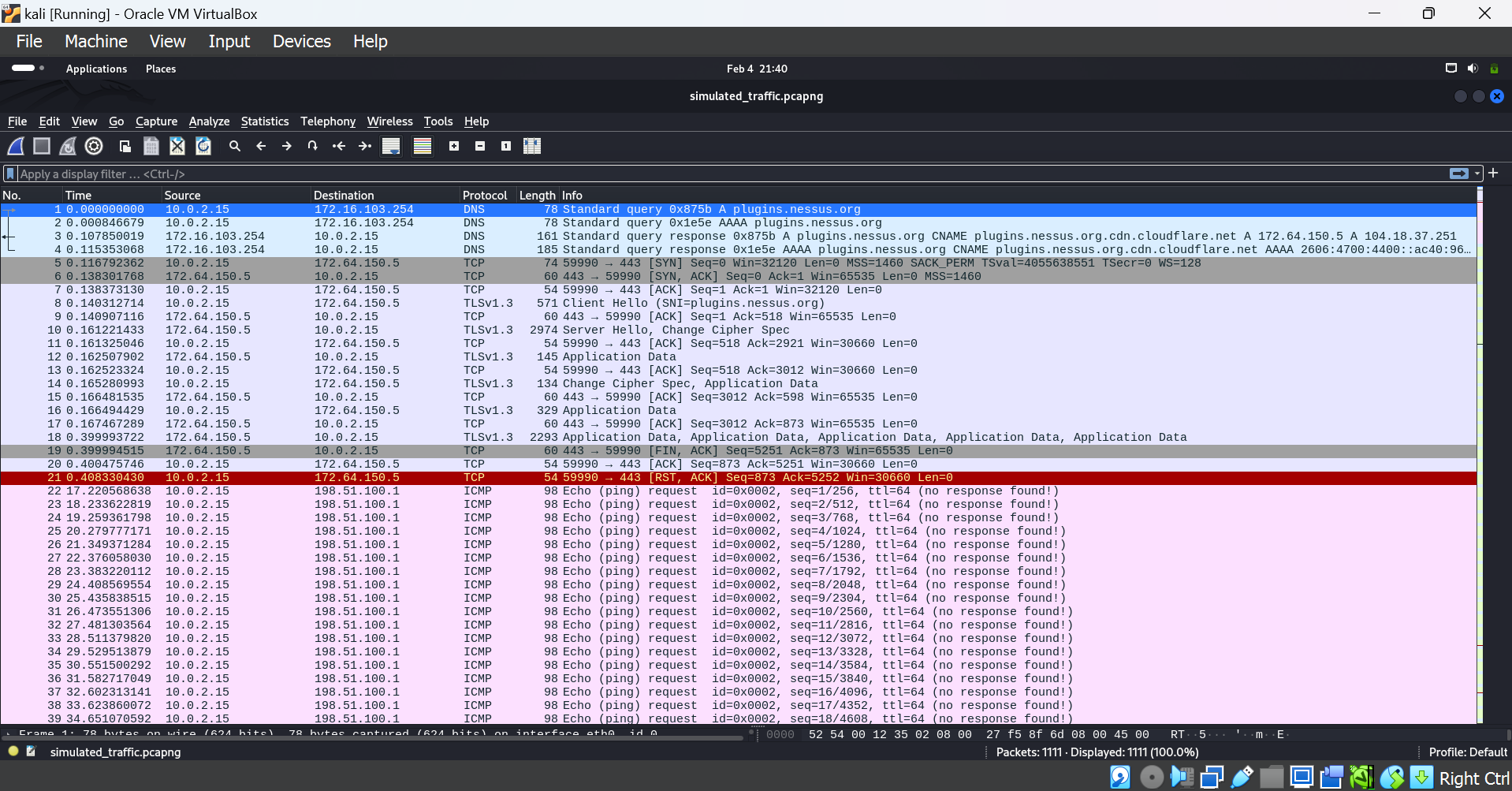




**3. Threat Simulations and Traffic Analysis**

This section details simulated cyber threats and their impact on network traffic.

|  |  |  |  |
| --- | --- | --- | --- |
| **THREAT** | **DESCRIPTION** | **TAFFIC PATTERN OBSERVED** | **PACKET CAPTURE FILNAMES** |
| BruteForce Attack | Multiple login attempts to a test server | Repeated HTTP POST requests | Bruteforce\_attack.pcap |
| Dos Attack(Syn Flood) | Overloading a target with excessive requests | High SYN packet volume | Dos\_attack.pcap |
| Malicious Packet Injection | Injecting fake malware data into the network | Unusual TCP payload | Malicious\_packet.pcap |
| DNS Amplification Attack | Flooding the network with DNS queries | High DNS traffic | Dns\_attack.pcap |



3.1 Brute Force Attack Simulation

|  |
| --- |
| Steps to Simulate:  import requests  url = "http://example.com/login" # Replace with actual test site  for i in range(100):  response = requests.post(url, data={"username": "admin", "password": f"password{i}"})  print(f"Attempt {i}: {response.status\_code}") |

Findings:

* Wireshark shows a flood of HTTP POST requests to /login.
* Detection: Set up Intrusion Detection System (IDS) rules to flag repeated login attempts.

📌 Insert Screenshot of Brute Force Traffic Here

---

3.2 DoS Attack (SYN Flood) Simulation

Steps to Simulate:

from scapy.all import \*

packet = IP(dst="192.168.1.1")/TCP(dport=80, flags="S")

send(packet, count=1000)

Findings:

Massive SYN packet flood detected.

Detection: Implement rate-limiting using firewall rules.

📌 Insert Screenshot of DoS Traffic Here

---

3.3 Malicious Packet Injection Simulation

Steps to Simulate:

from scapy.all import \*

packet = IP(dst="192.168.1.1")/TCP(dport=80)/Raw(load="malicious\_payload")

send(packet, count=50)

Findings:

Wireshark captured suspicious TCP packets with payload anomalies.

Detection: Inspect payloads for signs of malware.

📌 Insert Screenshot of Malicious Packet Here

---

4. Recommendations for Network Security

4.1 Firewall Rules Implementation

Block Suspicious IPs

sudo ufw deny from 198.51.100.1

Restrict Unused Ports

sudo ufw deny 8080

4.2 Preventing Brute Force Attacks

Install fail2ban:

sudo apt install fail2ban

Configure:

sudo nano /etc/fail2ban/jail.local

[ssh]

enabled = true

maxretry = 5

findtime = 600

bantime = 3600

Restart service:

sudo service fail2ban restart

4.3 Mitigating DoS Attacks

Limit SYN Flooding:

sudo iptables -A INPUT -p tcp --syn -m limit --limit 1/s --limit-burst 3 -j ACCEPT

4.4 DNS Security Enhancements

Monitor DNS activity:

sudo apt install dnstop

sudo dnstop eth0

Block unnecessary DNS requests:

sudo ufw deny out to any port 53

---

5. Conclusion

This project successfully analyzed real-world network traffic, simulated cyber threats, and applied defensive security measures. The analysis helped detect anomalies such as:

✅ High SYN request volumes (DoS attack)

✅ Excessive DNS queries (DNS attack)

✅ Repeated login attempts (Brute force attack)

✅ Suspicious packet payloads (Malware injection)

Next Steps

Automate real-time anomaly detection.

Implement machine learning models for proactive security monitoring.

Strengthen network defense by integrating SIEM solutions.

---

6. Attachments

📂 Captured Traffic Files (PCAP Format)

normal\_traffic.pcap

bruteforce\_attack.pcap

dos\_attack.pcap

malicious\_packet.pcap

dns\_attack.pcap

📸 Screenshots of Wireshark Analysis

Normal Traffic

Brute Force Attack

DoS Attack

Malicious Packets

