NETWORK PACKET SNIFFER & FIREWALL – REPORT

1. Introduction

This project involves the development of a real-time packet sniffer and dynamic firewall system with a graphical user interface (GUI). It is designed to monitor, capture, and filter incoming/outgoing network packets based on predefined or user-defined rules. The tool serves both as an educational resource and as a lightweight network analysis utility.

---

2. Abstract

The objective of this project is to integrate live packet capturing with a dynamic firewall rule system within a single desktop application. Built using Python, the project leverages Scapy for packet sniffing, PyQt5 for the GUI, and Matplotlib for live graphical representation of protocol distribution. Firewall rules are persistently stored in JSON format and can be added, removed, or auto-generated during runtime. The tool provides clear logging, colored alerts for blocked traffic, and intuitive rule management, making it suitable for beginners and network learners.

---

3. Tools Used:

- **Python 3.x** – Core language for scripting and integration
- **Scapy** – For capturing and parsing packets
- **PyQt5** – GUI framework for building interactive interfaces
- **Matplotlib** – For generating real-time protocol graphs
- **JSON** – To persist firewall rules across sessions
- **Threading & Queues** – For background packet capture and responsiveness

---

4. Steps Involved in Building the Project

1. **Initialized packet sniffing engine** using Scapy with real-time capture and filtering.
2. **Designed a PyQt5-based GUI** with multiple windows: live packet viewer, firewall manager, and protocol graph.
3. **Implemented real-time filtering** based on user input (protocols and ports).
4. **Integrated a rule manager** to allow IP, port, and protocol-based blocking with persistent JSON storage.
5. **Added a graph viewer** to show protocol distribution over time using Matplotlib.

6. **Auto-generated firewall rules** based on suspicious packet activity and showed real-time popup alerts.
7. **Ensured multithreading** for GUI responsiveness and continuous packet capture.

---

5. Conclusion

This project demonstrates how Python can be effectively used for network traffic analysis, firewall rule management, and visualization within a GUI application. It combines core concepts in cybersecurity, UI/UX design, and real-time systems. The system is extensible, easy to use, and provides foundational understanding of how packet sniffers and firewalls operate behind the scenes.