Project Report

Title: Personal Firewall using Python

Author: [Your Name]

Date: [Submission Date]

Supervisor/Mentor: [Name]

1. Introduction

In today's interconnected digital environment, securing systems from unauthorized access and malicious traffic is critical.
A firewall acts as the first line of defense by monitoring and controlling incoming and outgoing network traffic based on predefined rules.

This project focuses on developing a lightweight personal firewall in Python. The firewall is capable of monitoring packets, applying custom rules
for blocking/allowing traffic, logging suspicious activity, and optionally enforcing system-level blocking using iptables.
Additionally, a graphical user interface (GUI) has been implemented using Tkinter for live monitoring.

2. Objectives

- To design and implement a personal firewall using Python.
- To sniff incoming and outgoing packets using Scapy.
- To define customizable rule sets for filtering traffic based on IPs, Ports, and Protocols.
- To log suspicious or blocked packets for auditing.
- To optionally integrate with iptables for real system-level enforcement (Linux).
- To provide a Tkinter-based GUI for live monitoring and rule management.

3. Tools and Technologies

- Python 3.x
- Scapy (Packet Sniffing)
- Tkinter (GUI)
- iptables (Linux firewall)
- Logging module (for auditing)
- Operating System: Linux (Ubuntu/Kali/Debian recommended)

4. Methodology

Step 1: Captured live packets using Scapy.
Step 2: Defined firewall rules in JSON (block IPs, ports, protocols).
Step 3: Applied rules to allow/block packets.
Step 4: Logged allowed/blocked packets into firewall.log.
Step 5: Used iptables for real packet blocking (optional).
Step 6: Developed Tkinter GUI for live monitoring and rule management.

5. System Architecture

Packets → Scapy Sniffer → Rule Engine → (Allowed / Blocked)
Allowed packets are logged, while blocked packets are logged and optionally dropped using iptables.

6. Implementation

- Config File (rules.json) defines blocked IPs, ports, protocols.
- Python script (firewall.py) processes packets.
- Logging to firewall.log for auditing.
- Optional GUI with Tkinter for live monitoring and rule addition.

7. Results

- Successfully captured packets using Scapy.
- Applied custom rules for filtering.
- Logged allowed/blocked packets.
- Integrated iptables for real packet blocking.
- Developed Tkinter GUI for monitoring.

8. Conclusion

The project successfully demonstrates how a personal firewall can be built in Python.
It provides monitoring and enforcement features with an easy-to-use GUI.
The firewall is lightweight, customizable, and extensible for future improvements.

9. Future Enhancements

- Add allow-list rules.
- Web-based dashboard for remote monitoring.
- Signature-based intrusion detection.
- Real-time alerts (Email/Telegram).
- Cross-platform support (Windows firewall integration).

10. Reflection

This project enhanced my understanding of:
- Network packet structure (IP, TCP, UDP, ICMP).
- Firewall concepts (rules, filtering, logging).
- Practical use of Scapy and iptables.
- GUI design using Tkinter.
- Importance of logging and auditing in security tools.