PROJECT REPORT

PERSONAL FIREWALL USING PYTHON

---

1. INTRODUCTION

With the rapid growth of internet usage and cloud-based services, network security has become a critical concern for individuals and organizations. Unauthorized access, malware communication, and insecure network services pose serious risks. Firewalls act as the first line of defense by monitoring and controlling incoming and outgoing network traffic based on predefined security rules.

This project, **Personal Firewall Using Python**, focuses on building a lightweight firewall application that monitors network traffic, applies rule-based filtering, and logs suspicious activity. The project is designed as a **learning-oriented cybersecurity lab** to help beginners understand firewall logic, packet inspection, and SOC-style monitoring workflows.

---

2. PROBLEM STATEMENT

Traditional enterprise firewalls are complex and abstract for beginners to understand. Many cybersecurity learners struggle to visualize how packets are captured, analyzed, and filtered at a low level.

Key Challenges:

- Lack of hands-on understanding of packet-level traffic
- Difficulty relating firewall theory to real-world SOC operations
- Limited exposure to traffic logging and monitoring concepts

Proposed Solution:

Develop a **Python-based personal firewall** that demonstrates:

- Live packet sniffing
- IP and port-based filtering
- Security logging
- Modular firewall rule design

---

3. OBJECTIVES OF THE PROJECT

The main objectives of this project are:

1. To understand how network packets flow through a system
2. To capture and analyze packets in real time
3. To implement rule-based filtering for IPs and ports

4. To log suspicious network activity
5. To simulate basic firewall behavior similar to SOC environments
6. To strengthen Linux and Python fundamentals for cybersecurity roles

---

4. SCOPE OF THE PROJECT

Included in Scope:

- Packet sniffing using Python
- Rule-based traffic filtering
- Logging of security events
- Modular code structure
- Educational simulation of firewall behavior

Out of Scope:

- Enterprise-grade packet blocking
- Kernel-level firewall enforcement
- High-performance packet filtering
- Production deployment

This project is **strictly for educational purposes**.

---

5. SYSTEM REQUIREMENTS

Hardware Requirements:

- Processor: Dual-core or higher
- RAM: Minimum 4 GB
- Storage: 20 GB free space

Software Requirements:

- Operating System: Linux (Ubuntu recommended)
- Programming Language: Python 3
- Libraries: Scapy
- Tools: Terminal, Text Editor (Nano/VS Code)

---

6. TECHNOLOGIES USED

6.1 Python

Python is used due to its simplicity, readability, and extensive support for networking libraries.

6.2 Scapy

Scapy is a powerful packet manipulation and sniffing library that allows:

- Packet capture
- Protocol analysis
- Packet inspection

6.3 Linux (Ubuntu)

Linux provides:

- Raw socket access
- Strong security model
- Realistic SOC lab environment

---

7. PROJECT ARCHITECTURE

High-Level Architecture:

1. Network Interface
2. Packet Capture Engine (Scapy)
3. Rule Engine (IP & Port Rules)
4. Logging Module
5. Console Output (Monitoring)

Architecture Flow:

Incoming Traffic

↓

Packet Capture

↓

Rule Evaluation

↓

Allowed / Blocked

↓

Logging & Output

---

8. PROJECT MODULES

8.1 Firewall Rules Module (rules.py)

- Defines blocked IP addresses
- Defines blocked destination ports
- Easy to update without changing core logic

8.2 Logging Module (logger.py)

- Records security events
- Maintains audit trail
- Helps in incident investigation

8.3 Firewall Engine (firewall.py)

- Captures packets
- Applies rules
- Prints results
- Logs suspicious traffic

---

9. IMPLEMENTATION DETAILS

Packet Sniffing

The firewall listens to packets using Scapy's sniff() function. Each packet is analyzed for:

- Source IP
- Destination IP
- Protocol
- Destination port

Rule-Based Filtering

Rules are applied in the following order:

1. Source IP check
2. Destination port check
3. Allow traffic if no rule matches

Logging

Blocked or suspicious traffic is logged with:

- Timestamp

- Source IP
- Destination IP / Port
- Reason for blocking

---

10. SAMPLE OUTPUT

Allowed Traffic:

[ALLOWED] 192.168.1.5 -> 142.250.77.14

Blocked IP:

BLOCKED IP: 192.168.1.100 -> 192.168.1.5

Blocked Port:

BLOCKED PORT: 23 from 10.10.10.10

---

11. SECURITY CONCEPTS DEMONSTRATED

- Network traffic monitoring
- Packet-level inspection
- Firewall rule enforcement
- Logging and auditing
- Defensive (Blue Team) security

---

12. SOC WORKFLOW ALIGNMENT

This project aligns with SOC Level 1 responsibilities such as:

- Monitoring network traffic
- Identifying suspicious connections
- Logging security events
- Performing basic incident analysis
- Understanding firewall alerts

---

13. LIMITATIONS OF THE PROJECT

- Does not actively drop packets at kernel level
- Performance is limited compared to hardware firewalls
- Requires root privileges

- Suitable only for learning and labs

---

14. FUTURE ENHANCEMENTS

- GUI-based dashboard
- Dynamic rule updates
- Integration with system firewalls
- Alerting via email or dashboard
- MITRE ATT&CK mapping
- Export logs to SIEM

---

15. RISK ANALYSIS

Risk	Description	Mitigation
Misuse	Packet sniffing abuse	Educational disclaimer
Performance	High traffic overload	Limited lab use
Permissions	Root access required	Controlled environment

---

16. TESTING & VALIDATION

- Verified packet capture during browsing
- Tested blocked IP rules
- Tested blocked ports
- Validated logging functionality

---

17. LEARNING OUTCOMES

- Strong understanding of firewall fundamentals
- Improved Python scripting skills
- Exposure to Linux security concepts
- Practical SOC-style monitoring experience

---

18. CONCLUSION

The **Personal Firewall Using Python** project successfully demonstrates how basic firewall mechanisms work at a packet level. It provides valuable hands-on exposure to network monitoring,

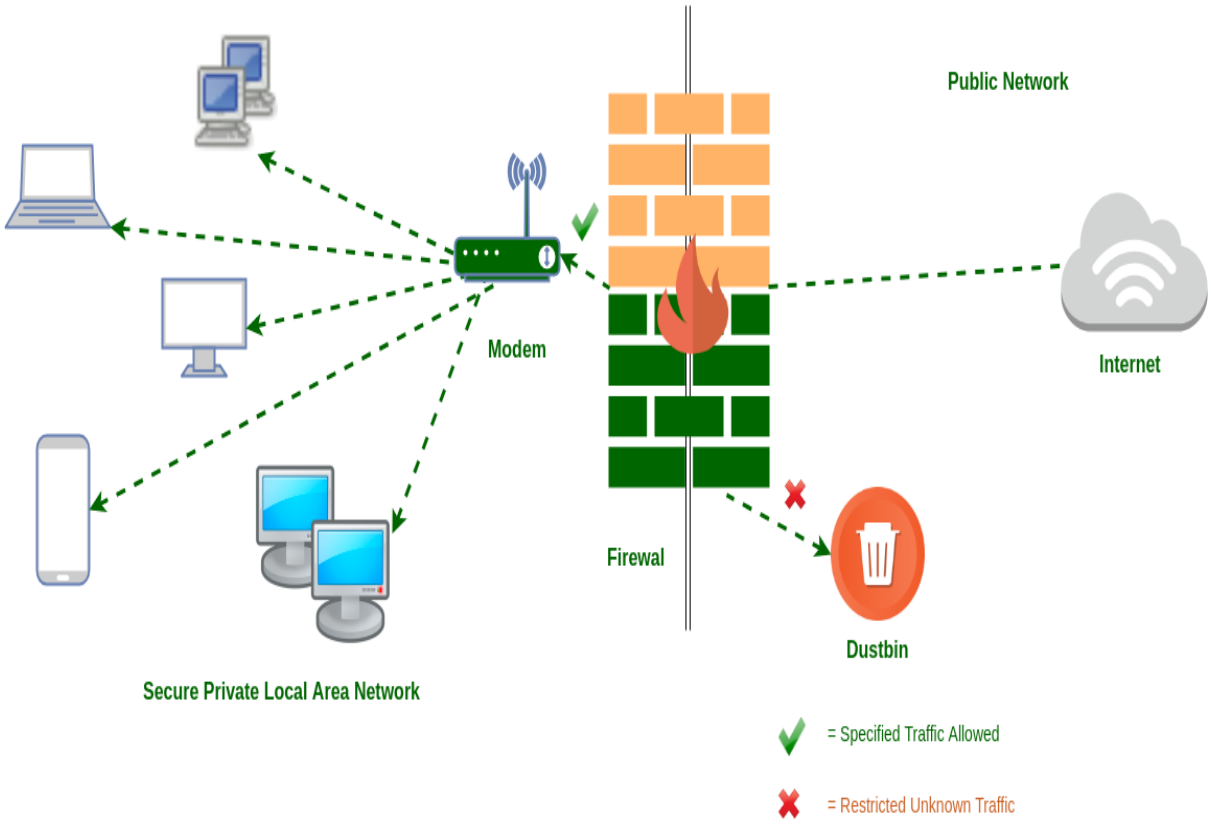
traffic filtering, and logging, making it an excellent foundational project for cybersecurity students and SOC analyst aspirants.

---

19. REFERENCES

- Python Official Documentation
- Scapy Documentation
- Linux Networking Concepts
- Firewall and Network Security Fundamentals

HIGH-LEVEL ARCHITECTURE DIAGRAM



This diagram illustrates the overall architecture of the Python-based personal firewall, showing how network traffic flows from the network interface through packet capture, rule evaluation, logging, and monitoring.

DATA FLOW DIAGRAM