Python Firewall Project Report

Introduction

This project presents the design and implementation of a Python-based Firewall capable of monitoring and controlling network traffic.
The firewall enforces user-defined security policies by blocking suspicious IP addresses, restricting specific protocols, and permitting only approved ports.
To enhance usability, the firewall includes an interactive Graphical User Interface (GUI) developed with Tkinter, enabling both technical and non-technical users to operate it effectively.
Additionally, a logging mechanism records blocked traffic for later analysis, supporting both transparency and traceability.

Abstract

The proposed system integrates packet sniffing, rule-based filtering, and logging into a cohesive framework with a user-friendly interface.
Leveraging Scapy for packet capture, the firewall inspects incoming traffic and applies customizable rules such as IP blocking, port restrictions, and protocol filtering.
Suspicious or disallowed packets are logged with timestamps for auditing.

The GUI provides essential firewall controls including start/stop operations, real-time log viewing, and dynamic rule addition.
This project demonstrates how fundamental concepts of network security, specifically rule-based traffic filtering, can be effectively implemented in Python.
While not designed to replace enterprise-grade firewalls, it offers an accessible learning platform for understanding how firewalls operate at the software level.

Tools and Technologies Used

- Python – Core programming language.
- Scapy – For packet sniffing and traffic inspection.
- Tkinter – For developing the graphical user interface.
- Logging (Python Standard Library) – For recording blocked activities into firewall.log.
- Threading – To run the firewall processes in the background while keeping the GUI responsive.

Implementation Steps

1. Rule Management (rules_manager.py)
- Manages security rules for blocking IPs, restricting protocols, and allowing specific ports.
- Provides functions to add, remove, and update rules dynamically.

2. Packet Filtering (firewall_core.py)
- Captures packets using Scapy.
- Applies rule-based filtering:
  - Blocks traffic from blacklisted IPs.
  - Allows communication only through whitelisted ports.
  - Restricts unwanted protocols (e.g., ICMP, Telnet).
- Logs blocked traffic with reasons for improved traceability.

3. Logging System (logger.py)
- Utilizes Python's built-in logging library.
- Records all blocked activities with timestamps in firewall.log.
- Supports log review for auditing and debugging purposes.

4. Graphical User Interface (gui.py)
- Provides interactive control with start/stop buttons.
- Displays current firewall status (ON/OFF).
- Enables adding/removing rules via popup dialogs.
- Offers a log viewer with real-time updates.

5. Main Application (main.py)
- Initializes the Tkinter GUI.
- Launches the firewall core in the background.
- Serves as the entry point for user interaction.

Conclusion

The project demonstrates the feasibility of building a functional Python-based software firewall that combines packet sniffing, rule enforcement, logging, and GUI-based management.
By simplifying complex firewall mechanisms into a learning-friendly implementation, this project enhances understanding of network security fundamentals.

While the prototype successfully delivers monitoring, control, and logging features, it remains a simplified version compared to enterprise firewalls. Future enhancements could include:

- Persistent rule storage in databases or configuration files.
- Advanced traffic analysis (e.g., deep packet inspection).
- Real-time alerting (email or desktop notifications).
- Integration with Intrusion Detection Systems (IDS).
- Role-based access control for administrative use.

This work not only provides a hands-on understanding of network defense mechanisms but also serves as a strong foundation for further research and development in cybersecurity tools.

System Architecture Diagram

Diagram illustrating flow: Incoming Packets → Firewall Core → Rule Matching → Allow/Block → Logging & GUI

Packet Filtering Workflow

Step 1: Capture Packet → Step 2: Apply Rules → Step 3: Allow or Block → Step 4: Log Action → Step 5: Display in GUI