ELEVATE LABS - Project Report

Abstract

This document presents three practical cybersecurity projects developed using Python to enhance understanding of network defense and threat analysis:

1. A Personal Firewall that monitors and filters network traffic based on user-defined rules.
2. A Cyber Threat Intelligence (CTI) Dashboard that aggregates and visualizes threat data from open intelligence sources.
3. A Honeypot Server designed to detect and analyze attack patterns. Each project demonstrates hands-on application of Python scripting, API integration, and basic security monitoring concepts.

Introduction

Cybersecurity involves proactive defense mechanisms to identify, prevent, and respond to digital threats. Through these projects, the aim was to design simple yet effective tools that emulate core functionalities used in enterprise-grade systems — including firewalls, CTI dashboards, and honeypots.
The implementation focused on network packet handling, threat data aggregation, and intrusion detection simulation, providing a foundation for deeper exploration in security automation and incident analysis.

Tools Used

- Programming Language: Python 3
- Libraries & Frameworks: Scapy, Flask, Requests, Pymongo, Tkinter
- APIs: VirusTotal API, AbuseIPDB API
- Databases: MongoDB
- System Utilities: iptables (Linux), SocketServer
- Additional Tools: Chart.js (for visualizations), Fail2Ban (optional integration)

Steps Involved in Building the Project

1. Personal Firewall

- Used Scapy to sniff and inspect incoming/outgoing packets.
- Created a rules.json file to define allowed or blocked IPs, ports, and protocols.
- Integrated iptables commands through Python to enforce packet filtering.
- Logged suspicious packets to a file for analysis.

- (Optional) Developed a Tkinter-based GUI for live monitoring of network activity.

2. Cyber Threat Intelligence Dashboard

- Built using Flask and MongoDB to manage threat data and user queries.
- Integrated VirusTotal and AbuseIPDB APIs to validate IPs and domains.
- Stored and visualized Indicators of Compromise (IOCs) with timestamps and threat levels.
- Implemented data visualization using Chart.js for trends and lookup frequency.
- Enabled real-time user input, tagging, and result export options.

3. Honeypot Server

- Developed a lightweight TCP-based honeypot simulating SSH-like service.
- Logged attacker IPs, connection attempts, and entered commands to honeypot_logs.jsonl.
- Optionally integrated with Fail2Ban to auto-block persistent malicious IPs.
- Analyzed logs to identify repeated attack patterns and IP geolocation trends.

Conclusion

The projects successfully demonstrate the foundational mechanisms of cybersecurity operations prevention, detection, and analysis.
The Personal Firewall reinforces packet inspection and rule-based filtering; the CTI Dashboard encourages proactive intelligence gathering; and the Honeypot Server aids in understanding real-world attacker behavior.
These implementations provide valuable, hands-on exposure to cybersecurity engineering principles and can be further expanded with automation, visualization, and integration into SIEM platforms for professional-level deployment.