7.NETWORK PACKET SNIFFING WITH ALERT SYSTEM

Introduction

As a beginner in cybersecurity and Python programming, this project was my first step into learning how real-world network monitoring and intrusion detection systems operate. I built a packet sniffer that listens to live network traffic, analyzes it for suspicious patterns, generates alerts, and logs everything persistently. This hands-on experience deepened my understanding of network protocols, attack detection methods, and data management.

Abstract

This project implements a real-time network packet sniffer using Python and the powerful Scapy library. Network packets captured from the system are dissected to reveal key details such as source and destination IP addresses, ports, protocol types, packet sizes, and TCP flags. The program applies basic anomaly detection techniques to identify potential threats including port scanning, packet flooding, and SYN flood denial-of-service attacks. Alert events, together with detailed packet data, are saved in a SQLite database file named *packets.db*. it detects the suspicious port and shows the alert To facilitate analysis, a complementary script retrieves and displays these records in readable tabular form. An optional GUI further enhances live monitoring and interaction.

How the Database Works

When the sniffer starts, it checks for the existence of the *packets.db* database file. If found, the file is deleted to ensure each run starts fresh, avoiding mixing previous sessions' data. The program then creates two fundamental tables:

- Packets table: Stores one row per captured packet, recording source and destination IPs, ports, the protocol used, packet size in bytes, TCP flag information (such as SYN or ACK), and a timestamp recording capture time.
- Alerts table**:** Logs security alerts triggered by suspicious network activity. Each alert includes the alert type (e.g., Port Scanning Attack), the IP address responsible, a descriptive message, the severity level (e.g., HIGH, CRITICAL), and the timestamp.

Persistently saving this information in a database enables reviewing, querying, and analyzing the data at any time — unlike transient console outputs, which vanish once the program stops.

CORE OF CODE'S

The program connects to the SQLite database and sets up the required tables on start, ensuring the schema is ready to store packets and alerts.

Detection Variables

Dictionaries maintain counters and sets to track individual IPs' nature, such as how many packets they sent, which ports they tried, and how many SYN flags have appeared without ACK responses. This state tracking is essential for identifying anomalies.

Alert Creation

Alerts are generated and saved to the database when suspicious patterns are detected. The system formats detailed messages showing severity, alert type, affected IP, description, and time, printing these both to the screen and GUI.

Anomaly Detection Functions

- Port Scanning: Monitors the count of unique ports contacted by an IP in a timeframe, triggering alerts when thresholds are exceeded.
- Packet Flooding: Tracks the volume of packets from each IP, raising warnings or alerts for unusually high rates.
- SYN Flood Detection: Detects high counts of SYN packets without corresponding ACKs, indicating a SYN flood.
- Suspicious Ports: Flags attempts to contact sensitive ports commonly targeted by attackers.

Packet Processing

Each sniffed packet is parsed for relevant information, saved into the database, and passed through the anomaly detection logic.

Reporting and Visualization

The program prints periodic summaries showing active IPs, protocol usage, and recent alerts. A GUI option presents live, scrollable logs and buttons for statistics display and data clearing.

Data Viewing Script

To access and review the stored data conveniently, a helper script reads the database, verifying its readiness, and displays the latest alerts and packets in well-formatted tables using the Pandas and Tabulate libraries. This greatly simplifies data analysis without needing to browse raw logs.

Conclusion

Building this sniffer taught me how packet capture, analysis, and alerting are combined into practical tools that security professionals use daily. The use of persistent databases ensures valuable evidence is retained for audits and investigations. The supplementary data viewer script highlights the need for clear presentation of complex network data. Overall, this project was a rewarding introduction to network security monitoring, bridging programming, protocol analysis, and cybersecurity principles.