Intrusion Detection Systems

B S Vishnu Chasanth AM.EN.U4ECE22113
Devika Vinod AM.EN.U4ECE22116
Niranjan AM.EN.U4ECE22133
Pavithra A P AM.EN.U4ECE22135
Abhijith K AM.EN.U4ECE22149

***Abstract*—As cyber threats evolve, traditional security options such as firewalls and antivirus are no longer adequate. This project demonstrates a lightweight, real-time Network-Based Intrusion Detection System (NIDS) on Wireshark for real-time packet capture and Python for automated detection. Python packages Scapy, PyShark, and Pandas are utilized to identify anomalies and known attack signatures in network traffic. When suspicious behavior is detected, the system records extensive alerts to facilitate proactive handling of threats. The solution provides an efficient and scalable way for open-source tool-based improvement of network security.**

***Index Terms*—Network Security, Intrusion Detection System (IDS), Wireshark, Python, Packet Analysis, Scapy, PyShark, Network Monitoring, Cybersecurity, Real-time Detection**

I. Introduction

In the current hyper-connected digital world, network security is at the top of the priority list for organizations and individuals alike. With a mounting frequency and sophistication of cyberattacks, conventional security practices—like firewalls and antivirus software—are no longer adequate on their own. To alleviate this mounting challenge, Intrusion Detection Systems (IDS) have become a key element of contemporary cybersecurity frameworks.

An IDS will keep an eye out for network traffic to identify malicious activity and possible dangers in real time. Between the two main types of IDS—Host-Based (HIDS) and Network-Based (NIDS)—the latter provides wider coverage by scanning traffic all over the network instead of concentrating on a single endpoint.

This work revolves around designing a Network-Based Intrusion Detection System (NIDS) based on Wireshark for capturing packets and Python for smart traffic analysis. Wireshark captures low-level, protocol-specific information via real-time packet sniffing, whereas Python, augmented with libraries such as Scapy, PyShark, and Pandas, allows for automated identification of anomalies and known attack patterns.

The key objective of the system is to provide a lightweight, scalable, and open-source solution for real-time threat detection. Through the integration of packet-level visibility and automated analysis, the system enables users to act ahead of threats by proactively detecting and responding to malicious activity before it becomes a big breach.

II. Objectives

The main goals of this project are as follows:

To implement and design a Network-Based Intrusion Detection System (NIDS): Use open-source software to scan and inspect network traffic for indication of harmful behavior.

To intercept live network packets with Wireshark or TShark: Allow protocol-level communication examination for raw data examination in real time.

To conduct smart packet analysis with Python: Utilize packages like Scapy, PyShark, and Pandas to parse, filter, and identify anomalies in captured network traffic.

To identify and log unwanted or malicious activities: Identify pre-defined attack patterns and traffic anomalies, and trigger alerts accordingly.

To enable a lightweight and scalable security solution: Make sure the system can be easily implemented in small to medium-scale network environments with low overhead.

To encourage proactive cybersecurity efforts: Enable users to take well-informed action based on early indications of threats to prevent data breaches and downtime.

III. Methodology

The creation of the suggested Network-Based Intrusion Detection System (NIDS) is organized into a set of well-defined steps, each playing a part toward the ultimate objective of real-time threat identification with open-source tools. The system utilizes Wireshark/TShark for packet collection and Python for anomaly detection and analysis of the data. The entire methodology is explained below:

A. Packet Capture using Wireshark/TShark

The initial step is to capture live network traffic from a chosen interface using Wireshark or its command-line equivalent, TShark.Wireshark offers a graphical user interface through which packets are captured and monitored in real time.TShark used in the background enables automation of packet capture into .pcap files.The active network interface is chosen by the user, and packets are captured in real-time or for some time duration.Captured data contain key attributes like source/destination IP, port numbers, protocol types, flags, and payloads.

B. Packet Parsing and Feature Extraction

Once the packet capture is done, Python is employed to parse the .pcap file through libraries like:

PyShark – Python wrapper for TShark to access packet attributes in an easy manner. Scapy – For dissecting, manipulating, and filtering packets with ease. Pandas – For rearranging parsed data in structured table formats (DataFrames) suitable for analysis. The system extracts the following important features: IP addresses (source/destination), Protocol type (TCP, UDP, ICMP, etc.), Packet size, Port numbers, TCP flags (SYN, ACK, FIN, etc.), Time interval between packets.

These features are essential for behavior analysis and anomaly detection.

C. C. Anomaly and Signature-Based Detection

The parsed and structured packet data is analyzed using two approaches:

- **Signature-Based Detection:** Known malicious patterns (e.g., multiple SYN flooding, port scanning, DNS spoofing) and predefined rules are utilized to mark behavior as suspicious.Example: Excessive SYN packets from a single host without completion of TCP handshakes would be an indication of a SYN flood attack.
- **Anomaly Detection:** Behavioral thresholds are defined for packet frequency, protocol usage, or traffic volume.Deviation from normal baseline traffic is labeled as anomalous.Pandas and NumPy statistical functions are utilized to calculate mean, variance, and deviation scores.

D. Logging and Alert Generation

Once an intrusion or anomaly is identified, the system acts immediately on the following steps:

- Log File Generation: Stores detailed information such as timestamp, source IP, protocol, and type of threat.
- Real-Time Alerts: Alerts are printed to the console or stored in JSON/CSV format for external dashboard or SIEM systems integration.
- Future releases can also incorporate email/SMS alerts or visual dashboards.

E. Performance Optimization and Testing:

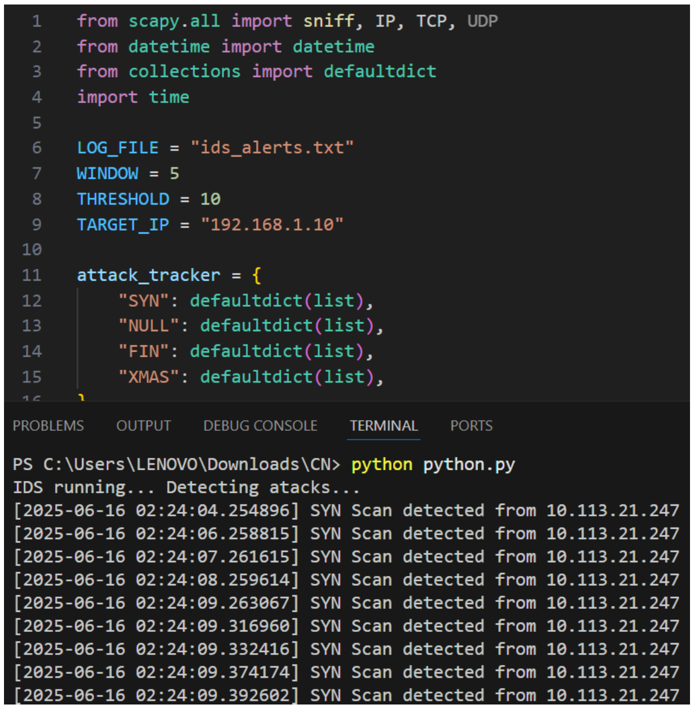
The last phase is to test the system against diverse network conditions with sample attack datasets (e.g., KDD Cup 99 or CIC-IDS2017) and real-time environments. The priority is:

- Minimizing false positives
- Minimizing false positives
- Improving compatibility in different network environments.

IV. Results

The implemented Network-Based Intrusion Detection System (NIDS) was run in a controlled environment to detect and analyze network traffic for reconnaissance operations, i.e., SYN scan attacks. The system was successful in capturing and examining live packets through scapy, implementing logic to identify patterns based on connection attempts without valid handshakes.

- **Attack Simulation:** SYN scan simulated via nmap or bespoke scripts
- **Target IP:** 192.168.1.10
- **Threshold:** 10 SYN packets in a 5-second window from the same IP



```python
from scapy.all import sniff, IP, TCP, UDP
from datetime import datetime
from collections import defaultdict
import time

LOG_FILE = "ids_alerts.txt"
WINDOW = 5
THRESHOLD = 10
TARGET_IP = "192.168.1.10"

attack_tracker = {
    "SYN": defaultdict(list),
    "NULL": defaultdict(list),
    "FIN": defaultdict(list),
    "XMAS": defaultdict(list),
```

```
PS C:\Users\LENOVO\Downloads\CN> python python.py
IDS running... Detecting atacks...
[2025-06-16 02:24:04.254896] SYN Scan detected from 10.113.21.247
[2025-06-16 02:24:06.258815] SYN Scan detected from 10.113.21.247
[2025-06-16 02:24:07.261615] SYN Scan detected from 10.113.21.247
[2025-06-16 02:24:08.259614] SYN Scan detected from 10.113.21.247
[2025-06-16 02:24:09.263067] SYN Scan detected from 10.113.21.247
[2025-06-16 02:24:09.316960] SYN Scan detected from 10.113.21.247
[2025-06-16 02:24:09.332416] SYN Scan detected from 10.113.21.247
[2025-06-16 02:24:09.374174] SYN Scan detected from 10.113.21.247
[2025-06-16 02:24:09.392602] SYN Scan detected from 10.113.21.247
```

The system precisely monitored the rate of suspicious SYN packets and detected scanning attempts with negligible delay. The system is easy to enhance to include support for other scan types (NULL, FIN, XMAS) and log results to facilitate in-depth post-analysis.

This validates the implementation and operation of an efficient, rule-based intrusion detection system for real-time network threat monitoring.

V. Conclusion

This project demonstrates effectively the design and development of a light-weight, real-time Network-Based Intrusion Detection System (NIDS) with the aid of open-source tools like Wireshark and Python. It achieves this by integrating Wireshark's powerful packet capture feature with the analytical potential of Python packages like Scapy, PyShark, and Pandas. The system effectively analyzes network traffic, identifies malicious events, and sends timely alerts.

The system is able to detect common patterns of attack and anomalies by examining protocol behavior and traffic patterns. The modular and scalable nature of the system allows it to be used in small to medium-sized network environments where resource limitation and real-time response are important.

With an ever-changing threat scenario in the field of cybersecurity, this project showcases the power of open-source technologies to create meaningful, smart, and flexible intrusion detection systems. To make it even more robust, future development may involve integrating machine learning, visualization dashboards, and the ability to analyze encrypted traffic to make network defense systems even more robust.

References

[1] Liao, H. J., Lin, C. H. R., Lin, Y. C., Tung, K. Y. (2013). Intrusion detection system: A comprehensive review. Journal of network and computer applications, 36(1), 16-24.

[2] Khraisat, A., Gondal, I., Vamplew, P., Kamruzzaman, J. (2019). Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity, 2(1), 1-22.