Network Traffic Analyzer Project Report

Anouar SAYAH

August 12, 2024

1 Introduction

The Network Traffic Analyzer project aims to develop a tool for monitoring and analyzing network traffic to detect potential security threats in real-time. The project leverages Flask for the web interface and Scapy for packet sniffing and analysis. The goal is to identify threats such as port scans and Distributed Denial of Service (DDoS) attacks by analyzing incoming network packets.

2 Methodology

2.1 Overview

The project consists of two main components:

1. A backend system implemented in Python using Scapy for network packet analysis.
2. A frontend system using Flask to provide a web interface for displaying alerts and monitoring traffic.

2.2 Backend Implementation

The backend is responsible for capturing and analyzing network packets. The system detects port scans and DDoS attacks based on packet frequency and source IP addresses.

2.2.1 Port Scan Detection

Port scans are detected by counting the number of packets from the same IP address to different ports. If the number of packets exceeds a threshold within a given timeframe, an alert is generated.

2.2.2 DDoS Attack Detection

DDoS attacks are detected by counting the number of packets from the same IP address within a specific timeframe. If the packet count exceeds a threshold, an alert is generated.

2.2.3 Code Snippet

```python
from flask import Flask, render_template, jsonify
from scapy.all import sniff, IP, TCP, UDP
import threading
import time
from collections import defaultdict

app = Flask(__name__)

packet_count = defaultdict(int)
ip_count = defaultdict(int)
alerts = []

def detect_port_scan(packet):
    if TCP in packet or UDP in packet:
        ip_src = packet[IP].src
        port = packet[TCP].dport if TCP in packet else packet[UDP].dport
        packet_count[(ip_src, port)] += 1
        if packet_count[(ip_src, port)] > 10:
            alert = f"Alert: Possible port scan detected from {ip_src}"
            alerts.append(alert)
            print(alert)

def detect_ddos(packet):
    ip_src = packet[IP].src
    ip_count[ip_src] += 1
    if ip_count[ip_src] > 100:
        alert = f"Alert: Possible DDoS attack detected from {ip_src}"
        alerts.append(alert)
        print(alert)

```

```python
def packet_callback(packet):
    if IP in packet:
        detect_port_scan(packet)
        detect_ddos(packet)
        ip_src = packet[IP].src
        ip_dst = packet[IP].dst
        print(f"Packet: {ip_src} -> {ip_dst}")

def reset_counters():
    global packet_count, ip_count
    while True:
        time.sleep(60)
        packet_count = defaultdict(int)
        ip_count = defaultdict(int)

def start_sniffing():
    sniff(prn=packet_callback, store=0)

sniff_thread = threading.Thread(target=start_sniffing)
sniff_thread.daemon = True
sniff_thread.start()

reset_thread = threading.Thread(target=reset_counters)
reset_thread.daemon = True
reset_thread.start()

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/alerts')
def get_alerts():
    return jsonify(alerts)

if __name__ == '__main__':
    app.run(debug=True)
```

Listing 1: Python Code for Network Traffic Analyzer

2.3 Frontend Implementation

The frontend is implemented using Flask and provides a web interface to display alerts. The page automatically refreshes every 5 seconds to show the latest alerts.

2.3.1 HTML and JavaScript Code

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Network Traffic Analyzer</title>
    <style>
        body { font-family: Arial, sans-serif; margin: 0; padding: 0; background-color: #f4f4f4; }
        .container { width: 80%; margin: auto; overflow: hidden; }
        #alerts { margin: 20px 0; padding: 20px; background: #fff; border-radius: 8px; box-shadow: 0 0 10px rgba(0, 0, 0, 0.1); }
        .alert { padding: 10px; margin: 10px 0; border-left: 6px solid #ff6f61; background-color: #ffe6e6; color: #333; }
    </style>
</head>
<body>
    <div class="container">
        <h1>Network Traffic Analyzer</h1>
        <div id="alerts">
            <h2>Alerts</h2>
            <div id="alert-list"></div>
        </div>
    </div>
    <script>
        function fetchAlerts() {
            fetch('/alerts')
                .then(response => response.json())
                .then(data => {
                    const alertList = document.getElementById('alert-list');
                    alertList.innerHTML = '';
                    data.forEach(alert => {
                        const alertDiv = document.createElement('div');
                        alertDiv.className = 'alert';
                        alertDiv.innerText = alert;
                        alertList.appendChild(alertDiv);
                    });
                });
        }
        setInterval(fetchAlerts, 5000);
    </script>
</body>
```

```
</html>
```

Listing 2: HTML Code for Frontend

3 Results

The Network Traffic Analyzer successfully detects port scans and DDoS attacks based on network traffic. Alerts are generated and displayed in real-time on the web interface. The detection thresholds are configurable, allowing for adaptation to different network environments.

4 Conclusion

The project demonstrates a practical application of network traffic analysis for security purposes. The combination of Flask for the web interface and Scapy for packet analysis provides a robust solution for monitoring network traffic and detecting potential threats.

5 Future Work

Future enhancements could include:

- Integration with a database for persistent storage of alerts.
- More sophisticated detection algorithms and heuristics.
- User authentication and access control for the web interface.