Name:Ankita Kolapte

Intern Number:270

Submitted To: Digisuraksha Parhari Foundation

Proof of Concept (PoC) - Lightweight Intrusion Prevention System (IPS)

1. Introduction

This Proof of Concept (PoC) demonstrates the development of a lightweight Intrusion Prevention System (IPS) using Python, Scapy, and iptables. The IPS is capable of detecting and blocking malicious traffic patterns such as ICMP floods, SYN floods, SQL injection attempts, and suspicious TCP connections. The goal is to provide a simple but functional IPS that prevents common network-based attacks in real-time.

2. Objectives

The IPS should:

- - Block ICMP ping floods.
- - Drop repeated TCP SYN floods or half-open connections.
- - Prevent simple scan patterns (SYN/NULL/FIN scans, repeated port attempts).
- - Detect and block simple malicious payloads (e.g., SQL injection strings).
- - Dynamically block and unblock IP addresses using iptables.

3. Implementation

The IPS is implemented in Python using the Scapy library for packet sniffing and inspection. Malicious traffic is identified based on thresholds and payload inspection. Detected malicious IPs are blocked using iptables rules. IP addresses are automatically unblocked after a fixed timeout period.

4. Core IPS Code (Simplified)

Below is the core Python implementation of the IPS logic:

```python
from scapy.all import sniff, IP, TCP, ICMP
import os
from collections import defaultdict
import time

# Track packets per IP
packet_count = defaultdict(int)
last_time = defaultdict(float)

# Thresholds
ICMP_THRESHOLD = 20
SYN_THRESHOLD = 30
BLOCK_TIME = 60
blocked_ips = {}

def block_ip(ip):
    os.system(f"iptables -A INPUT -s {ip} -j DROP")
    blocked_ips[ip] = time.time()

def unblock_ips():
    current = time.time()
    for ip in list(blocked_ips.keys()):
        if current - blocked_ips[ip] > BLOCK_TIME:
            os.system(f"iptables -D INPUT -s {ip} -j DROP")
            del blocked_ips[ip]

def detect(packet):
    if IP in packet:
        src_ip = packet[IP].src
        current_time = time.time()
        if current_time - last_time[src_ip] > 5:
            packet_count[src_ip] = 0
            last_time[src_ip] = current_time
        packet_count[src_ip] += 1
        if ICMP in packet and packet_count[src_ip] > ICMP_THRESHOLD:
            block_ip(src_ip)
        if TCP in packet and packet[TCP].flags == "S":
```

```
        if packet_count[src_ip] > SYN_THRESHOLD:
            block_ip(src_ip)
    unblock_ips()

sniff(prn=detect, store=0)
```

5. Demo & Testing

The IPS was tested using PCAPs and live traffic generation tools:

- - Normal traffic PCAP (allowed).
- - ICMP flood traffic (blocked).
- - SYN flood traffic (blocked).
- - SQL injection payload in HTTP (blocked).

6. Deliverables

The PoC produces the following deliverables:

- - Demo execution with normal and malicious PCAPs.
- - Short report (this document).
- - Unit/Integration tests for detection and blocking functions.

7. Conclusion

This PoC successfully demonstrates a basic Intrusion Prevention System implemented in Python. It shows real-time detection and blocking of ICMP floods, SYN floods, and SQL injection attempts. The system is lightweight and can be further extended with more advanced detection rules, logging, and integration with enterprise-grade security monitoring tools.