Network IDS

Name: Harshali Kasurde

Intern id: 299

Network IDS- detect pings, connection attempts and basic exploit/ scan activity

- **Build a lightweight Network Intrusion Detection System (IDS) prototype that:**
  1. To implement a basic Intrusion Detection System (IDS) in Python that can analyse packet capture (PCAP) files and detect suspicious activities such as: (Flooding meaning Sending a huge number of packets/requests to a target system in a very short time. It Overload the system so it becomes slow, crashes, or can't serve real users.)
     - ICMP Flooding (Ping Floods) = Attacker sends tons of ICMP ping requests. Target spends all its time replying → system slows down.
     - SYN Flooding / SYN Scans =Attacker sends many TCP SYN requests but never completes the handshake. Server keeps waiting, uses up memory/resources, and gets overloaded.
  2. The IDS should correctly differentiate between:
     - Normal traffic PCAP (benign browsing/pings).
     - Attack traffic PCAP (scans, floods).

- **Why this is useful:**
  1. Attackers often use ping sweeps and port scans as the first step before launching bigger attacks.
  2. Flooding attacks (ICMP flood, SYN flood) can overload servers and cause Denial of Service (DoS).
  3. IDS tools like Snort and Suricata detect such attacks in real networks.
  4. This PoC shows how to detect them at a small-scale using Python.

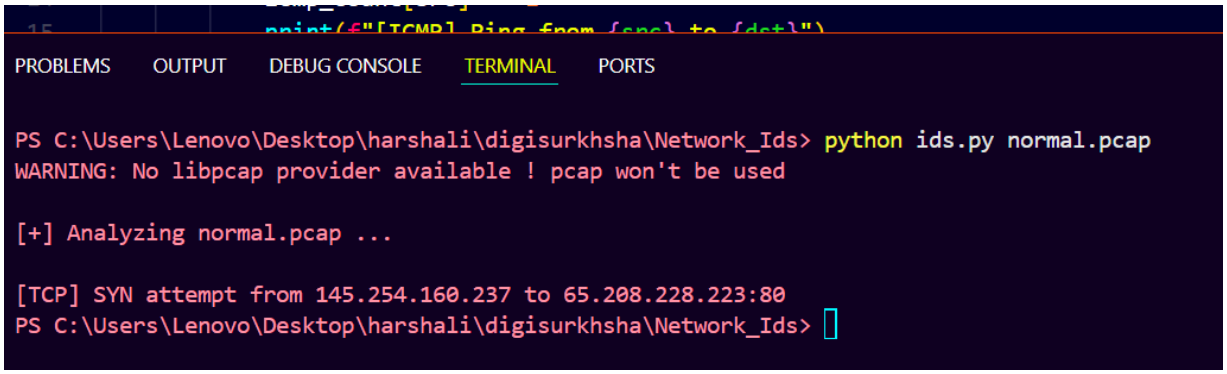
- **Tools & Setup:**
  1. Language: Python
  2. Python Library: Scapy → for packet parsing
  3. Test Data: PCAP files (downloaded from Wireshark Sample Captures , store same folder of a python code and rename file name with extension .pcap)
     - normal.pcap → contains benign traffic (like browsing or a single connection).
     - test.pcap → contains SYN attempts (possible scanning).

- **IDS Code (ids.py):**

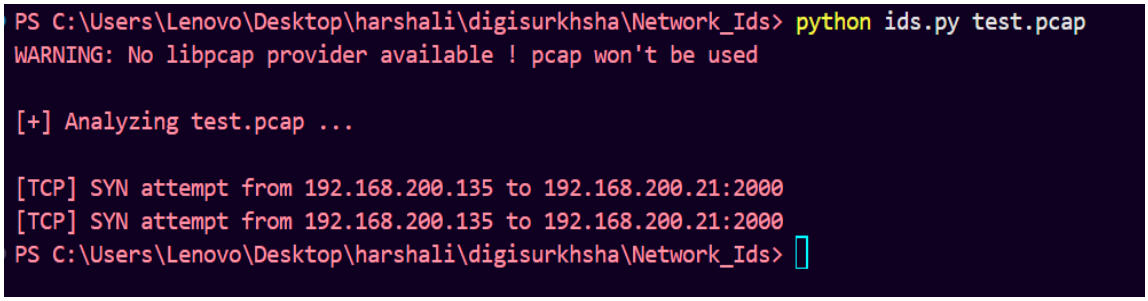
```python
import sys
from scapy.all import rdpcap, IP, ICMP, TCP
from collections import defaultdict

icmp_count = defaultdict(int)
syn_count = defaultdict(int)

# ----------- ICMP Detection -----------
def detect_icmp(pkt):
    if pkt.haslayer(ICMP):
        if pkt[ICMP].type == 8:  # Echo Request (Ping)
            src = pkt[IP].src
            dst = pkt[IP].dst
            icmp_count[src] += 1
            print(f"[ICMP] Ping from {src} to {dst}")

            if icmp_count[src] > 5:
                print(f"[ALERT] Possible ICMP flood from {src}")

# ----------- TCP Detection -----------
def detect_tcp(pkt):
    if pkt.haslayer(TCP):
        flags = pkt[TCP].flags
        src = pkt[IP].src
        dst = pkt[IP].dst
        dport = pkt[TCP].dport

        if flags == "S":  # SYN
            syn_count[src] += 1
            print(f"[TCP] SYN attempt from {src} to {dst}:{dport}")
            if syn_count[src] > 10:
                print(f"[ALERT] Possible SYN scan/flood from {src}")

        elif flags == 0:  # NULL scan
            print(f"[TCP] NULL scan detected from {src} to {dst}:{dport}")

        elif flags == "F":  # FIN scan
            print(f"[TCP] FIN scan detected from {src} to {dst}:{dport}")

# ----------- Analyze PCAP File -----------
def analyze_pcap(filename):
    print(f"\n[+] Analyzing {filename} ...\n")
    packets = rdpcap(filename)
    for pkt in packets:
        if pkt.haslayer(IP):
            detect_icmp(pkt)
            detect_tcp(pkt)
```

```python
# ----------- Main Program -----------
if __name__ == "__main__":
    if len(sys.argv) < 2:
        print("Usage: python ids.py <pcapfile>")
    else:
        pcap_file = sys.argv[1]
        analyze_pcap(pcap_file)

```

- **Execution Steps:**
  1. Run IDS on normal traffic PCAP
     Normal traffic PCAP → should show *no alerts* (just maybe a few packets logged).
     python ids.py normal.pcap



  2. Run IDS on attack traffic PCAP
     Attack traffic PCAP (scans/pings) → should trigger *alerts* (Possible SYN flood, ICMP flood, etc.).
     python ids.py test.pcap



- **Observations:**
  1. The IDS correctly did not flag normal traffic.
  2. The IDS raised alerts for attack traffic (SYN scan/flood).
  3. This demonstrates the ability to differentiate between safe and malicious traffic.

- **Conclusion:**

1. On **normal.pcap**, the IDS showed no alerts (safe behaviour).
2. On **test.pcap**, the IDS flagged suspicious SYN flood activity.