

**Barath Raj M**

**241901015**

## **EXPT: 6 IMPLEMENTATION OF PACKET SNIFFING USING RAW SOCKETS IN PYTHON**

### **Introduction:**

Packet sniffing reads raw network packets from your NIC so you can see headers (Ethernet/IP/TCP/UDP) and a bit of payload. This simple experiment uses Linux raw sockets to capture and print a brief summary for each IPv4 packet.

### **Aim:**

Write a minimal Python program that captures packets using a raw socket and prints source/destination IP, protocol, ports (if TCP/UDP) and a short hex dump of the payload.

### **Algorithm :**

1. Open a raw AF\_PACKET socket (capture all EtherTypes).
2. Loop: receive a packet.
3. Parse Ethernet header; if IPv4, parse IP header.
4. If TCP/UDP, parse ports. Print a one-line summary + short hex of payload.
5. Repeat until Ctrl+C.

### **Code:**

```
import socket
import struct
import binascii
import textwrap

def main():
    # Get host
    host = socket.gethostbyname(socket.gethostname())

    print('IP: {}'.format(host))

    # Create a raw socket and bind it
```

```

conn = socket.socket(socket.AF_INET, socket.SOCK_RAW, socket.IPPROTO_IP)

conn.bind((host, 0))

# Include IP headers
conn.setsockopt(socket.IPPROTO_IP, socket.IP_HDRINCL, 1)

# Enable promiscuous mode
conn.ioctl(socket.SIO_RCVALL, socket.RCVALL_ON)

while True:

    # Recive data

    raw_data, addr = conn.recvfrom(65536)

    # Unpack data

    dest_mac, src_mac, eth_proto, data = ethernet_frame(raw_data)

    print("\nEthernet Frame:")

    print("Destination MAC: {}".format(dest_mac))

    print("Source MAC: {}".format(src_mac))

    print("Protocol: {}".format(eth_proto))

# Unpack ethernet frame
def ethernet_frame(data):

    dest_mac, src_mac, proto = struct.unpack('!6s6s2s', data[:14])

    return get_mac_addr(dest_mac), get_mac_addr(src_mac), get_protocol(proto), data[14:]

# Return formatted MAC address AA:BB:CC:DD:EE:FF
def get_mac_addr(bytes_addr):

    bytes_str = map('{:02x}'.format, bytes_addr)

    mac_address = ':'.join(bytes_str).upper()

    return mac_address

# Return formatted protocol ABCD
def get_protocol(bytes_proto):

    bytes_str = map('{:02x}'.format, bytes_proto)

```

```
protocol = ".join(bytes_str).upper()

return protocol

main()
```

**Output:**

```
C:\Windows\System32>cd "C:\Users\a8282\OneDrive\Documents

C:\Users\a8282\OneDrive\Documents>python packetsniff.py
IP: 10.77.0.213

Ethernet Frame:
Destination MAC: 45:00:00:34:BC:49
Source MAC: 40:00:80:06:61:39
Protocol: 0A4D

Ethernet Frame:
Destination MAC: 45:00:01:6E:D3:7B
Source MAC: 00:00:01:11:00:00
Protocol: 0A4D

Ethernet Frame:
Destination MAC: 45:00:01:6E:D3:7B
Source MAC: 00:00:01:11:F8:E6
Protocol: 0A4D

Ethernet Frame:
Destination MAC: 45:00:00:45:D3:7C
Source MAC: 00:00:01:11:00:00
Protocol: 0A4D

Ethernet Frame:
Destination MAC: 45:00:00:45:D3:7C
Source MAC: 00:00:01:11:FA:0E
Protocol: 0A4D

Ethernet Frame:
Destination MAC: 45:00:00:45:D3:7D
Source MAC: 00:00:01:11:00:00
Protocol: 0A4D
```

**Result:**

The Python program for packet sniffing using raw sockets was executed successfully. It captured live network packets and displayed the source IP, destination IP, protocol type, port numbers, and part of the data in hexadecimal form.

