

Aim

The aim is to receive network packets directly from the interface, decode Ethernet frame information (like source MAC, destination MAC, and protocol), and print these details. This method is often used in packet sniffing, analysis, or learning how network protocols work.

Introduction

This program uses raw sockets in Python to capture and display Ethernet frames from the network interface. It helps understand how network data travels at the lowest (data link) layer and is useful for learning about networking and cybersecurity.

Procedure

- The program finds the computer's IP address and prints it.
- It creates a raw socket, binds it to the network interface, and enables a mode to receive all network traffic, not just what is meant for the local device.
- It runs in an infinite loop, receiving network frames.
- For each received frame, it unpacks header details, then prints out the MAC addresses and protocol information of that frame for analysis.

Program:

```
import socket
import struct
import binascii
import textwrap

def main():
    host = socket.gethostname(socket.gethostname())
    print('IP: {}'.format(host))

    conn = socket.socket(socket.AF_INET, socket.SOCK_RAW, socket.IPPROTO_IP)
    conn.bind((host, 0))
    conn.setsockopt(socket.IPPROTO_IP, socket.IP_HDRINCL, 1)
    conn.ioctl(socket.SIO_RCVALL, socket.RCVALL_ON)

    while True:
        raw_data, addr = conn.recvfrom(65536)
        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))

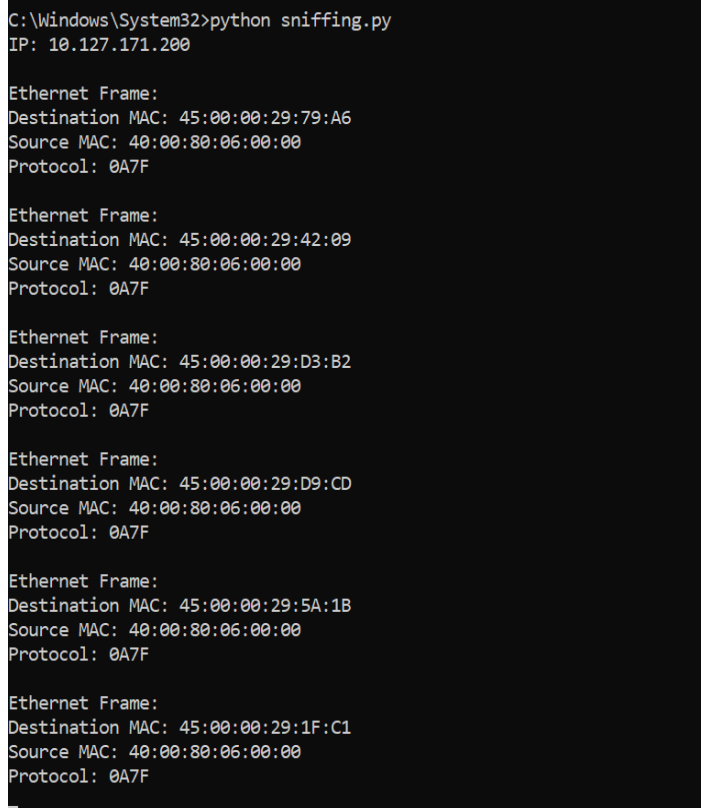
def ethernet_frame(data):
    dest_mac, src_mac, proto = struct.unpack('!6s6s2s', data[:14])
    return get_mac_addr(dest_mac), get_mac_addr(src_mac), get_protocol(proto), data[14:]

def get_mac_addr(bytes_addr):
    bytes_str = map('{:02x}'.format, bytes_addr)
    mac_address = ':'.join(bytes_str).upper()
    return mac_address

def get_protocol(bytes_proto):
    bytes_str = map('{:02x}'.format, bytes_proto)
    protocol = ''.join(bytes_str).upper()
    return protocol

main()

```



```

C:\Windows\System32>python sniffing.py
IP: 10.127.171.200

Ethernet Frame:
Destination MAC: 45:00:00:29:79:A6
Source MAC: 40:00:80:06:00:00
Protocol: 0A7F

Ethernet Frame:
Destination MAC: 45:00:00:29:42:09
Source MAC: 40:00:80:06:00:00
Protocol: 0A7F

Ethernet Frame:
Destination MAC: 45:00:00:29:D3:B2
Source MAC: 40:00:80:06:00:00
Protocol: 0A7F

Ethernet Frame:
Destination MAC: 45:00:00:29:D9:CD
Source MAC: 40:00:80:06:00:00
Protocol: 0A7F

Ethernet Frame:
Destination MAC: 45:00:00:29:5A:1B
Source MAC: 40:00:80:06:00:00
Protocol: 0A7F

Ethernet Frame:
Destination MAC: 45:00:00:29:1F:C1
Source MAC: 40:00:80:06:00:00
Protocol: 0A7F

```

RESULT:

Thus With help of python sniffing program ,we got the ethernet frames.