network.py 3 ●

```python
from scapy.all import *
from scapy.layers.inet import IP, TCP, UDP, ICMP

# Function to analyze packets
def analyze_packet(packet):
    # Check if the packet has an IP layer
    if IP in packet:
        ip_layer = packet[IP]
        src_ip = ip_layer.src
        dst_ip = ip_layer.dst
        proto = ip_layer.proto

        # Determine the protocol
        if proto == 6:
            protocol = "TCP"
        elif proto == 17:
            protocol = "UDP"
        elif proto == 1:
            protocol = "ICMP"
        else:
            protocol = "Other"

        print(f"Source IP: {src_ip}")
        print(f"Destination IP: {dst_ip}")
        print(f"Protocol: {protocol}")

        # Check if the packet has a TCP/UDP/ICMP layer and display payload data
        if TCP in packet and protocol == "TCP":
            tcp_layer = packet[TCP]
            print(f"Source Port: {tcp_layer.sport}")
            print(f"Destination Port: {tcp_layer.dport}")
            print(f"Payload: {bytes(tcp_layer.payload)}")
        elif UDP in packet and protocol == "UDP":
            udp_layer = packet[UDP]
            print(f"Source Port: {udp_layer.sport}")
            print(f"Destination Port: {udp_layer.dport}")
            print(f"Payload: {bytes(udp_layer.payload)}")
        elif ICMP in packet and protocol == "ICMP":
```

```
    # Check if the packet has a TCP/UDP/ICMP layer and display payload data
    if TCP in packet and protocol == "TCP":
        tcp_layer = packet[TCP]
        print(f"Source Port: {tcp_layer.sport}")
        print(f"Destination Port: {tcp_layer.dport}")
        print(f"Payload: {bytes(tcp_layer.payload)}")
    elif UDP in packet and protocol == "UDP":
        udp_layer = packet[UDP]
        print(f"Source Port: {udp_layer.sport}")
        print(f"Destination Port: {udp_layer.dport}")
        print(f"Payload: {bytes(udp_layer.payload)}")
    elif ICMP in packet and protocol == "ICMP":
        icmp_layer = packet[ICMP]
        print(f"Type: {icmp_layer.type}")
        print(f"Code: {icmp_layer.code}")
        print(f"Payload: {bytes(icmp_layer.payload)}")

    print("-" * 80)

# Function to start sniffing
def start_sniffing(interface):
    print(f"Starting packet sniffing on {interface}...")
    sniff(iface=interface, prn=analyze_packet, store=False)

if __name__ == "__main__":
    # Replace 'eth0' with the network interface you want to sniff
    # You can find the interface name using the 'ifconfig' command on Unix-based systems or 'ipconfig' on Windows
    interface = "eth0"
    start_sniffing(interface)
```

```
[Running] python -u "c:\Users\alokr\Downloads\network.py"
WARNING: Wireshark is installed, but cannot read manuf !
Starting packet capture...

Timestamp: 1722266554.010646
Non-IP Packet
Payload: b'`\x01\xb6i\x00:\x11@\xfe\x80\x00\x00\x00\x00\x00\x00\xc4AO&\x83\xc3\x10\xf5\xfe\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01\xe3\xab\x005\x00:\x83
[\xb3r\x01\x00\x00\x01\x00\x00\x00\x00\x00\x00\x06mobile\x06events\x04data\tmicrosoft\x03com\x00\x00\x01\x00\x01'

Timestamp: 1722266554.011004
Non-IP Packet
Payload: b'`\x02\xc1\xe0\x00:\x11@\xfe\x80\x00\x00\x00\x00\x00\x00\xc4AO&
\x83\xc3\x10\xf5\xfe\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01\xea\x9f\x005\x00:A\xb1\xed\xe8\x01\x00\x00\x01\x00\x00\x00\x00\x00\x00\x06mobile\x06events\x04data\tmicroso
ft\x03com\x00\x00A\x00\x01'

Timestamp: 1722266554.23292
Source IP: 34.95.145.254
Destination IP: 192.168.1.3
Protocol: 6
Protocol: TCP
Source Port: 443
Destination Port: 51562
Flags: A
Payload: b''

Timestamp: 1722266554.232994
Source IP: 192.168.1.3
Destination IP: 34.95.145.254
Protocol: 6
Protocol: TCP
Source Port: 51562
Destination Port: 443
Flags: A
Payload: b''
```

```
Timestamp: 1722266554.707047
Non-IP Packet
Payload: b"`\x00\x00\x00\x00\xb3\x11@\xfe\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01\xfe\x80\x00\x00\x00\x00\x00\x00\xc4AO&
\x83\xc3\x10\xf5\x005\xe3\xab\x00\xb38'\xb3r\x81\x80\x00\x01\x00\x03\x00\x00\x00\x00\x06mobile\x06events\x04data\tmicrosoft\x03com\x00\x00\x01\x00\x01\xc0\x0c\x00\x05\x00\x01\x00\x00\x00S\
x00'\x06mobile\x06events\x04data\x0etrafficmanager\x03net\x00\xc0>\x00\x05\x00\x01\x00\x00\x00\x17\x00*\x10onedscolprduks02\x07uksouth\x08cloudapp\x05azure\xc0)
\xc0q\x00\x01\x00\x01\x00\x00\x00\x03\x00\x043\x84\xc1h"

Timestamp: 1722266554.707304
Non-IP Packet
Payload: b"`\x00\x00\x00\x00\xdf\x11@\xfe\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01\xfe\x80\x00\x00\x00\x00\x00\x00\xc4AO&
\x83\xc3\x10\xf5\x005\xea\x9f\x00\xdf~\xb5\xed\xe8\x81\x80\x00\x01\x00\x02\x00\x01\x00\x00\x06mobile\x06events\x04data\tmicrosoft\x03com\x00\x00A\x00\x01\xc0\x0c\x00\x05\x00\x01\x00\x00\x0
0S\x00'\x06mobile\x06events\x04data\x0etrafficmanager\x03net\x00\xc0>\x00\x05\x00\x01\x00\x00\x00\x17\x00*\x10onedscolprduks02\x07uksouth\x08cloudapp\x05azure\xc0)
\xc0\x82\x00\x06\x00\x01\x00\x00\x00\x17\x000\x06ns1-01\tazure-dns\xc0)\x06msnhst\xc0\x1f\x00\x00\x00\x01\x00\x00\x03\x84\x00\x00\x01,\x00\t:\x80\x00\x00\x00<"

Timestamp: 1722266554.708642
Source IP: 192.168.1.3
Destination IP: 51.132.193.104
Protocol: 6
Protocol: TCP
Source Port: 51602
Destination Port: 443
Flags: S
Payload: b''

Timestamp: 1722266554.853898
Source IP: 51.132.193.104
Destination IP: 192.168.1.3
Protocol: 6
Protocol: TCP
Source Port: 443
Destination Port: 51602
Flags: SA
Payload: b''
```

```
Timestamp: 1722266554.854055
Source IP: 192.168.1.3
Destination IP: 51.132.193.104
Protocol: 6
Protocol: TCP
Source Port: 51602
Destination Port: 443
Flags: A
Payload: b''

Timestamp: 1722266554.85455
Source IP: 192.168.1.3
Destination IP: 51.132.193.104
Protocol: 6
Protocol: TCP
Source Port: 51602
Destination Port: 443
Flags: PA
Payload: b'\x16\x03\x01\x02b\x01\x00\x02^\x03\x03w\xe6\x11\xc3g\x12H\x17\x93a\x80\xe8?\x9b\xea\xaf};\xf2?5K \xd0M^c\x94\xd9\x8c|\x91
t\x92\nI\x04\xc4\xc1\xb5Y\xb0\xe9w\x03\x84\xfd\x83\x93\x15\x0b\x93T\xbc\xc2\xc4w\xa7\x04C>t\xe24\x00 \x9a\x9a\x13\x01\x13\x02\x13\x03\xc0+\xc0/\xc0,
\xc00\xcc\xa9\xcc\xa8\xc0\x13\xc0\x14\x00\x9c\x00\x9d\x00/
\x005\x01\x00\x01\xf5zz\x00\x00\x00r\x00\x12\x00\x10\x04\x03\x08\x04\x04\x01\x05\x03\x08\x05\x05\x01\x08\x06\x06\x01\xfe\r\x00\xba\x00\x00\x01\x00\x01\xbc\x00
\xa2\xbd\x83\x81K\xc7=\xbd\xb9\x8e\xdc\xc0\xc0\x9d\xb4\x15%h\xf9W\xd3w\xe3g\xebe5\xb1\x03\x8c\xedk\x00\x90\x84Q\xfb\xea\x03p\x88\xb0B\xbcc\xe1m\xfc\x0b\xef\x07\x1c\x14TRy\xef\xc1\x12;
\x7f\x1c3\xa3\x85$&j\xa7\xfd\xdd\xe2M\xc5!o\n\xf5\x8c\x82MI<\ra\x94\xc9\xe8\x90\x18R\xfb^\x8c\xd6\x04S\xc6\xf4Y\x1cs\xb8\x96Z\x8f\xe3@/\x91\x1fD\x02
(B\x00q5\xda\xc4\x8e\xa6\x13h\xc4\xe1m\xf1\xc2\xac_0p<Z\x1c\xcc\xbb\x9a\x97-F]l\xdd\x80AS-\xcc\ng\xf2!\xf8)] \n\x12\xb1\x86\x88\xd4~\x07\xa98\xcdS*+\x0b\x9c\xd6M\xd5w\x003\x00+\x00)
\x9a\x9a\x00\x01\x00\x00\x1d\x00 \xba\xd7\x8b\x0c[~\x8f\x9c\xd5=W\x8e\xf3\x0c\x12\x884\x076\x9dl\x8cLy\xe1#\xcf$\x9e\xbb\xe7t\x00\x17\x00\x00\xff\x01\x00\x01\x00\x00#\x00\x00\x00
+\x00\x07\x06jj\x03\x04\x03\x03\x00-\x00\x02\x01\x01\x00\x1b\x00\x03\x02\x00\x02\x00\x10\x00\x0e\x00\x0c\x02h2\x08http/1.1\x00\x00\x00%\x00#\x00\x00 mobile.events.data.microsoft.
com\x00\n\x00\n\x00\x08\x9a\x9a\x00\x1d\x00\x17\x00\x18Di\x00\x05\x00\x03\x02h2\x00\x12\x00\x00\x00\x0b\x00\x02\x01\x00\x00\x05\x00\x05\x01\x00\x00\x00\x00\xea\xea\x00\x01\x00\x00)\x00
[\x00&\x00
\x87@\x00\x00cS\xb5\xcd\xb9\xdb-\xf6\x18\xb0\xba0\xa3\x91=\\dR\xe6\xcf6\xe6\xf9\xc7N\xdcM\xfa\xecTq\x04\x0010z\x8e\x93B\x06\xdd\xeeV\t\xe2W\x9a\xf2\xcf\\\x8e\x8f\xbd\xa0\xf8Mf\xb15\xcb\xed
\xe8\xab\xdfn9]\xb2;\xa0\xdet\xedU\xac\x87%\x0c\xae\\\xdb\x11\t'
```