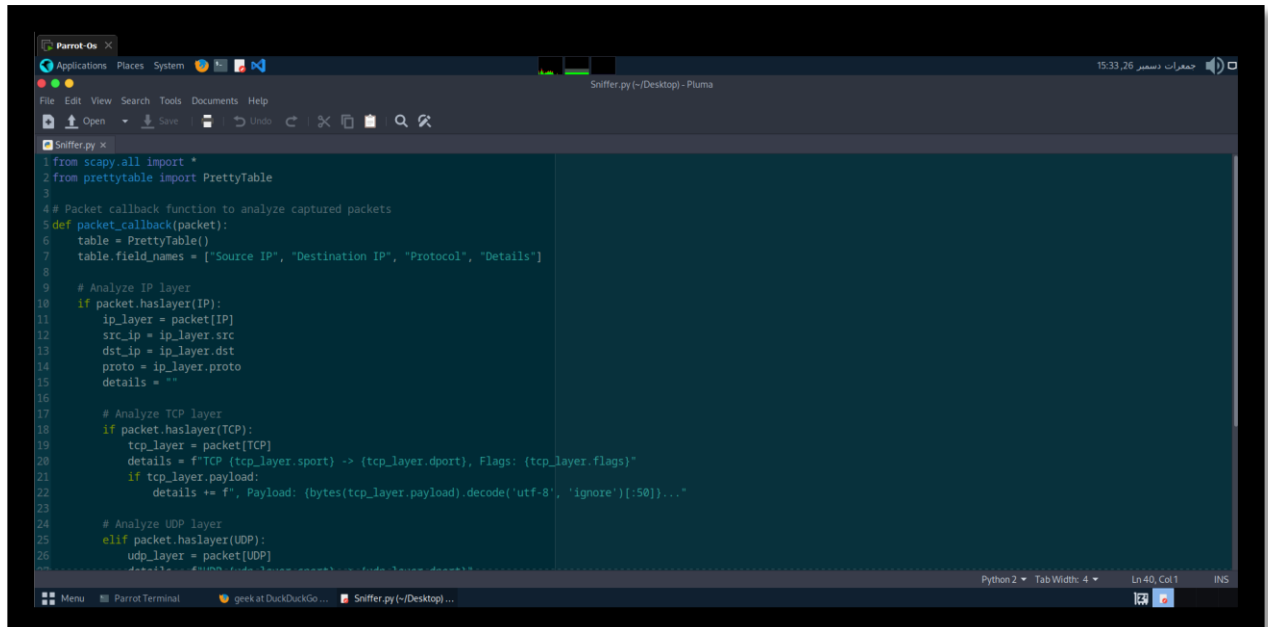


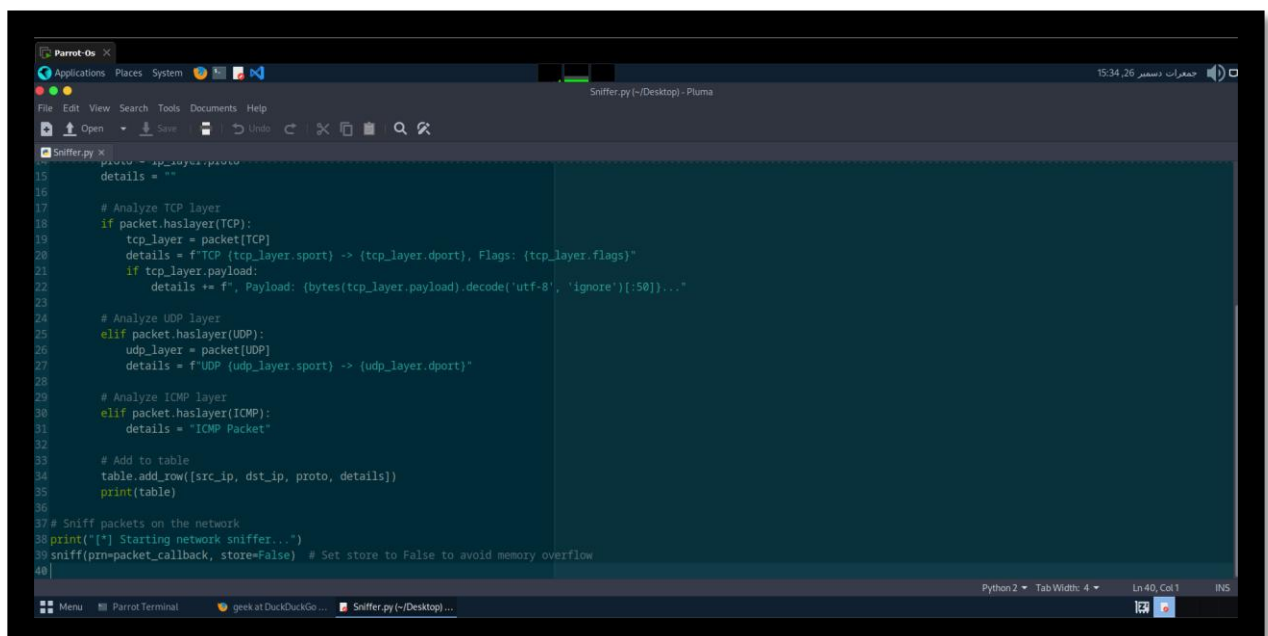
TASK

Build a network sniffer in Python that captures and analyzes network traffic. This project will help you understand how data flows on a network and how network packets are structured.

Code

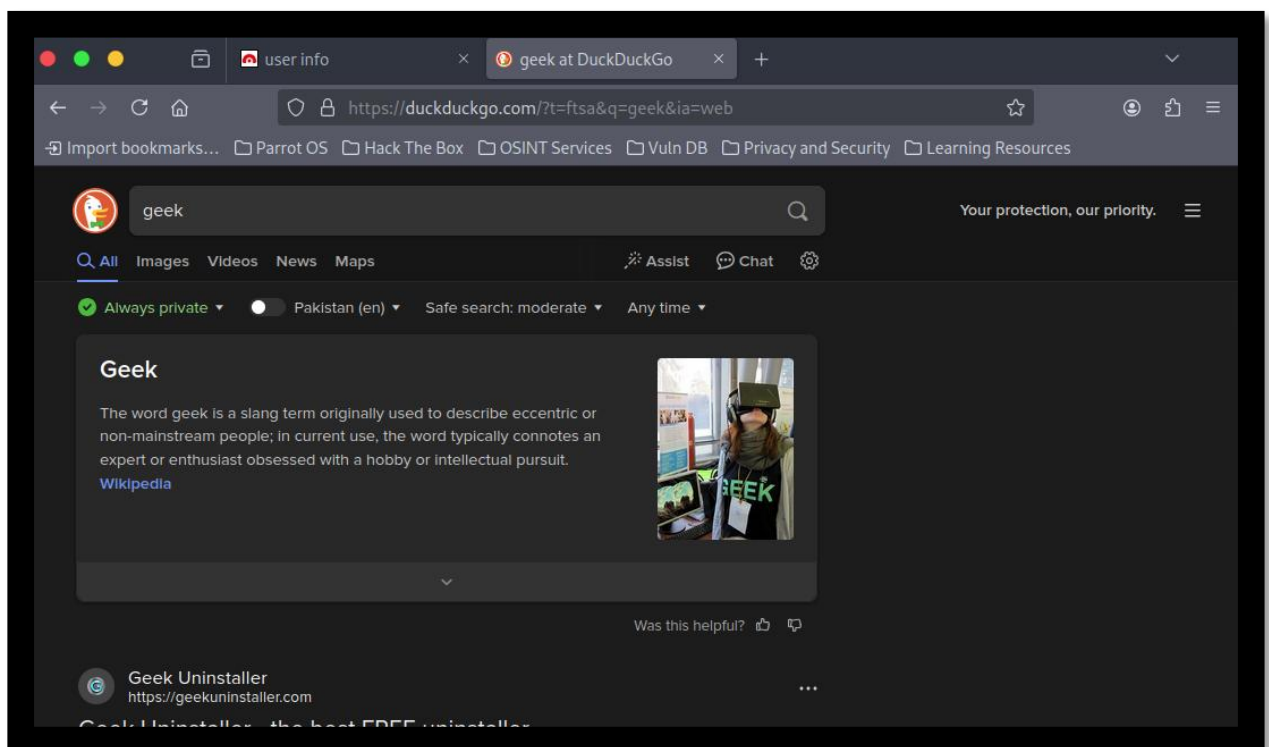
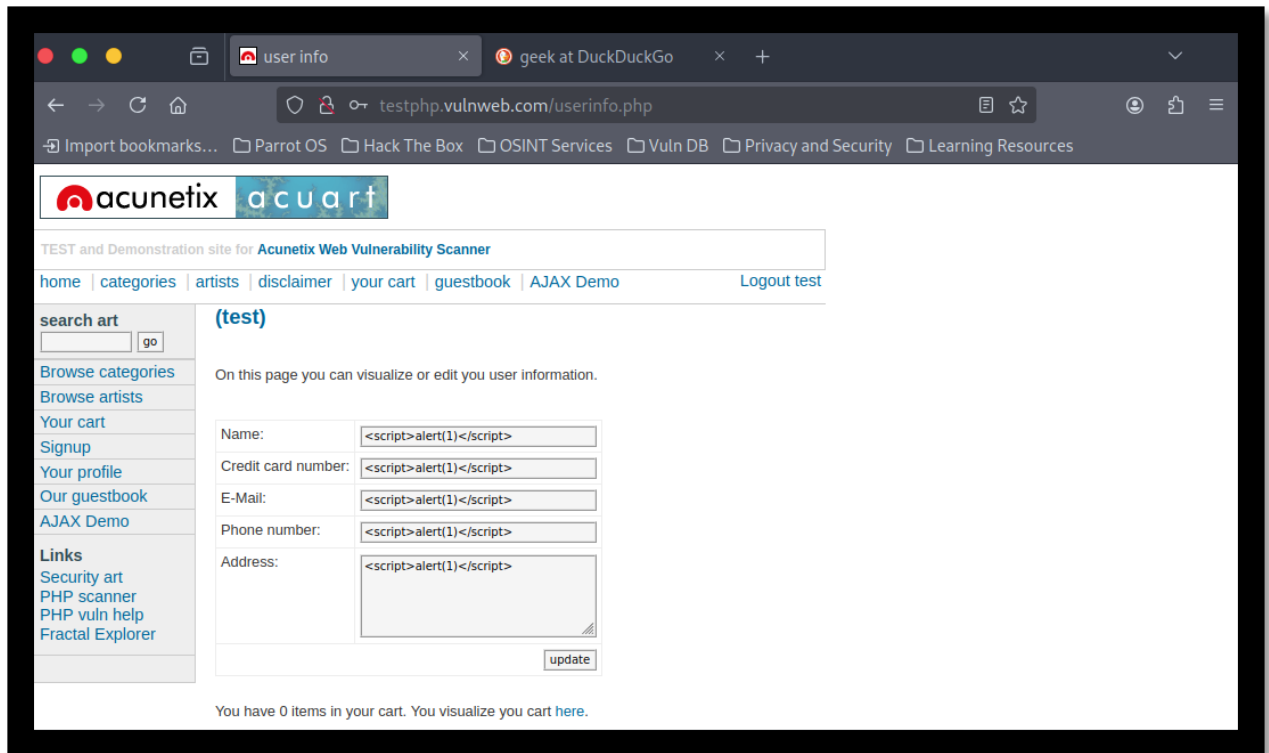


```
1 from scapy.all import *
2 from prettytable import PrettyTable
3
4 # Packet callback function to analyze captured packets
5 def packet_callback(packet):
6     table = PrettyTable()
7     table.field_names = ["Source IP", "Destination IP", "Protocol", "Details"]
8
9     # Analyze IP layer
10    if packet.haslayer(IP):
11        ip_layer = packet[IP]
12        src_ip = ip_layer.src
13        dst_ip = ip_layer.dst
14        proto = ip_layer.proto
15        details = ""
16
17    # Analyze TCP layer
18    if packet.haslayer(TCP):
19        tcp_layer = packet[TCP]
20        details = f"TCP (tcp_layer.sport) -> (tcp_layer.dport), Flags: (tcp_layer.flags)"
21        if tcp_layer.payload:
22            details += f", Payload: (bytes(tcp_layer.payload).decode('utf-8', 'ignore'))[:50]..."
23
24    # Analyze UDP layer
25    elif packet.haslayer(UDP):
26        udp_layer = packet[UDP]
27        details = f"UDP (udp_layer.sport) -> (udp_layer.dport)"
28
29    # Analyze ICMP layer
30    elif packet.haslayer(ICMP):
31        details = "ICMP Packet"
32
33    # Add to table
34    table.add_row([src_ip, dst_ip, proto, details])
35    print(table)
36
37 # Sniff packets on the network
38 print("[*] Starting network sniffer...")
39 sniff(prn=packet_callback, store=False) # Set store to False to avoid memory overflow
40
```



```
15     details = ""
16
17    # Analyze TCP layer
18    if packet.haslayer(TCP):
19        tcp_layer = packet[TCP]
20        details = f"TCP (tcp_layer.sport) -> (tcp_layer.dport), Flags: (tcp_layer.flags)"
21        if tcp_layer.payload:
22            details += f", Payload: (bytes(tcp_layer.payload).decode('utf-8', 'ignore'))[:50]..."
23
24    # Analyze UDP layer
25    elif packet.haslayer(UDP):
26        udp_layer = packet[UDP]
27        details = f"UDP (udp_layer.sport) -> (udp_layer.dport)"
28
29    # Analyze ICMP layer
30    elif packet.haslayer(ICMP):
31        details = "ICMP Packet"
32
33    # Add to table
34    table.add_row([src_ip, dst_ip, proto, details])
35    print(table)
36
37 # Sniff packets on the network
38 print("[*] Starting network sniffer...")
39 sniff(prn=packet_callback, store=False) # Set store to False to avoid memory overflow
40
```

Visited Website



Output

```
+-----+-----+-----+-----+
| Source IP | Destination IP | Protocol | Details |
+-----+-----+-----+-----+
| 192.168.10.137 | 192.168.10.2 | 17 | UDP 35092 -> 53 |
+-----+-----+-----+-----+
| Source IP | Destination IP | Protocol | Details |
+-----+-----+-----+-----+
| 192.168.10.137 | 192.168.10.2 | 17 | UDP 35092 -> 53 |
+-----+-----+-----+-----+
| Source IP | Destination IP | Protocol | Details |
+-----+-----+-----+-----+
| 192.168.10.2 | 192.168.10.137 | 17 | UDP 53 -> 35092 |
+-----+-----+-----+-----+
| Source IP | Destination IP | Protocol | Details |
+-----+-----+-----+-----+
| 192.168.10.2 | 192.168.10.137 | 17 | UDP 53 -> 35092 |
+-----+-----+-----+-----+
| Source IP | Destination IP | Protocol | Details |
+-----+-----+-----+-----+
| 192.168.10.137 | 20.204.6.105 | 6 | TCP 57176 -> 443, Flags: S |
+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+
| 192.168.10.137 | 20.204.6.105 | 6 | TCP 57188 -> 443, Flags: S |
+-----+-----+-----+-----+
| Source IP | Destination IP | Protocol | Details |
+-----+-----+-----+-----+
| 192.168.10.137 | 20.204.244.192 | 6 | TCP 57340 -> 443, Flags: PA, Payload: fsfy)Y
B'ijmVp... | 6R+~<BvA3EFm, 艱c!G)+~S
+-----+-----+-----+-----+
| Source IP | Destination IP | Protocol | Details |
+-----+-----+-----+-----+
| 20.204.244.192 | 192.168.10.137 | 6 | TCP 443 -> 57340, Flags: A, Payload: ... |
+-----+-----+-----+-----+
| Source IP | Destination IP | Protocol | Details |
+-----+-----+-----+-----+
| 20.204.6.105 | 192.168.10.137 | 6 | TCP 443 -> 57176, Flags: SA, Payload: ... |
+-----+-----+-----+-----+
| Source IP | Destination IP | Protocol | Details |
+-----+-----+-----+-----+
| 192.168.10.137 | 20.204.6.105 | 6 | TCP 57176 -> 443, Flags: A |
+-----+-----+-----+-----+
| Source IP | Destination IP | Protocol | Details |
+-----+-----+-----+-----+
| 192.168.10.137 | 20.204.6.105 | 6 | TCP 57176 -> 443, Flags: PA, Payload: 0U1uB/<Ad Dpg4G`LK&U)P"|t"/... |
+-----+-----+-----+-----+
```