Code Explanation

1. Importing the Required Module

a) from scapy.all import sniff

- scapy.all: Imports all functionalities of 'Scapy', a powerful library for handling network packets.
- sniff(): Function that captures network packets in real-time.

2. Defining the Packet Handling Function

b) def packet_callback(packet):

- This function is called each time a packet is captured.
- 'packet': Represents the captured packet.

3. Displaying the Packet Summary

c) print(packet.summary())

- 'packet.summary()' displays a quick summary of the captured packet (e.g., protocol type, ports, etc.).

4. Checking if the Packet Contains an IP Layer

d) if packet.haslayer('IP'):

- Checks if the packet contains an 'IP' (Internet Protocol) layer.
- If so, extracts the source and destination IP addresses.

5. Displaying Source and Destination IP Addresses

e) print(f'Source IP: {packet['IP'].src} -> Destination IP: {packet['IP'].dst}')

- packet['IP'].src: Sender's IP address.
- packet['IP'].dst: Recipient's IP address.

6. Checking and Displaying TCP Information

a) if packet.haslayer('TCP'):

b) print(f'TCP Packet - Source Port: {packet['TCP'].sport} -> Destination Port: {packet['TCP'].dport}')

- 'haslayer('TCP')': Checks if the packet contains a 'TCP' layer.
- 'packet['TCP'].sport' : Displays the TCP source port.
- 'packet['TCP'].dport' : Displays the TCP destination port.

7. Checking and Displaying UDP Information

a) if packet.haslayer('UDP'):

b) print(f'UDP Packet - Source Port: {packet['UDP'].sport} -> Destination Port: {packet['UDP'].dport}')

- Similar concept as TCP, but for UDP packets.

8. Capturing Packets with 'sniff()'

a) print('Starting network sniffer...')

b) sniff(prn=packet_callback, store=False)

- 'sniff()' starts packet capture.
- 'prn=packet_callback': Calls the 'packet_callback()' function for each captured packet.
- 'store=False': Does not store packets in memory (prevents overload).