**Assignment- 7**

**Question: 1**

**Problem Statement:**

**Traceroute Implementation**

Traceroute is a network diagnostic tool used to track the route that packets take from the source to a destination. It sends packets with increasing Time-to-Live (TTL) values and observes the ICMP “Time Exceeded” responses from intermediate routers. Scapy allows you to implement traceroute easily by sending ICMP packets with varying TTL values and analysing the responses.

* Write a Python program that implements the traceroute functionality using Scapy.
* The program should take a destination IP address as input and send a series of ICMP packets with varying Time-to-Live (TTL) values to trace the route to the destination.
* Display the IP addresses of the routers along the path.

In your code, define a function traceroute() that takes the destination IP address and the maximum number of hops as inputs. Run a loop from TTL 1 to max hops, creating ICMP echo request packets with the corresponding TTL values and sending them using sr1() (send and receive in one function) from Scapy. Consider a timeout period of 1 second for the response.

* If you receive no response within the timeout, we print \* to indicate no response from that hop.
* If you receive an ICMP Echo Reply, it means we have reached the destination, and we print the destination IP address.
* If you receive an ICMP Time Exceeded, it indicates that the packet has reached an intermediate router, and we print the router’s IP address.

Please note that the actual number of hops may be less than max hops, depending on the network topology and firewall configurations. Also, some routers might be configured to not respond to ICMP Time Exceeded messages, which can result in incomplete traceroute information.

**Solution approach:**

For this problem first we need to create ICMP echo packets with variable ttl values. This can be easily done by using a loop to iterate through ttl values from 1 to given MAX\_HOPS value.

Now this packet needs to be sent to the given destination/target ip address. Now here we will use sr1() method of Scapy so that we just receive a single reply for the ICMP echo message we just sent.

Now we need to check for replay form the destination machine. If we don’t receive any reply then we will print a ‘\*’ , denoting that the intermediate target was not reachable. If we receive reply then the source IP of the replay packet will denote the IP of the intermediate router or receiver. So we can then print that IP address.

Lastly, if the source IP address of the reply packet is same as the destination IP we took as input then we are sure that we have reached the final target IP and the host is reachable. Otherwise we will continue until MAX\_HOPS times. If still not found then we will conclude that destination host is not reachable from out machine.

**Source code:**

import scapy.all as scapy

def traceroot(ipAddr, maxHops):

print("\nhops\t Routers-IP")

for i in range (1, maxHops+1):

pack = scapy.IP(dst=ipAddr, ttl=i)/scapy.ICMP()/"Hello"

rep = scapy.sr1(pack, timeout=1, verbose=False)

if(rep == None):

print(i, "\t", "\*")

else:

x = rep[0]

xSrc = x[scapy.IP].src

print(i, "\t", xSrc)

if(xSrc == ipAddr):

return

print("Destination not found within ", maxHops, " hops")

ipAddr = input("Enter target IP: ")

maxHops = int(input("Enter max hops: "))

traceroot(ipAddr, maxHops)

**Output:**

**Case 1:**

Enter target IP: 142.250.77.100

Enter max hops: 20

hops Routers-IP

1 192.168.0.1

2 172.23.215.1

3 \*

4 192.168.199.170

5 202.78.239.62

6 142.251.227.211

7 142.251.55.231

8 142.250.77.100

**Case 2:**

Enter target IP: 202.165.107.49

Enter max hops: 30

hops Routers-IP

1 192.168.0.1

2 172.23.215.1

3 \*

4 192.168.199.53

5 203.171.240.1

6 \*

7 180.87.36.9

8 180.87.36.41

9 180.87.54.7

10 210.176.138.23

11 203.84.209.77

12 106.10.128.7

13 106.10.131.216

14 106.10.128.246

15 202.165.107.49

**Case 3:**

Enter target IP: 142.250.0.0

Enter max hops: 10

hops Routers-IP

1 192.168.0.1

2 172.23.215.1

3 \*

4 192.168.199.170

5 202.78.239.62

6 74.125.242.129

7 74.125.242.138

8 172.253.74.53

9 142.251.248.62

10 108.170.228.156

Destination not found within 10 hops