Q1.

tracert : tracert (short for "trace route") is a network diagnostic tool used to trace the path that packets take from your computer to a destination (such as a website or another network device). It helps identify the route taken by the data packets across the network and provides information about each hop (intermediate router or gateway) along the way. This is useful for diagnosing network issues, such as pinpointing where a connection is slow or where packets are being dropped.

Syntax : tracert [options] dst\_hostname

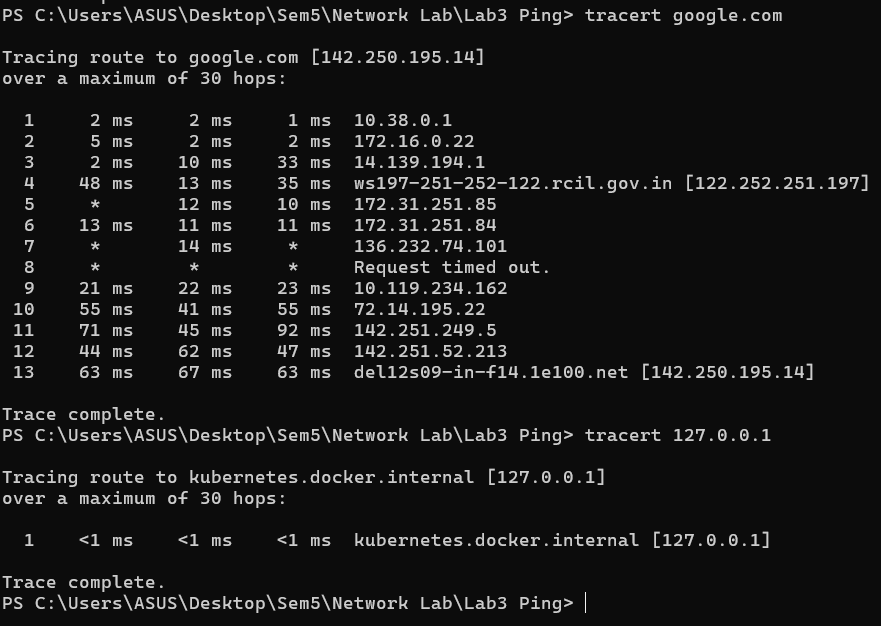
Examples :

tracert google.com

tracert 127.0.0.1 // localhost

Q2.

tracert google.com



**Hop Number**: The first column indicates the number of the hop.

**Round-Trip Times (RTT)**: The next three columns show the time (in milliseconds) that it took for the packet to travel to that hop and back. Multiple times are shown to give an average.

**IP Address / Hostname**: The last column displays the IP address or hostname of the router or gateway that responded to the packet at that hop.

Q3.

tracert options :

 **-d**: Prevents tracert from resolving IP addresses to hostnames. This speeds up the operation by avoiding DNS lookups.

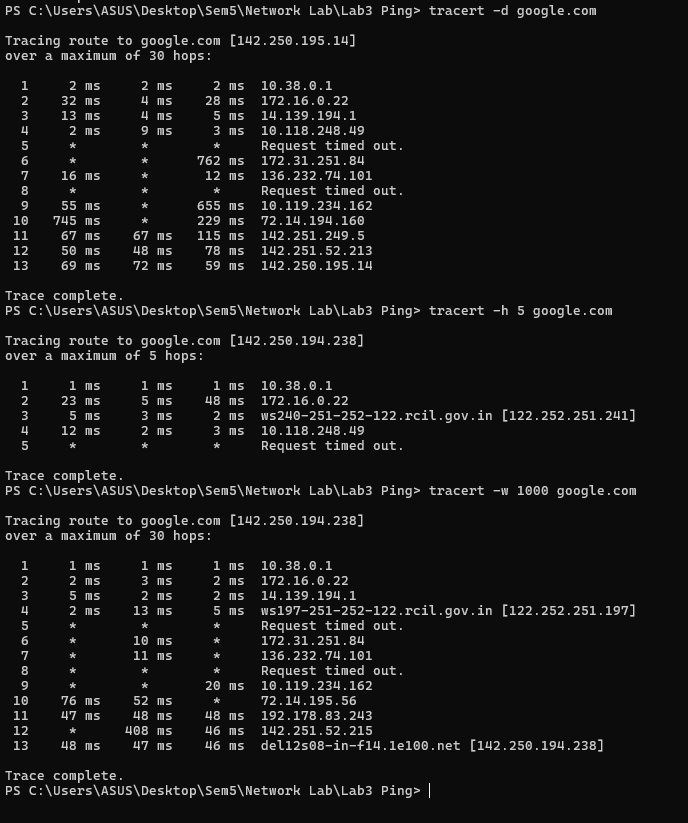
tracert -d google.com

 **-h**: Specifies the maximum number of hops (TTL values) to search for the target. The default is 30.

tracert -h 10 google.com

 **-w**: Sets the timeout, in milliseconds, for each reply. The default is 4000 milliseconds (4 seconds).

tracert -w 1000 google.com



Q4.

**Situation**

Q5. **You notice that your internet connection is unusually slow, especially when trying to access certain websites like google.com. You suspect that the issue might be due to network congestion or a problem with one of the intermediate routers between your computer and the destination server.**

### Using tracert to Diagnose the Issue

1. **Run tracert to Identify Latency**  
   First, use tracert to trace the route to the slow website:  
   tracert google.com
   * The output will display each hop between your computer and the destination server, along with the round-trip time (RTT) for each hop.
   * High RTT values at a specific hop could indicate network congestion or an issue with that router.
   * If multiple timeouts (\* \* \* Request timed out.) occur at any hop, it might suggest a significant problem with that router or an upstream network.
2. **Use -w Option to Adjust Timeout**
   * If certain hops aren't responding (showing \* \* \*), you can increase the timeout to give the routers more time to respond:  
     tracert -w 2000 google.com
   * The -w option extends the time each hop waits for a response, helping to determine if the issue is due to slow-responding routers rather than completely unresponsive ones.
3. **Use -h Option to Limit the Number of Hops**
   * If you suspect the issue is with a router close to your network (e.g., within your ISP), you can limit the number of hops to focus on the initial part of the route:  
     tracert -h 10 google.com
   * This limits the trace to the first 10 hops, allowing you to concentrate on potential issues within your ISP's network.
4. **Analyze the Results**
   * **High RTT Values**: If an intermediate router shows significantly higher RTT compared to previous hops, that router might be the source of the delay.
   * **Multiple Timeouts**: Consistent timeouts beyond a certain hop may indicate a problem with that specific router or network segment.

**Conclusion**

The tracert (short for "trace route") utility is a powerful tool for diagnosing network connectivity issues by tracing the path data packets take from your computer to a destination server. It operates by sending ICMP Echo Request packets to the target and displays the series of hops (routers or gateways) the packets pass through, along with the time each hop takes to respond.

**Key Applications:**

* **Diagnosing Connectivity Issues**: tracert helps identify where a connection is failing or slowing down by pinpointing which router or segment of the network is causing delays or packet loss.
* **Analyzing Network Paths**: It provides insights into the route data takes across the internet, which can be useful for optimizing network configurations or understanding how data travels between different geographic locations.
* **Troubleshooting Network Problems**: Network administrators and IT professionals use tracert to troubleshoot issues such as slow network speeds, intermittent connectivity, or routing problems.

**Limitations and Potential Issues with tracert**

While tracert is a valuable diagnostic tool, it has some limitations and potential issues:

1. **ICMP Blocking**: Some routers and firewalls block ICMP Echo Request packets (used by tracert), leading to timeouts or incomplete traces. This can make it difficult to diagnose problems accurately.
2. **Asymmetrical Routing**: The path that data takes from your computer to a destination might not be the same as the path back. tracert only shows the forward path, which can be misleading in cases where the issue lies on the return route.
3. **Variable Network Load**: The times reported by tracert (round-trip times) can vary significantly depending on network load at the time the trace is performed. This variability can make it hard to identify consistent issues.
4. **Limited Information**: tracert provides information on the route and timing but doesn't give detailed insights into the nature of the problem. For example, it won't explain why a router is slow or identify packet loss causes beyond simple timeouts.
5. **Limited Reach in Complex Networks**: In complex networks, especially those involving VPNs, MPLS, or cloud environments, tracert might not accurately reflect the real path data takes due to network abstraction and encapsulation techniques.

Objective 2.

Code :

from scapy.all import IP, ICMP, sr1, conf

import time

def tracert(dest\_ip, max\_ttl=30, packet\_size=64, timeout=2, src\_ip=None, num\_pings=3, delay=1, output\_file=None):

    try:

        # Set the source IP if specified

        if src\_ip:

            conf.src = src\_ip

        # Prepare to save output if specified

        if output\_file:

            f = open(output\_file, 'w')

        else:

            f = None  # Set f to None if no output file is specified

        print(f"Tracing route to {dest\_ip} over a maximum of {max\_ttl} hops:")

        if f:

            f.write(f"Tracing route to {dest\_ip} over a maximum of {max\_ttl} hops:\n")

        for ttl in range(1, max\_ttl + 1):

            rtt\_times = []

            packet\_sent = 0

            packet\_received = 0

            for i in range(num\_pings):

                packet\_sent += 1

                pkt = IP(dst=dest\_ip, ttl=ttl)/ICMP()/("X"\*packet\_size)

                start\_time = time.time()

                reply = sr1(pkt, verbose=False, timeout=timeout)

                end\_time = time.time()

                if reply:

                    packet\_received += 1

                    rtt = (end\_time - start\_time) \* 1000  # RTT in milliseconds

                    rtt\_times.append(rtt)

                    print(f"{ttl}\t{reply.src}\t{round(rtt, 2)} ms")

                    if f:

                        f.write(f"{ttl}\t{reply.src}\t{round(rtt, 2)} ms\n")

                    if reply.src == dest\_ip:

                        print("Trace complete.")

                        if f:

                            f.write("Trace complete.\n")

                        break

                else:

                    print(f"{ttl}\t\*\tRequest timed out.")

                    if f:

                        f.write(f"{ttl}\t\*\tRequest timed out.\n")

                time.sleep(delay)

            if reply and reply.src == dest\_ip:

                break

            packet\_loss = ((packet\_sent - packet\_received) / packet\_sent) \* 100

            if rtt\_times:

                avg\_rtt = sum(rtt\_times) / len(rtt\_times)

                min\_rtt = min(rtt\_times)

                max\_rtt = max(rtt\_times)

            else:

                avg\_rtt = min\_rtt = max\_rtt = None

            print(f"Hop {ttl}: Sent={packet\_sent}, Received={packet\_received}, Loss={packet\_loss:.2f}%, min/avg/max RTT = {min\_rtt}/{avg\_rtt}/{max\_rtt} ms")

            if f:

                f.write(f"Hop {ttl}: Sent={packet\_sent}, Received={packet\_received}, Loss={packet\_loss:.2f}%, min/avg/max RTT = {min\_rtt}/{avg\_rtt}/{max\_rtt} ms\n")

        if f:

            f.close()

    except ValueError as ve:

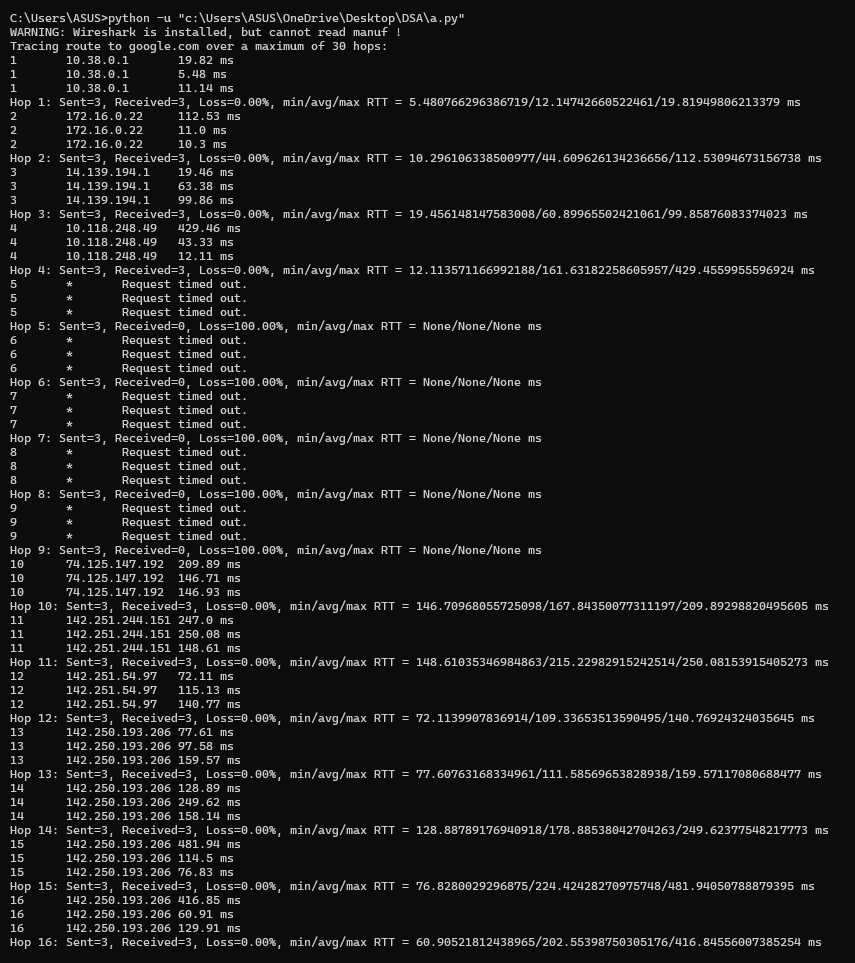
        print(f"Input Error: {ve}")

    except Exception as e:

        print(f"An error occurred: {e}")

if \_\_name\_\_ == "\_\_main\_\_":

    tracert("google.com", num\_pings=3, delay=1, output\_file="tracert\_output.txt")



A screen shot of a computer

Description automatically generated