**Name :- Shanu Kumar**

**Roll no. :- 2201CS67**

**Networks Lab Assignment 4 Solution**

**Objective 1: Tracert Utility Analysis**

**1. Tracert Basics**

**Purpose of the Tracert Utility:**

* The tracert (short for "trace route") utility is a network diagnostic tool used to trace the path that packets take from your computer to a destination host. It provides a list of hops (routers) between your computer and the target, helping to identify where delays or failures occur.

**Basic Syntax:**

* The basic syntax of the tracert command is:

tracert [options] target\_host

* + target\_host: The domain name or IP address of the destination.

**Examples:**

* To trace the route to a website (e.g., google.com):

Command :- tracert google.com

* A screenshot of a computer

  Description automatically generated

**To trace the route to a local host:**

Code :- tracert 127.0.0.1

A screen shot of a computer

Description automatically generated

**2. Tracert Output Analysis**

**Running Tracert:**

* **Command:** tracert google.com
* **Sample Output:**
* A screenshot of a computer

  Description automatically generated

css

Copy code

Tracing route to google.com [142.250.193.238] over a maximum of 30 hops:

**Output Explanation:**

* **Hop Number:** The sequence number of the router the packet passes through.
* **IP Address:** The IP address of the router.
* **RTT (Round-Trip Time):** The time it takes for a packet to go from the source to the destination and back, measured in milliseconds.

**Local Host Tracert:**

* **Command:** tracert 127.0.0.1
* **Output :** A screen shot of a computer

  Description automatically generated
* **Output Explanation:**
  + Since 127.0.0.1 is the loopback address, the output will typically show just one hop with minimal RTT.

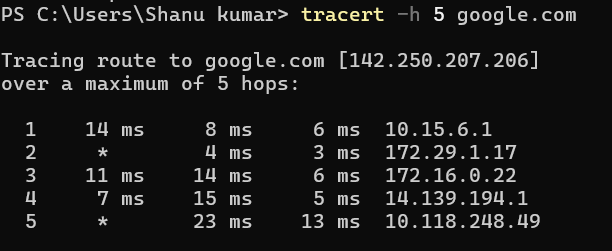
**3. Tracert Options**

**-d (Do not resolve hostnames):**

* **Description:** This option prevents the tracert utility from resolving IP addresses to their corresponding domain names, which can speed up the trace process.
* **Example:** tracert -d google.com
* **OUTPUT**
* A screenshot of a computer

  Description automatically generated

**-h (Maximum number of hops):**

* **Description:** This option allows you to set the maximum number of hops (routers) to be traced before the utility stops.
* **Example:** tracert -h 5 google.com
* **OUTPUT **

**-w (Timeout in milliseconds):**

* **Description:** This option sets the wait time in milliseconds for each reply before moving on to the next hop.
* **Example:** tracert -w 500 google.com
* **OUTPUT**
* A screenshot of a computer

  Description automatically generated

**4. Troubleshooting with Tracert**

**Scenario:**

* **Problem:** A user is experiencing slow network speeds when accessing a particular website.
* **Using Tracert:**
  + **Command:** tracert google.com
  + **Analysis:** The tracert output can show if there is a specific hop that is causing delays, indicating a possible network bottleneck or misconfiguration at a specific router.

**Options to Use:**

* **-h:** To limit the number of hops traced if the destination is known to be within a few hops.
* **-d:** To speed up the process by skipping hostname resolution.

**5. Conclusion**

**Summary:**

* The tracert utility is a powerful tool for network diagnostics, helping identify where delays or failures occur along a packet's route to its destination.

**Limitations:**

* Tracert may not work effectively if ICMP traffic is blocked by routers, or if the destination is unreachable, leading to incomplete or misleading results.

**Objective 2: Scapy-based Tracert Utility**

**1. Basic Functionality**

**Testing:**

* Ensure the provided Scapy-based tracert code works with various inputs, such as different destination IPs, max TTL values, packet sizes, timeouts, and source IPs.

**2. Additional Features**

**Implementation:**

* **Number of pings per hop:**

ping\_per\_hop = 3 # Number of pings

* **Delay between pings:**

delay\_between\_pings = 0.5 # Delay in seconds

* **Save output to a file:**

with open("tracert\_output.txt", "w") as file:

file.write(output)

**3. Error Handling**

**Try-Except Blocks:**

* **Invalid Destination IP:**

try:

ip = socket.gethostbyname(destination)

except socket.error:

print("Invalid IP address.")

* **Invalid Max TTL Value:**

if not (1 <= max\_ttl <= 255):

raise ValueError("Invalid TTL value.")

* **Invalid Packet Size:**

if packet\_size < 0:

raise ValueError("Packet size must be a positive integer.")

**4. Output Formatting**

**Improved Output:**

* **Example:**

print(f"Hop {hop}: {ip} | RTT: {rtt:.2f} ms | Loss: {loss}%")

**Submission**

* **Modified Code:**
  + Include the updated Scapy-based tracert code with the additional features.
* **Brief Report:**
  + Describe the new features, error handling approach, and sample outputs.