Ping and Traceroute Report

Name: Samyak Rajesh Shah

Email: ss4604@rit.edu

1. Introduction

This document provides a detailed explanation of the Python implementations of custom ping and traceroute commands for network diagnostics. The scripts were developed using Python and the Scapy library.

The goal of this assignment was to understand ICMP communication, TTL-based routing, and the use of raw sockets to interact with network layers.

2. Project Structure

The project contains the following files:

File Name	Description
my_ping.py	Custom ping implementation using ICMP
my_traceroute.py	Custom traceroute implementation using UDP/ICMP
requirements.txt	List of Python dependencies
README.md	Instructions for running the scripts
docs/	Sphinx documentation

3. Environment Setup

- Python 3.7 or above
- Scapy library (scapy==2.6.1)

Installation:

pip install -r requirements.txt

4. Ping Command (my_ping.py)

4.1 Description

my_ping.py simulates the Linux ping command. It sends ICMP echo request packets to a specified host and waits for a reply.
It measures round-trip time (RTT) and determines network connectivity.

4.2 Command-line Arguments

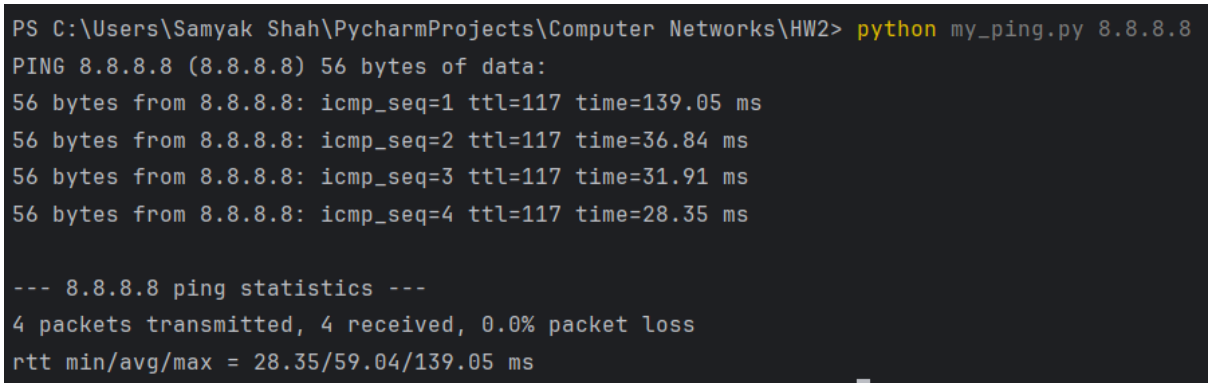
Option	Description
-c	Number of packets to send (default: 5)
-i	Interval in seconds between packets (default: 1)
-s	Packet payload size in bytes (default: 56)
-t	Timeout for the ping operation in seconds (default: 10)
destination	**Required.** Hostname or IP to ping

4.3 Example Usage

1. **Ping Google DNS with default parameters:**

   python my_ping.py 8.8.8.8

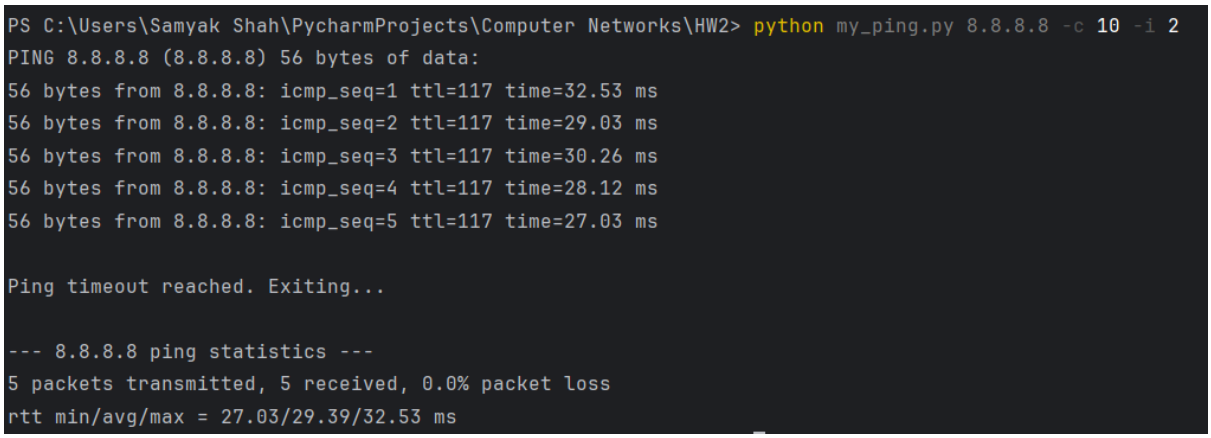
Screenshot:



2. **Ping 8.8.8.8 with 10 packets, 2-second interval:**

   python my_ping.py 8.8.8.8 -c 10 -i 2

Screenshot:



3. **Ping with larger packet size (128 bytes):**

   python my_ping.py 8.8.8.8 -s 128

Screenshot:

```
PS C:\Users\Samyak Shah\PycharmProjects\Computer Networks\HW2> python my_ping.py 8.8.8.8 -s 128
PING 8.8.8.8 (8.8.8.8) 128 bytes of data:
128 bytes from 8.8.8.8: icmp_seq=1 ttl=117 time=35.01 ms
128 bytes from 8.8.8.8: icmp_seq=2 ttl=117 time=28.33 ms
128 bytes from 8.8.8.8: icmp_seq=3 ttl=117 time=27.54 ms
128 bytes from 8.8.8.8: icmp_seq=4 ttl=117 time=29.58 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0.0% packet loss
rtt min/avg/max = 27.54/30.12/35.01 ms
```

5. Traceroute Command (my_traceroute.py)

5.1 Description

my_traceroute.py simulates the Linux traceroute command.
It sends UDP packets with increasing TTL values to trace the route to a destination.
Intermediate routers reply with ICMP messages when TTL expires, revealing the path.

5.2 Command-line Arguments

Option	Description
-n	Print hop addresses numerically only
-q	Number of probes per TTL (default: 3)
-S	Print summary of unanswered probes per hop
destination	**Required.** Hostname or IP to trace

5.3 Example Usage

1. **Traceroute to Google DNS with default probes:**

   python my_traceroute.py 8.8.8.8

Screenshot:

```
PS C:\Users\Samyak Shah\PycharmProjects\Computer Networks\HW2> python my_traceroute.py 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 3 probes per hop
1  192.168.0.1                                          125.12 ms 5.30 ms 4.94 ms
2  int-1.hnrtnyaf01m.netops.charter.com (142.254.218.125)       12.98 ms 14.09 ms 12.96 ms
3  lag-63.hnrtnyaf01h.netops.charter.com (24.58.232.209)        34.22 ms 32.29 ms 156.89 ms
4  lag-46.mcr11hnrtnyaf.netops.charter.com (24.58.49.70)        23.63 ms 15.16 ms 12.43 ms
5  lag-28.rcr01albynyyf.netops.charter.com (24.58.32.70)        18.88 ms 22.37 ms 42.88 ms
6  lag-416-10.nycmny837aw-bcr00.netops.charter.com (66.109.6.10)        27.53 ms 27.39 ms 35.80 ms
7  209.85.172.44                                        27.71 ms 29.10 ms 29.07 ms
8  * * *
9  dns.google (8.8.8.8)                                 56.91 ms 26.83 ms 26.57 ms
```

2. **Traceroute to Amazon numerically:**

   python my_traceroute.py amazon.com -n

Screenshot:

```
PS C:\Users\Samyak Shah\PycharmProjects\Computer Networks\HW2> python my_traceroute.py amazon.com -n
traceroute to amazon.com (52.94.236.248), 30 hops max, 3 probes per hop
1  192.168.0.1                                          17.20 ms 10.05 ms 6.39 ms
2  142.254.218.125                                      12.90 ms 14.36 ms 12.66 ms
3  24.58.232.209                                        26.03 ms 21.01 ms 28.01 ms
4  24.58.49.70                                          16.62 ms 12.96 ms 12.87 ms
5  24.58.32.70                                          22.39 ms 19.11 ms 20.55 ms
6  66.109.6.10                                          28.24 ms 28.32 ms 27.29 ms
7  66.109.9.5                                           32.94 ms 27.35 ms 28.00 ms
8  * * *
9  * * *
10 * * *
11 * * *
12 * * *
```

3. **Traceroute using 5 probes per hop and showing unanswered summary:**

   python my_traceroute.py 8.8.8.8 -q 5 -S

Screenshot:

```
PS C:\Users\Samyak Shah\PycharmProjects\Computer Networks\HW2> python my_traceroute.py 8.8.8.8 -q 5 -S
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 5 probes per hop
1  192.168.0.1                                        10.98 ms 5.88 ms 5.60 ms 4.77 ms 6.13 ms
2  int-1.hnrtnyaf01m.netops.charter.com (142.254.218.125)       13.20 ms 13.68 ms 14.46 ms 13.78 ms 12.24 ms
3  lag-63.hnrtnyaf01h.netops.charter.com (24.58.232.209)        30.87 ms 35.00 ms 28.22 ms 31.21 ms 30.67 ms
4  lag-46.mcr11hnrtnyaf.netops.charter.com (24.58.49.70)        13.02 ms 12.99 ms 12.90 ms 16.04 ms 14.29 ms
5  lag-28.rcr01albynyyf.netops.charter.com (24.58.32.70)        23.65 ms 67.15 ms 19.40 ms 21.55 ms 19.36 ms
6  lag-416-10.nycmny837aw-bcr00.netops.charter.com (66.109.6.10)        28.31 ms 27.60 ms 29.53 ms 29.32 ms 52.57 ms
7  209.85.172.44                                      30.33 ms 27.21 ms 30.53 ms 28.79 ms 27.57 ms
8  * * * (5 not answered)
9  dns.google (8.8.8.8)                               26.36 ms 30.81 ms 35.89 ms 28.55 ms 28.24 ms
```

6. Conclusion

The custom ping and traceroute scripts successfully demonstrate network connectivity tests and path discovery. The project reinforces understanding of ICMP, TTL, and low-level packet handling in Python.