Homework 2: Ping and Traceroute Report

Marvynn Talusan

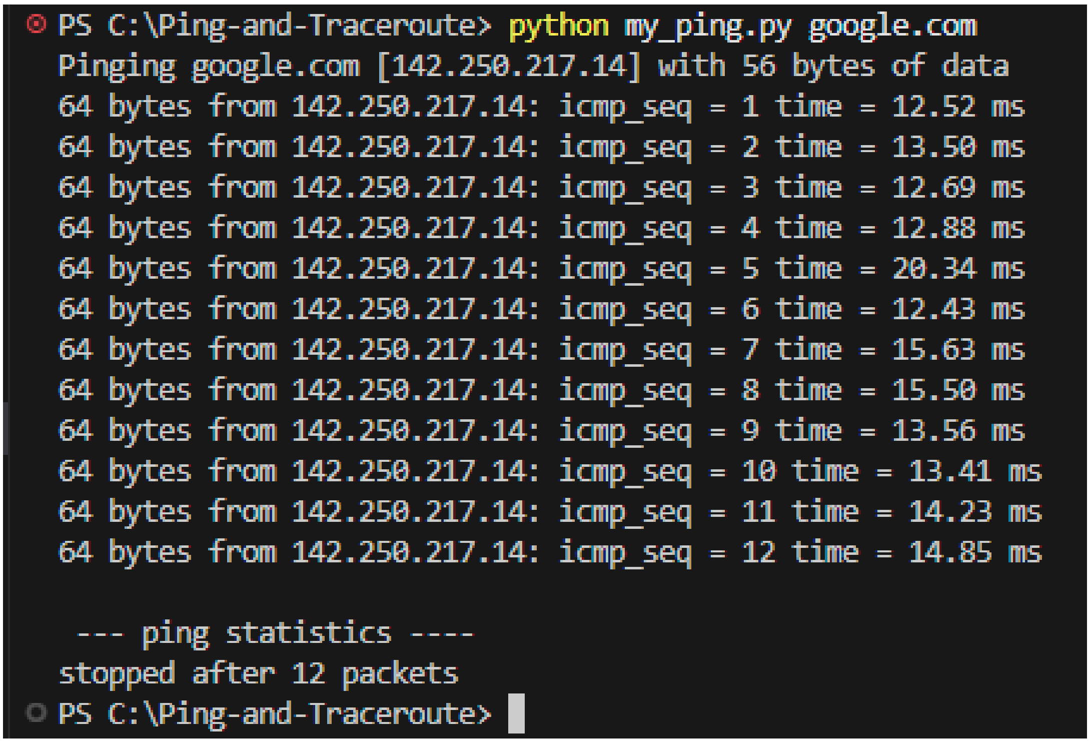
February 28, 2026

1 Introduction

This report details the different flags implemented in tools: `my_ping.py` and `my_traceroute.py`.

2 Ping Implementation Testing

2.1 Basic Connectivity and Statistics

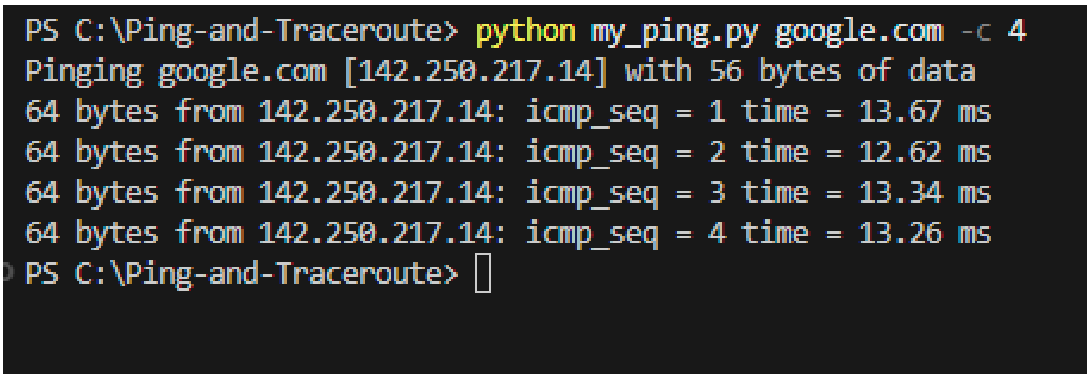


```
PS C:\Ping-and-Traceroute> python my_ping.py google.com
Pinging google.com [142.250.217.14] with 56 bytes of data
64 bytes from 142.250.217.14: icmp_seq = 1 time = 12.52 ms
64 bytes from 142.250.217.14: icmp_seq = 2 time = 13.50 ms
64 bytes from 142.250.217.14: icmp_seq = 3 time = 12.69 ms
64 bytes from 142.250.217.14: icmp_seq = 4 time = 12.88 ms
64 bytes from 142.250.217.14: icmp_seq = 5 time = 20.34 ms
64 bytes from 142.250.217.14: icmp_seq = 6 time = 12.43 ms
64 bytes from 142.250.217.14: icmp_seq = 7 time = 15.63 ms
64 bytes from 142.250.217.14: icmp_seq = 8 time = 15.50 ms
64 bytes from 142.250.217.14: icmp_seq = 9 time = 13.56 ms
64 bytes from 142.250.217.14: icmp_seq = 10 time = 13.41 ms
64 bytes from 142.250.217.14: icmp_seq = 11 time = 14.23 ms
64 bytes from 142.250.217.14: icmp_seq = 12 time = 14.85 ms

 --- ping statistics ----
stopped after 12 packets
PS C:\Ping-and-Traceroute>
```

Figure 1: Standard ping execution showing basic connectivity.

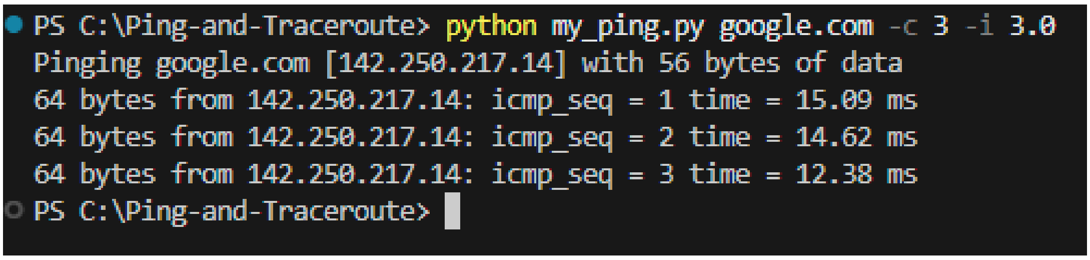
2.2 Flag Demonstration

Below demonstrate that the program can take in user defined constraints for packet counts, intervals, and timeouts



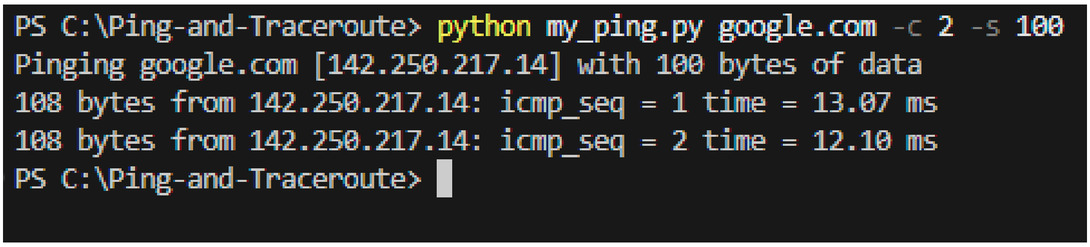
```
PS C:\Ping-and-Traceroute> python my_ping.py google.com -c 4
Pinging google.com [142.250.217.14] with 56 bytes of data
64 bytes from 142.250.217.14: icmp_seq = 1 time = 13.67 ms
64 bytes from 142.250.217.14: icmp_seq = 2 time = 12.62 ms
64 bytes from 142.250.217.14: icmp_seq = 3 time = 13.34 ms
64 bytes from 142.250.217.14: icmp_seq = 4 time = 13.26 ms
PS C:\Ping-and-Traceroute>
```

Figure 2: Count Flag (-c): Stopping after exactly 4 packets



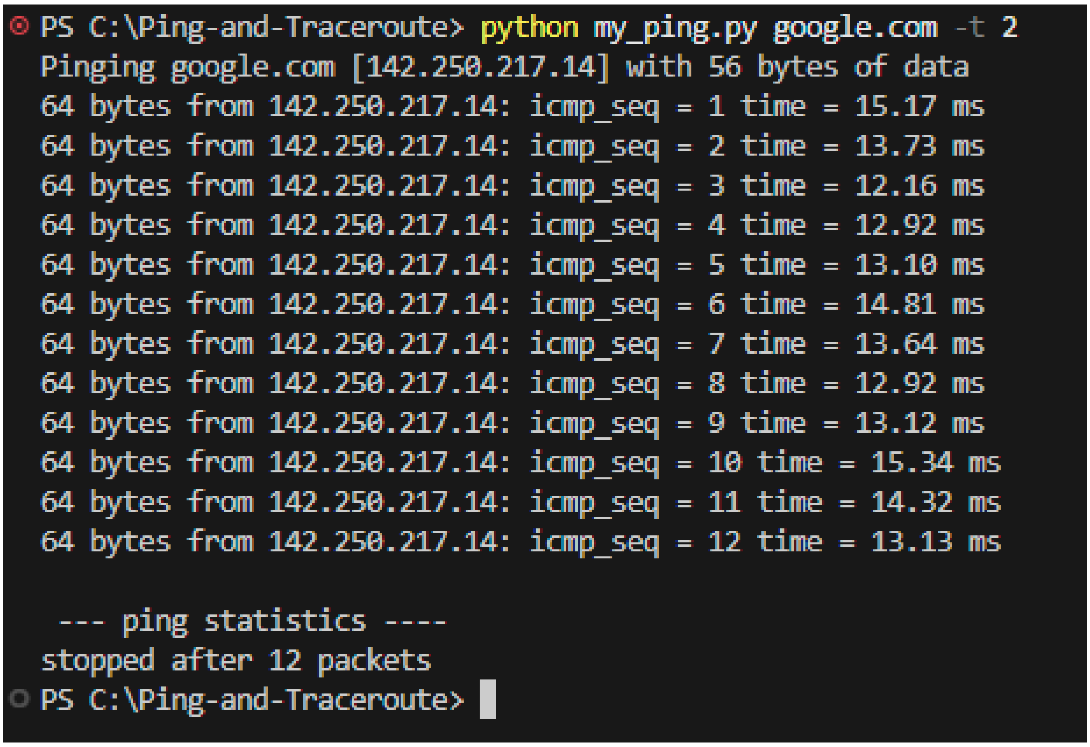
```
PS C:\Ping-and-Traceroute> python my_ping.py google.com -c 3 -i 3.0
Pinging google.com [142.250.217.14] with 56 bytes of data
64 bytes from 142.250.217.14: icmp_seq = 1 time = 15.09 ms
64 bytes from 142.250.217.14: icmp_seq = 2 time = 14.62 ms
64 bytes from 142.250.217.14: icmp_seq = 3 time = 12.38 ms
PS C:\Ping-and-Traceroute>
```

Figure 3: Wait Flag (-i): Demonstrating a 3 second interval between pings



```
PS C:\Ping-and-Traceroute> python my_ping.py google.com -c 2 -s 100
Pinging google.com [142.250.217.14] with 100 bytes of data
108 bytes from 142.250.217.14: icmp_seq = 1 time = 13.07 ms
108 bytes from 142.250.217.14: icmp_seq = 2 time = 12.10 ms
PS C:\Ping-and-Traceroute>
```

Figure 4: Packet Size (-s): Sending 100 data bytes



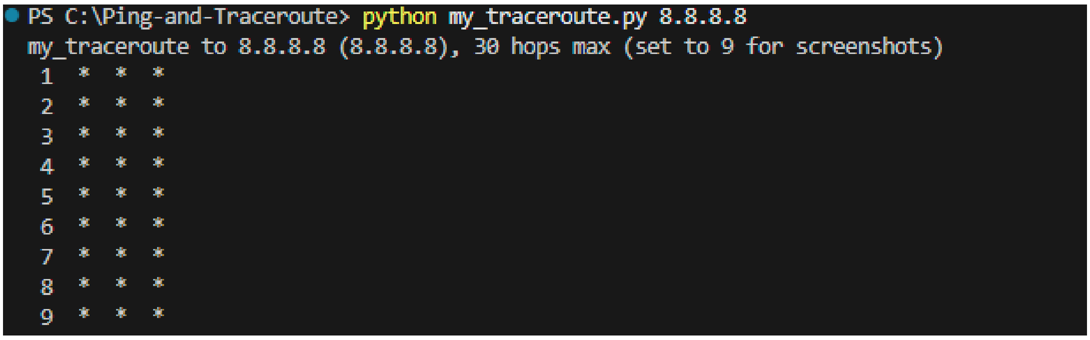
```
PS C:\Ping-and-Traceroute> python my_ping.py google.com -t 2
Pinging google.com [142.250.217.14] with 56 bytes of data
64 bytes from 142.250.217.14: icmp_seq = 1 time = 15.17 ms
64 bytes from 142.250.217.14: icmp_seq = 2 time = 13.73 ms
64 bytes from 142.250.217.14: icmp_seq = 3 time = 12.16 ms
64 bytes from 142.250.217.14: icmp_seq = 4 time = 12.92 ms
64 bytes from 142.250.217.14: icmp_seq = 5 time = 13.10 ms
64 bytes from 142.250.217.14: icmp_seq = 6 time = 14.81 ms
64 bytes from 142.250.217.14: icmp_seq = 7 time = 13.64 ms
64 bytes from 142.250.217.14: icmp_seq = 8 time = 12.92 ms
64 bytes from 142.250.217.14: icmp_seq = 9 time = 13.12 ms
64 bytes from 142.250.217.14: icmp_seq = 10 time = 15.34 ms
64 bytes from 142.250.217.14: icmp_seq = 11 time = 14.32 ms
64 bytes from 142.250.217.14: icmp_seq = 12 time = 13.13 ms

 --- ping statistics ----
stopped after 12 packets
PS C:\Ping-and-Traceroute>
```

Figure 5: Timeout Flag (-t): Program exiting after the specified time.

3 Traceroute Implementation Testing

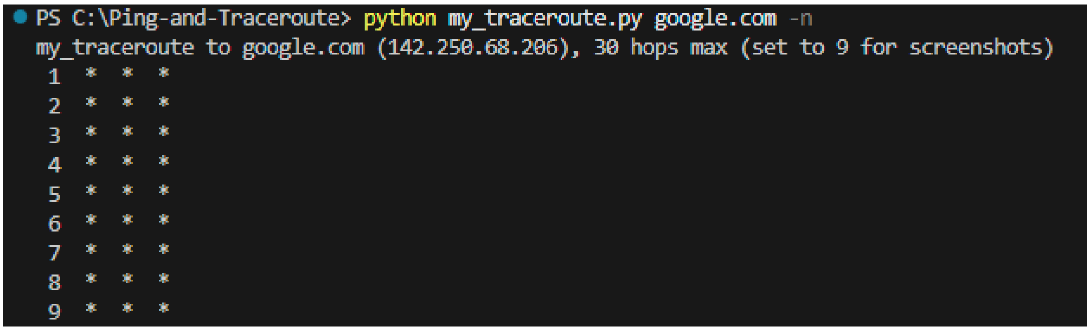
Below demonstrates the route discovery logic and flag implementation for `my_traceroute.py`

3.1 Route Discovery and Formatting



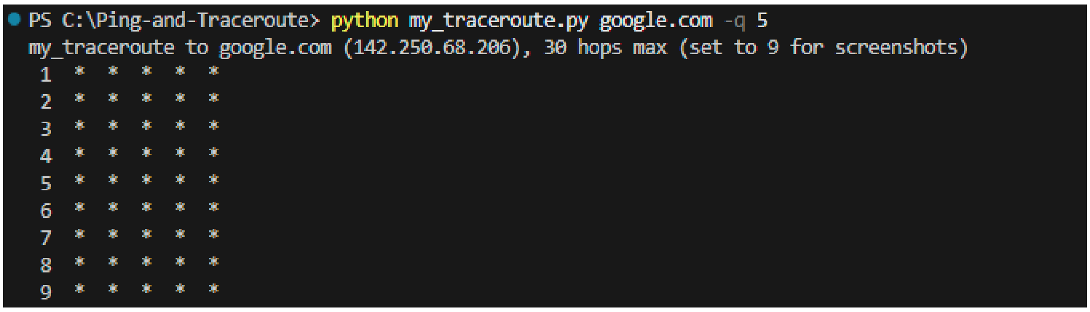
```
PS C:\Ping-and-Traceroute> python my_traceroute.py 8.8.8.8
my_traceroute to 8.8.8.8 (8.8.8.8), 30 hops max (set to 9 for screenshots)
 1  *  *  *
 2  *  *  *
 3  *  *  *
 4  *  *  *
 5  *  *  *
 6  *  *  *
 7  *  *  *
 8  *  *  *
 9  *  *  *
```

Figure 6: Standard traceroute output showing hop discovery



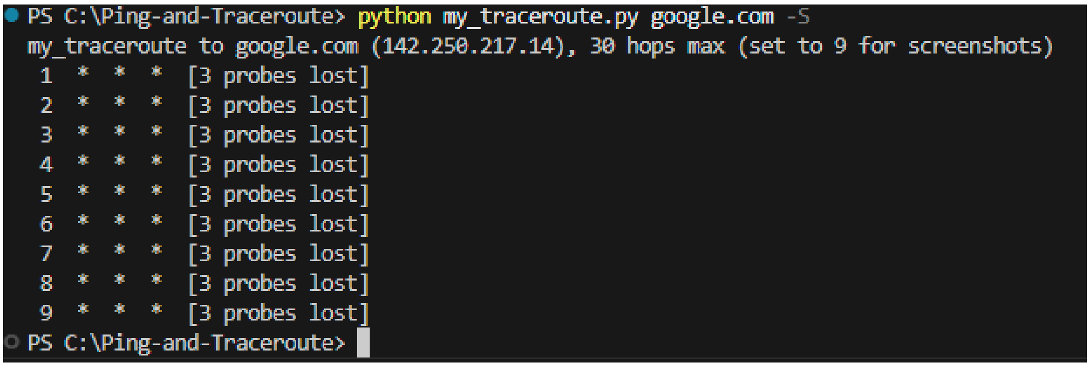
```
PS C:\Ping-and-Traceroute> python my_traceroute.py google.com -n
my_traceroute to google.com (142.250.68.206), 30 hops max (set to 9 for screenshots)
 1  *  *  *
 2  *  *  *
 3  *  *  *
 4  *  *  *
 5  *  *  *
 6  *  *  *
 7  *  *  *
 8  *  *  *
 9  *  *  *
```

Figure 7: Numeric Flag (-n): Displaying IP addresses only



```
PS C:\Ping-and-Traceroute> python my_traceroute.py google.com -q 5
my_traceroute to google.com (142.250.68.206), 30 hops max (set to 9 for screenshots)
 1  *  *  *  *  *
 2  *  *  *  *  *
 3  *  *  *  *  *
 4  *  *  *  *  *
 5  *  *  *  *  *
 6  *  *  *  *  *
 7  *  *  *  *  *
 8  *  *  *  *  *
 9  *  *  *  *  *
```

Figure 8: Queries Flag (-q): Showing 5 RTT measurements per hop



```
PS C:\Ping-and-Traceroute> python my_traceroute.py google.com -S
my_traceroute to google.com (142.250.217.14), 30 hops max (set to 9 for screenshots)
  1  *  *  *  [3 probes lost]
  2  *  *  *  [3 probes lost]
  3  *  *  *  [3 probes lost]
  4  *  *  *  [3 probes lost]
  5  *  *  *  [3 probes lost]
  6  *  *  *  [3 probes lost]
  7  *  *  *  [3 probes lost]
  8  *  *  *  [3 probes lost]
  9  *  *  *  [3 probes lost]
PS C:\Ping-and-Traceroute>
```

Figure 9: Summary Flag (-S): Tracking and reporting probes lost per hop

4 Technical Observation: Network Timeouts

While testing traceroute on the RIT campus network (`wireless.rit.edu`), most of the intermediate hops returned `* * *` in the output. This indicates that no ICMP response was received for those probes.

After researching the issue, I found that many university networks block or filter UDP probes and ICMP "Time Exceeded" messages for security reasons. Since traceroute depends on those responses to identify intermediate routers, the probes time out and display asterisks instead.

I was not able to bypass or disable this network filtering on the campus network

The program handles these situations correctly by:

1. Printing an asterisk for probes that time out
2. Using the `-S` flag to report how many probes were unanswered per hop
3. Continuing to increment the TTL value until the destination is reached or the maximum hop limit (30) is reached