02/20/2023 Shreyas Belkune sb2660

PROJECT 2 : PING AND TRACEROUTE

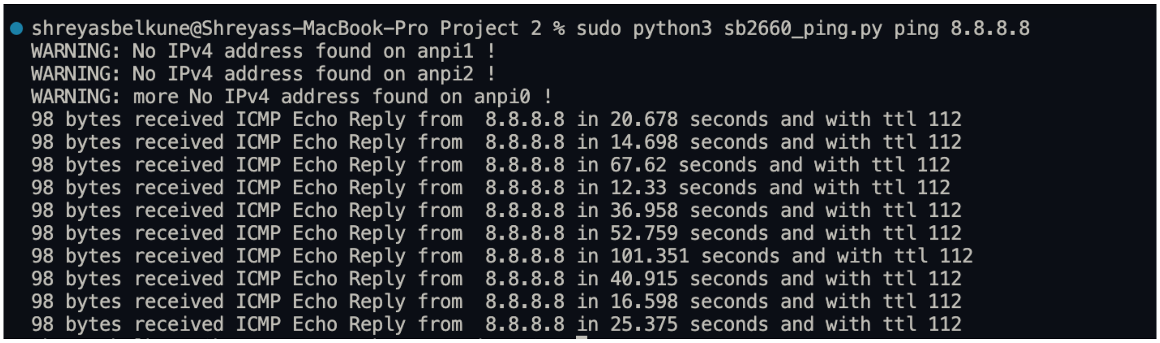
1. PING PROGRAM

This program sends an ICMP echo request to the destination address and expects a response from the destination in general.

A. Running the code

i. To simply run the code and to ping a particular destination continuously-

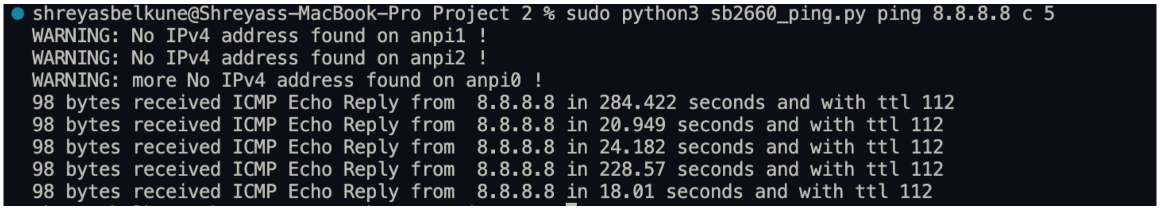
1. *"**sudo python3 sb2660_ping.py ping 8.8.8.8**".*
2. *The above line pings google DNS infinitely until stopped.*
3. *Use control + c to stop the code if running on terminal.*
4. *Output –*



```
● shreyasbelkune@Shreyass-MacBook-Pro Project 2 % sudo python3 sb2660_ping.py ping 8.8.8.8
  WARNING: No IPv4 address found on anpi1 !
  WARNING: No IPv4 address found on anpi2 !
  WARNING: more No IPv4 address found on anpi0 !
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 20.678 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.698 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 67.62 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 12.33 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 36.958 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 52.759 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 101.351 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 40.915 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 16.598 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 25.375 seconds and with ttl 112
```

ii. To run the ping a destination for a certain number of times-

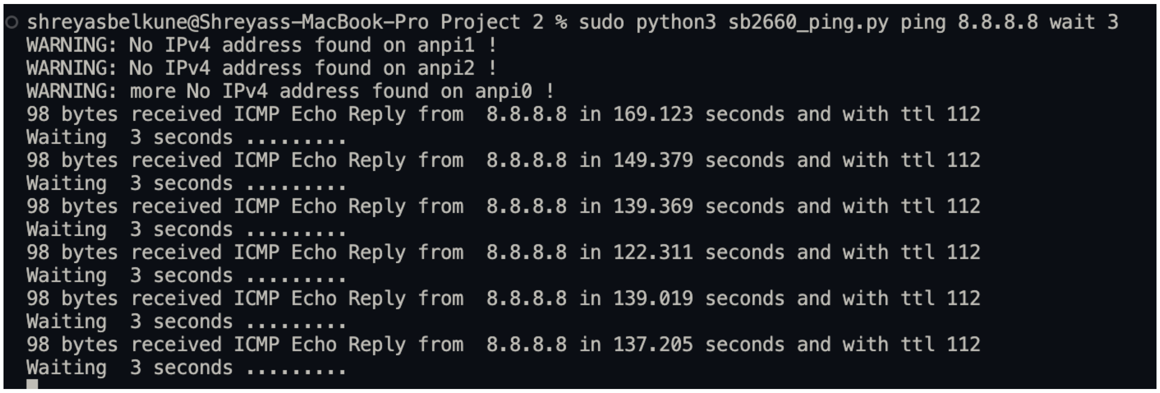
1. *"**sudo python3 sb2660_ping.py ping 8.8.8.8 c 5**".*
2. *The above line pings google DNS 5 times.*
3. *Output –*



```
● shreyasbelkune@Shreyass-MacBook-Pro Project 2 % sudo python3 sb2660_ping.py ping 8.8.8.8 c 5
  WARNING: No IPv4 address found on anpi1 !
  WARNING: No IPv4 address found on anpi2 !
  WARNING: more No IPv4 address found on anpi0 !
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 284.422 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 20.949 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 24.182 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 228.57 seconds and with ttl 112
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 18.01 seconds and with ttl 112
```

iii. To have a delay between every ping packet sent –

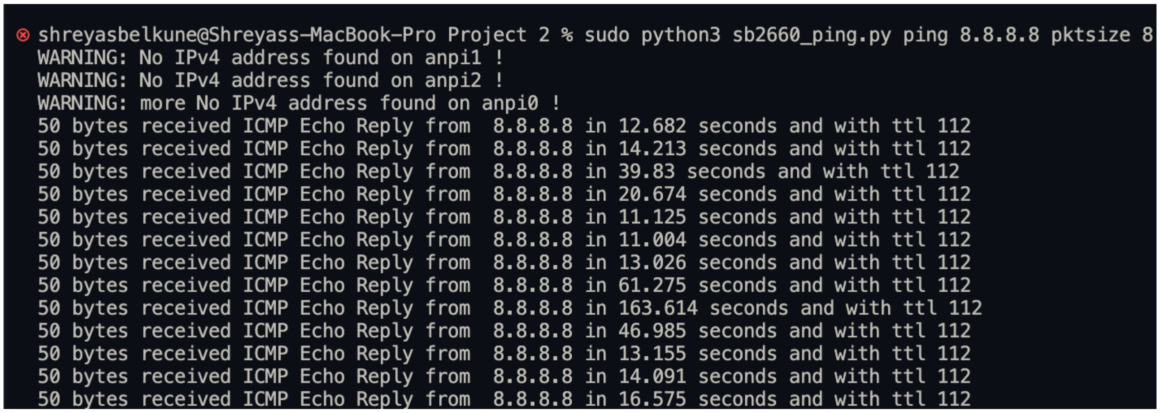
1. ***“sudo python3 sb2660_ping.py ping 8.8.8.8 wait 3”***.
2. *The above line pings google DNS infinitely but will wait 3 seconds before sending the next packet.*
3. *Output –*



```
○ shreyasbelkune@Shreyass-MacBook-Pro Project 2 % sudo python3 sb2660_ping.py ping 8.8.8.8 wait 3
  WARNING: No IPv4 address found on anpi1 !
  WARNING: No IPv4 address found on anpi2 !
  WARNING: more No IPv4 address found on anpi0 !
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 169.123 seconds and with ttl 112
  Waiting  3 seconds .........
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 149.379 seconds and with ttl 112
  Waiting  3 seconds .........
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 139.369 seconds and with ttl 112
  Waiting  3 seconds .........
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 122.311 seconds and with ttl 112
  Waiting  3 seconds .........
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 139.019 seconds and with ttl 112
  Waiting  3 seconds .........
  98 bytes received ICMP Echo Reply from  8.8.8.8 in 137.205 seconds and with ttl 112
  Waiting  3 seconds .........
```

iv. To send a packet with a specific size –

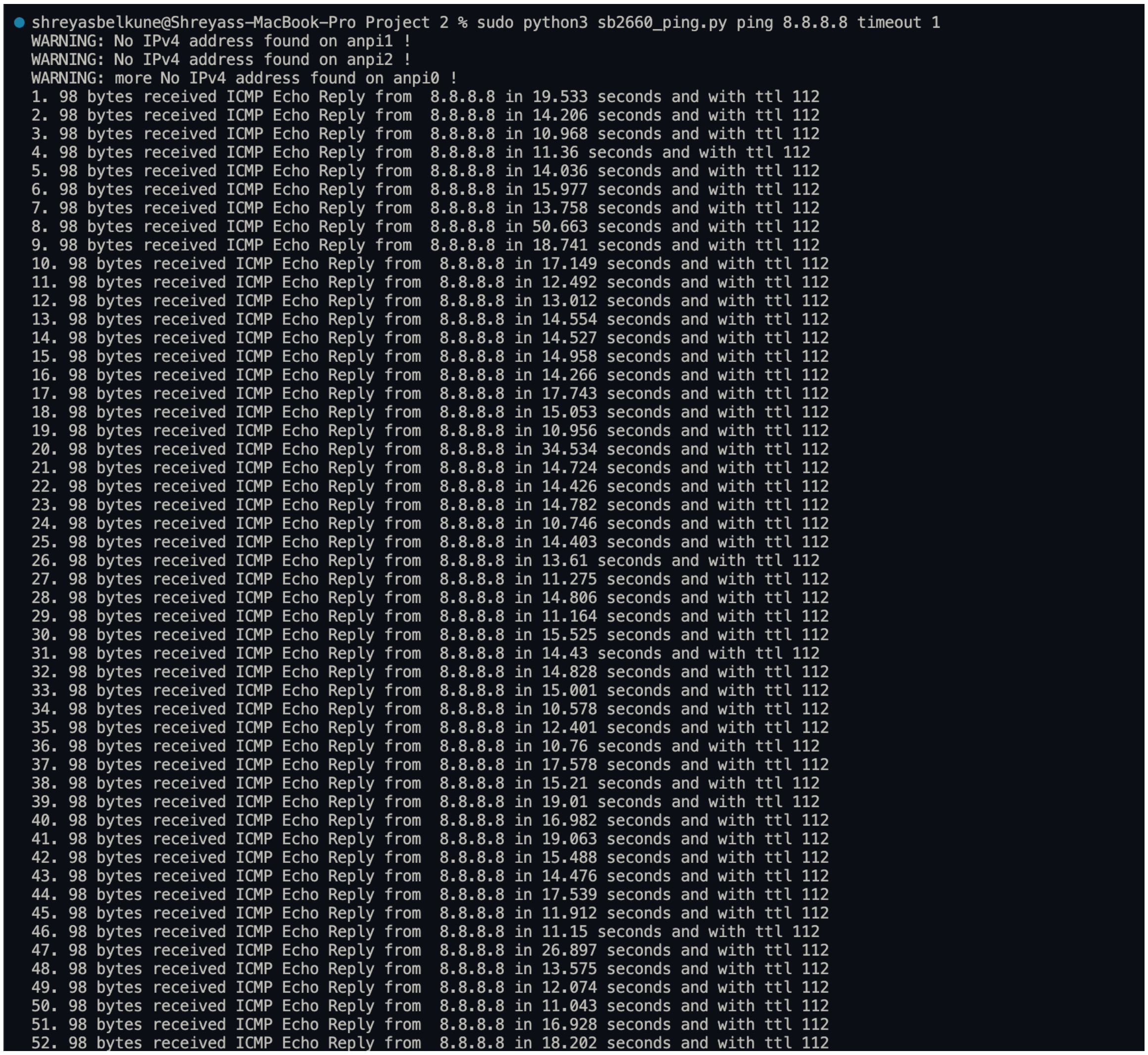
1. ***“sudo python3 sb2660_ping.py ping 8.8.8.8 pktsize 8***”.
2. *Note that the default packet size sent by the program is 98 bytes (Ethernet Header = 14 Bytes, IP Header = 20 Bytes, ICMP Header = 8 Bytes, Data = 56 Bytes) Note that data is also calculated in the protocol header.*
3. *The above input would just send 8 bytes of data making the packet size 50 in this case.*
4. *Output*



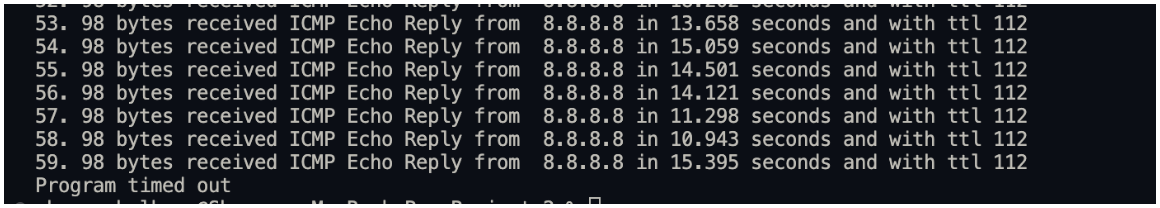
```
⊗ shreyasbelkune@Shreyass-MacBook-Pro Project 2 % sudo python3 sb2660_ping.py ping 8.8.8.8 pktsize 8
  WARNING: No IPv4 address found on anpi1 !
  WARNING: No IPv4 address found on anpi2 !
  WARNING: more No IPv4 address found on anpi0 !
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 12.682 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 14.213 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 39.83 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 20.674 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 11.125 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 11.004 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 13.026 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 61.275 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 163.614 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 46.985 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 13.155 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 14.091 seconds and with ttl 112
  50 bytes received ICMP Echo Reply from  8.8.8.8 in 16.575 seconds and with ttl 112
```

v. To timeout the code after certain time-

1. ***"sudo python3 sb2660_ping.py ping 8.8.8.8 timeout 5"***.
2. *The above input would terminate the code after 5 seconds, if an echo request is sent just before the timeout, and if the response comes after timeout that packet won't be displayed.*
3. *Output*



```
● shreyasbelkune@Shreyass-MacBook-Pro Project 2 % sudo python3 sb2660_ping.py ping 8.8.8.8 timeout 1
  WARNING: No IPv4 address found on anpi1 !
  WARNING: No IPv4 address found on anpi2 !
  WARNING: more No IPv4 address found on anpi0 !
  1. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 19.533 seconds and with ttl 112
  2. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.206 seconds and with ttl 112
  3. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 10.968 seconds and with ttl 112
  4. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 11.36 seconds and with ttl 112
  5. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.036 seconds and with ttl 112
  6. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 15.977 seconds and with ttl 112
  7. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 13.758 seconds and with ttl 112
  8. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 50.663 seconds and with ttl 112
  9. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 18.741 seconds and with ttl 112
  10. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 17.149 seconds and with ttl 112
  11. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 12.492 seconds and with ttl 112
  12. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 13.012 seconds and with ttl 112
  13. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.554 seconds and with ttl 112
  14. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.527 seconds and with ttl 112
  15. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.958 seconds and with ttl 112
  16. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.266 seconds and with ttl 112
  17. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 17.743 seconds and with ttl 112
  18. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 15.053 seconds and with ttl 112
  19. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 10.956 seconds and with ttl 112
  20. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 34.534 seconds and with ttl 112
  21. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.724 seconds and with ttl 112
  22. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.426 seconds and with ttl 112
  23. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.782 seconds and with ttl 112
  24. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 10.746 seconds and with ttl 112
  25. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.403 seconds and with ttl 112
  26. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 13.61 seconds and with ttl 112
  27. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 11.275 seconds and with ttl 112
  28. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.806 seconds and with ttl 112
  29. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 11.164 seconds and with ttl 112
  30. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 15.525 seconds and with ttl 112
  31. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.43 seconds and with ttl 112
  32. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.828 seconds and with ttl 112
  33. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 15.001 seconds and with ttl 112
  34. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 10.578 seconds and with ttl 112
  35. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 12.401 seconds and with ttl 112
  36. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 10.76 seconds and with ttl 112
  37. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 17.578 seconds and with ttl 112
  38. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 15.21 seconds and with ttl 112
  39. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 19.01 seconds and with ttl 112
  40. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 16.982 seconds and with ttl 112
  41. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 19.063 seconds and with ttl 112
  42. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 15.488 seconds and with ttl 112
  43. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.476 seconds and with ttl 112
  44. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 17.539 seconds and with ttl 112
  45. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 11.912 seconds and with ttl 112
  46. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 11.15 seconds and with ttl 112
  47. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 26.897 seconds and with ttl 112
  48. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 13.575 seconds and with ttl 112
  49. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 12.074 seconds and with ttl 112
  50. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 11.043 seconds and with ttl 112
  51. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 16.928 seconds and with ttl 112
  52. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 18.202 seconds and with ttl 112
```



```
 53. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 13.658 seconds and with ttl 112
 54. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 15.059 seconds and with ttl 112
 55. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.501 seconds and with ttl 112
 56. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 14.121 seconds and with ttl 112
 57. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 11.298 seconds and with ttl 112
 58. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 10.943 seconds and with ttl 112
 59. 98 bytes received ICMP Echo Reply from  8.8.8.8 in 15.395 seconds and with ttl 112
 Program timed out
```

2. TRACEROUTE PROGRAM

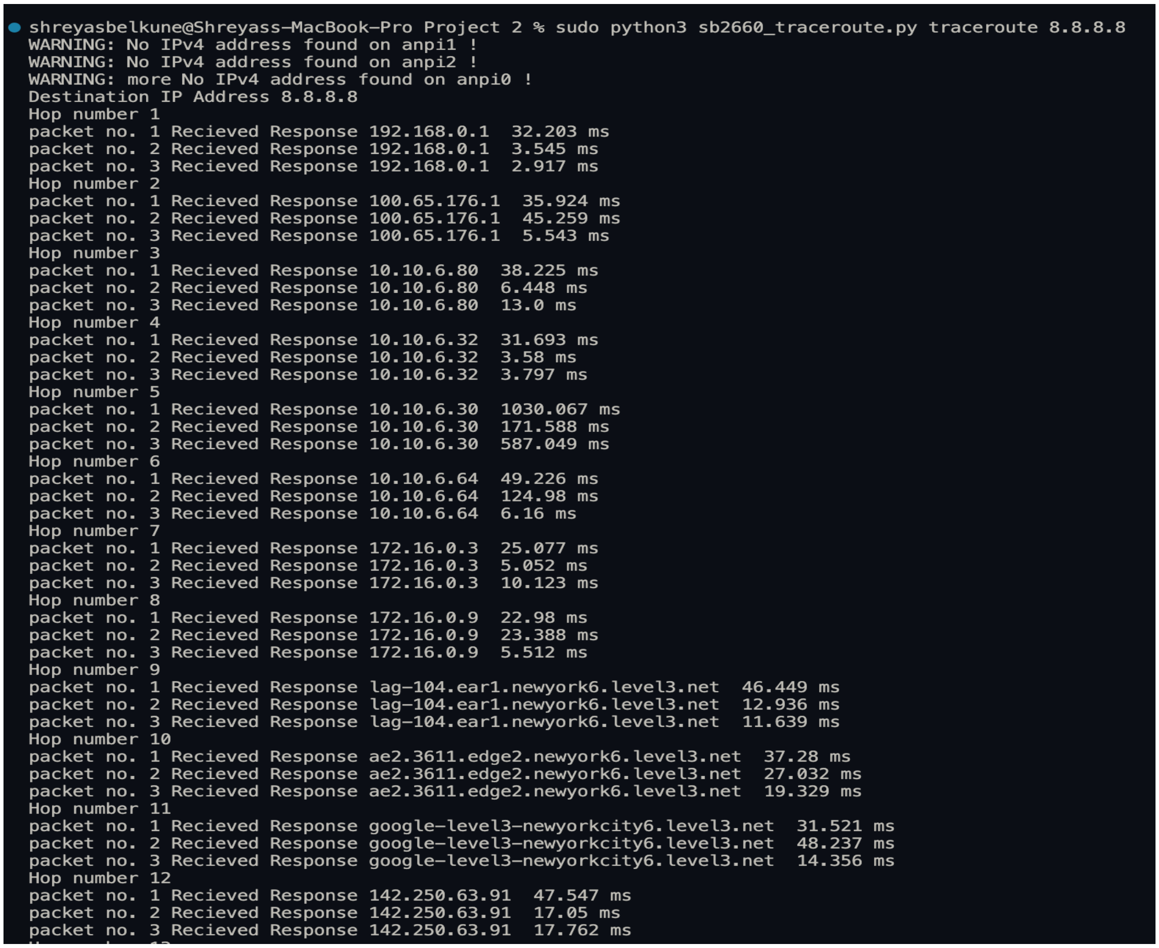
This program finds the route a packet takes to reach the destination address, in this code you’re essentially finding out the route a normal ping packet takes to reach its destination.

At every hop it sends 3 ICMP packet and expects a response from the first hop router/device to tell us the sender that the packet we’re sending won’t reach the desired destination as the TTL(Time to live) has decremented to 0. With every hop a packet takes the TTL decrements by 1.

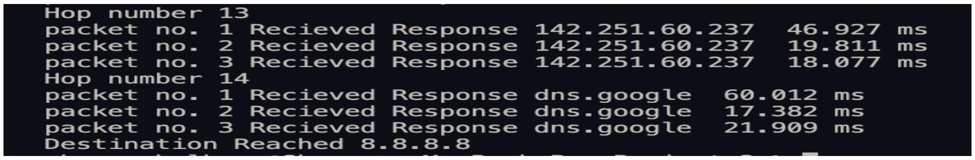
A. Running the code

i. To generally find the route –

1. *“**sudo python3 sb2660_traceroute.py traceroute 8.8.8.8**”.*
2. *We’re trying to find the route a packet takes to reach Google DNS Server.*
3. *Output*



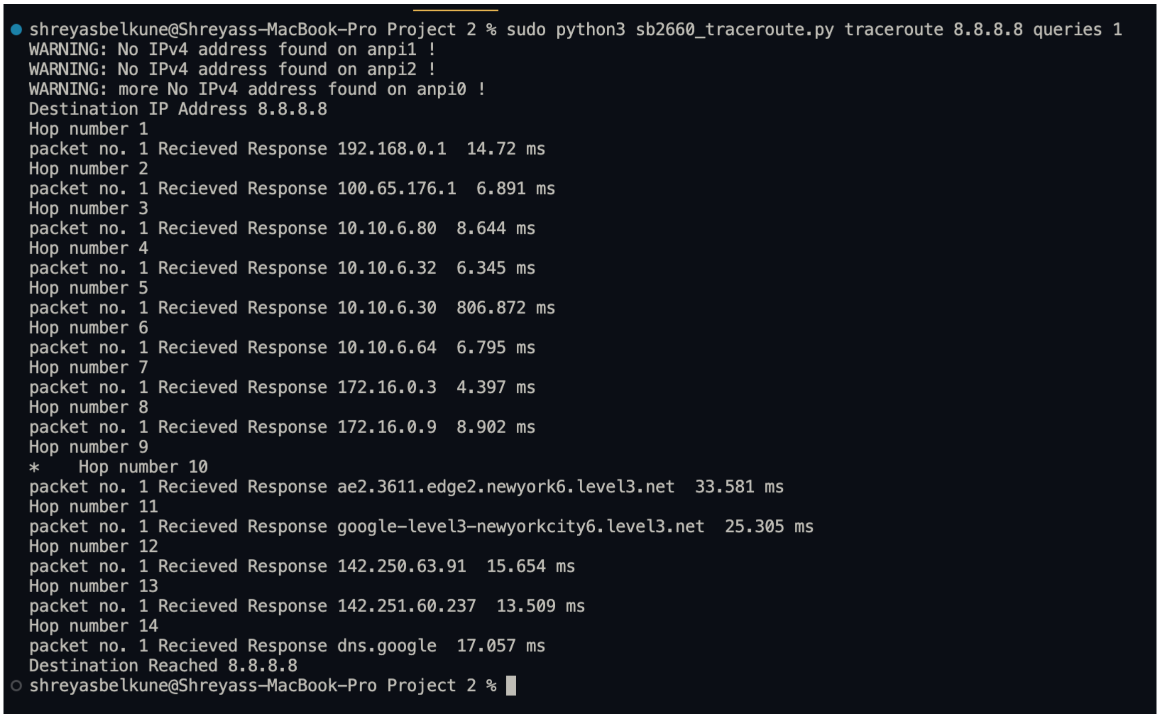
```
shreyasbelkune@Shreyass-MacBook-Pro Project 2 % sudo python3 sb2660_traceroute.py traceroute 8.8.8.8
WARNING: No IPv4 address found on anpi1 !
WARNING: No IPv4 address found on anpi2 !
WARNING: more No IPv4 address found on anpi0 !
Destination IP Address 8.8.8.8
Hop number 1
packet no. 1 Recieved Response 192.168.0.1  32.203 ms
packet no. 2 Recieved Response 192.168.0.1  3.545 ms
packet no. 3 Recieved Response 192.168.0.1  2.917 ms
Hop number 2
packet no. 1 Recieved Response 100.65.176.1  35.924 ms
packet no. 2 Recieved Response 100.65.176.1  45.259 ms
packet no. 3 Recieved Response 100.65.176.1  5.543 ms
Hop number 3
packet no. 1 Recieved Response 10.10.6.80  38.225 ms
packet no. 2 Recieved Response 10.10.6.80  6.448 ms
packet no. 3 Recieved Response 10.10.6.80  13.0 ms
Hop number 4
packet no. 1 Recieved Response 10.10.6.32  31.693 ms
packet no. 2 Recieved Response 10.10.6.32  3.58 ms
packet no. 3 Recieved Response 10.10.6.32  3.797 ms
Hop number 5
packet no. 1 Recieved Response 10.10.6.30  1030.067 ms
packet no. 2 Recieved Response 10.10.6.30  171.588 ms
packet no. 3 Recieved Response 10.10.6.30  587.049 ms
Hop number 6
packet no. 1 Recieved Response 10.10.6.64  49.226 ms
packet no. 2 Recieved Response 10.10.6.64  124.98 ms
packet no. 3 Recieved Response 10.10.6.64  6.16 ms
Hop number 7
packet no. 1 Recieved Response 172.16.0.3  25.077 ms
packet no. 2 Recieved Response 172.16.0.3  5.052 ms
packet no. 3 Recieved Response 172.16.0.3  10.123 ms
Hop number 8
packet no. 1 Recieved Response 172.16.0.9  22.98 ms
packet no. 2 Recieved Response 172.16.0.9  23.388 ms
packet no. 3 Recieved Response 172.16.0.9  5.512 ms
Hop number 9
packet no. 1 Recieved Response lag-104.ear1.newyork6.level3.net  46.449 ms
packet no. 2 Recieved Response lag-104.ear1.newyork6.level3.net  12.936 ms
packet no. 3 Recieved Response lag-104.ear1.newyork6.level3.net  11.639 ms
Hop number 10
packet no. 1 Recieved Response ae2.3611.edge2.newyork6.level3.net  37.28 ms
packet no. 2 Recieved Response ae2.3611.edge2.newyork6.level3.net  27.032 ms
packet no. 3 Recieved Response ae2.3611.edge2.newyork6.level3.net  19.329 ms
Hop number 11
packet no. 1 Recieved Response google-level3-newyorkcity6.level3.net  31.521 ms
packet no. 2 Recieved Response google-level3-newyorkcity6.level3.net  48.237 ms
packet no. 3 Recieved Response google-level3-newyorkcity6.level3.net  14.356 ms
Hop number 12
packet no. 1 Recieved Response 142.250.63.91  47.547 ms
packet no. 2 Recieved Response 142.250.63.91  17.05 ms
packet no. 3 Recieved Response 142.250.63.91  17.762 ms
```



```
Hop number 13
packet no. 1 Recieved Response 142.251.60.237  46.927 ms
packet no. 2 Recieved Response 142.251.60.237  19.811 ms
packet no. 3 Recieved Response 142.251.60.237  18.077 ms
Hop number 14
packet no. 1 Recieved Response dns.google  60.012 ms
packet no. 2 Recieved Response dns.google  17.382 ms
packet no. 3 Recieved Response dns.google  21.909 ms
Destination Reached 8.8.8.8
```

ii. To alter the number of queries sent per hop –

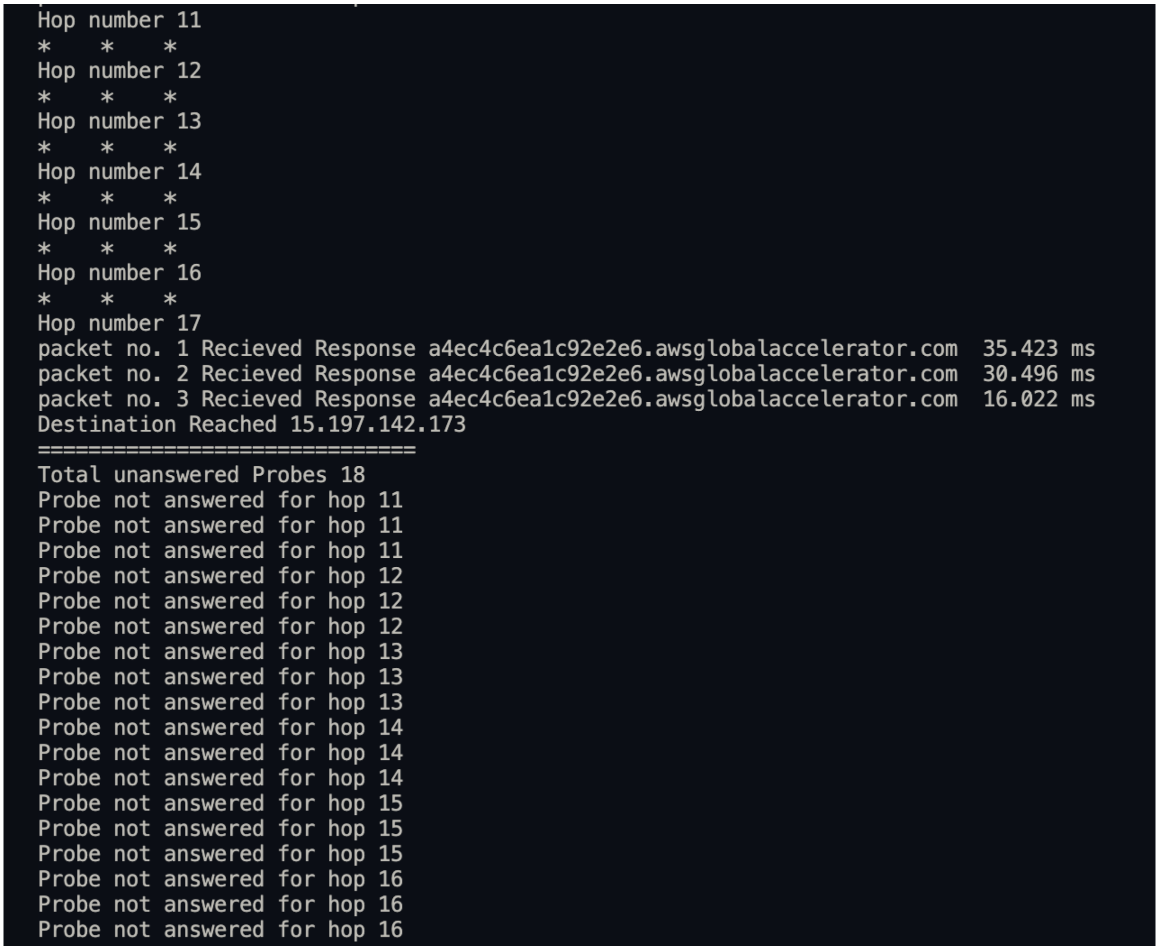
1. *“**sudo python3 sb2660_traceroute.py traceroute 8.8.8.8 queries 1**”.*
2. *In general, traceroute send 3 probes per hop, in the above given input we change that to 1.*
3. *Output*



```
● shreyasbelkune@Shreyass-MacBook-Pro Project 2 % sudo python3 sb2660_traceroute.py traceroute 8.8.8.8 queries 1
  WARNING: No IPv4 address found on anpi1 !
  WARNING: No IPv4 address found on anpi2 !
  WARNING: more No IPv4 address found on anpi0 !
  Destination IP Address 8.8.8.8
  Hop number 1
  packet no. 1 Recieved Response 192.168.0.1  14.72 ms
  Hop number 2
  packet no. 1 Recieved Response 100.65.176.1  6.891 ms
  Hop number 3
  packet no. 1 Recieved Response 10.10.6.80  8.644 ms
  Hop number 4
  packet no. 1 Recieved Response 10.10.6.32  6.345 ms
  Hop number 5
  packet no. 1 Recieved Response 10.10.6.30  806.872 ms
  Hop number 6
  packet no. 1 Recieved Response 10.10.6.64  6.795 ms
  Hop number 7
  packet no. 1 Recieved Response 172.16.0.3  4.397 ms
  Hop number 8
  packet no. 1 Recieved Response 172.16.0.9  8.902 ms
  Hop number 9
  *    Hop number 10
  packet no. 1 Recieved Response ae2.3611.edge2.newyork6.level3.net  33.581 ms
  Hop number 11
  packet no. 1 Recieved Response google-level3-newyorkcity6.level3.net  25.305 ms
  Hop number 12
  packet no. 1 Recieved Response 142.250.63.91  15.654 ms
  Hop number 13
  packet no. 1 Recieved Response 142.251.60.237  13.509 ms
  Hop number 14
  packet no. 1 Recieved Response dns.google  17.057 ms
  Destination Reached 8.8.8.8
○ shreyasbelkune@Shreyass-MacBook-Pro Project 2 %
```

iii. To print the summary of how many probes weren’t answered-

1. *“**sudo python3 sb2660_traceroute.py traceroute 8.8.8.8 summary**”.*
2. *The above code will still print the route the packet took to reach the destination, but in the end, it’ll tell you on which hop the probe wasn’t answered.*
3. *An unanswered probe is denoted by a “*”, if a packet is sent and a response is not received in 5 seconds, that probe has timed out.*
4. *If 3 probes at a certain hop, it’ll increase the TTL by 1 and moves on to the next hop.*
5. *Output-*



```
Hop number 11
*    *    *
Hop number 12
*    *    *
Hop number 13
*    *    *
Hop number 14
*    *    *
Hop number 15
*    *    *
Hop number 16
*    *    *
Hop number 17
packet no. 1 Recieved Response a4ec4c6ea1c92e2e6.awsglobalaccelerator.com  35.423 ms
packet no. 2 Recieved Response a4ec4c6ea1c92e2e6.awsglobalaccelerator.com  30.496 ms
packet no. 3 Recieved Response a4ec4c6ea1c92e2e6.awsglobalaccelerator.com  16.022 ms
Destination Reached 15.197.142.173
==============================
Total unanswered Probes 18
Probe not answered for hop 11
Probe not answered for hop 11
Probe not answered for hop 11
Probe not answered for hop 12
Probe not answered for hop 12
Probe not answered for hop 12
Probe not answered for hop 13
Probe not answered for hop 13
Probe not answered for hop 13
Probe not answered for hop 14
Probe not answered for hop 14
Probe not answered for hop 14
Probe not answered for hop 15
Probe not answered for hop 15
Probe not answered for hop 15
Probe not answered for hop 16
Probe not answered for hop 16
Probe not answered for hop 16
```

3. RESOURCES USED AND RESTRICTIONS

A. Libraries Used

i. Scapy – packet creation (https://scapy.readthedocs.io/en/latest/usage.html)

ii. Socket – to send and receive packet, also for altering TTL (https://docs.python.org/3/library/socket.html)

iii. Time – For calculating round trip time .

iv. Struct – To unpack the receiving packet

v. Sys – To take in command line arguments

vi. Signal – To terminate the entire code after a certain time. (https://docs.python.org/3/library/signal.html)

B. Restrictions

i. The programs don't take in multiple filters

ii. The program will will display a host name whenever it found for a particular ip, else it'll print just the ip.