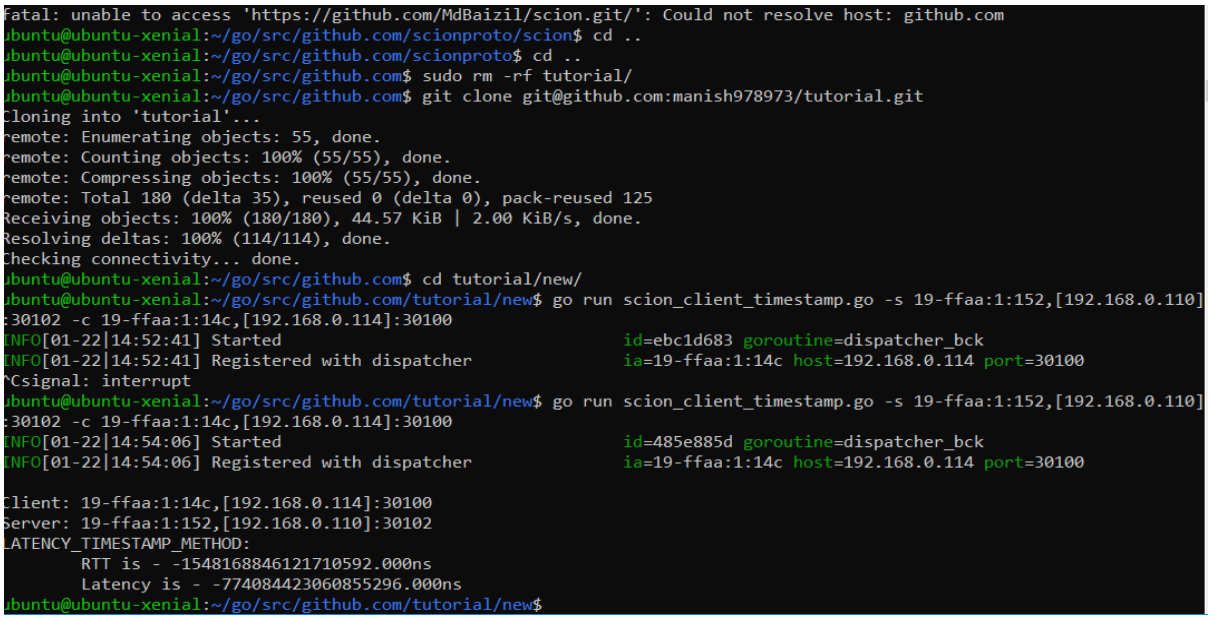
Latency

Timestamp (Task 1) (Screenshot attached)



The code has been written and implemented. I have referred the sensor and homework codes as a reference.

1. For the Ideal condition, the times in both the clocks (client and server side) should be same so that there is no hindrance in computing latency.

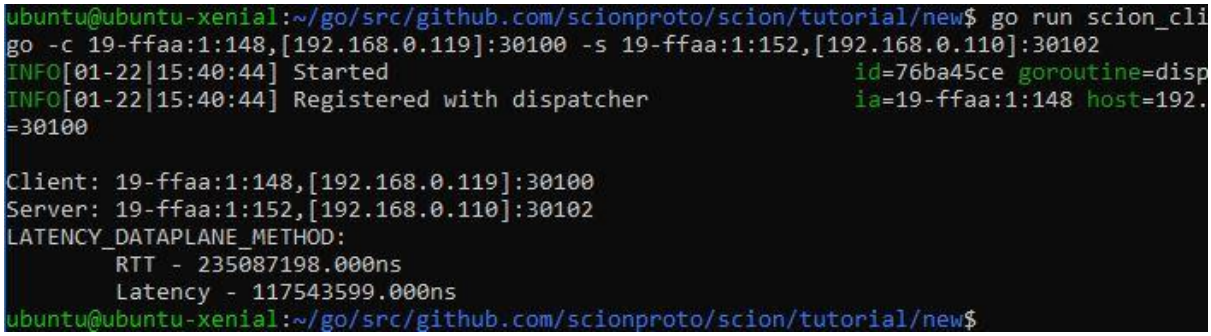
2. Latency = time_received - time_sent where time is computed from Bandwidth and the size of the packet.

3. Latency could be affected by bandwidth, size of the packet and time syncing.

4. These overhead are dynamic since bandwidth ,size,applications ,time varies.

5. The overheads depends on client and server applications which is not easy to overcome because of their complex code and timings .

Data Plane (task 2)



The code has been written and implemented. I have referred the sensor and homework codes as a reference

1. In todays Non SCION Architecture the response from the server to the client would take some different path which would result in different timings and hence RTT would be the double of Latency.However in SCION Path(client to server) = Path(server to client)...Hence RTT would be the double of Latency.

2. The UDP connection (LISTENS TO SCION) in server and ensures the response back to the client only once some response or number is received.

Control Plane(task 3) (code not implemented)

I didn't implement this since I couldn't understand the logic behind its implementation.

1. B responses to A only when A requests for the echo which ensures B's echos are not sent before A's requests.

2. B can counterfeit the results hence extending the RTT. Hence it can be varied with changes done from B's side.

BandWidth

Assumptions. No screenshot since there are several errors in client code.

The server code has been implemented however the client code is incomplete. I am encountering several errors.

In the code (The main function determines the path used for establishing the scion communication and the udp connection is established. A map is created using the send and recieved packets. This map is used in calculating the average bottleneck bandwidth. The output is converted to Mbps after sorting the average time interval consumed for packet transmission ie send and recieved.)

At bottleneck link, the two packets should queue one after the other and not at some other link after that. If both packet reach at almost same time, they could queue together at bottleneck line

•

There wont be packet loss in between. If its lost, there will be big time difference. Moreover, the router treats all packets equally. Once first packet reaches, it will take sometime to find routing direction. Then second one is expected to have same characteristics so it will get processed fast