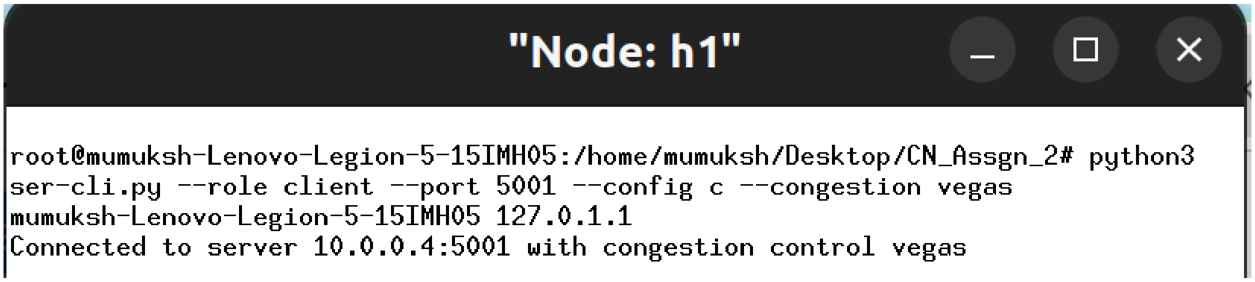
**Q2)**

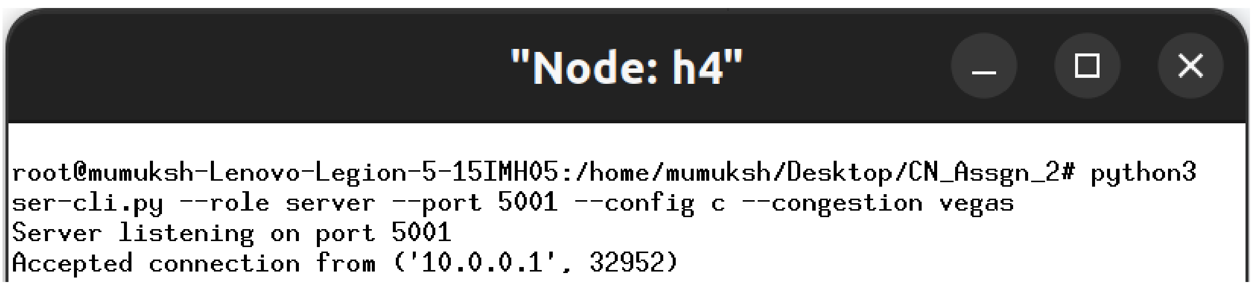
**(a)** Python implementation for custom topology and server-client script uploaded on the github repository with steps

XTerm interface for client




Client side command to run

XTerm interface for server

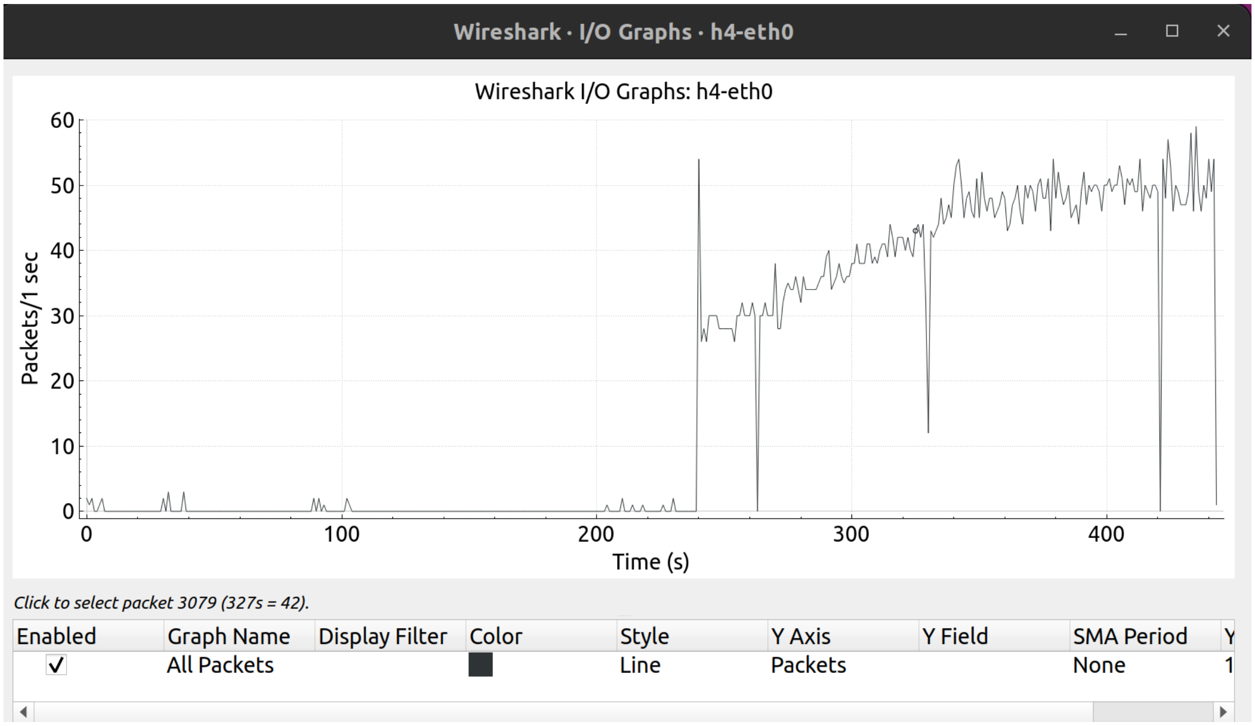



Server side command to run

**(b)** Throughput for various congestion control schemes (Vegas, Reno, Cubic, BBR) (Config=b for all of the following throughputs)

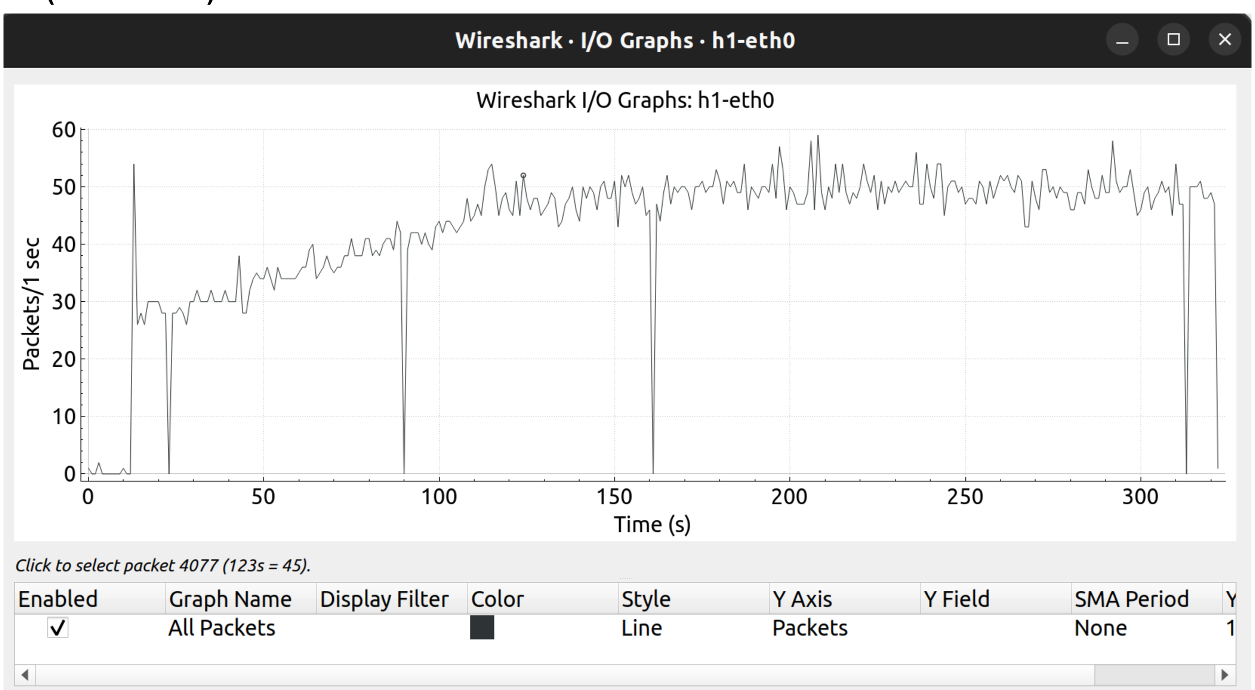
**Congestion=Vegas**

**H4 (Server Side)**




Packets/sec received vs Time plot for server side (Initial rate of 0 is because of no packets being transferred in that duration)

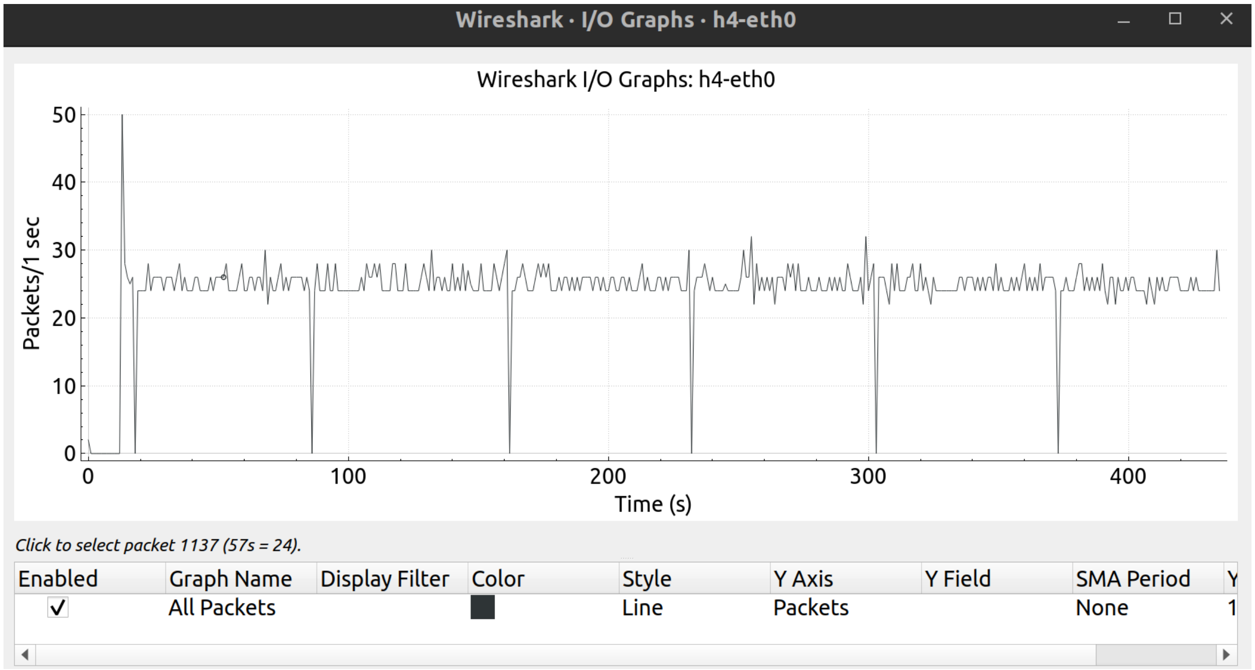
**H1 (Client Side)**




Packets/sec sent vs Time plot for client side

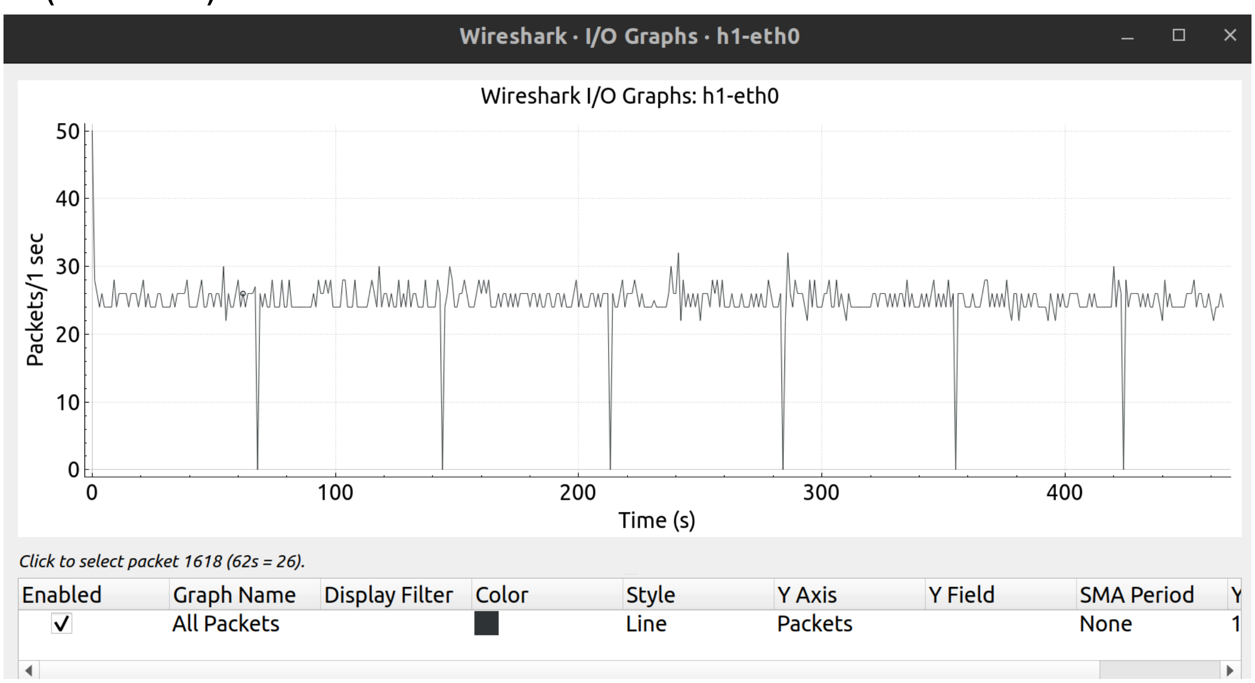
## Congestion=Reno

### H4 (Server Side)




Packets/sec received vs Time plot for server side

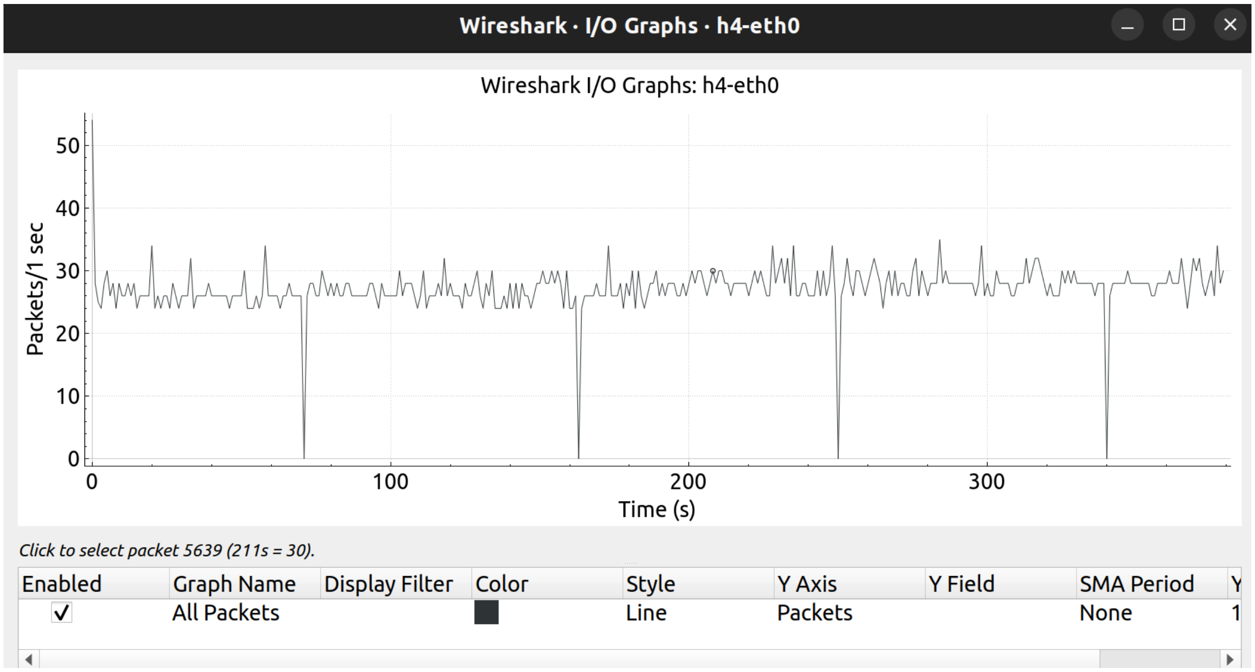
### H1 (Client Side)




Packets/sec sent vs Time plot for client side

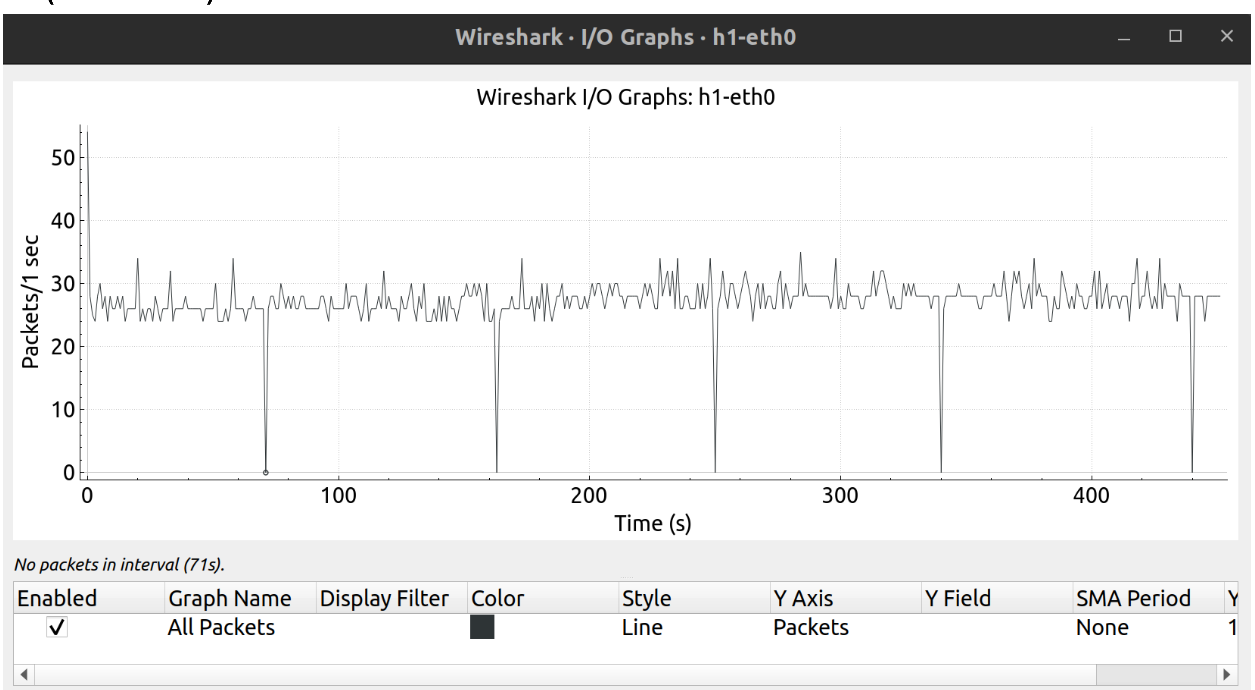
# Congestion=Cubic

## H4 (Server Side)




Packets/sec received vs Time plot for server side

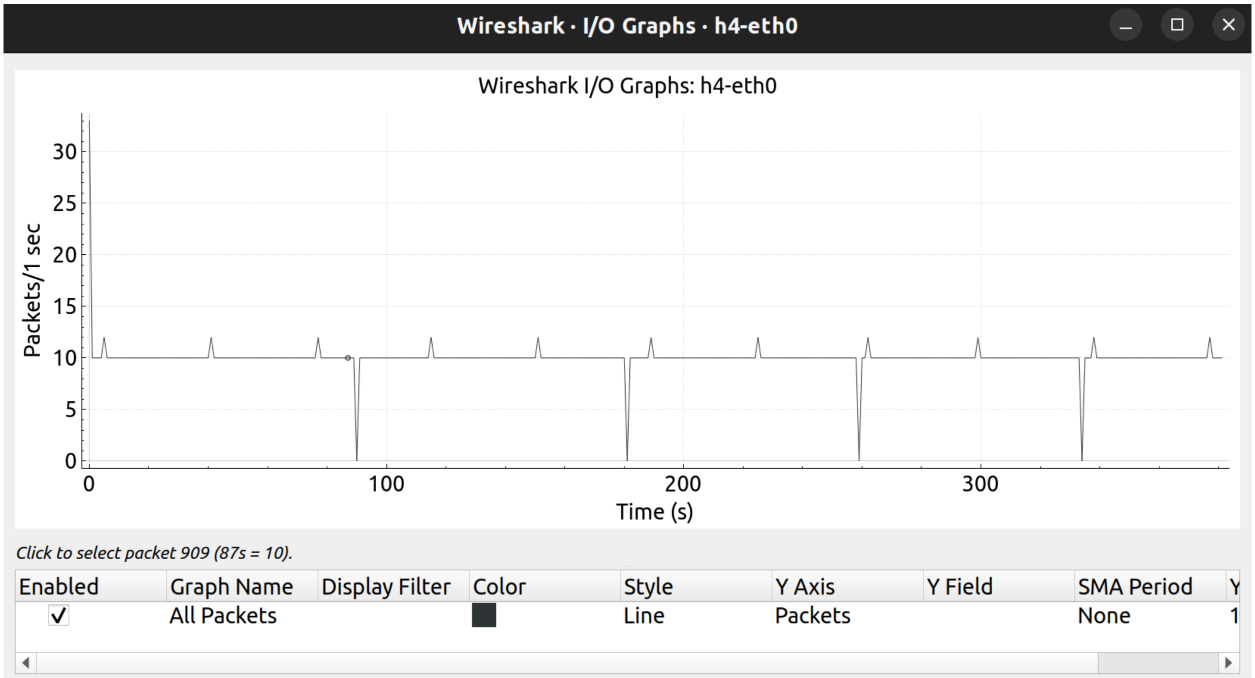
## H1 (Client Side)




Packets/sec sent vs Time plot for client side

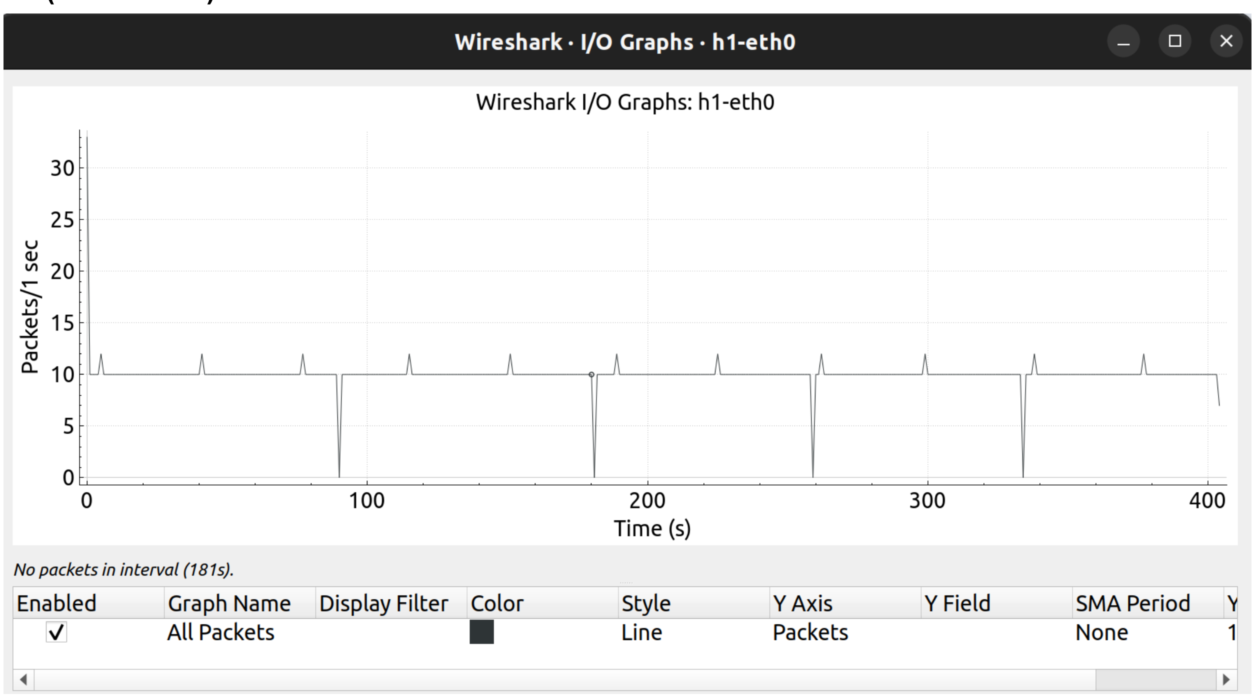
## Congestion=BBR

### H4 (Server Side)




Packets/sec received vs Time plot for server side

### H1 (Client Side)




Packets/sec sent vs Time plot for client side

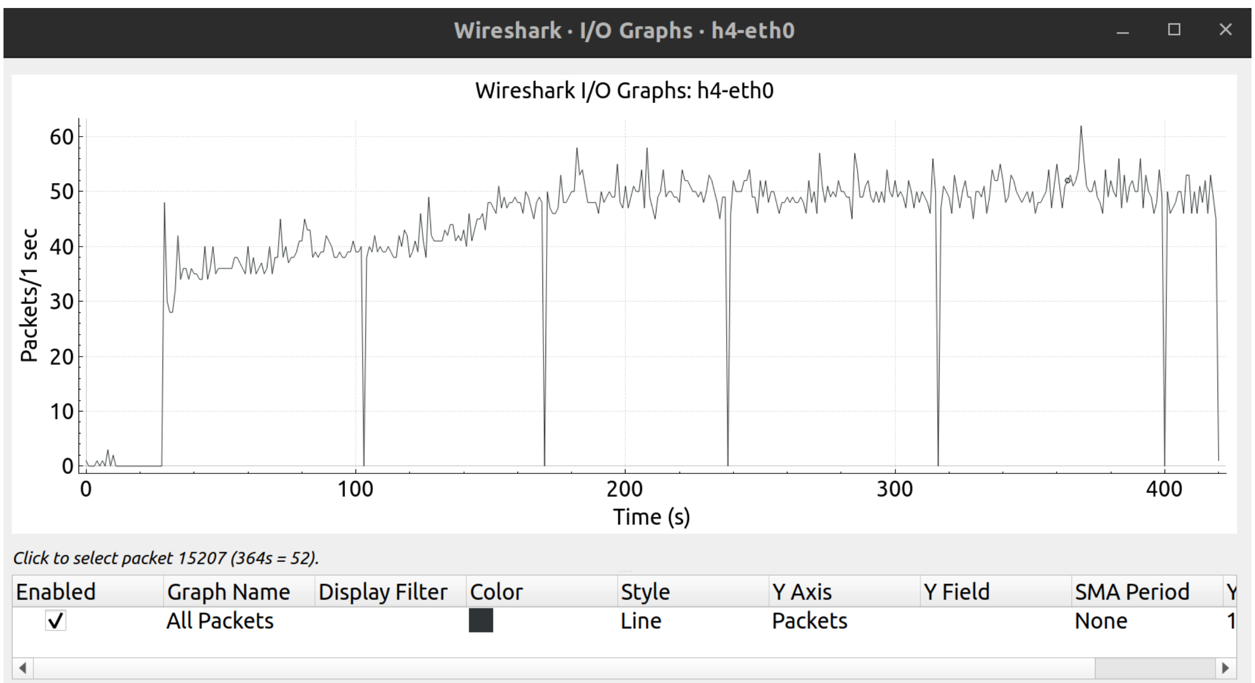
**Part (b) Result:** From all the plots that we can see, Vegas congestion control scheme has offered the highest throughput rate whereas the BBR congestion control scheme has offered the least throughput rate. However, the packet transfer rate has remained relatively conservative for reno congestion control scheme when compared to others whereas cubic has been relatively aggressive.

**(c)**

**(d)** Throughput for various congestion control schemes (Vegas, Reno, Cubic, BBR) but with link loss of 1% and 3% (Config=b for all of the following throughputs) <u>(link loss to be changed in custom topology script)</u>

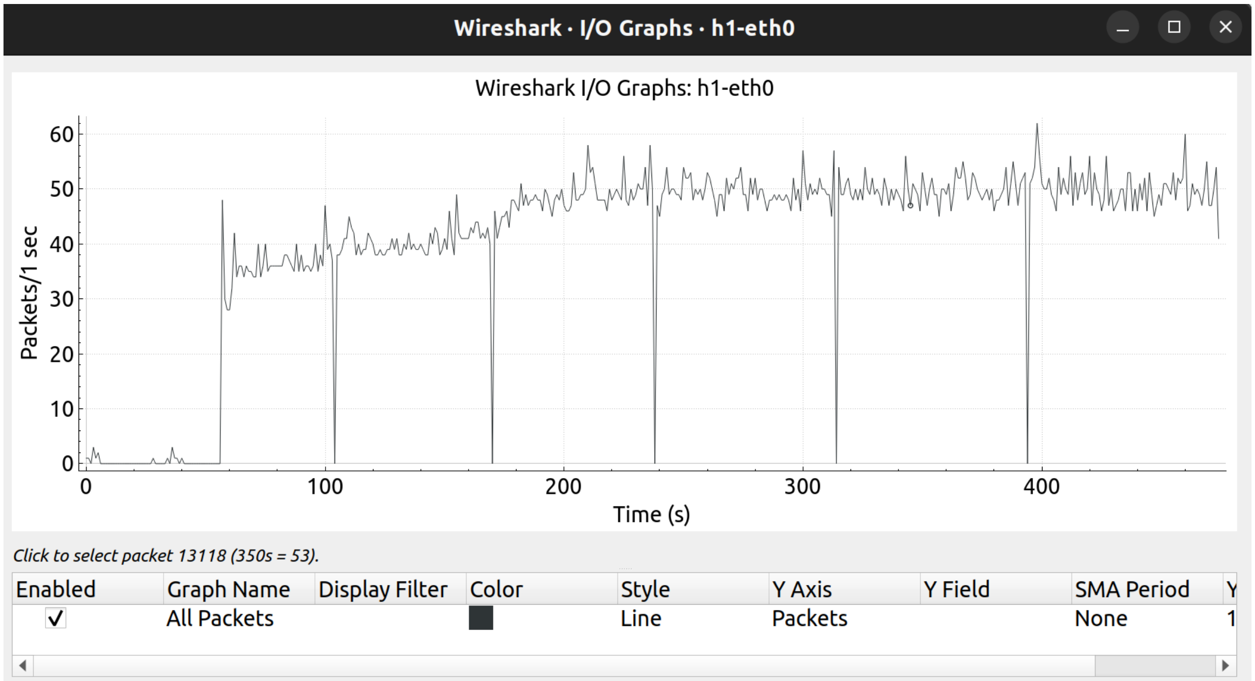
**<u>Congestion=Vegas, loss=1%</u>**

**H4 (Server Side)**




<u>Packets/sec received vs Time plot for server side</u>

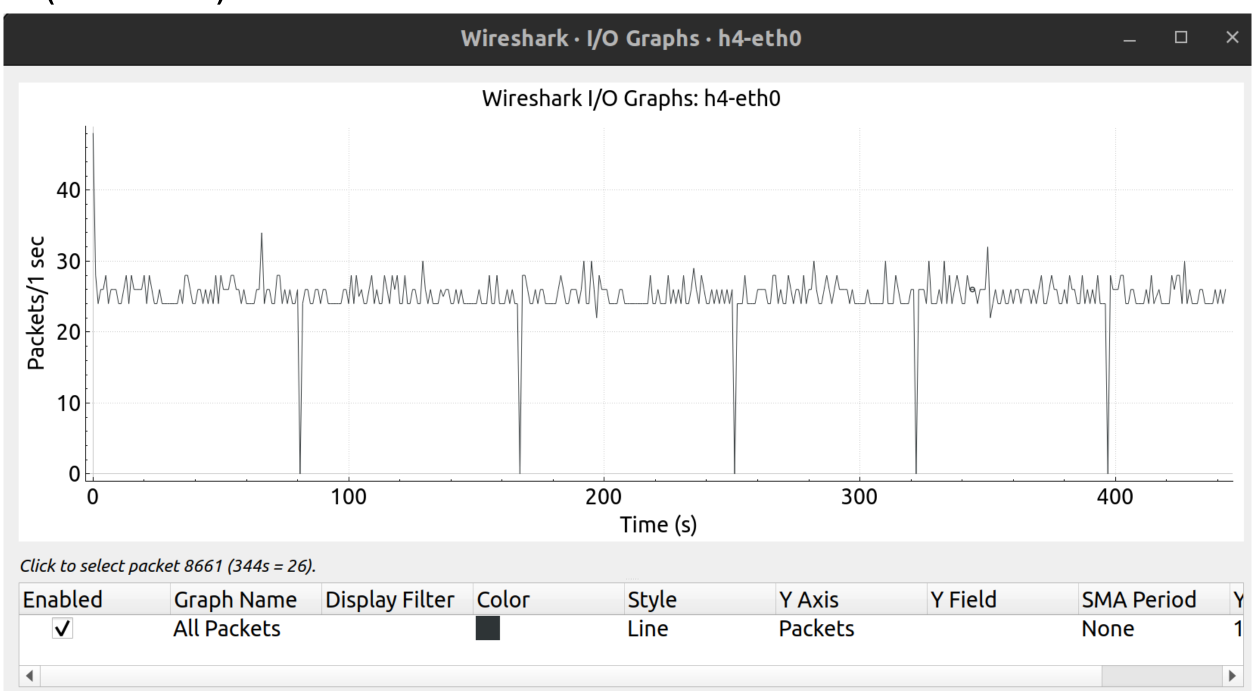
**H1 (Client Side)**




Packets/sec sent vs Time plot for client side

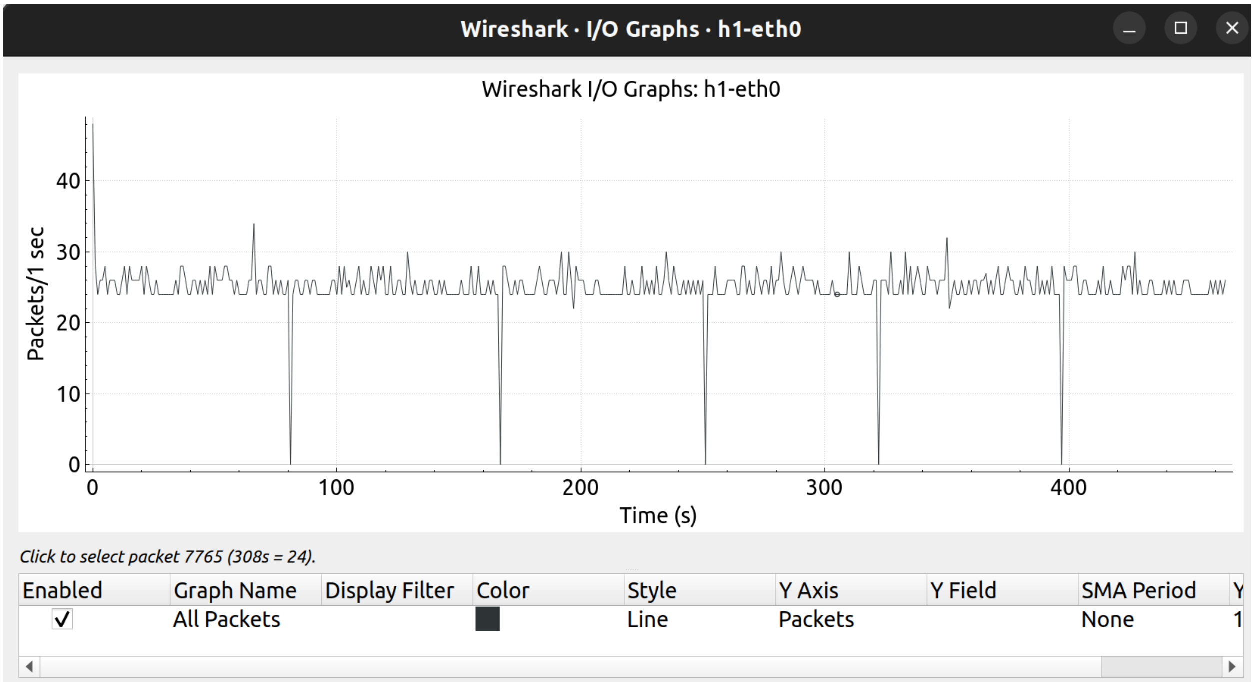
## Congestion=Reno, loss=1%

### H4 (Server Side)




Packets/sec received vs Time plot for server side

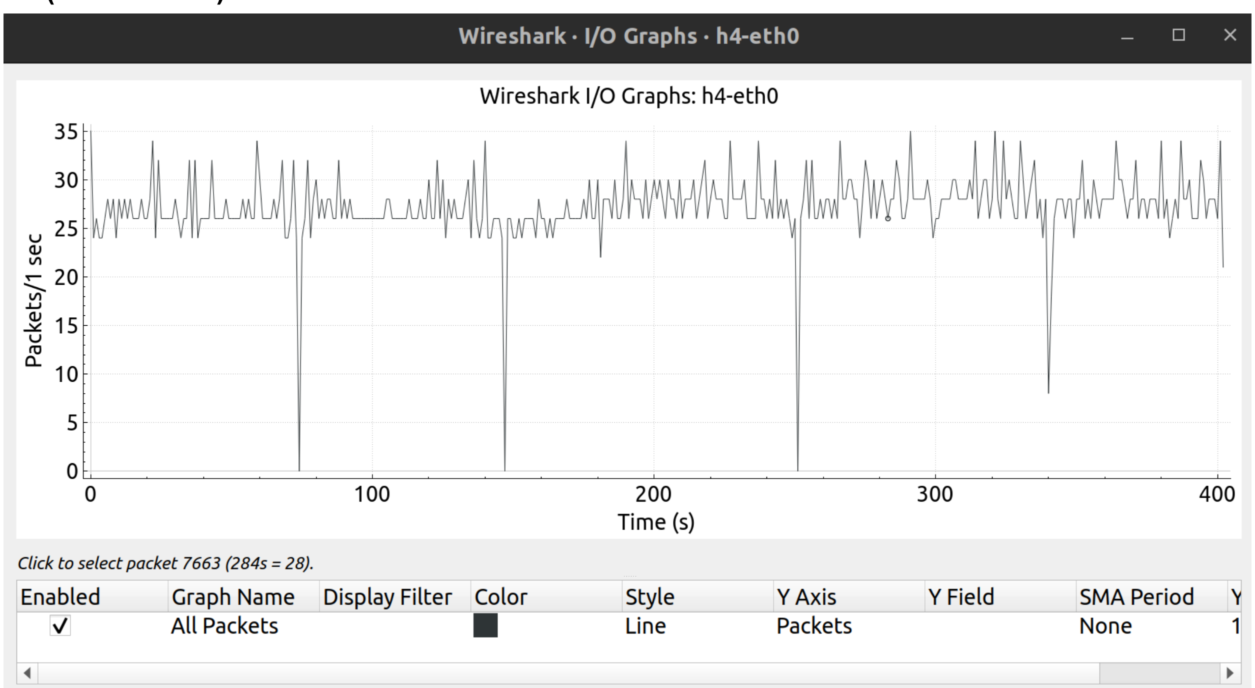
### H1 (Client Side)




Packets/sec sent vs Time plot for client side

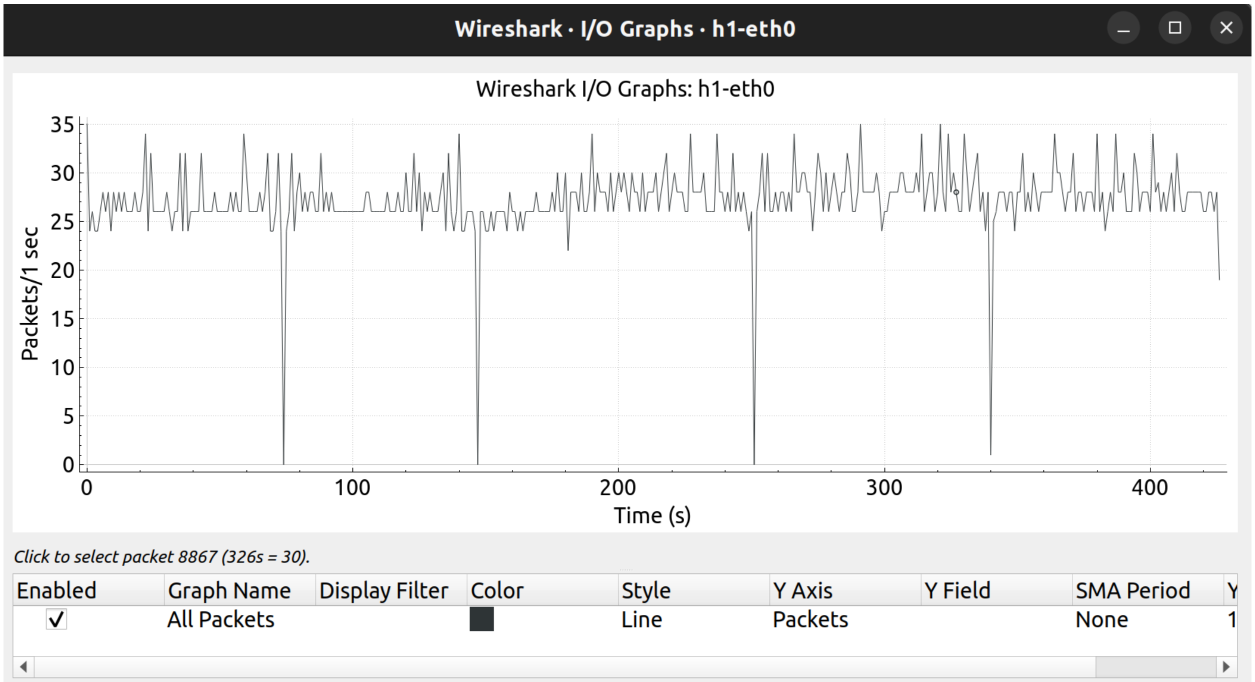
**Congestion=Cubic, loss=1%**

### H4 (Server Side)




Packets/sec received vs Time plot for server side

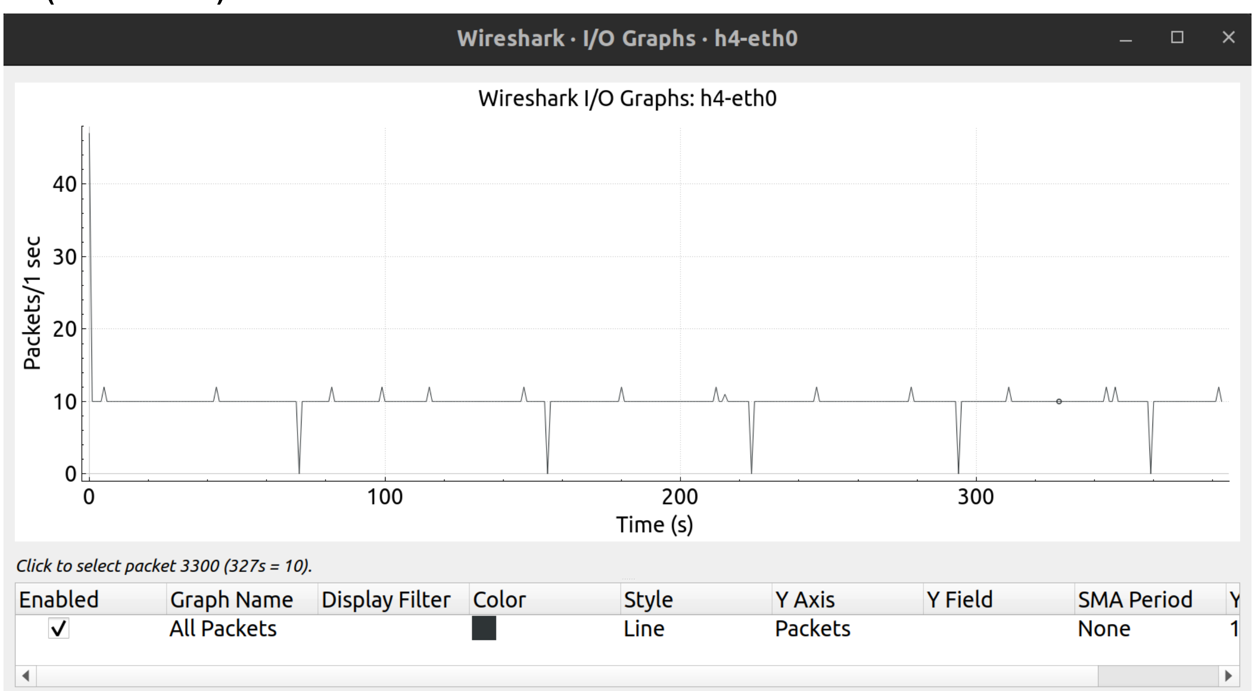
### H1 (Client Side)




Packets/sec sent vs Time plot for client side

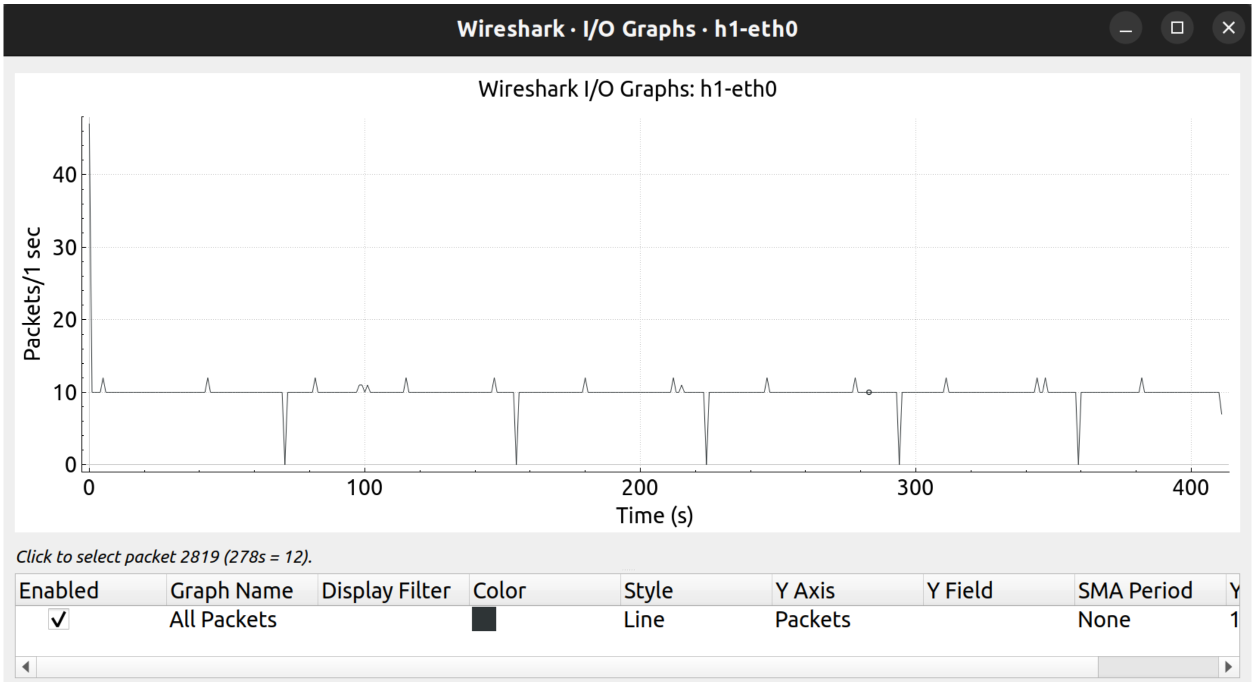
**Congestion=BBR, loss=1%**

### H4 (Server Side)




Packets/sec received vs Time plot for server side

### H1 (Client Side)

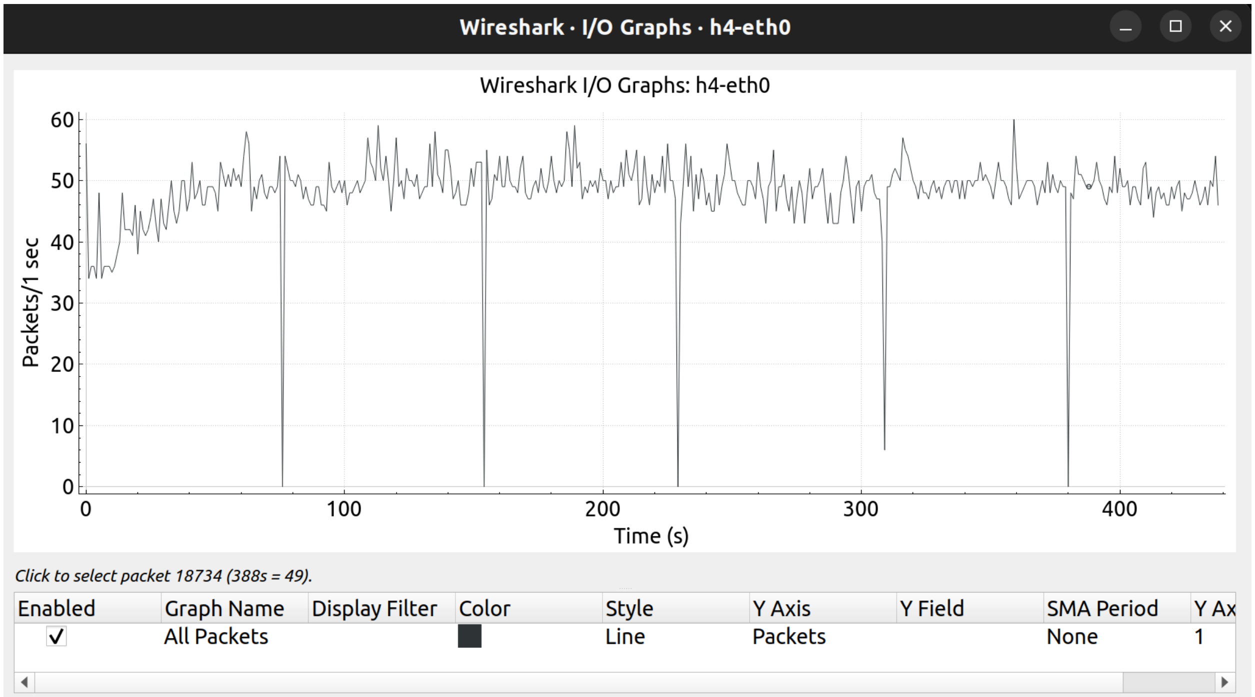



Packets/sec sent vs Time plot for client side

**Result (d (loss=1%)):** The overall observations are somewhat similar to what we observed in part (b), however, 1% link loss between the switches has caused more inconsistent throughput rates, i.e.more frequent fluctuations, is higher for all 4 congestion control schemes. This is because lost packets are supposed to be retransmitted which causes a more dynamicity in send rates.

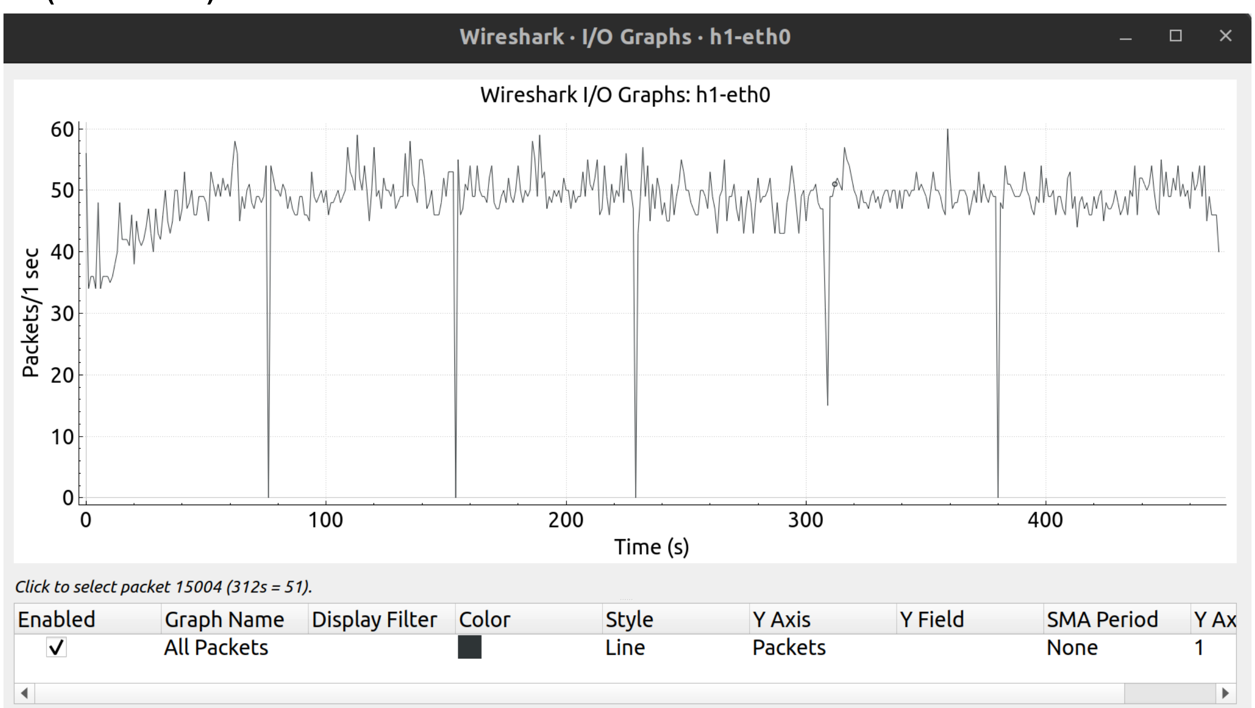
**Congestion=Vegas, loss=3%**

**H4 (Server Side)**




<u>Packets/sec received vs Time plot for server side</u>

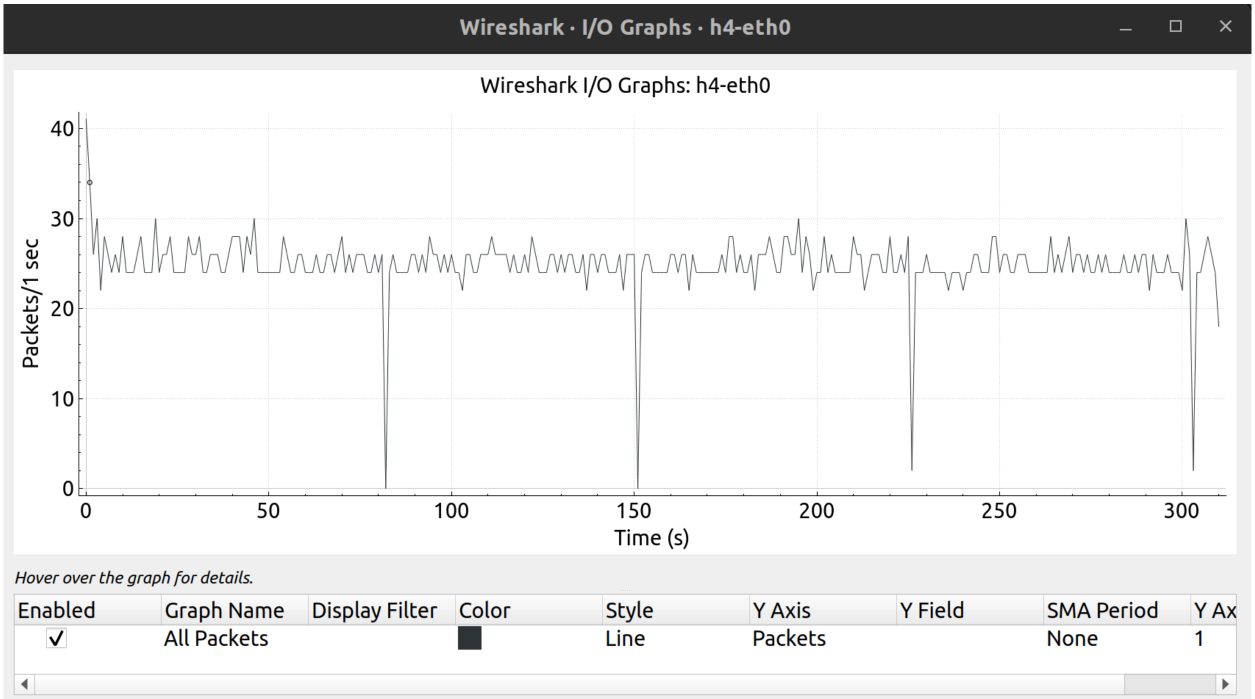
**H1 (Client Side)**




<u>Packets/sec sent vs Time plot for client side</u>

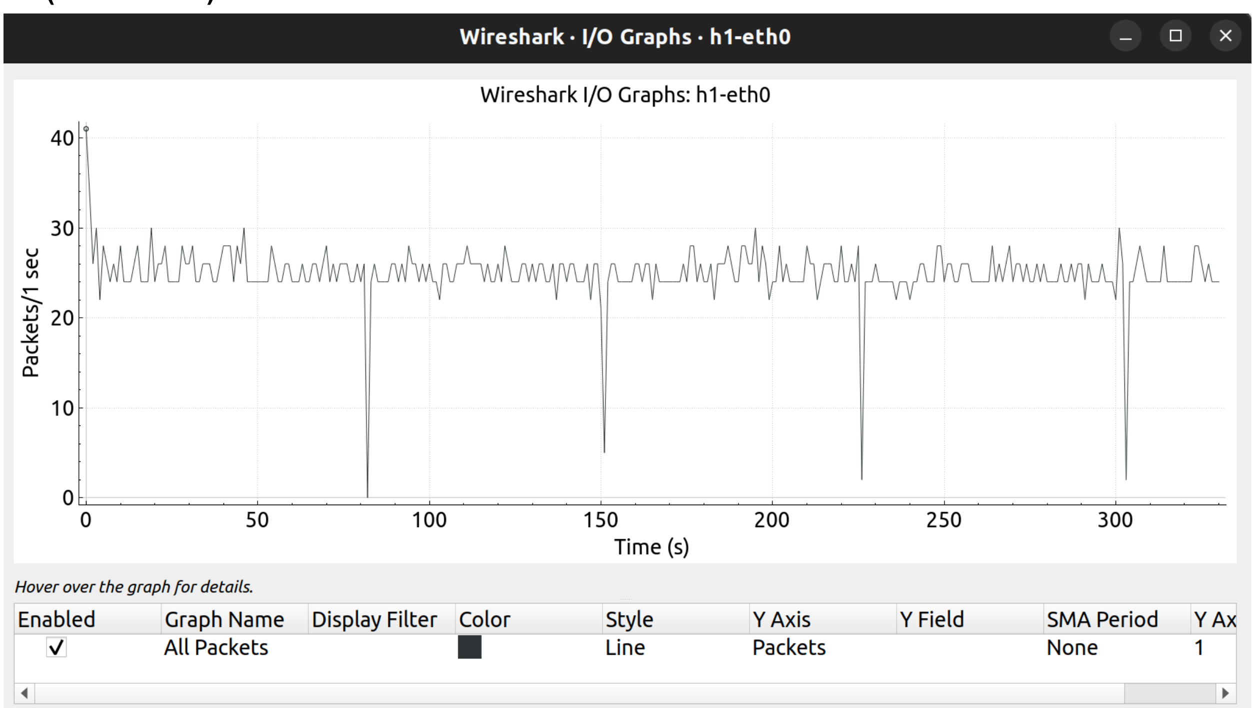
**<u>Congestion=Reno, loss=3%</u>**

**H4 (Server Side)**




<u>Packets/sec received vs Time plot for server side</u>

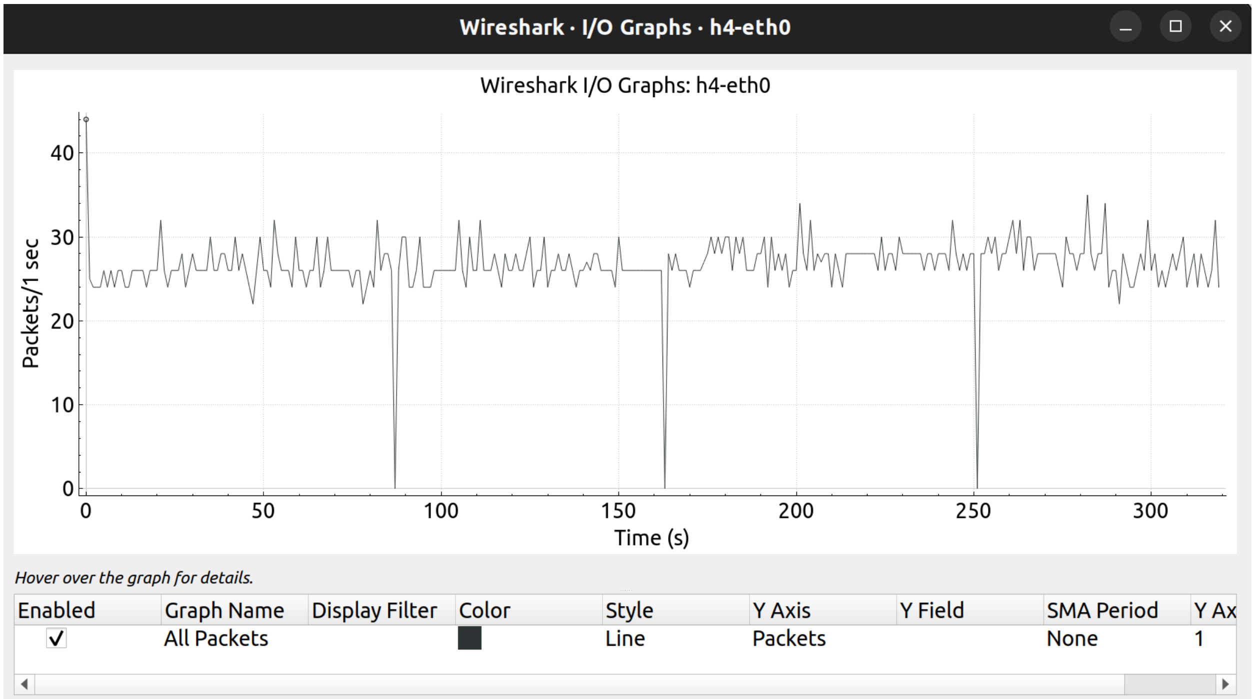
**H1 (Client Side)**




<u>Packets/sec sent vs Time plot for client side</u>

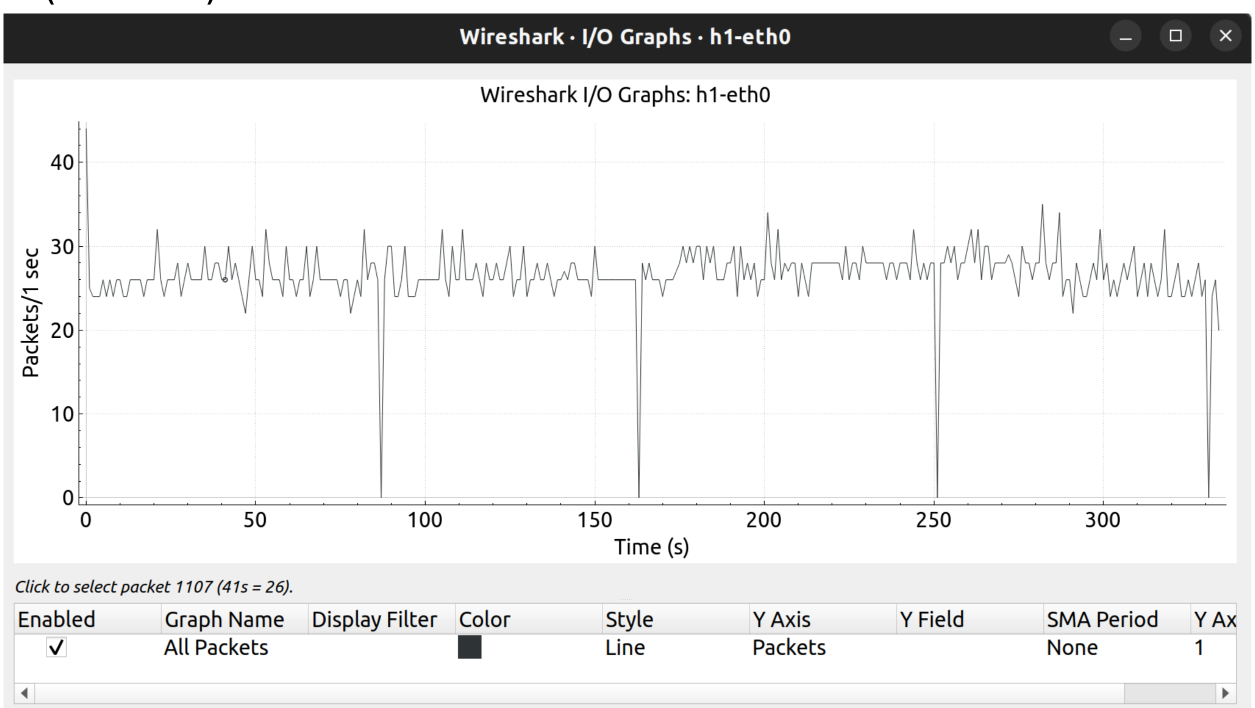
**<u>Congestion=Cubic, loss=3%</u>**

### H4 (Server Side)




Packets/sec received vs Time plot for server side

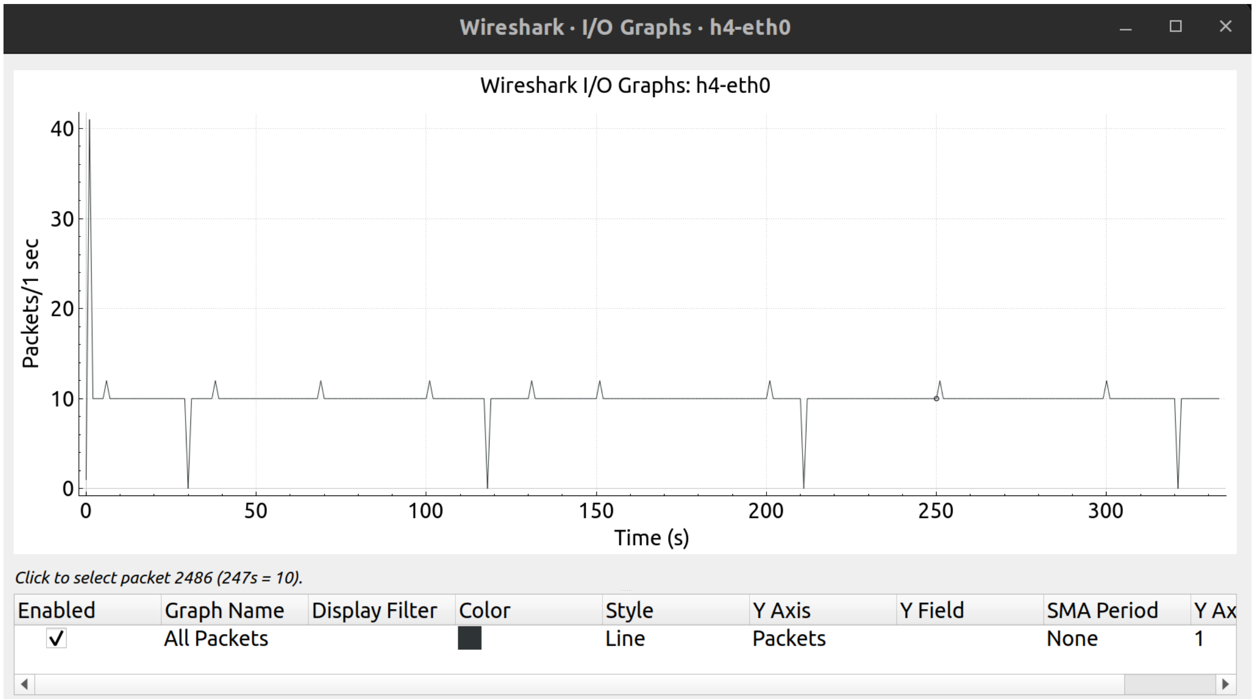
### H1 (Client Side)




Packets/sec sent vs Time plot for client side

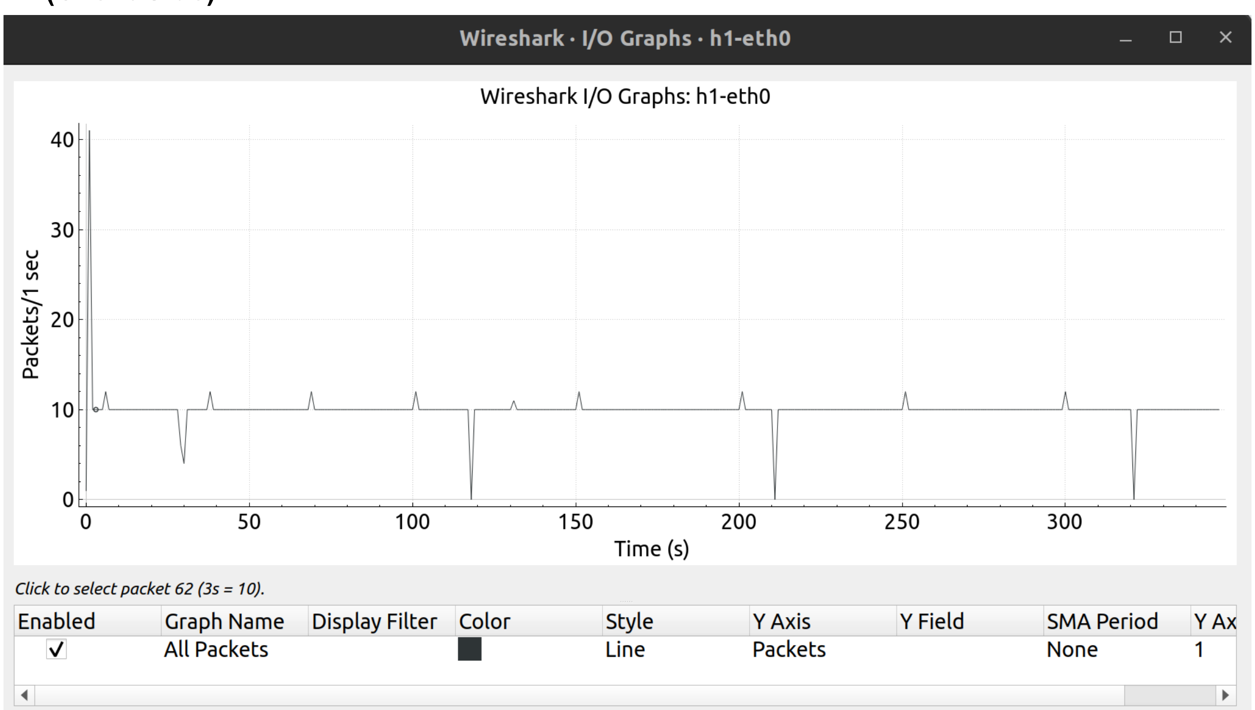
<u>**Congestion=BBR, loss=3%**</u>

### H4 (Server Side)




<u>Packets/sec received vs Time plot for server side</u>

### H1 (Client Side)




<u>Packets/sec sent vs Time plot for client side</u>