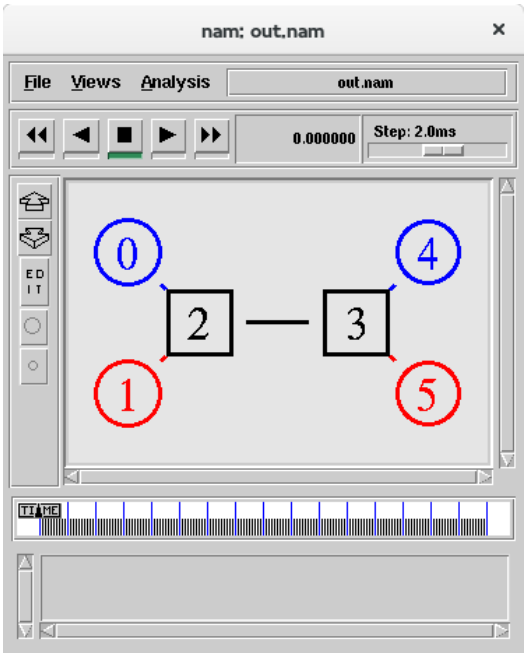
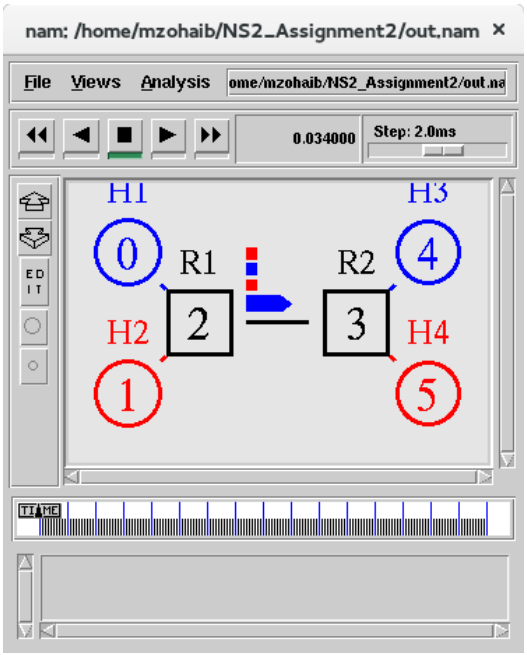
# Setup:

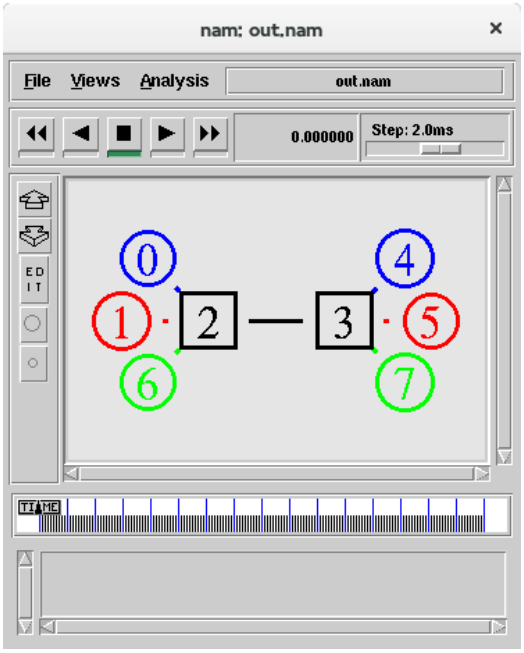
## For Scenario 1

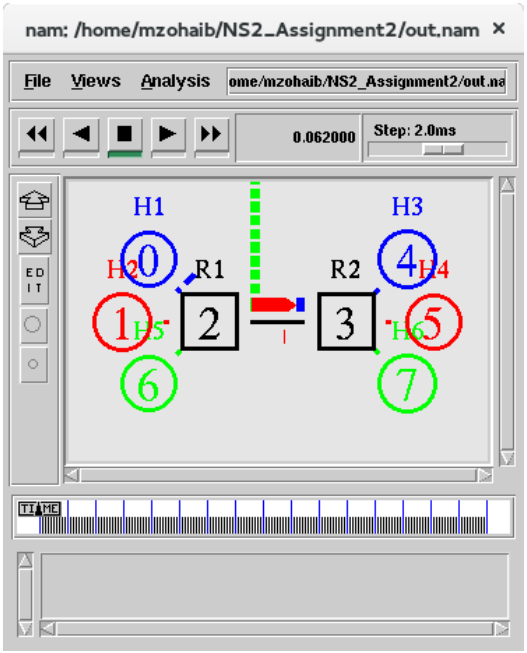



There are two logical TCP flows: One from H1 to H3 and other from H2 to H4. Both sources are atatched with FTP sources using default parameters. All the queues are droptail.

## For Scenario 2




There are two logical TCP flows: One from H1 to H3 and other from H2 to H4. There is another UDP flow from H5 to H6. Sources H1 and H2 are atatched with FTP sources. H5 is constant bit rate source. FIFO queue with Random Early Detection is implemented at R1. The rest of queues are droptail.

.

# Procedure:

First 30 seconds of simulation are ignored so that transients may die down and we reach steady state. Results are recorded starting from t=30 to 180 sec. All the sources start sending packets to their respective sinks at t=30 sec and stop at 180 sec.

### Calculating Average Throughput:

Every 1 second starting from t=30 sec, all the sinks calculate the number of bytes received and subtract number of bytes received until t=30 from this value to calculate average number of bytes received from t=30 to that time. Then these bytes are divided by time interval from 30 to this time to calculate average throughput until this time.

### Calculating Instantaneous Throughput:

For calculating instantaneous throughput, all the sinks record number of bytes received in each 1 sec interval starting from t=30 sec. These bytes are then divided by 1 sec to obtain instantaneous throughput.

# Results:

## For Droptail

| Scenario | Average Throughput Sink1(kbps) | Average Throughput Sink2(kbps) | Average Throughput Sink3(kbps) |
|---|---|---|---|
| 1 | 500 | 500 | NA |
| 2 | 0 | 0 | 1000 |

## For RED

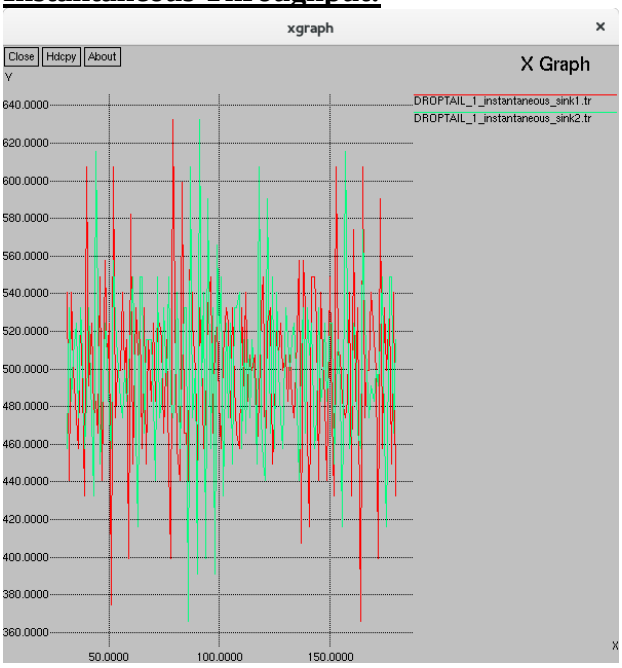
| Scenario | Average Throughput Sink1(kbps) | Average Throughput Sink2(kbps) | Average Throughput Sink3(kbps) |
|---|---|---|---|
| 1 | 479 | 518 | NA |
| 2 | 22 | 26 | 950 |

# Throughput vs time graphs:

## Droptail scenario 1:

### Average Throughput:


### Instantaneous Throughput:

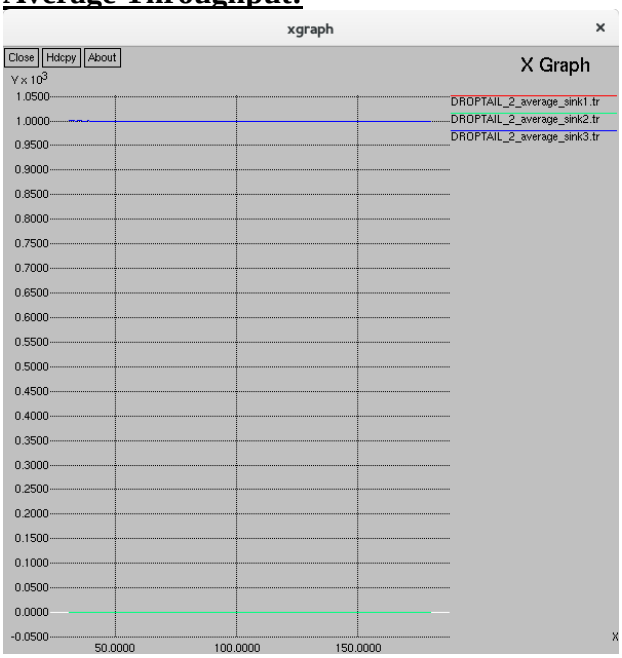



In Droptail scenario 1, both the sources are identical with identical link delays. Therefore, average throughput for both the sources is same and is equal to half the capacity of central 1Mbps link.

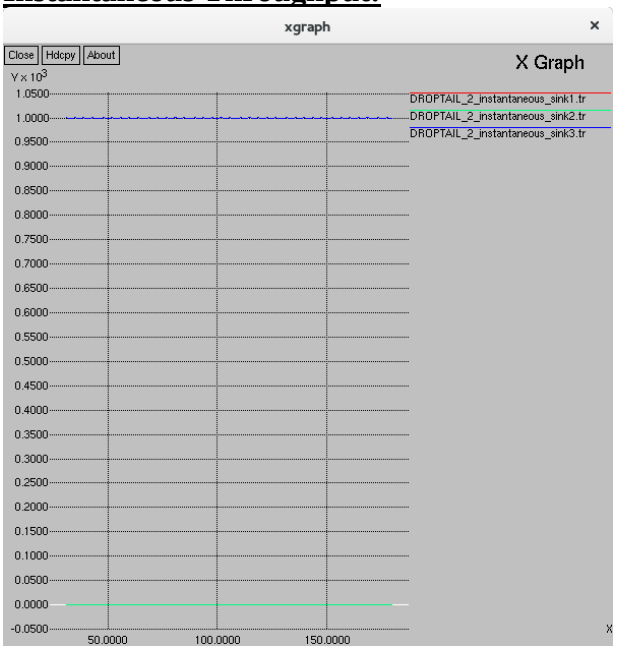
From instantaneous graphs we see that throughput at the sinks varies but the sum remains equal to 1Mbps.

# Droptail scenario 2:

## Average Throughput:




## Instantaneous Throughput:




In Droptail scenario 2, both the TCP sources are identical with identical link delays. But UDP source is constantly sending at data rate equal to the capacity of the lowest capacity link i.e. 1 Mbps. We have ignored initial 30 sec results where the TCP sources try to send data but back off (decrease their window size) due to packet drop due to buffer overflow. In the long run,

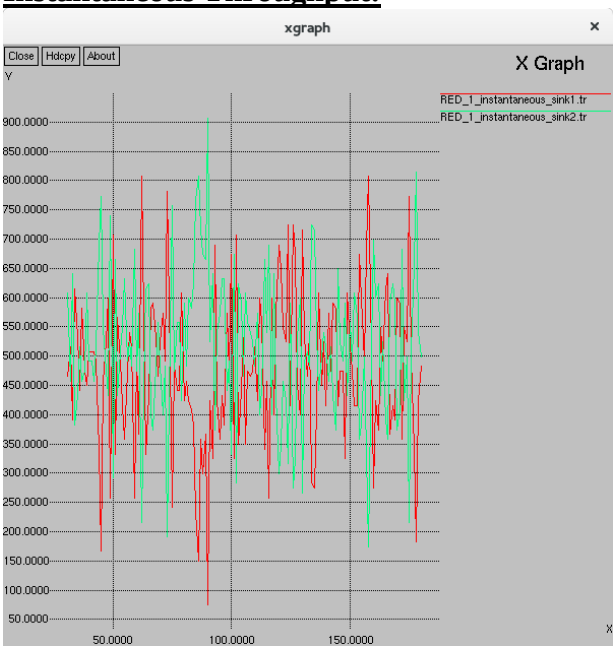
persistent unresponsive UDP source keeps on sending at maximum data rate. Therefore, average throughput for both the TCP sources is zero. UDP is getting all the 1Mbps bandwidth.

From instantaneous graphs we see that throughput at the TCP sinks is zero because transient period has finished where sources would send data and will timeout and resend. Here only UDP source is sending and gets instantaneous throughput of 1Mbps at all instants.

## Red scenario 1:

### Average Throughput:

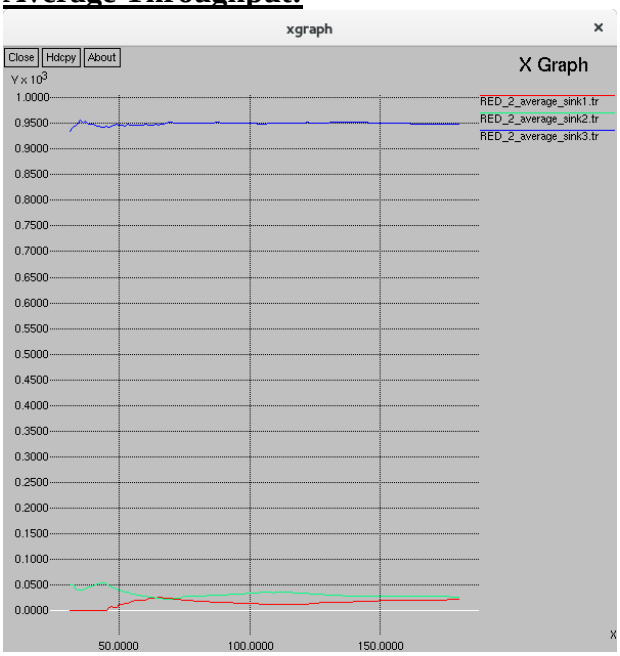

### Instantaneous Throughput:



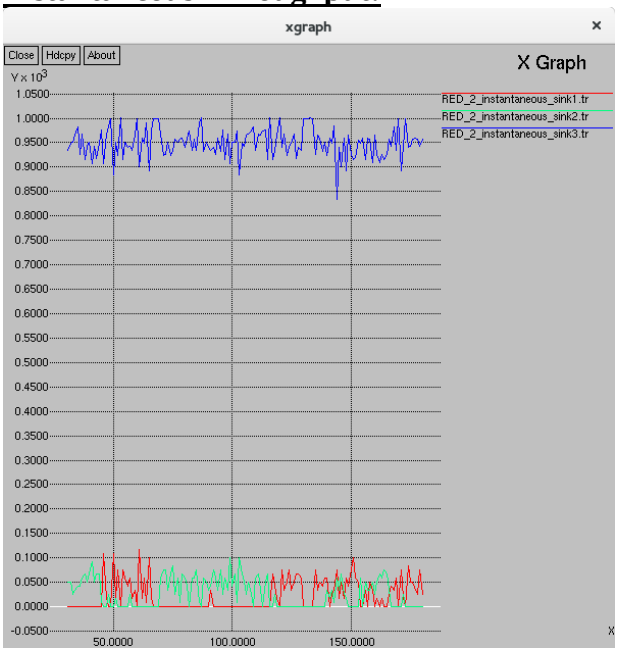
The results are almost similar to scenario 1, because both the sources are identical. The principle of RED is that it drops packets before actual buffer overflows take place. In our simulations, buffer size is 20 packets, but minimum threshold for RED is 10 packets and max threshold is 15 packets. RED keeps track of running average of number of packets in queue .If average number of packets remains below 10 (in our case), the packet is accepted in queue. If the average number of packets in queue is between 10 and 15 (in our case), the packet is dropped with probability as a function of 1/linterm (1/50 = 0.02), maximum and minimum threshold and current value of average queue size. If the average size is above 15, the packet is definitely dropped. TCP sources would go into fast retransmit and fast recovery mode, when their packets are dropped and would slow down in response to congestion. But we would get almost similar result as droptail scenario because both sources are identical with identical links.

## Red scenario 2:

### Average Throughput:



**Instantaneous Throughput:**




In this case, UDP source would not decrease its sending rate in response to dropped packets unlike TCP sources. Using RED would slightly increase throughput for TCP sources because some of there would be space available for TCP packets in buffer because of definite dropping of UDP packets when average queue size exceeds 15. These sources would get throughput of 22 and 26 kbps in contrast to 0 in droptail case. Throughput of UDP source would slightly decrease to 950 from 1000 kbps.

## Conclusion:

Using aggressive UDP source with gentle TCP sources results in very unfair allocation of resources. UDP data rate is equal to capacity of slowest link in the network, so it will receive its desired capacity. UDP source does not decrease its data rate in response to both droptail policy of packet drop or earlier packet drop as in RED. We are ignoring the initial 30 sec results where TCP sources would get some share of bandwidth. But in steady state they have been completely left out.