SIMULATION AND ANALYSIS OF TCP CONGESTION CONTROL VARIANTS

Introduction:

- ***Congestion*** is a situation in Communication Networks in which too many packets are present in a network and it subsequently leads to performance degradation.
- ***Congestion Control*** refers to techniques that can either prevent congestion, before it happens, or remove congestion after it has happened. Congestion control modulates traffic entry into a network in order to avoid congestive collapse resulting from oversubscription.

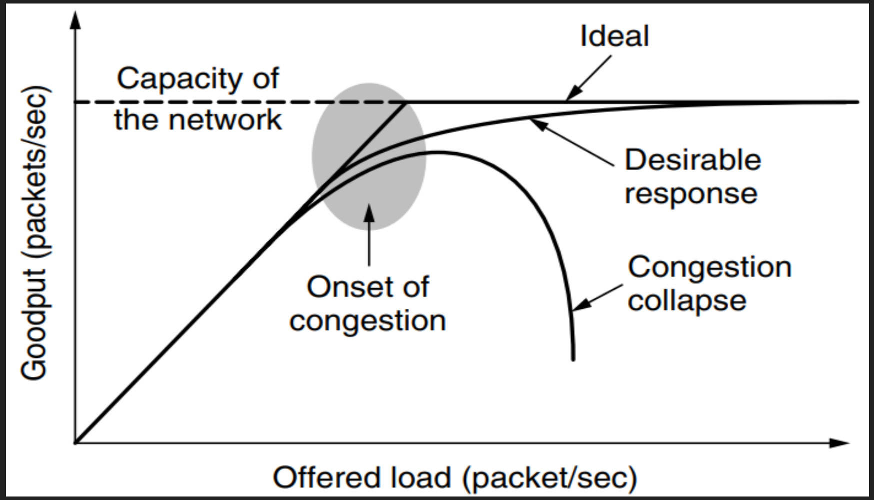


Fig. 1: Performance Drop due to Congestion

- ***Transmission Control Protocol (TCP)*** is a standard that defines how to establish and maintain a network communication through which application programs can exchange data. TCP works with the Internet Protocol (IP), which defines how computers send packets of data to each other. It is a connection-oriented communications protocol that is used for exchanging data between two end devices in a network. It is very powerful in dealing with congestion control and retransmission.

TCP Congestion Control:

TCP Congestion Control techniques prevent congestion or help mitigate the congestion after it occurs. The basis of TCP congestion includes:

- ***ADDITIVE INCREASE MULTIPLICATIVE DECREASE (AIMD)***
- ***RETRANSMISSION TIMER***
- ***SLOW-START MECHANISM***
- ***ACK-CLOCK***

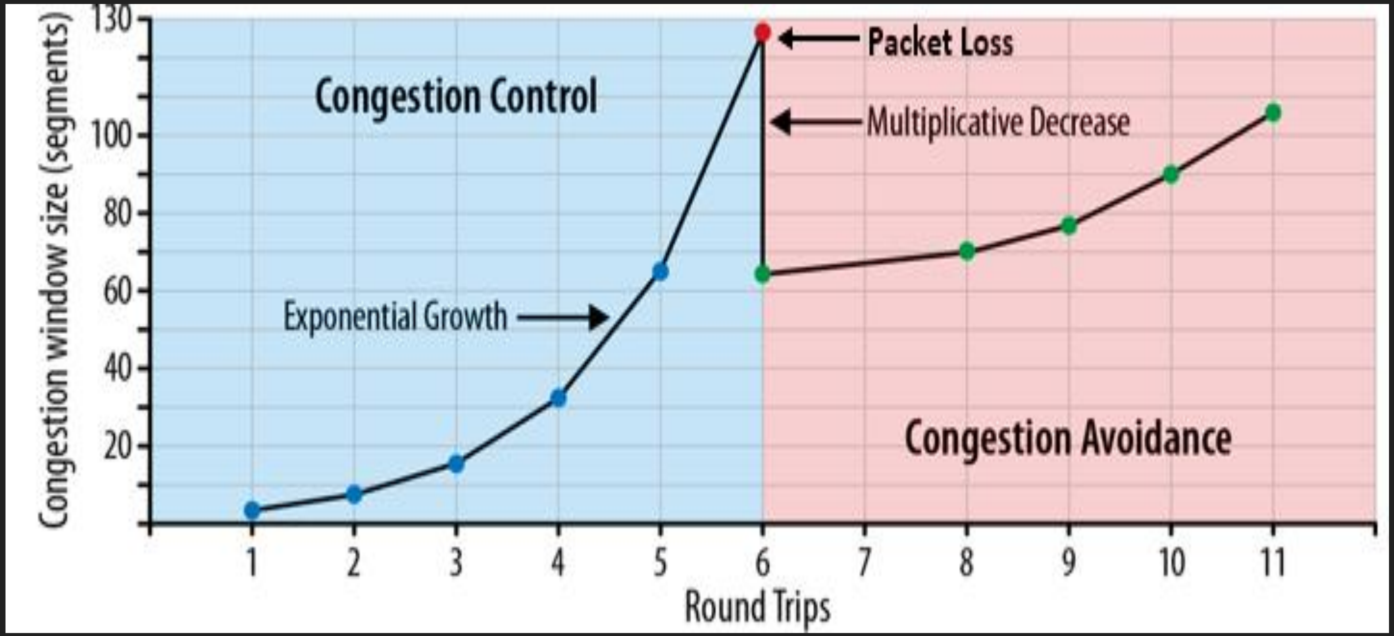


Fig. 2: TCP Congestion Control

TCP Variants:

Some well-known variants of TCP Congestion Control are:

- ***TCP Tahoe***: It adds a number of new algorithms and refinements to earlier TCP implementations. The new algorithms include Slow-Start, Congestion Avoidance, and Fast Retransmit.
- ***TCP Reno***: It retains the enhancements incorporated into Tahoe TCP but modified the Fast Retransmit operation to include Fast Recovery. It prevents the communication channel from going empty after Fast Retransmit.
- ***TCP New Reno***: It is a slight modification over TCP-Reno. It is able to detect multiple packet losses and thus is much more efficient that Reno in the event of multiple packet losses.
- ***TCP Vegas***: It adopts a more sophisticated bandwidth estimation scheme. It uses the difference between expected and actual flow rates to estimate the available bandwidth in the network.
- ***TCP SACK***: TCP with Selective Acknowledgment (SACK) preserves the properties of TCP Tahoe and Reno and uses retransmit timeouts as a recovery method.

Additional Concepts:

- ***THROUGHPUT***
- ***PACKET DROP RATE***
- ***LATENCY***
- ***TCP FAIRNESS***
- ***DROPTAIL***
- ***RANDOM EARLY DETECTION (RED)***

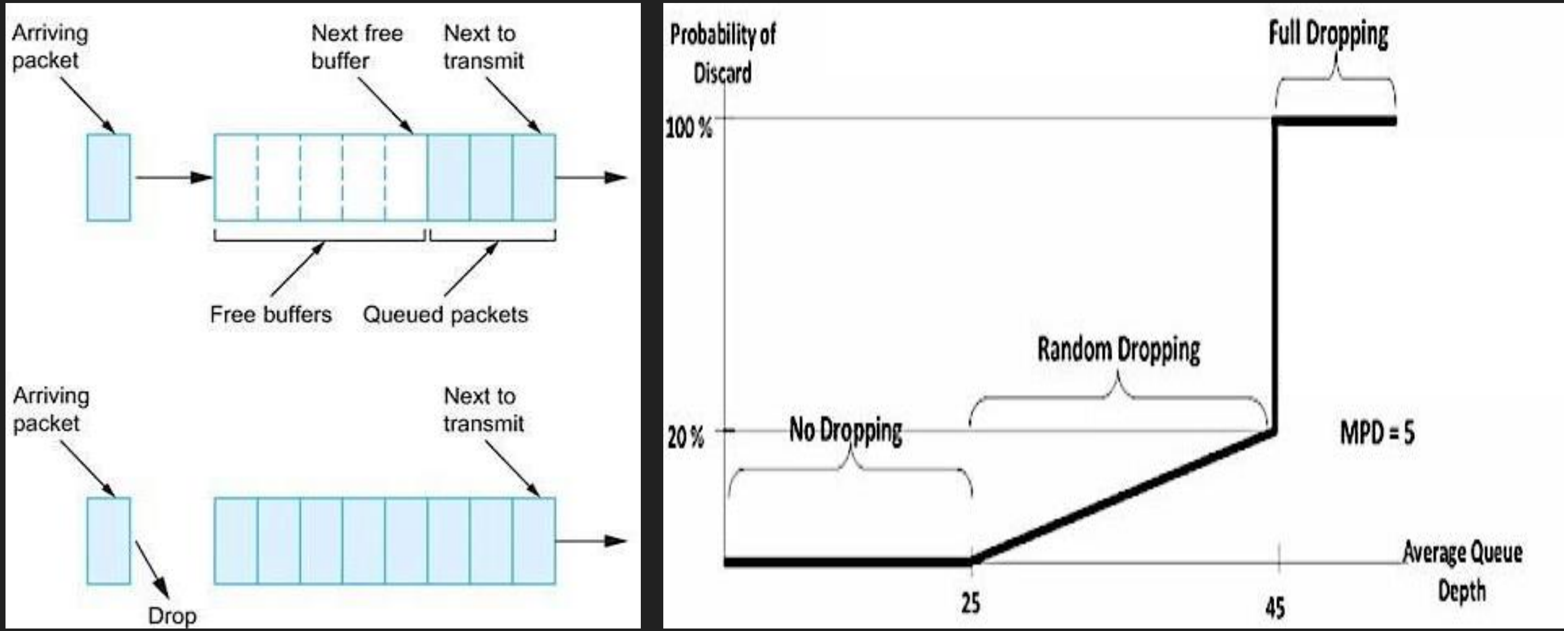


Fig. 3: DropTail and Random Early Detection (RED)

Problem Statement:

The objective of this project is to provide a comprehensive analysis of TCP Congestion Control techniques which includes:

- ***Simulate TCP Congestion in a Network*** using Cisco Packet Tracer by setting up an appropriate topology involving client and server.
- ***Analyze and compare the performance of TCP variants*** by means of performance metrics such as Throughput, Packet Drop Rate and Latency under different load conditions by linearly increasing Constant Bit Rate (CBR) from 1 Mbps to 10 Mbps between two nodes.
- ***Analyze and compare the Fairness of TCP variants*** among each other in a network by means of Throughput under different load conditions by linearly increasing Constant Bit Rate (CBR) from 1 Mbps to 10 Mbps between two nodes. Pair of TCP variants analyzed are Reno-Reno, New Reno-Reno, Vegas-Vegas and New Reno-Vegas.
- ***Analyze and compare the influence of the Queuing Disciplines*** namely: DropTail and Random Early Drop (RED) to TCP Reno with and without Selective Acknowledgement (SACK) by means of Average Bandwidth and Average Latency.

Simulation of TCP Congestion:

In this section, we have simulated TCP Congestion using ***Cisco Packet Tracer.*** The Network Topology consists of ***6 Client*** devices ***and 1 Server***. The Server is connected to ***Central Router*** through FastEthernet (100 Mbps) Port. Three Clients are connected with each ***Switch*** using FastEthernet Ports. The Switches are connected to Central Router through GigabitEthernet (1000 Mbps) Ports. The IP Addresses of all the devices are listed alongside them. The Default Gateways are indicated along the connections.

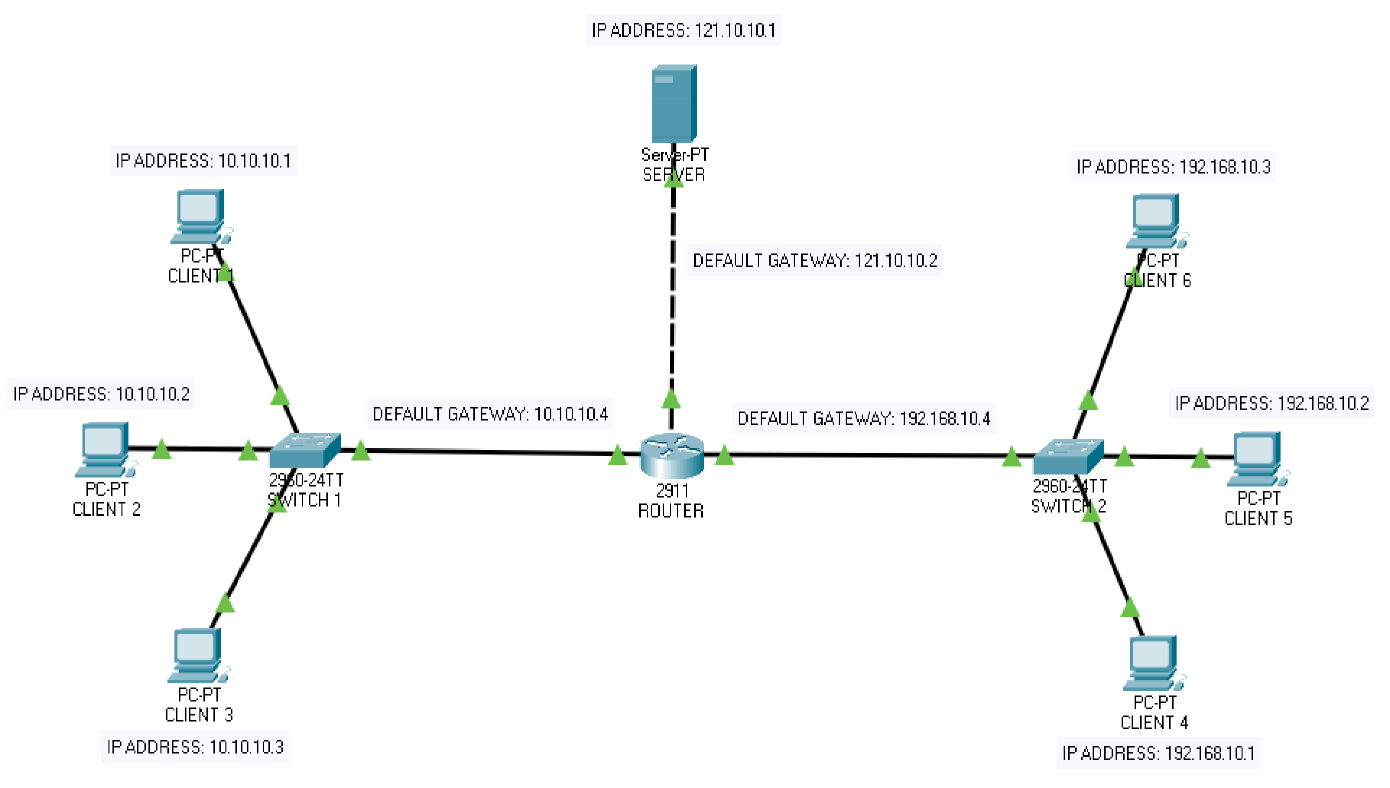


Fig. 4: Network Topology for TCP Congestion Simulation

- After setting up the network, ***HTTP traffic*** is generated from each Client to the Server.
- The Packets are queued at Switches before leaving for Central Router.
- A large amount of HTTP Traffic leads to ***Packet drops at Central Router as well as Server.***
- Packet transmission time from Switches to Router is very less due to use of ***Gigabit connections***. This leads to large ***Packet Queuing at Central Router*** and thus floods the Server with Packets leading to Packet Loss.
- The ***Dropped Packet*** belongs to Client 1 with destination being the server having Destination Port 80.

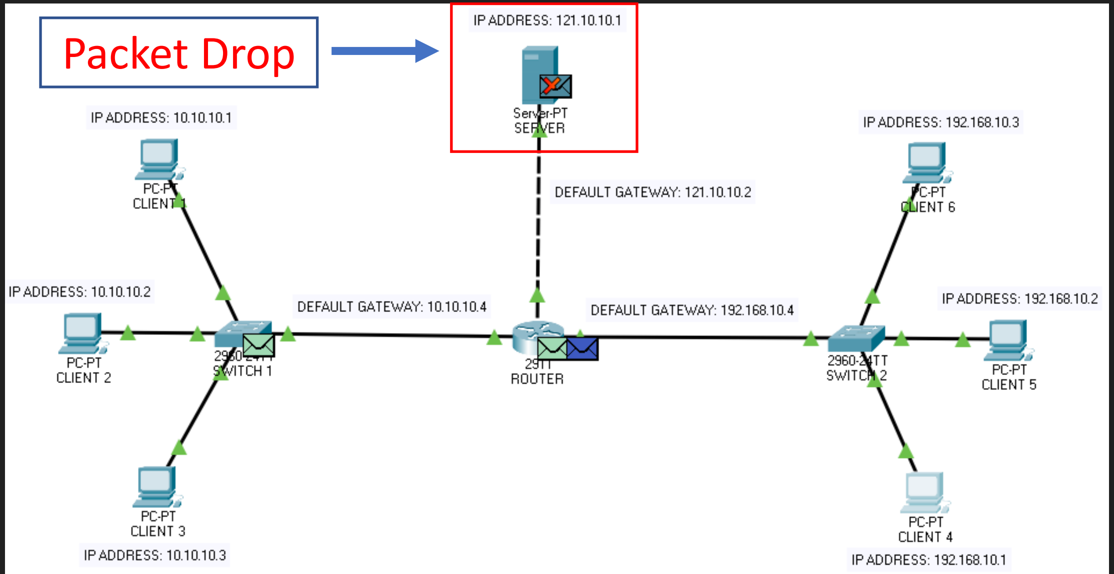


Fig. 5: Packet Drop at Server due to Congestion

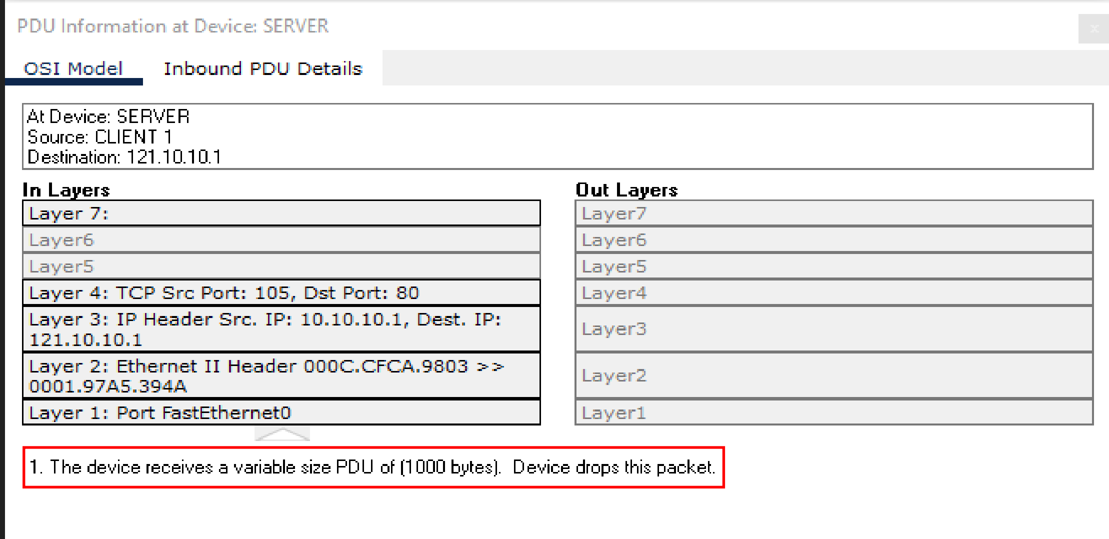
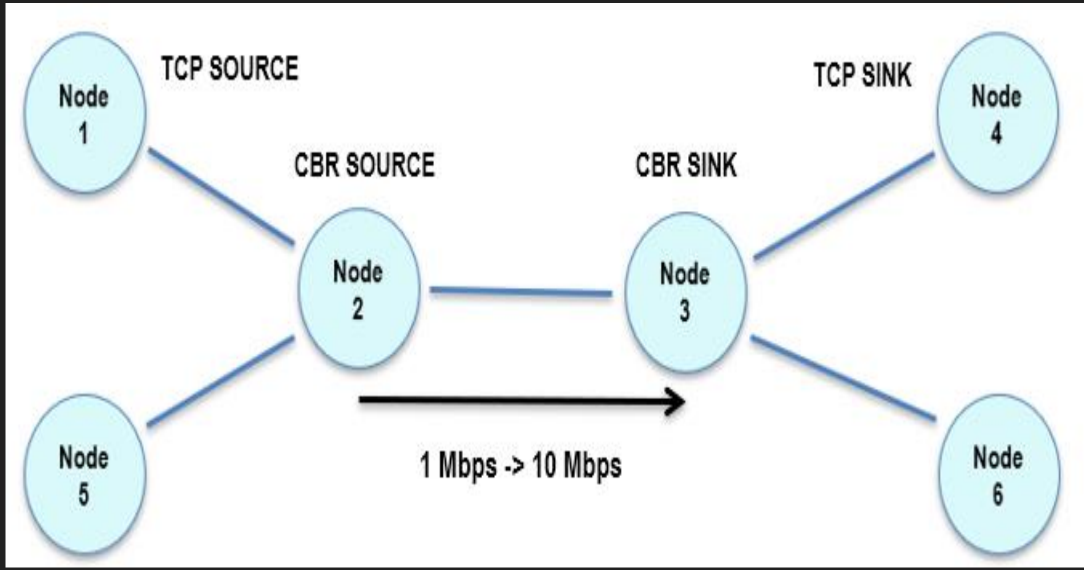


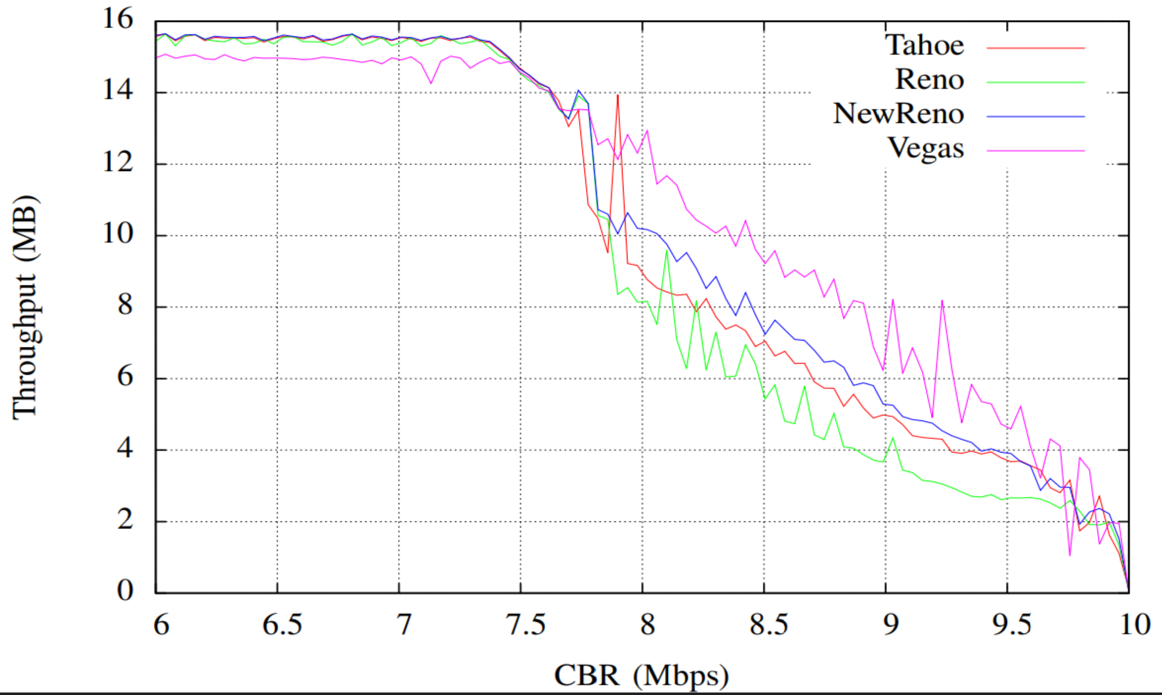
Fig. 6: Information of the Dropped Packet at Server

Performance Comparison of TCP Variants:

In this section, we have analyzed and compared the performance of TCP variants by means of ***Throughput, Packet Drop Rate*** and ***Latency*** under different load conditions by linearly increasing ***Constant Bit Rate (CBR)*** from 1 Mbps to 10 Mbps. Figure below shows the ***Network Topology*** and ***Flow Setup*** used. The CBR flow from Node 2 to Node 3 is the only varying quantity.

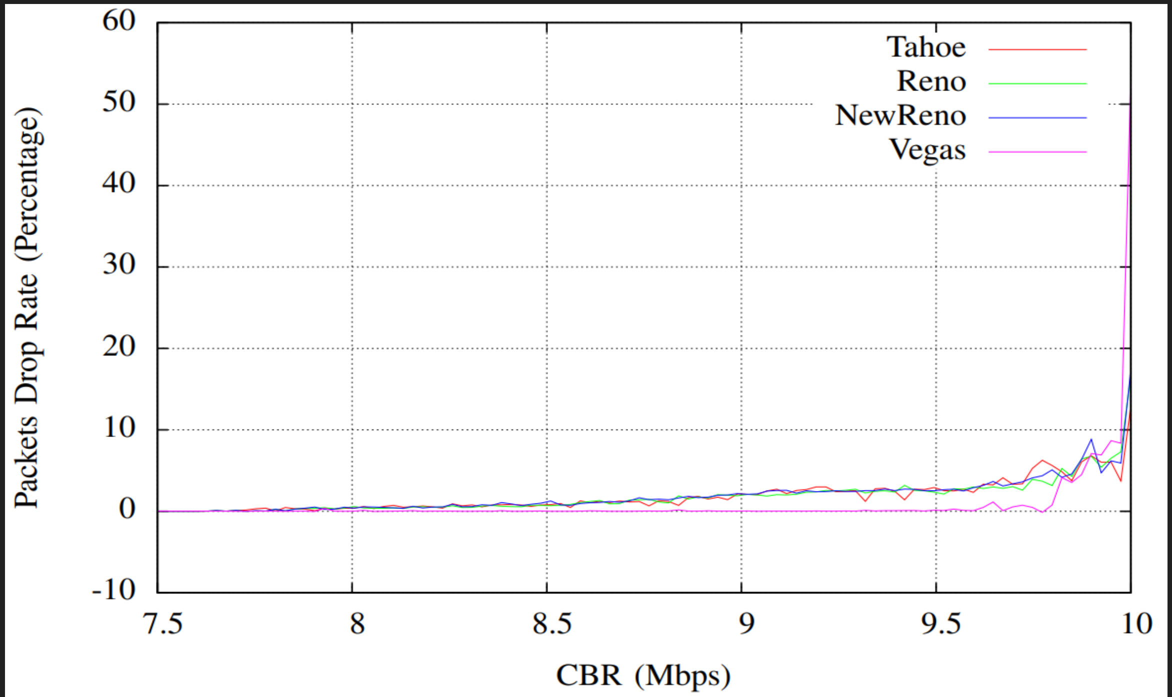


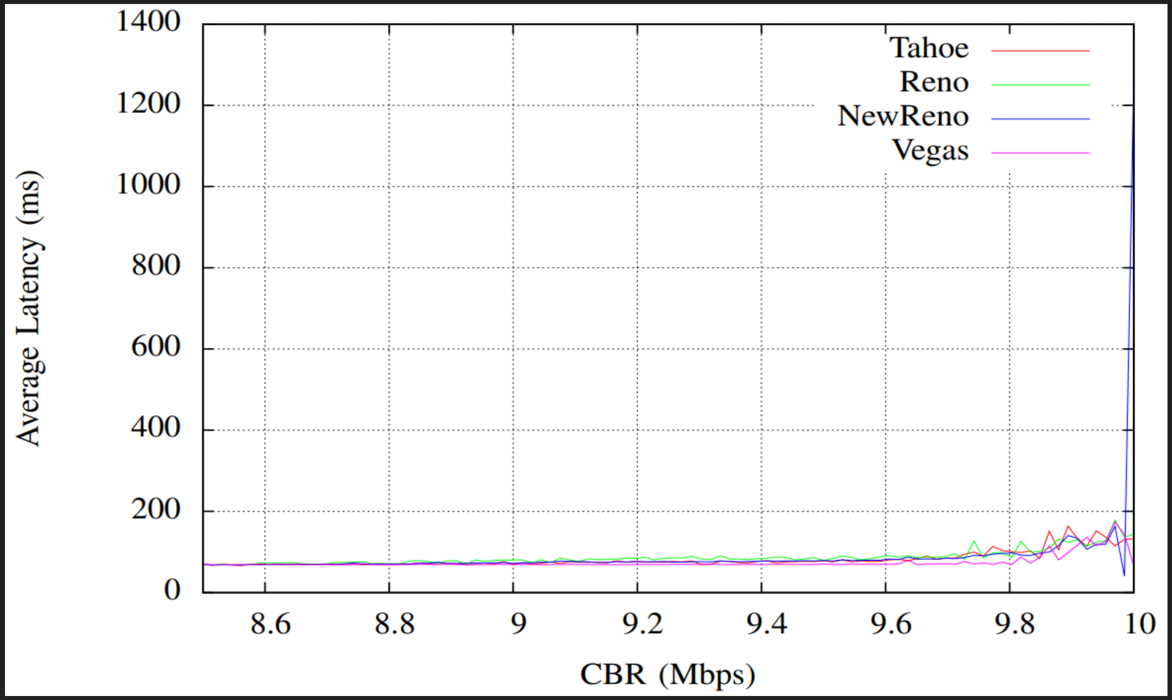
THROUGHPUT: When CBR between Node 2 and 3 is less than 7.5Mbps, the throughput for all TCP variants is almost similar. However, when the CBR flow increases and the network start to congest, TCP Vegas performs the best and have the highest average throughput while TCP Reno has the lowest throughput.



PACKET DROP RATE: Initially, when the CBR is less than 7.5Mbps, the packet drop rate is very less. When CBR flow gets dominant but has not occupied all the bandwidth in the network, TCP Vegas performs better by dropping less packets while the other three has similar higher drop rate. TCP Vegas runs steadier than others with the increase of CBR. When CBR increases to the bandwidth limitation, Vegas has a 50% drop rate compared with the other three.

LATENCY: All TCP variants exhibit the same latency till CBR is less than 8.5 Mbps, but it varies afterwards. TCP Vegas detects congestion at the initial stage, and it queues the packets when the RTT received is greater than the base RTT. Hence, Vegas has the lowest average latency. The other three variants have comparatively higher latency than TCP Vegas. Only TCP New Reno has a latency blast when CBR reaches to the bandwidth limitation.

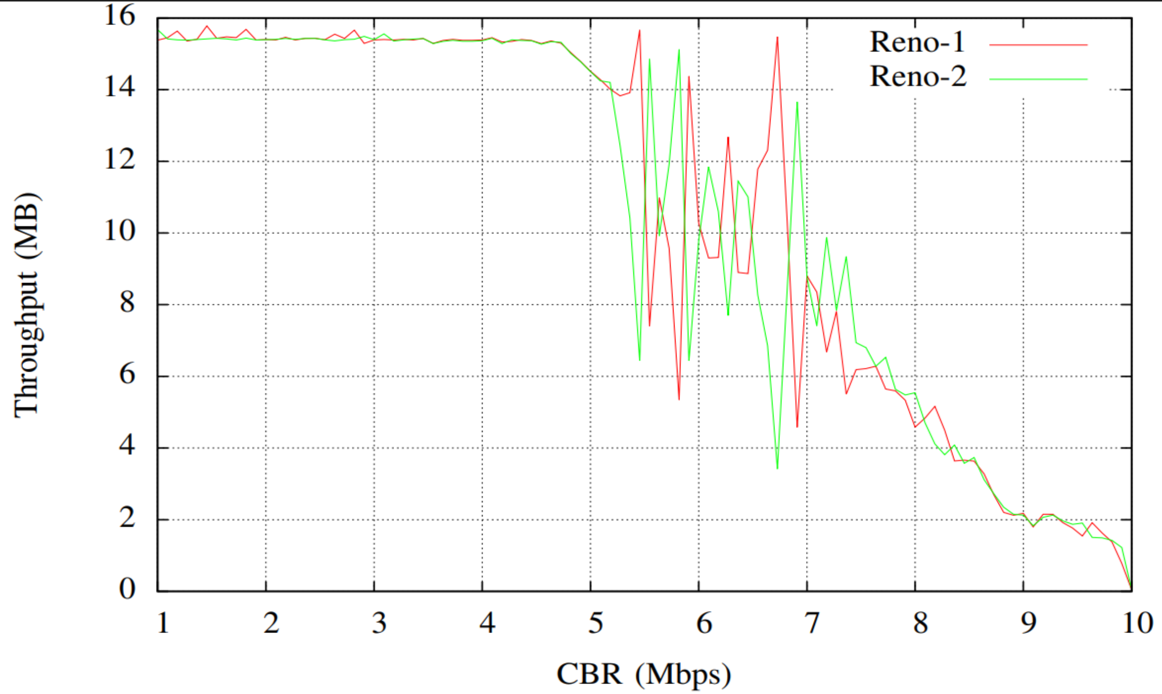




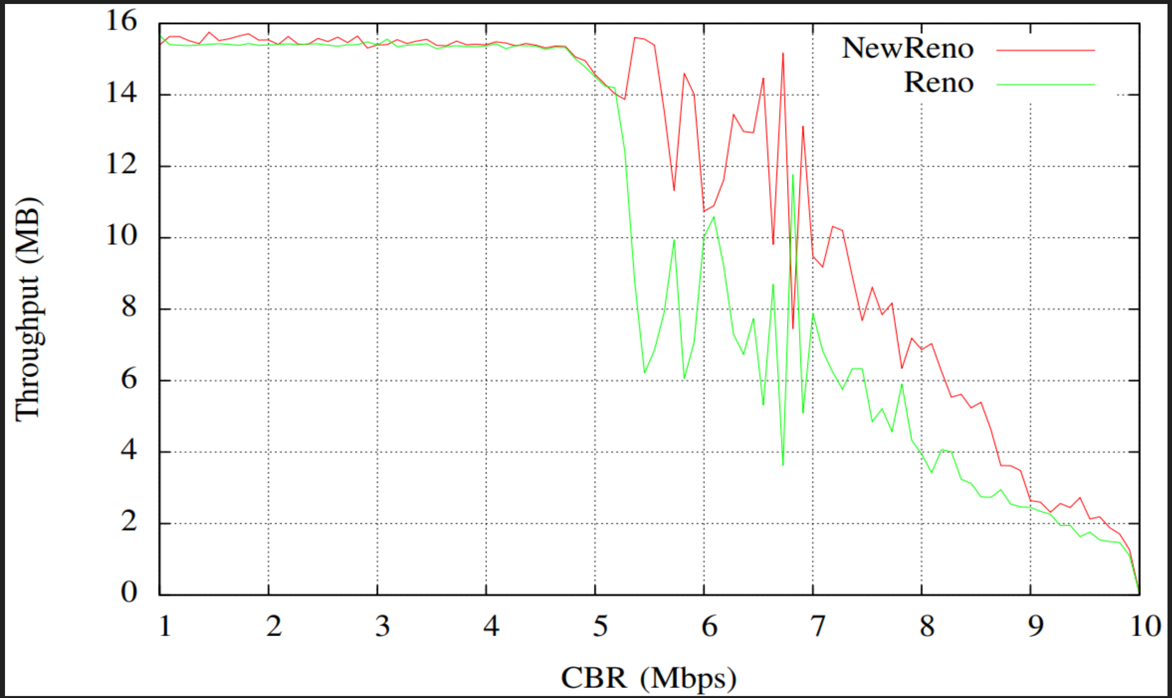
Fairness Between TCP Variants:

In this section, we have analyzed and compared the ***Fairness of TCP variants*** among each other by linearly increasing Constant Bit Rate (CBR). One TCP variant is used along Node 1 to Node 4 and other is used along Node 5 to Node 6.

RENO-RENO: There are certain oscillations in the throughput, the two Reno lines follow similar tracks and keep close with the increase of CBR. Hence, Reno-Reno TCP pair is fair.

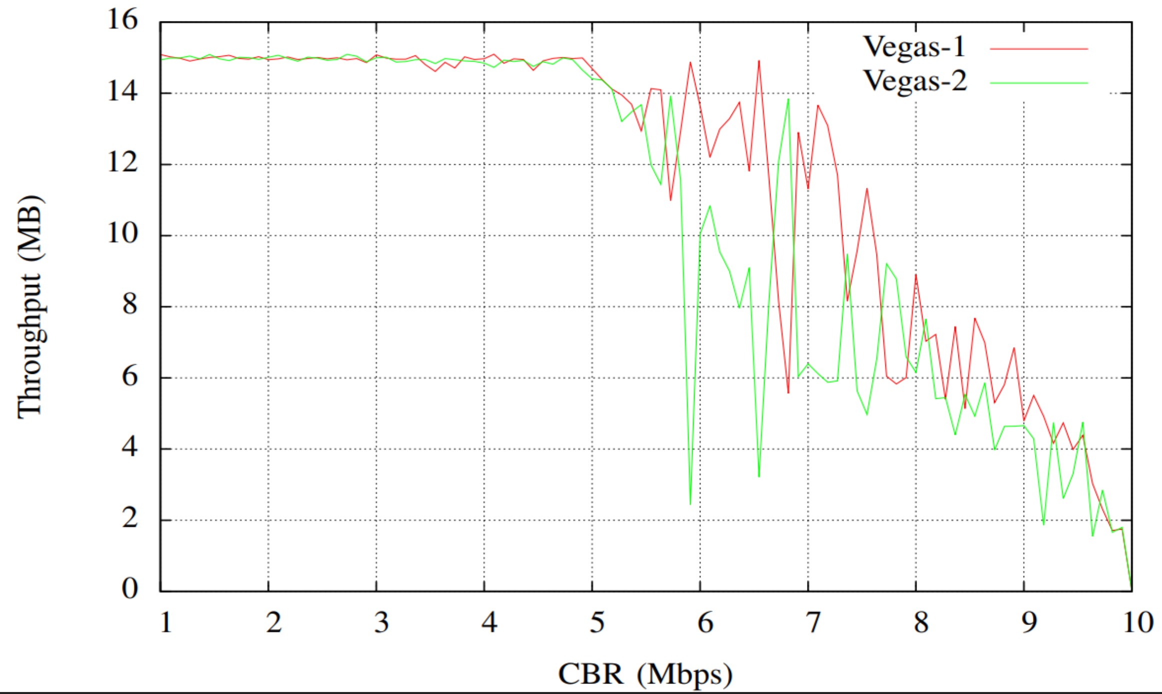


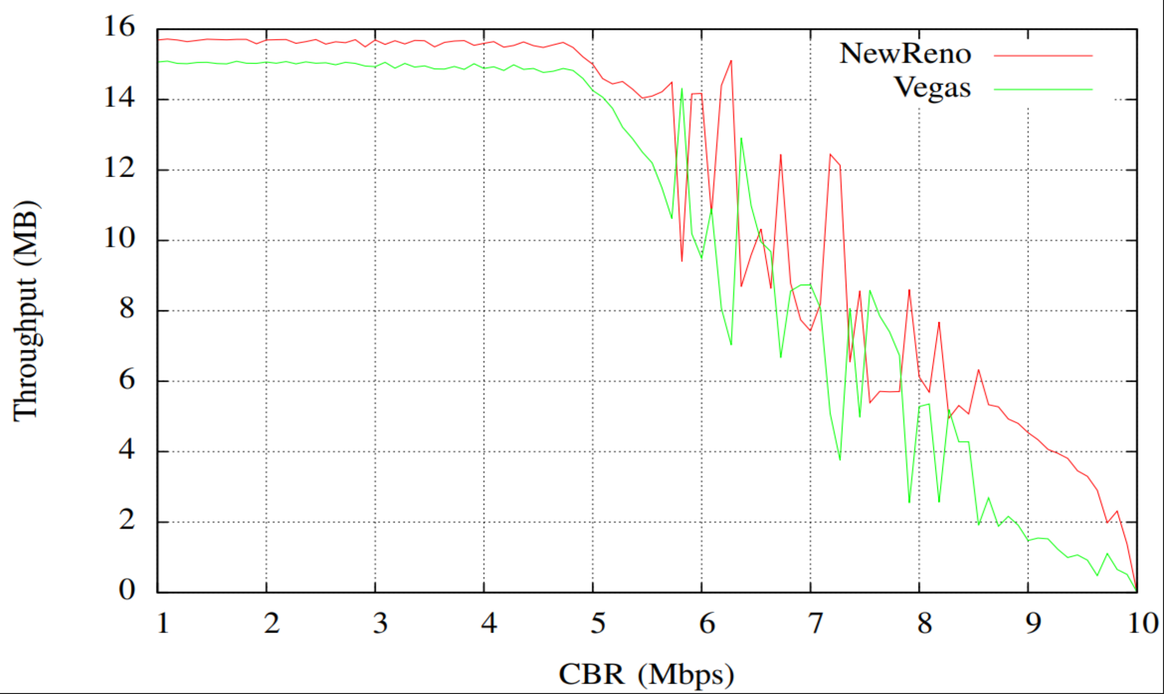
NEW RENO-RENO: TCP Reno has lower throughput than New Reno when CBR gets larger than 5Mbps. So New Reno is not that fair to Reno TCP, but it does not suppress Reno too much.



VEGAS-VEGAS: There are certain oscillations in the throughput, the two Vegas lines follow similar tracks and keep close with the increase of CBR. The oscillation is acceptable, since only one TCP can have higher bandwidth. Hence, Vegas-Vegas TCP pair is fair.

NEW RENO-VEGAS: With the increase of CBR, New Reno has higher throughput than TCP Vegas. The gap between the two variant grows larger with increase in CBR. TCP New Reno runs steadier and loses fewer packets than Vegas. Thus, New Reno is unfair to Vegas.



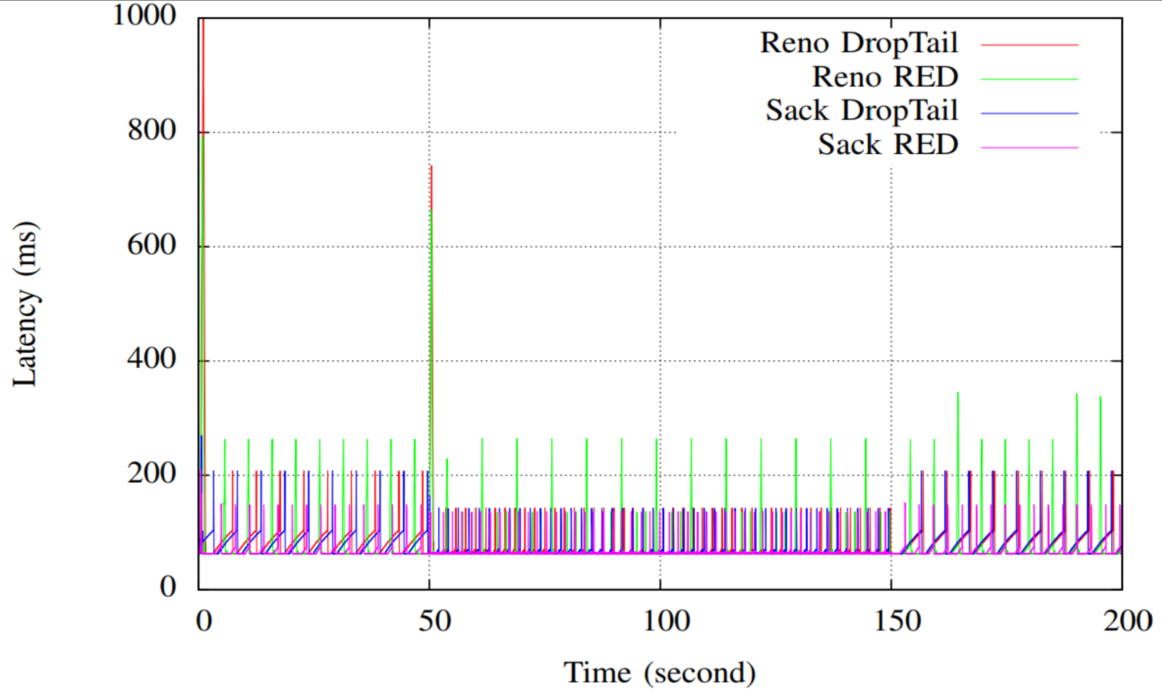


The analysis concludes that same TCP variant is usually fair to each other on throughput, while different variants cannot be completely fair to the co-existing variants. Hence, Reno-Reno and Vegas-Vegas are fair to each other in same network, but on the other hand, New Reno-Reno and New Reno-Vegas are not fair to each other.

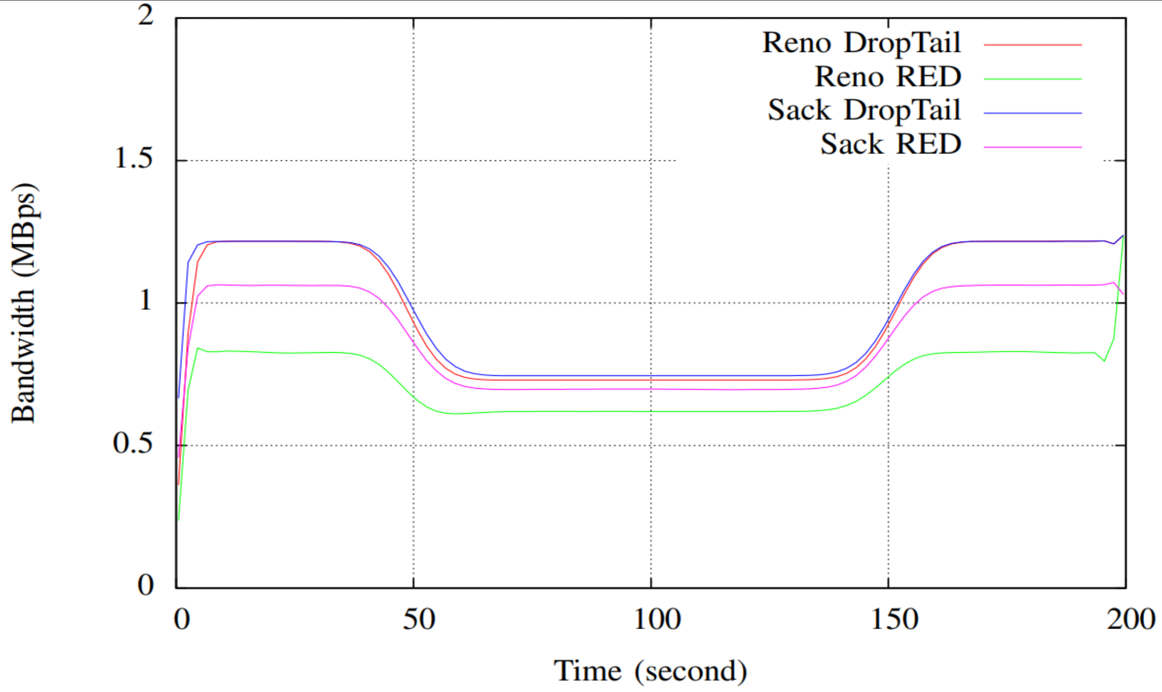
Influence of Queuing Algorithms:

In this section, we analyzed the ***influence of the queuing disciplines*** namely: ***DropTail*** and ***Random Early Drop (RED)*** on TCP Reno with and without ***Selective Acknowledgement (SACK)*** through Average Latency and Average Bandwidth.

AVERAGE LATENCY: TCP Reno has a smaller latency with DropTail than with RED. TCP SACK has larger latency with DropTail than with RED.



AVERAGE BANDWIDTH: Both TCP variants has higher bandwidth with DropTail than RED. Reno with RED queue type has the worst performance in bandwidth.



Conclusion And Learning:

CONCLUSION

In this project, we have:

- ***Simulated TCP Congestion*** in a Network using ***Cisco Packet Tracer Simulator***
- ***Analyzed and Compared Performance of TCP variants.***
- ***Analyzed Fairness between TCP variants***
- ***Analyzed the influence of the queuing disciplines on TCP Reno with and without Selective Acknowledgement (SACK).***

LEARNING

In this project, we studied and simulated different TCP variants for TCP Congestion Control. By understanding these algorithms, we came to know how these variants could be used to solve real life network congestion problems. This project helped us to have a better ***understanding of TCP Congestion Control*** and become familiar with different types of variants available to deal with various types of network congestion situations. We performed ***TCP Congestion Simulations*** using ***Cisco Packet Tracer Simulator***. During the course of this project, we used ***Network Simulator*** for analytical purposes.

THANK YOU