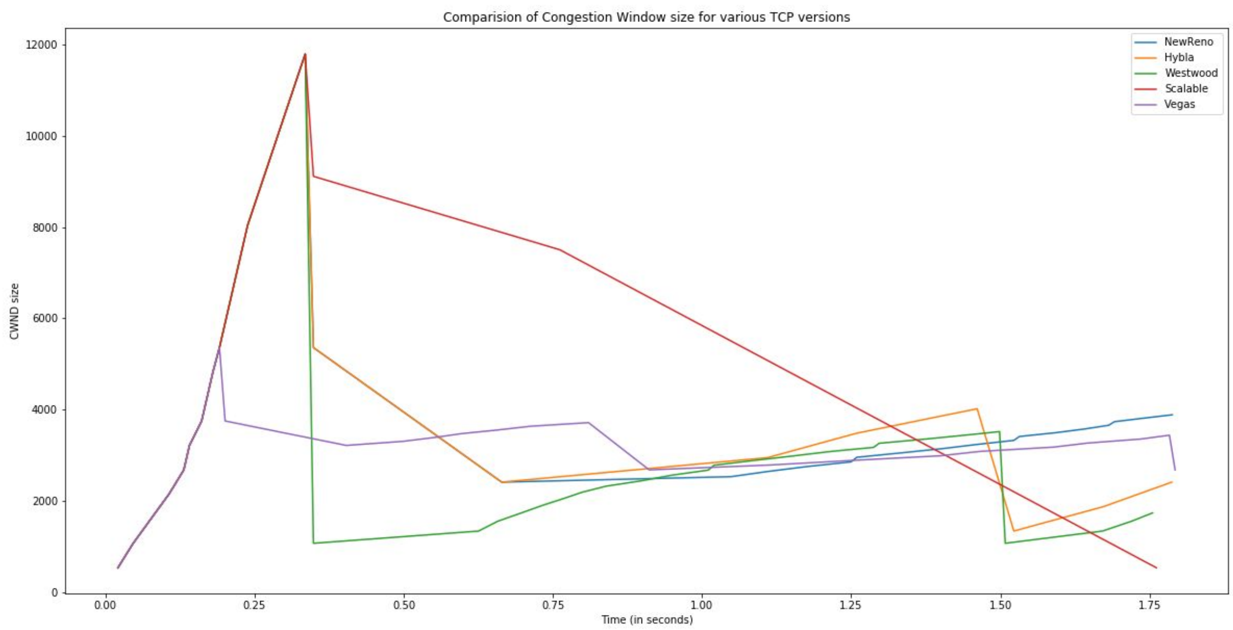
CS349 : Assignment - 4
Network Simulation Using NS-3
Group : 09

Akul Agarwal 160101085 | Deepak Kumar Gouda 160123054 | Yash Kothari 160123044

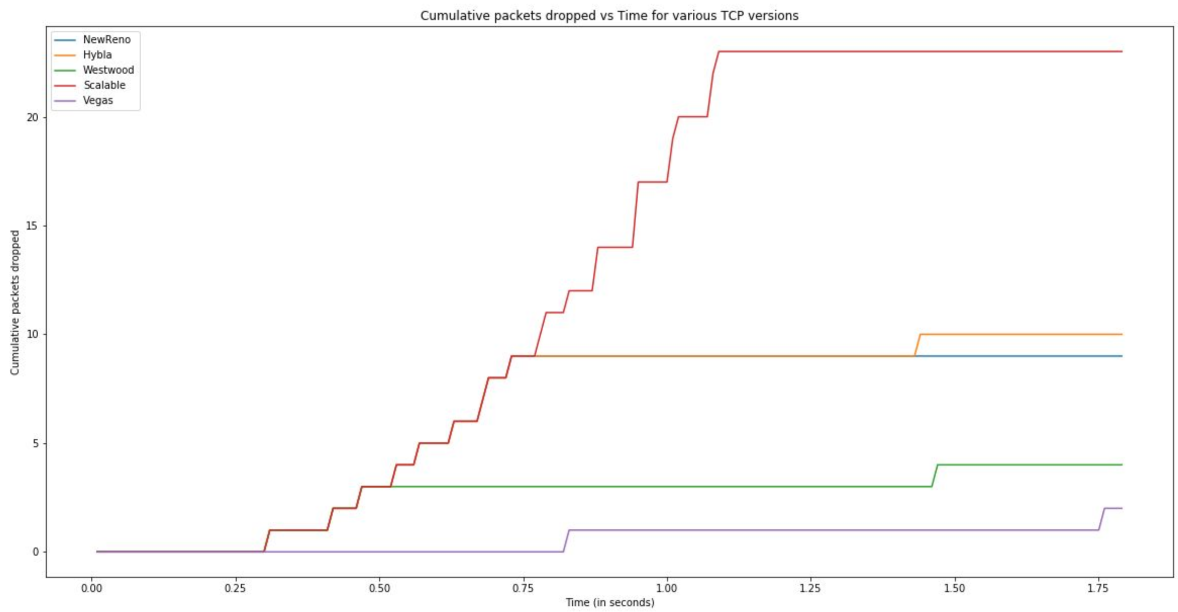
Graphs with explanation

1. **TCP Congestion window plot:**



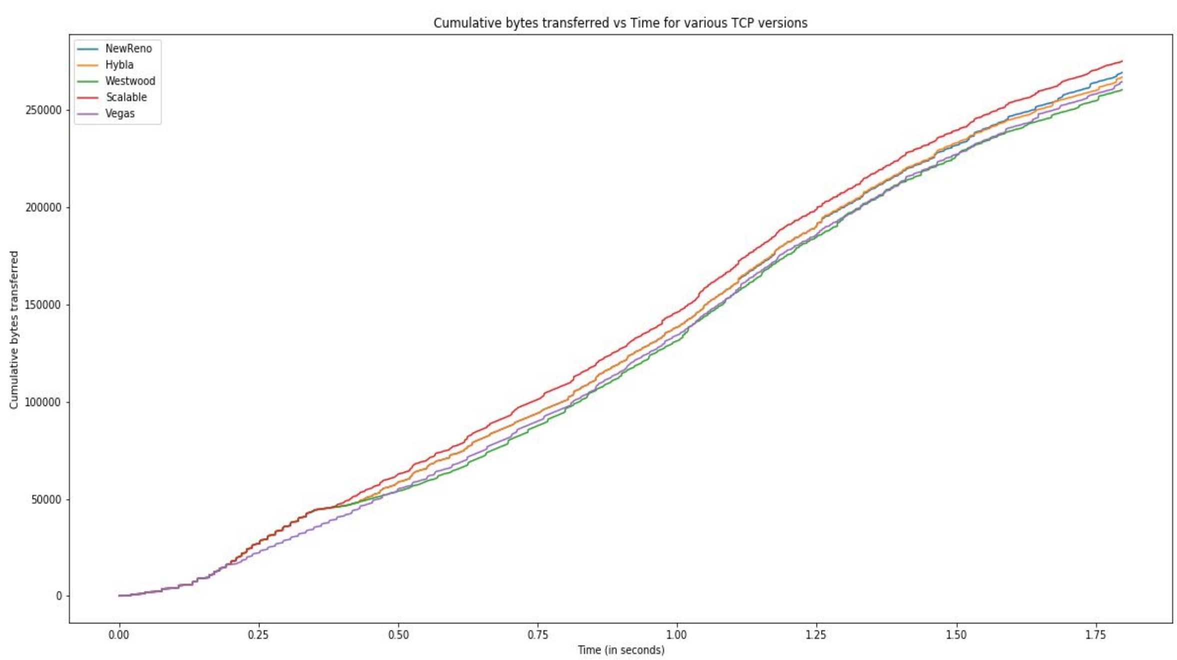
It can be observed that the first packet drop which occurs at *0.31 seconds* causes the change in congestion window of the TCP versions. While *New Reno* and *Hybla* drop to **half** the congestion window, *Westwood* starts from **1** and *Scalable* drops by **1/8**th only. Due to the larger size congestion window of *Scalable*, it also faces higher packet drops as indicated in the later plots.

2. Cumulative packets dropped vs Time



It can be observed that the packet drops of TCP *Scalable* see rapid increase between **1s** and **1.2s** as all UDP connections are active in this time frame. Other TCP versions do not indicate large packet drops as their congestion window is smaller than TCP Scalable.

3. Cumulative bytes transferred vs Time



Important questions we thought needed answering

1) Why are we using MyApp for TCP and not CBR Traffic ?

The congestion window is obtained by tracing **CongestionWindow** of a socket. While using **BulkSendHelper**, socket is created at run time and the actual packet transmission doesn't start at *0 seconds* precisely. If we wish to get congestion window, we can either schedule a function to start tracing at some time in future because socket creation takes some time after starting of application or we could use **MyApp[Source]** which explicitly creates a socket and then creates an application. In latter case we can start tracing from beginning of simulation itself.

In UDP, we don't have any requirement to trace a socket, hence we can directly use OnOffHelper.

2) What does DropPackets measures ?

FlowMonitor records stats of Layer 3 and above packets. FlowMonitor returns packets dropped classified by **ReasonCodes**. Following are possible ReasonCode

Enumerator	
DROP_NO_ROUTE	Packet dropped due to missing route to the destination.
DROP_TTL_EXPIRE	Packet dropped due to TTL decremented to zero during IPv4 forwarding.
DROP_BAD_CHECKSUM	Packet dropped due to invalid checksum in the IPv4 header.
DROP_QUEUE	Packet dropped due to queue overflow. Note: only works for NetDevices that provide a TxQueue attribute of type Queue with a Drop trace source. It currently works with Csma and PointToPoint devices, but not with WiFi or WiMax.
DROP_QUEUE_DISC	Packet dropped by the queue disc.

All the packet are dropped correspond to reasonCode 4(PROP_QUEUE_DISK) which is due to DropTailQueue.

3) What is our approach for the problem ?

- Create two nodes for our PointToPoint channel.
- Assign IP addresses to the nodes.
- Create an FTP application using MyApp because of the reasons already mentioned above.
- Create a sink application at node 1 to receive FTP data.
- Create OnOffHelper to configure CBR settings
- Create 5 UDP applications to simulate CBR Agents and send data at different times as per problem statement.
- Create a sink to receive packets from different CBR Agents
- Record data:
    1. Using congestion window trace on tcp socket
    2. Using Packets Dropped data from flowMonitor
    3. Using Tx trace

References

1) **Details of all the TCP models present in ns-3**
   https://www.nsnam.org/docs/models/html/tcp.html
2) **Introduction to all the common objects in ns-3**

   https://www.nsnam.org/docs/tutorial/singlehtml/index.html