Wireless Network Project
on
Study of TCP Variants

Shashank Rustagi[1] **Lasani Hussain**[2] **Mona Singh**[3]
Department of Computer Science and Engineering
IIIT-D

{ [1]shashank21081@iiitd.ac.in, [2]lasani21042@iiitd.ac.in, [3]mona21053@iiitd.ac.in }

1 Introduction

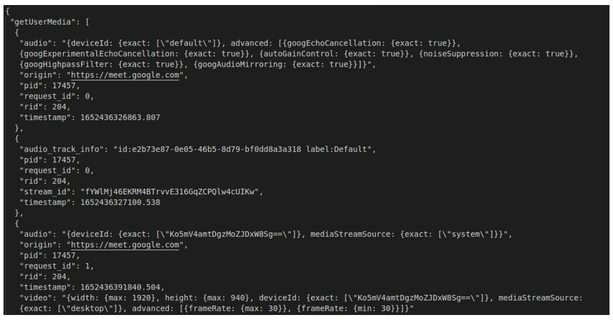
We analysed webrtc JSON dumps created over google meet with HD video presented by both participants, using python for different TCP variants in real time as well as we did simulation via NS3 and performed throughput vs payload metrics comparison for all the TCP variants like TCP Westwood, Vegas, Reno, Veno, Illinois etc.

2 Analysis of webrtc json dumps for different TCP variants

We analyzed the webrtc dumps created for google meet. WebRTC is an HTML5 specification that we can use to add real time media communications between browser and devices.

To get a detailed description of a video/audio session conducted over google meet, we need to open another tab and direct it to this “internal” URL: chrome://webrtc-internals/

webrtc-internals allows us to download the trace as a large JSON file that we can look at. The JSON looks like this :



```
{
 "getUserMedia": [
  {
   "audio": "{deviceId: {exact: [\"default\"]}, advanced: [{googEchoCancellation: {exact: true}},
   {googExperimentalEchoCancellation: {exact: true}}, {autoGainControl: {exact: true}}, {noiseSuppression: {exact: true}},
   {googHighpassFilter: {exact: true}}, {googAudioMirroring: {exact: true}}]}",
   "origin": "https://meet.google.com",
   "pid": 17457,
   "request_id": 0,
   "rid": 204,
   "timestamp": 1652436326863.807
  },
  {
   "audio_track_info": "id:e2b73e87-0e05-46b5-8d79-bf0dd8a3a318 label:Default",
   "pid": 17457,
   "request_id": 0,
   "rid": 204,
   "stream_id": "fYWlMj46EKRM4BTrvvE316GqZCPQlw4cUIKw",
   "timestamp": 1652436327100.538
  },
  {
   "audio": "{deviceId: {exact: [\"Ko5mV4amtDgzMoZJDxW8Sg==\"]}, mediaStreamSource: {exact: [\"system\"]}}",
   "origin": "https://meet.google.com",
   "pid": 17457,
   "request_id": 1,
   "rid": 204,
   "timestamp": 1652436391840.504,
   "video": "{width: {max: 1920}, height: {max: 940}, deviceId: {exact: [\"Ko5mV4amtDgzMoZJDxW8Sg==\"]}, mediaStreamSource:
   {exact: [\"desktop\"]}, advanced: [{frameRate: {max: 30}}, {frameRate: {min: 30}}]}"
```

Figure 1: WEBRTC DUMP

The json contains very detailed information about the connections, such as PeerConnections filed which is actually a nested dictionary . RTCPeerConnection API traces are a very powerful tool that contains information related to every second of the communication between peers such as audio level, packet loss, jitter, frame rates , type of encryption used during that time instant.We are interested in audio and video fields which detail the minute statistics. For the purpose of this study we have focussed exclusively on statistics related to video , and dependence/relfection of one parameter values to another, for example how packet width/height changes in response to jitter in TCP reno vs TCP illinois etc.

For conducting this study , we collected the webrtc dumps by having 2 participants in a google meet and both presenting their screens with both playing HD video , so as to generate saturated traffic and observe the behavior of packet loss/ frame width reduction etc in response to jitter.

We analyzed the json file using python code. Focus was on extracting meaningful data like audio and video metrics , and graphs were plotted for video metrics like frame rate, resolution, NACKs . We tried to figure out how in different TCP variants, the video quality changes with respect to jitter in the network and how slow/fast these parameters changed in response to jitter rates.

3 TCP VARIATIONS

3.1 TCP BBR(WestWood)

TCP Westwood (TCPW) is a sender-side modifica- tion of the TCP congestion window algorithm that improves upon the performance of TCP Reno in wired as well as wireless networks.

Observations : In spite of jitter and resulting drop in frame rates , the video frame width/height didn't suffer .

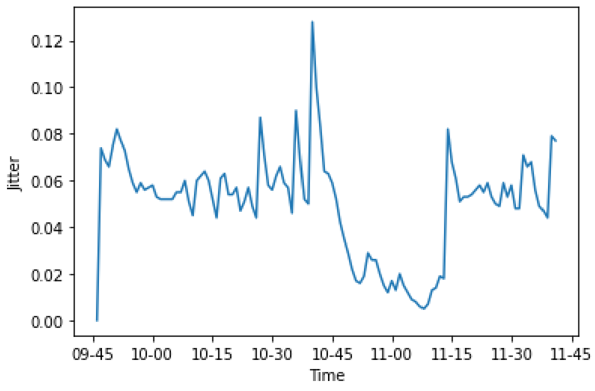


Figure 2: BBR Jitter

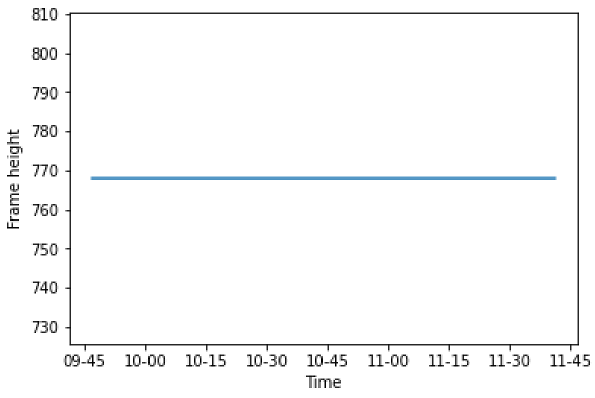


Figure 3: BBR Frame height

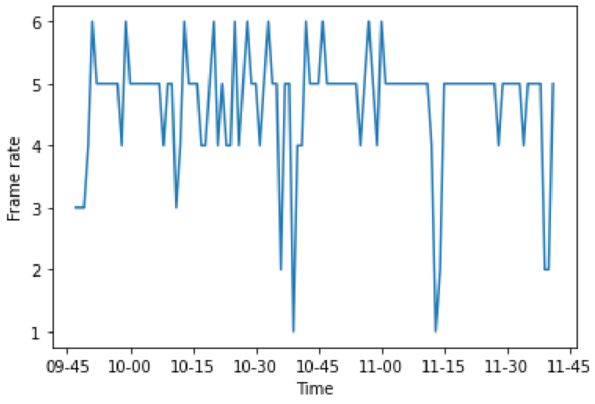


Figure 4: BBR Frame Rate

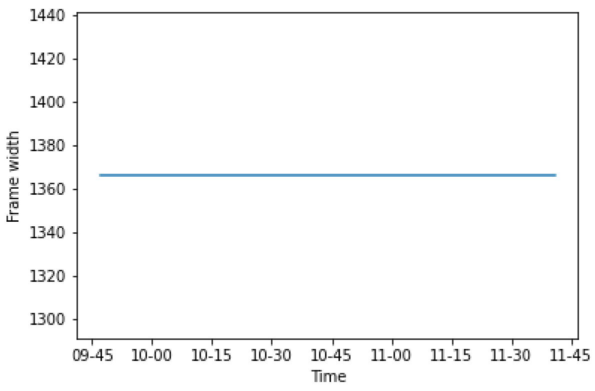


Figure 5: BBR Frame width

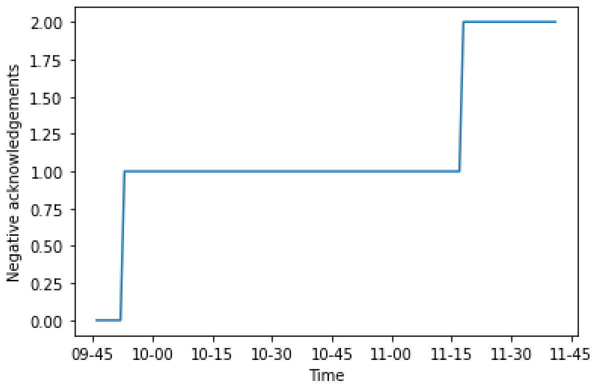


Figure 6: BBR NAK

3.2 TCP Illinois

TCP-Illinois achieves high throughput, allocates the network resource fairly, and is incentive com- patible with standard TCP.

Observations : There was a delay of 30 seconds between the jitter and video quality downgrading to lower resolution

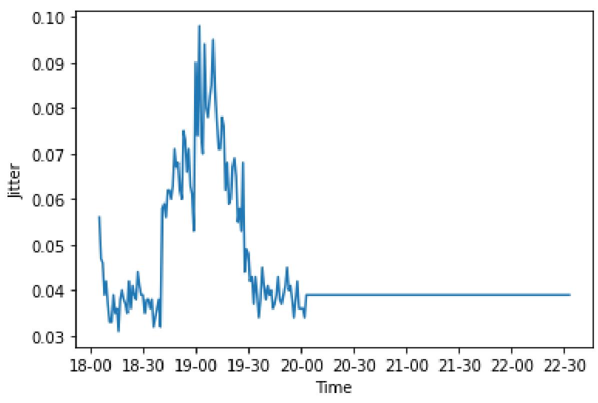


Figure 7: Illinois Jitter

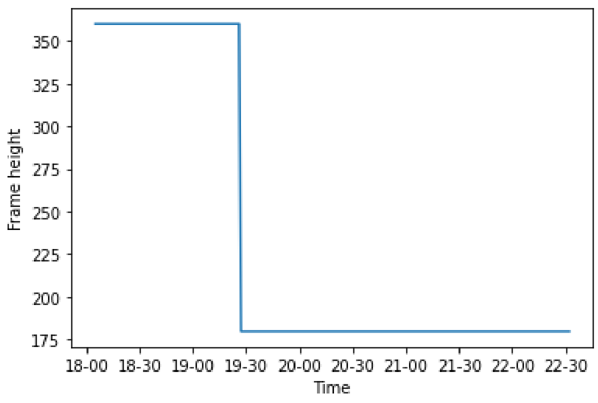


Figure 8: Illinois Frame height

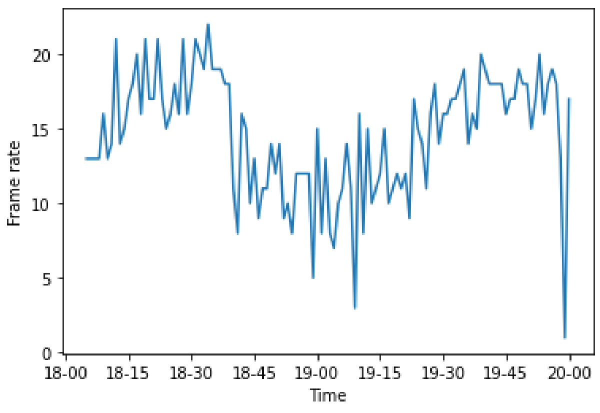


Figure 9: Illinois Frame Rate

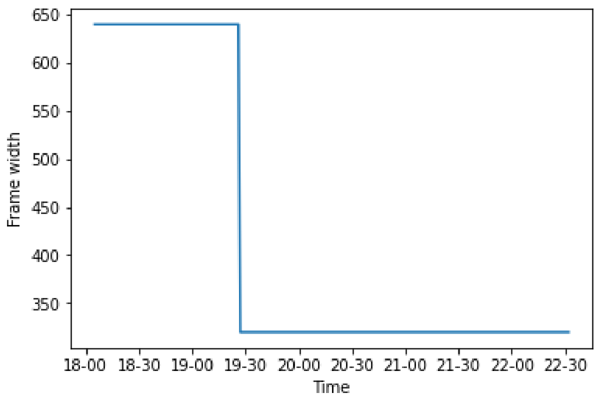


Figure 10: Illinois Frame width

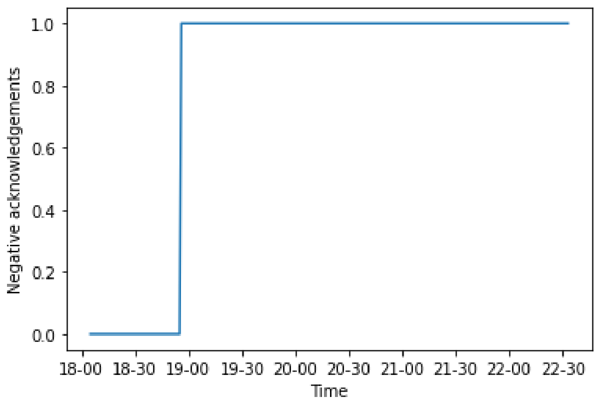


Figure 11: Illinois NAK

3.3 TCP NewReno/Reno

TCP Reno includes algorithms named Fast Retrans- mit and Fast Recovery for congestion control.

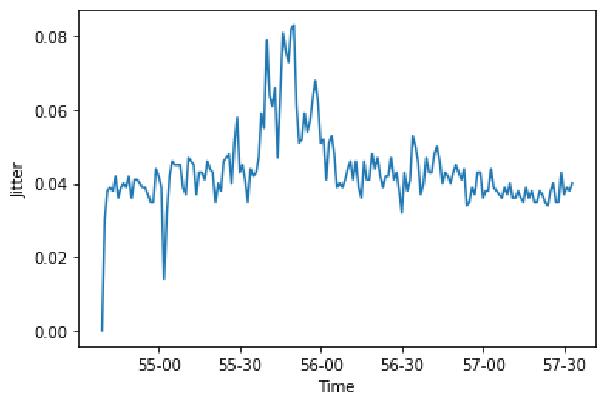


Figure 12: Reno Jitter

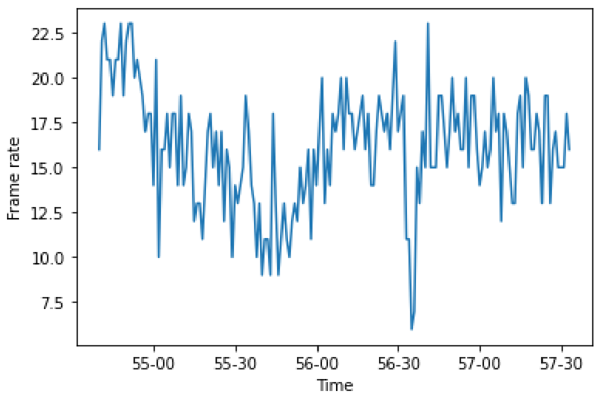


Figure 14: Reno Frame Rate

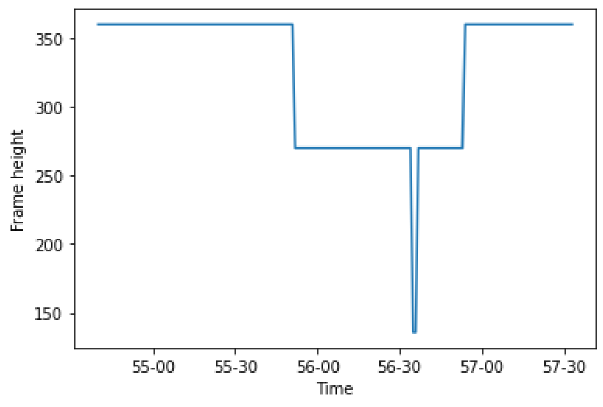


Figure 13: Reno Frame height

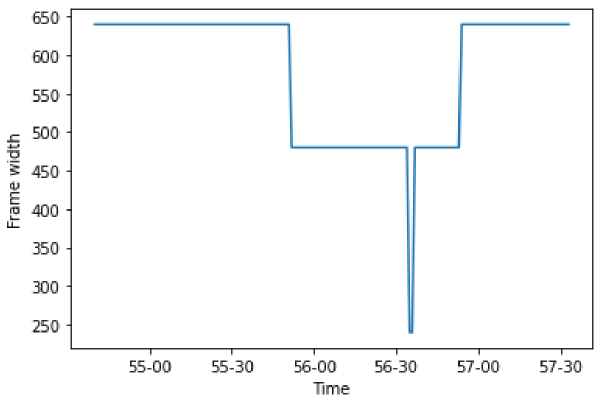


Figure 15: Reno Frame width

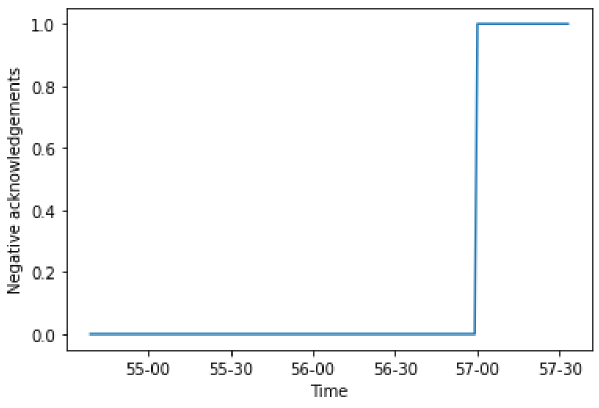


Figure 16: Reno NAK

3.4 TCP Veno

Observations : NACK rates increase much before actual jitter happening in network

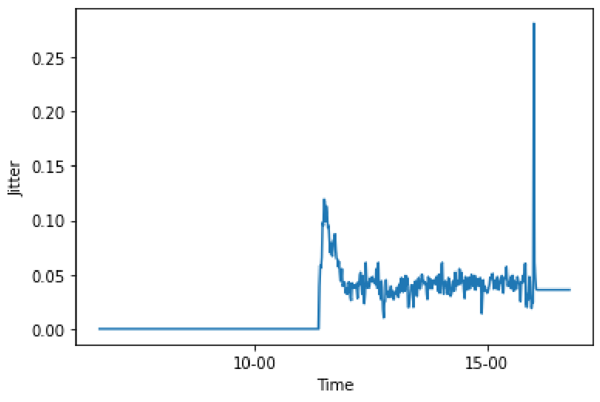


Figure 17: Veno Jitter

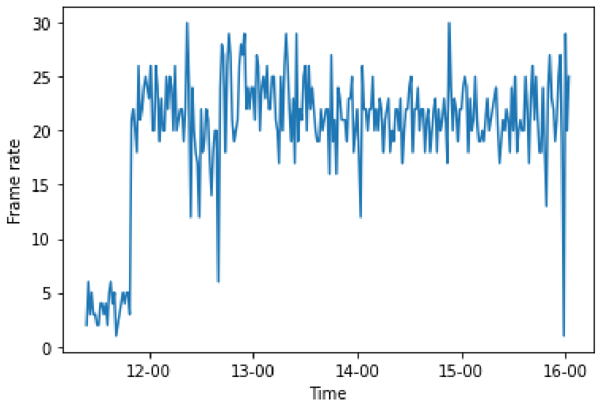


Figure 19: Veno Frame Rate

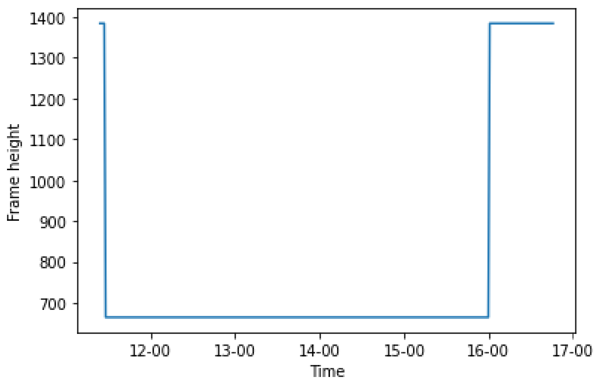


Figure 18: Veno Frame height

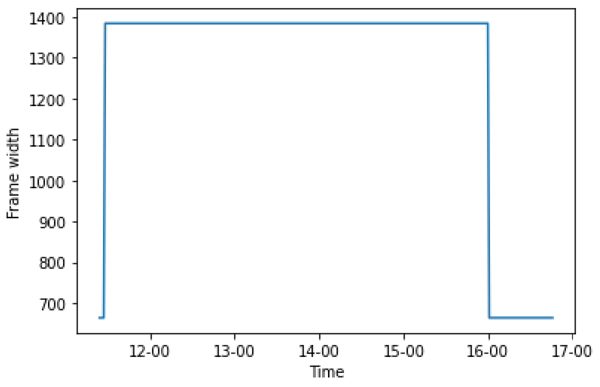


Figure 20: Veno Frame width

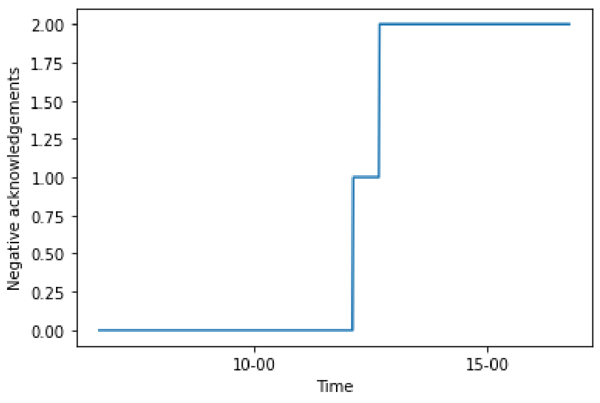


Figure 21: Veno NAK

3.5 TCP Vegas

TCP Vegas is a TCP congestion avoidance algo- rithm that emphasizes packet delay, rather than packet loss, and also help to determine the rate at which we can send packets.

Observations : Video resolution falls only for short duration during peak jitter , then restores quickly back to previous values

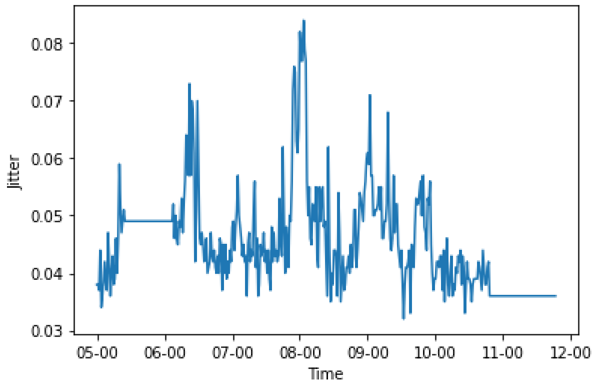


Figure 22: Vegas Jitter

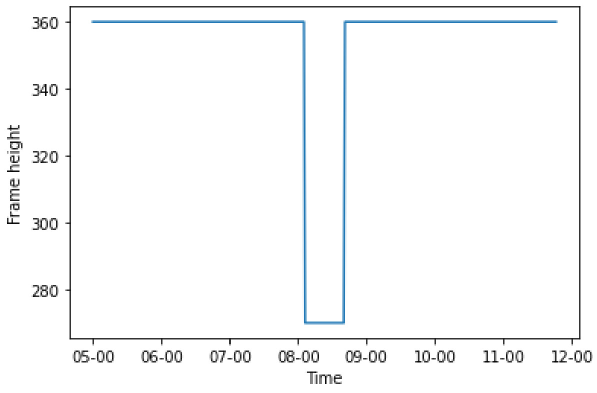


Figure 23: Vegas Frame height

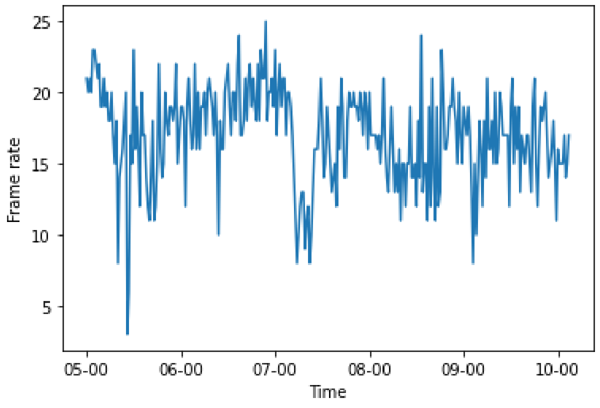


Figure 24: Vegas Frame Rate

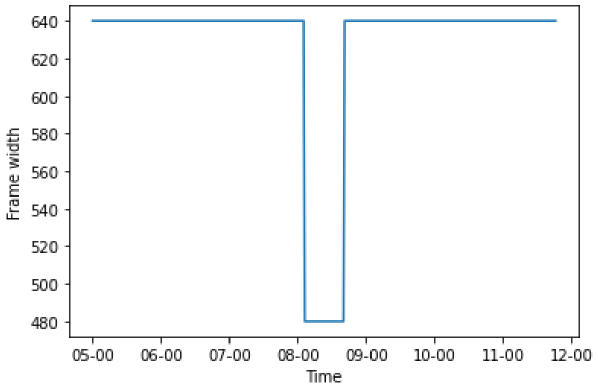


Figure 25: Vegas Frame width

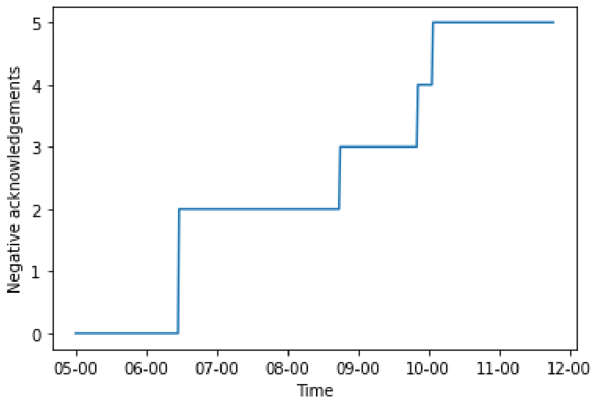


Figure 26: Vegas NAK

3.6 TCP Scalable

As the name suggests, this variant of TCP has very high throughput and is scalable Observations : Decrease in frame width and increase in frame height despite relatively low jitter.

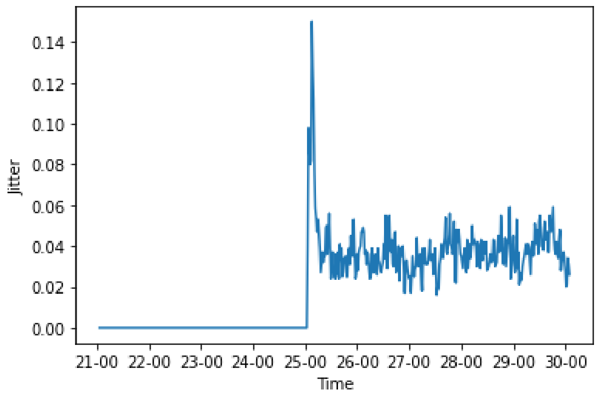


Figure 27: Scalable Jitter

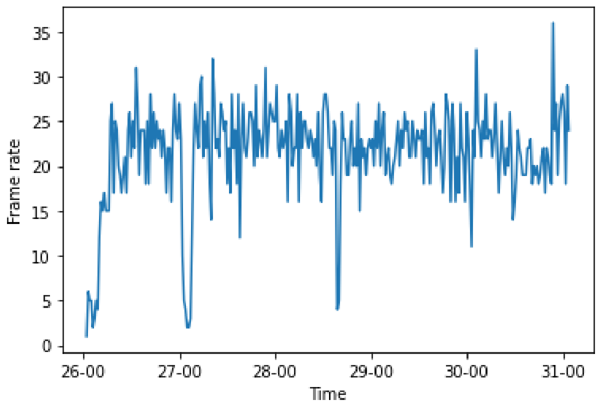


Figure 29: Scalable Frame Rate

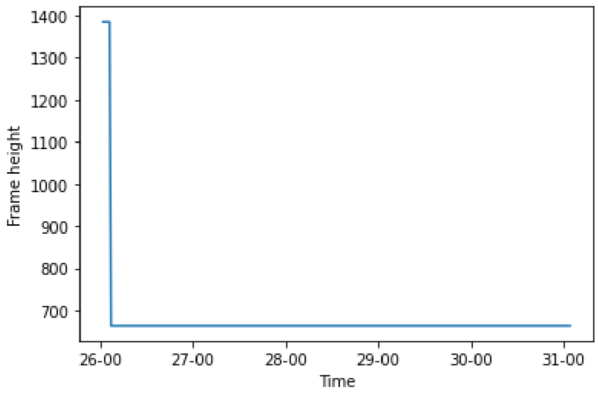


Figure 28: Scalable Frame height

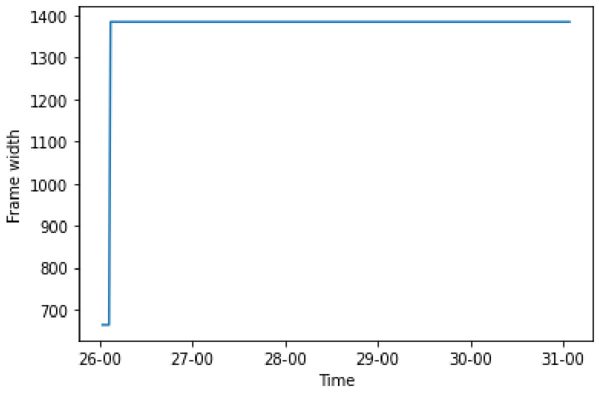


Figure 30: Scalable Frame width

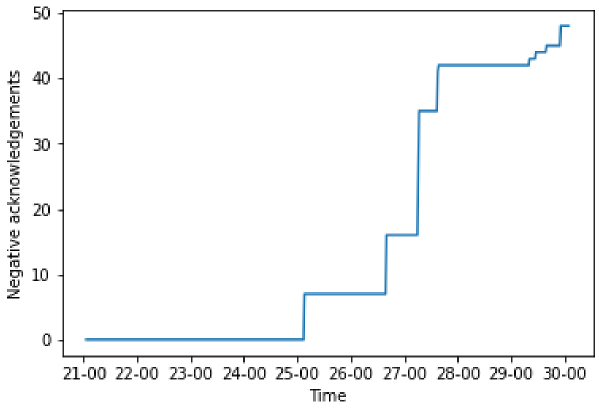


Figure 31: Scalable NAK

4 Throughput vs Payload Metrics comparison

Comparison of various TCP protocol using NS3 simulation.The wifi-tcp file was run with different payload size and the throughput was recorded for different tcp variants. The analysis was done by plotting graph using gnuplot.

4.1 TCP Scalable Throughput vs Payload

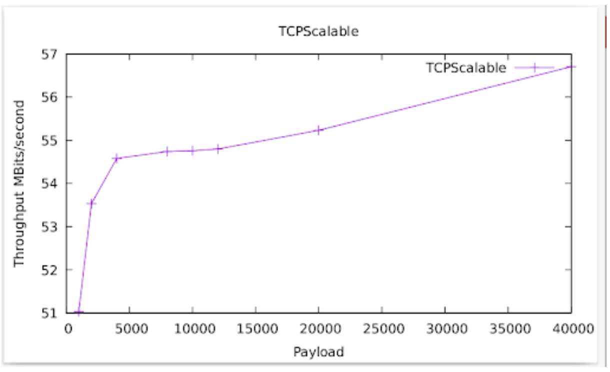


Figure 32: Scalable throughput vs payload

4.2 TCP WestWood Throughput vs Payload

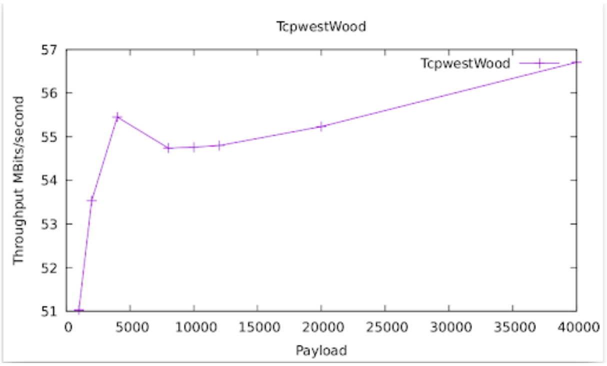


Figure 33: WestWood throughput vs payload

4.3 TCP Reno THroughput vs Payload

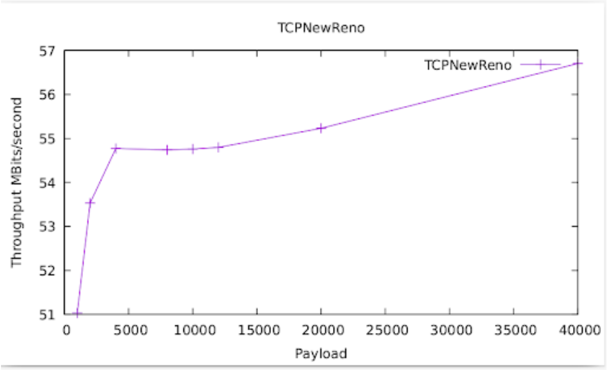


Figure 34: Reno throughput vs payload

4.4 TCP Veno Throughput vs Payload

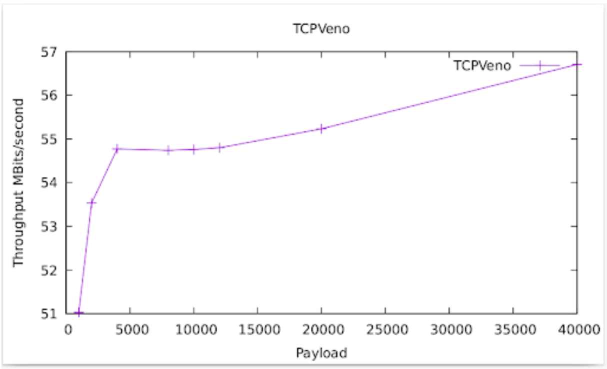


Figure 35: Veno throughput vs payload

4.5 TCP Vegas Throughput vs Payload

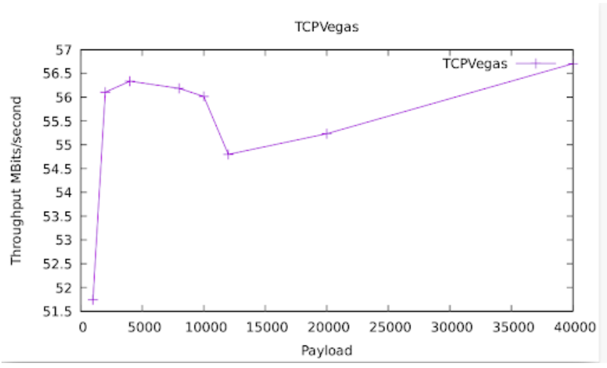
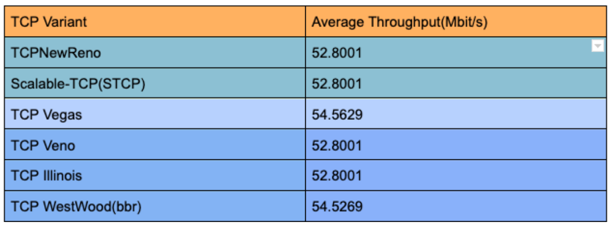


Figure 36: Vegas throughput vs payload



TCP Variant	Average Throughput(Mbit/s)
TCPNewReno	52.8001
Scalable-TCP(STCP)	52.8001
TCP Vegas	54.5629
TCP Veno	52.8001
TCP Illinois	52.8001
TCP WestWood(bbr)	54.5269

Figure 37: Table for TCP variant vs Avg Throughput

5 Conclusion

Different TCP variants react differently to jitter in the network, with TCP vegas syncing faster to changing jitter values. In TCP BBR video resolution didnt change despite high jitter in the network, this can be either be attributed to better congestion control or our lack of longer data duration.

6 Future work

As we can see all metrics here are time series data. We can use complex machine learning models such as

RNN's, LSTM's which remember the context of data patterns over long periods of time, to predict in case of jitter happening in future , how would frame resolution and rates change

7 References

https://webrtc.org/
https://testrtc.com/webrtc-internals-parameters/
https://www.youtube.com/watch?v=I8jn4vKm5QA