DOI: 10.21917/ijct.2011.0037

OPTIMISATION OF BUFFER SIZE FOR ENHANCING QOS OF VIDEO TRAFFIC USING CROSS LAYERED HYBRID TRANSPORT LAYER PROTOCOL APPROACH

S. Matilda[1] and B. Palaniappan[2]
[1]*Department of Information Technology, IFET College of Engineering, Tamil Nadu, India*
E-mail: matildags@yahoo.com
[2]*Department of Computer Science and Engineering, Annamalai University, Tamil Nadu, India*
E-mail: bpau2002@yahoo.co.in

Abstract
Video streaming is gaining importance, with the wide popularity of multimedia rich applications in the Internet. Video streams are delay sensitive and require seamless flow for continuous visualization. Properly designed buffers offer a solution to queuing delay. The diagonally opposite QoS metrics associated with video traffic poses an optimization problem, in the design of buffers. This paper is a continuation of our previous work [1] and deals with the design of buffers. It aims at finding the optimum buffer size for enhancing QoS offered to video traffic. Network-centric QoS provisioning approach, along with hybrid transport layer protocol approach is adopted, to arrive at an optimum size which is independent of RTT. In this combinational approach, buffers of routers and end devices are designed to satisfy the various QoS parameters at the transport layer. OPNET Modeler is used to simulate environments for testing the design. Based on the results of simulation it is evident that the hybrid transport layer protocol approach is best suited for transmitting video traffic as it supports the economical design.

Keywords:
Buffer Size, GoP (Group of Pictures), Hybrid Transport Layer Protocol, QoS (Quality of Service)

1. INTRODUCTION

Video Streams are the major traffic in today's Internet and is bound to grow at a very high rate in near future. With the amount of data and video traffic in the Internet escalating at a fast rate, routers and end devices must be capable of handling multiple Gigabit connections at given time. End-to-end latency is an important metric which affects the QoS from the users' point of view, as video traffic is delay sensitive. This end-to-end latency has three components: transmission delay, propagation delay and queuing delay. Of these queuing delay is the only variable component and is the single biggest cause of uncertainty [2].

An optimum buffer size should take into consideration QoS parameters such as, queuing delay, link utilization, end-to-end throughput and packet loss. To enhance performance, a large throughput is desired and to reduce delay, data in the buffer is to be made as less as possible. While the former case, demands an increase in the average sending rate and thereby the buffer size, the latter case requires reduction in buffer size. These goals are contradictory and thereby, call for a multi-criteria optimization solution [3].

Most of the work done considers only one or two parameters for design of buffer in core routers or edge routers, while the rest of the criteria are at stake [3], [4], [5]. Very few solutions are offered using multi-criteria optimization for all devices from source to destination.

Real-time video is streamed using Real Time Streaming Protocol (RTSP) at the application layer and TCP or UDP at the transport layer. Transmitting real-time video using the above method results in the links being underutilized and lays a limitation on the number of clients that can be serviced [4]. In addition as the number of clients increase, the bandwidth is insufficient and the quality of service decreases [6]. These drawbacks are overcome using cross layered hybrid transport layer protocol approach [1]. The hybrid approach uses TCP to transmit the I frames and UDP to transmit the B and P frames. This novel approach increases the number of clients serviced simultaneously and improves network performance.

This paper deals with the design of buffers and aims at finding the optimum buffer size for video traffic. Network-centric QoS provisioning approach is adopted along with hybrid transport layer protocol approach. In this combinational approach, buffers of routers and end devices are designed to satisfy the various QoS parameters at the transport layer. As the QoS parameters are contradictory, prioritization of the parameters is done to find the optimum solution. End-to-end delay is given the highest priority as the traffic under consideration is video stream. The next in order are throughput and link utilization. Least preference is given to packet loss and initial delay as their effect on video is not critical.

In the OPNET simulation, video trace encoded using H.264 (SVC) encoder is preprocessed to suit the design and is imported as traffic into the simulated network. A bottle-neck link is introduced and twelve scenarios are used to compare and analyze the QoS parameters. Comparison of QoS parameters is done between traditional TCP and cross layered hybrid transport layer protocol for all the twelve scenarios for various buffer values.

2. STREAMING VIDEO USING CROSS LAYERED HYBRID TRANSPORT LAYER PROTOCOL APPROACH

2.1 BASICS OF VIDEO FRAMES

Videos are a sequence of frames (images) displayed at a constant rate and each frame contains spatial (within) or temporal (between images) redundancy. Video coding schemes reduce the raw video content size by exploiting this redundancy. Video formats may range from 128 x 120 pixels to 1920 x 1080 pixels with various color depths. Common Internet video formats (YouTube) uses 320 x 240 pixel resolution. The commonly-used high definition standard is 1280 × 720 pixels and hence this size is chosen for analysis. The higher the video

frame resolution or pixel color depth, the larger is the raw video content size.

Consider an encoded video with a GoP (Group of Pictures) pattern G10B2. This pattern has 10 a frame of which one is an I frame, three are P frames and six are B frames. (shown in Fig.1)

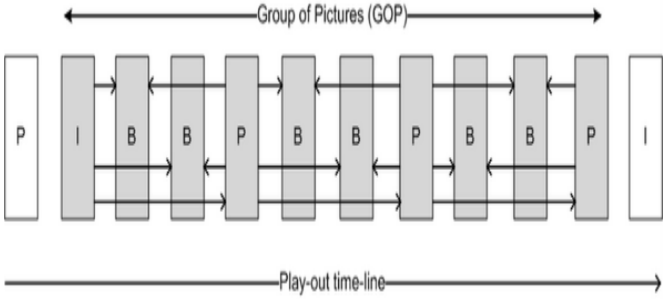

Fig.1. Frames in a GoP pattern G10B2

Using a larger GoP, the bit rate can be reduced but the quality of video decreases. Smaller GoP value increases the bandwidth but gives a good video quality.

2.2 CHOICE OF TRANSPORT LAYER PROTOCOL

Under normal circumstances real time video is transmitted using RTSP in the application layer and TCP or UDP in the transport layer. RTSP is a stateful protocol and a session identifier is used to keep track of sessions when needed. TCP ensures reliability and offers a good quality of service, but the delay occurring during the three way handshake and acknowledgments makes it unsuitable for real-time video. UDP does not guarantee delivery of the media stream. If there is data loss, the stream may suffer a "dropout". However this is the simplest protocol available.

The size of an I frame in encoded video is large when compared to P or B frame. Loss of an I frame can distort the quality of video, as the following B or P frames cannot be decoded, while loss of P or B frame will have negligible effect (Fig 2). Hence, to retain quality, all the I frames are to arrive intact at the receiver.

To improve reliability, quality of video, link utilization and reduce end-to-end delay both TCP and UDP are used in the transport layer [1], [6], [7], [8]. To capitalize the advantages of both the transport layer protocols, TCP is used to transmit the I frames, while P and B frames are transmitted using UDP [6]. This stems from the fact that RTSP is transport independent and the default port for both TCP and UDP is 554 [9], [10], [11]. This cross layered hybrid approach increases link utility and reliability, while it minimizes end-to-end delay and buffer requirement. Thus the overall performance of video transmission is improved and the quality of service is enhanced [12].

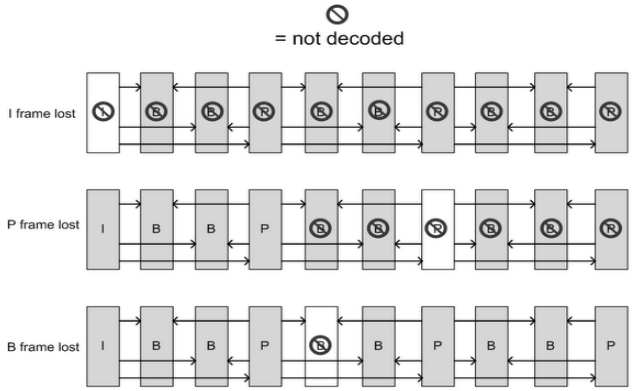

Fig.2. Effects of decoding due to lost frames

3. NECESSITY OF BUFFERS

The transport layer is responsible for flow control and congestion control in the Internet. Flow control ensures that the receiver is not overwhelmed and congestion control ensures that the network is not throttled with packets. When a connection is established sockets are created at both the sending and receiving ends and buffer space is allocated.

TCP uses adaptive sliding window scheme for implementing flow control. The maximum congestion window size is related to the amount of buffer space available at a given time in each socket. On the sender side bytes between LastByteAcked and LastByteWritten must be buffered and on the receiver side bytes between NextByteRead and LastByteRcvd must be buffered. UDP provides best effort delivery service and does not have any flow control mechanism. However buffer space is allocated when UDP sockets are created.

Video playback rates range from 10 frames to 30 frames per second. Network conditions affect the arrival rates of these frames, but, it is critical that the client playback the video at this speed to maintain the quality. If the frames are played at the speed at which they arrive, discontinuity in sequence will be observed by the user, because of the delay in arrival caused by a bottle-neck link or network congestion. The only alternative is to queue the packets and make them readily available for playout. In the case of video, decoding can be done only if the entire GoP is available. Hence all the packets which constitute the GoP have to be stored, for decoding the entire GoP. Buffering helps to smooth out differences, between packet entry or delivery rate and exit or consumption rate, in any networking device, between the sender and the receiver. It aids in seamless playout at the receiving end. However, like any buffering, socket buffers can add latency if the size of the queue is large. The delay in the buffer is computed by dividing the size of the socket buffer, by the data rate. Properly designed buffers reduce delay and increases throughput simultaneously. Default buffers space allotted for each socket connection create a problem rather than improving the QoS parameters.

3.1 BUFFER SIZING PROBLEM

The functionality of the router varies based on its position in the network. Backbone routers support, routing at high data rate over a few links. Enterprise routers support a large number of links for a rich set of value added services. In addition it has to

support high bandwidth and delay guarantees. The efficiency and performance of the router buffers play a critical role and if the buffers are not properly designed the performance of the entire network drops. Buffer size should be larger enough to reduce packet loss and at the same time be small enough to reduce queuing delay. The buffer sizing problem is a stochastic, non-linear problem with an integer decision vector. It is a hard combinatorial optimization problem, which is made more difficult by the fact that it is not obtainable in a closed form to interrelate diagonally opposite metrics like throughput, delay, link utilization and packet loss. Reduction in buffer size also reduces the cost and saves memory space inside the buffer

3.2 EXISTING BUFFER DESIGNS FOR ROUTERS

3.2.1 Rule of the thumb:

This was derived based on Villamizar and Song's first experiments in 1994 and assumes a single long-lived TCP flow going through the bottleneck link. The rule is defined by the expression (1)

$$\text{Buffer Size} = RTT \times C \quad (1)$$

where RTT is the Roundtrip time and C is the Channel capacity of the bottle-neck link [13].

This rule cannot give a single constant value to the size of the buffer, as the value of RTT varies for different networks depending on the link speed, type of network and many other characteristics of the network. Even for the same network the value of RTT varies with time and destination. Hence arriving at a single value of RTT is not possible and calculating buffer size based on RTT is suitable only for adaptive buffer sizing .Further related studies show that, the size of the buffer is very high for a single flow.

3.2.2 Small Buffers Model:

Appenzeller et al. proposed this model, wherein the buffer size is reduced by a factor $\sqrt{N}$ when N long-lived TCP flows share a link.

$$\text{Buffer Size} = \frac{RTT \times C}{\sqrt{N}} \quad (2)$$

For a 50Mbps link with 200 TCP flows and 60 ms. RTT, the buffer size is 1500 packets according to the rule of the thumb and 100 packets as per small buffers rule [14]. Though the size of the buffer is reduced considerably and large number of flows is accommodated, the drawback of using RTT still exists. Moreover, the number of flows is also a variable and hence, this design is suitable only for adaptive buffer sizing solutions.

3.2.3 Tiny Buffers Model:

This model suggests a buffer size of just 20 -50 packets. This rule assumes that the user is willing to sacrifice the throughput and link utilization by 10 -15%. It is designed for optical routers, where link is not a bottleneck but buffer is the bottle neck [15]. All these designs are for TCP connections as they occupy 80- 85 % of the Internet traffic [16].

3.3 EXISTING BUFFER DESIGNS FOR SENDER AND RECEIVER

UDP receive buffering is done in the operating system kernel, wherein the kernel allocates a fixed-size buffer for each socket receiving UDP. Buffer space is consumed for every UDP packet that has arrived, but has not yet been delivered to the consuming application. The memory required for the kernel to do UDP buffering is a scarce resource and the kernel allocates a modest-size buffer when a UDP socket is created. UDP buffer sizes should be large enough to allow an application to endure the normal variance in CPU scheduling latency without suffering packet loss. They should also be small enough to prevent the application from having to read through excessively old data following an unusual spike in CPU scheduling latency. Too little UDP buffer space causes the operating system kernel to discard UDP packets.

Buffer Size for UDP Receiver is given as the product of Maximum Latency and Average Rate. UDP receive buffer space is allocated based on policy settings. However, UDP senders do not monitor available UDP receive buffer space in the receiving kernel. UDP receivers simply discard incoming packets, if the available buffer space is exhausted. UDP is "sender-paced" because the sending application can send whenever it requires without regard to available buffer space in the receiving kernel [17].

The operating system kernel allocates TCP receive buffer space based on policy settings, available memory, and other factors. TCP in the sending kernel continuously monitors available receive buffer space in the receiving kernel [17]. When a TCP receive buffer fills up, the sender calls for flow control and stops sending further packets. Thus TCP sender is "receiver-paced".

TCP sockets are considerably larger than UDP sockets as there are more connection state information to maintain. TCP sockets also require both receive and transmit buffer, whereas UDP sockets require only a receive buffer [12]. The variable TCP_BUF_SIZE determines the TCP buffer size and depends on the operating system. Windows XP has a default TCP buffer size of 17,520 bytes. UDP_BUF_SIZE determines the UDP buffer size, the default value being 4096 bytes [12].

4. ANALYSIS OF VIDEO FRAMES

The size of a P frame is 50% that of an I frame and the size of a B frame is approximately 25% that of an I frame. The maximum payload size of a TCP packet is 1500 bytes.

$$\text{Frame size} = \text{Pixel Width} \times \text{Pixel Height} \times \text{Bit Depth} \quad (3)$$

The normal bit depth is 24 and compression ratio of H.264 is 60:1. The number of segments and the overhead associated with a GoP is evaluated for various transport layer protocols.

Case1: If TCP is used as the Transport layer Protocol

For a frame resolution of 1280 × 720 pixels (commonly-used high definition) the approximate size of I frame calculated using Eq. (3) is 45 KB /frame. Hence an I frame has to be roughly segmented into 31 parts, for being transmitted using TCP. This implies that a P frame has to be broken into 16 segments and B

frame into 8 segments. Hence one GoP contains 55 segments and is approximately 82,500 bytes. TCP headers are 20 bytes long and the overhead in this case is 1100 bytes. This implies that approximately 1 KB is sent as an overhead along with each GoP.

Case 2: Using Cross layered Hybrid Transport layer Protocol approach

As I frame is to be transmitted using TCP, it is to be segmented into 31 portions, as discussed in case 1. The maximum size of a UDP datagram is 65535 bytes or 64 KB. If the size of an I frame is 45KB then the size of the P frame will be 22.5 KB and the size of a B frame will be 11.25 KB approximately. Hence three B frames and a P frame can be transmitted using a single UDP datagram. The remaining B and P frames of the GoP can be transmitted using a maximum of 3 UDP datagrams. UDP headers are 8 bytes long and the overhead is only 24 bytes. The total overhead in this case is 644 bytes and the overall reduction in overhead compared to the first case is approximately 43 %.

Case 3: Using UDP as the Transport layer Protocol

This case has not been taken for discussion as reliability is not ensured when UDP is used. This affects most of the QoS parameters offered by the transport layer.

5. OPTIMAL BUFFER SIZE DESIGN

To achieve the best quality of service, it is critical to use optimal buffer sizes. Assuming that packet is not lost due to network congestion, the network throughput is directly related to TCP buffer size and latency. For economical design, buffer at any time should hold the number of packets required for immediate consumption by the application. For example, in an application where every tenth packet is counted or if only the tenth packet has the required information, it is sufficient to design a buffer with a capacity to hold 10 packets. Parallel to this analogy the buffer capacity is calculated for different approaches.

5.1. PACKETS TRANSMITTED USING TCP

In the current application video is played frame by frame – the general playback speed of video being 30 frames per second. The general policy is that transmission rate must be greater than or equal to consumption rate for continuous playback. Hence it is sufficient if the buffer can hold only packets required for assembling a single frame, provided the bottleneck link supports a speed greater than 30 frames per second.

In case of video, where a constant GoP of 10 is assumed, the first frame is usually an I frame and can be decoded when the first 31 packets arrive. But the following frames are P and B frames and depend on the previous and successive frames for being decoded. All the 10 frames can be decoded only if 55 packets are available. Thus the minimum buffer size should be equal to 55 packets. Buffer size of 81 KB is ideal to hold 55 packets, both at the transmitting and receiving end. The size of 30 frames in 3 GoP's approximates to 247500 bytes. Link speed of 256 Kbps at the bottle-neck link will support this frame rate for the estimated buffer size.

5.2. HYBRID TRANSPORT PROTOCOL APPROACH

In this approach, 31 packets which make up the I frame uses TCP, while the remaining 24 frames use UDP. The size of the sender end buffer is negligible in UDP, as it supports asymmetric traffic. At the sender end, buffer, with a capacity to accommodate 31 packets is sufficient. But as in the earlier case, buffer at the receiving end should hold 55 packets at a given time. Hence at the sending end, buffer size of 45 KB is sufficient, while 81 KB is required at the receiving end. As UDP packets take different routes the dedicated link transmits only 31 packets for each GoP. The minimum bandwidth which the bottle–neck link will support for this transmission is 145 Kbps. Thus adapting the hybrid transport layer protocol approach shows a considerable reduction in the buffer size and bandwidth requirement at the bottle-neck link while transmitting video traffic.

In case of multiple flows the size of buffer at server and client can be maintained at one GoP, while the buffer size at the routers can be calculated as shown,

If $C_i \leq C_o$, then B = 1 GoP

else

$$\text{If } C_i > C_o, \text{then } B = 1 \text{ GoP} \times \frac{Ci}{Co} \qquad (4)$$

where C_i is the sum capacity of all incoming links and C_O is the sum capacity of all outgoing links. Incoming link in this case is defined as the link closer to the server and outgoing links are those links closer to the client.

6. SIMULATION ENVIRONMENT

OPNET [18] is the simulator used to design the Internet environment as it is flexible and facilitates the requirements for this study. It supports importing video traces from external sources to generate traffic in the simulated environment. Applications could be configured to use TCP and UDP simultaneously. OPNET provides a GUI for the topology design, and offers a varied choice of real-time network components and devices in its object palette. It allows for realistic simulation of networks and has performance data collection and display module. Scenarios can be duplicated easily and the same network can be run for different parameters and protocols. Another advantage of using OPNET is that, it has been used extensively and there is wide confidence in the validity of the results it produces.

6.1. DETAILS OF VIDEO TRACE USED

For the simulation, traffic sources are required to create a realistic environment. [19], [20] demonstrate that for video traffic, usage of traces is a good choice. Sources of encoded video traces with various GoP, frame size, resolution and playback speed (frames/sec) are readily available in the research sites of many universities. H.264 SVC (Scalable Video Coding) codecs produce much higher traffic variability. H.264 SVC is primarily designed for scalable video encoding, but can also be used for encoding video into a single layer. The resulting single-layer encoding has typically temporal scalability built in, that is,

a single-layer SVC stream can be played out with different frame rates. The video trace was preprocessed to improve resolution before being imported into OPNET Modeler. Profile of video trace used is given in Fig.3 and Characteristics of Video trace used are given in Table.1.

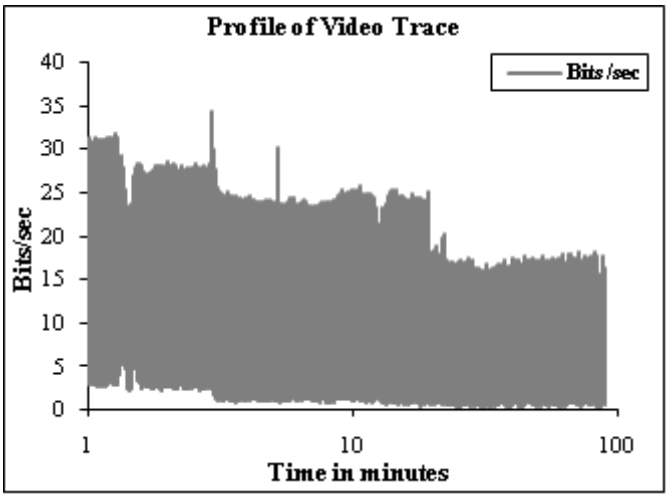

Fig.3. Profile of Video Trace

Table.1. Characteristics of Video Trace

Parameters	Values
Resolution	1280×720 pixels
Codec	H.264 SVC
Frame Compression ratio	60
Max Frame Size (Bytes)	45079
Min Frame size (Bytes)	48
GoP pattern	G10B3
Maximum GoP Size (Bytes)	85909
Minimum GoP Size (Bytes)	419
Frame Rate	30 fps
Number of frames	322979
Average Bit Rate	241.67 Kbps
Peak Rate	343.2 Kbps
Min Bit Rate	7.2 Kbps
Total Duration	181.49 minutes

6.2. SIMULATION SCENARIOS

A sample network with one video server, four edge routers, three core routers and six subnets was designed using OPNET Modeler version 14.0. The video server, served as the source of video described in section 6.1. Each edge router supports two subnets which are 100 Base T switched LANs (100BaseT_LAN in the object Palette Tree). Each LAN is a unit of 10 systems with a switching speed of 500,000 packets per second. Hence delay caused in the LAN due to the switching environment is negligible. The application layer supported video of all formats and the transport layer supported both TCP and UDP. This setup was used to test for the multicast environment. To maintain the same characteristics in a unicast environment the scenario was duplicated. The subnet connected to edge router 4 was replaced by a single video client and all the other subnets were failed.

The buffer size of video server and clients are set to hold 1 GoP and buffer size of the routers is calculated using expression (4). The buffer design was tested for the scenarios, given in Table.2. Fig. 4 and Fig. 5 are the screenshots of the unicasting and multicasting environment.

Table.2. List of Scenarios used for Simulation

Scenario Number	Transmission type	Protocol used	Buffer Size (Kb)
1a	Unicast	TCP	64
1b	Unicast	TCP	85
1c	Unicast	TCP	128
2a	Unicast	TCP+UDP	64
2b	Unicast	TCP+UDP	85
2c	Unicast	TCP+UDP	128
3a	Multicast	TCP	64
3b	Multicast	TCP	85
3c	Multicast	TCP	128
4a	Multicast	TCP+UDP	64
4b	Multicast	TCP+UDP	85
4c	Multicast	TCP+UDP	128

The link between Edge router1 and Core router1 was configured to act as the bottle-neck link with a bit rate of 256 Kbps. Qos parameters across this link was measured for comparison. Results were observed for buffer sizes of 64 KB, 85 KB and 128 KB at the client and server end. Fig 4 and Fig 5 are screenshots of the unicast and multicast simulation environments. For transmission using TCP, parameters were set in the video server, video client and in the application definition. The Windows XP version / flavour of TCP was chosen to make it similar to real world environment. In the application definition, the frame rate was set to 30 frames/s and type of service was set to 156. (Fig 6) This ensured video streaming application using TCP. The preprocessed video trace was imported to generate the required traffic in the network. The frames are counted as they are imported. GoP pattern is G10B3 indicating that the I frame arrives at position 0,10,20,30 etc. The following algorithm was used to segregate the frames to be sent through TCP and UDP.

Algorithm 1: Segregation of Frames using size of GoP

Initialise Count =0, gop =10

for Count = Count

If mod(Count/gop)=0,

transmit frame using tcp

end

else

transmit frame using udp

end

Count = Count + 1

Continue for loop till end of trace.

end

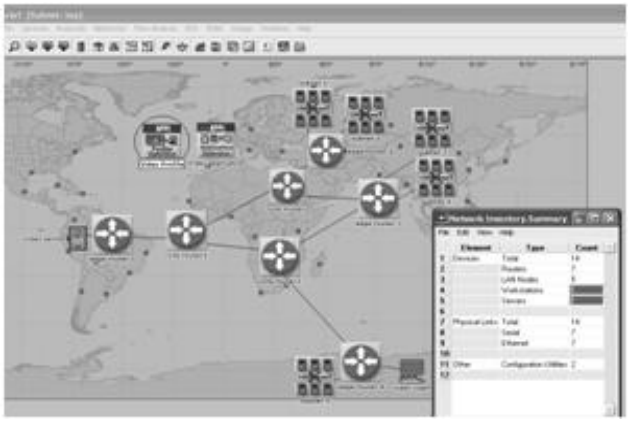

Fig.4.Unicasting environment with summary of network inventory

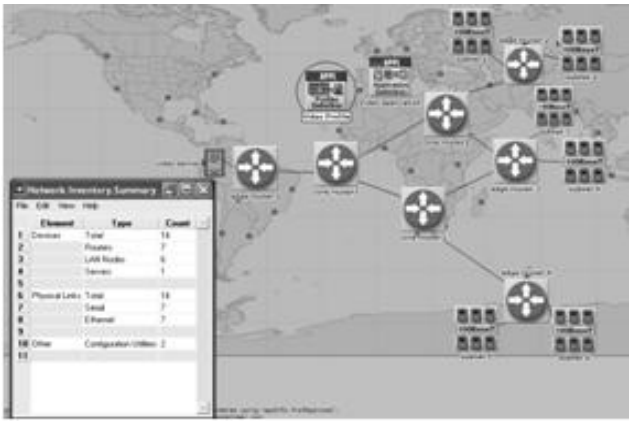

Fig.5. Multicasting environment with summary of network inventory

Fig.6. Configuring the Simulation Environment for Video Transmission using TCP with ToS 156

7. SIMULATION RESULTS

7.1 DELAY PARAMETERS

7.1.1 Delay in Core Router:

Processing delay which was inherent in routers was observed during simulation. The value was 0.0000040 seconds for all routers in the network throughout the entire period. Hence delay due to processing is negligible. Fig.7a. shows the output graph for processing delay in core router1. As the link between the edge router1 and core router1 (L1) was configured to act as the bottle-neck link, delay parameters across the link was observed. Fig .7b. shows the queuing delay in the bottle-neck link as taken from the object statistics of OPNET Modeler. The results for various scenarios are shown from Fig. 8a. to Fig. 8d.End-to-end delay was also observed for various scenarios using TCP and result is shown in Fig.8e.

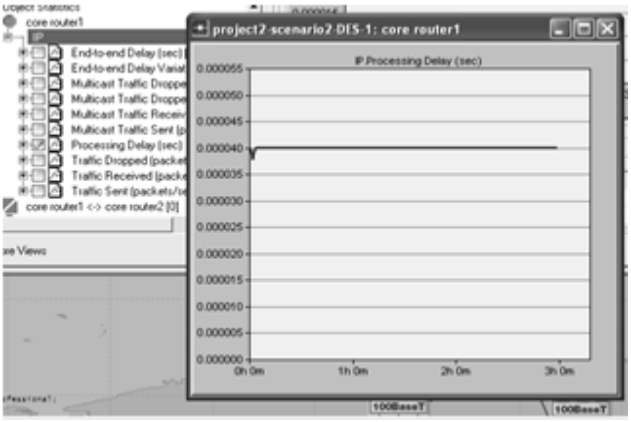

Fig.7a. Processing Delay in Core router 1 as viewed in OPNET

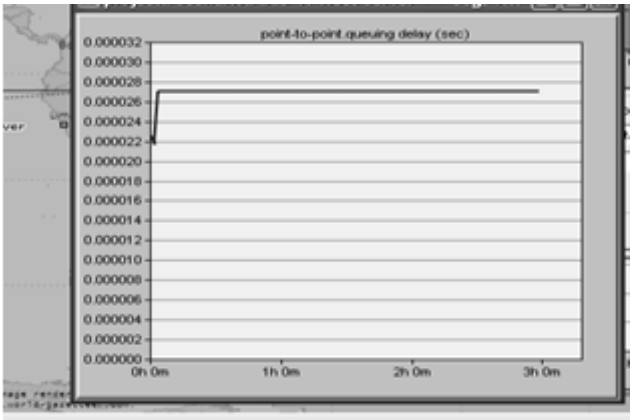

Fig.7b. Queuing Delay between Edge Router1 and Core router 1 as viewed in OPNET.

7.1.2 Queuing Delay at Bottle-Neck link:

Queuing Delay for all the scenarios described in Table.1 are shown from Fig.8a. to Fig.8d.

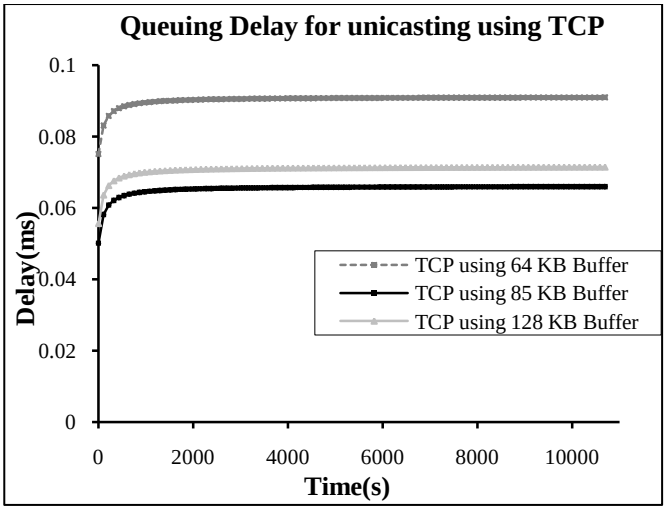

Fig.8a. Queuing Delay for Unicasting using TCP for different Buffer Sizes

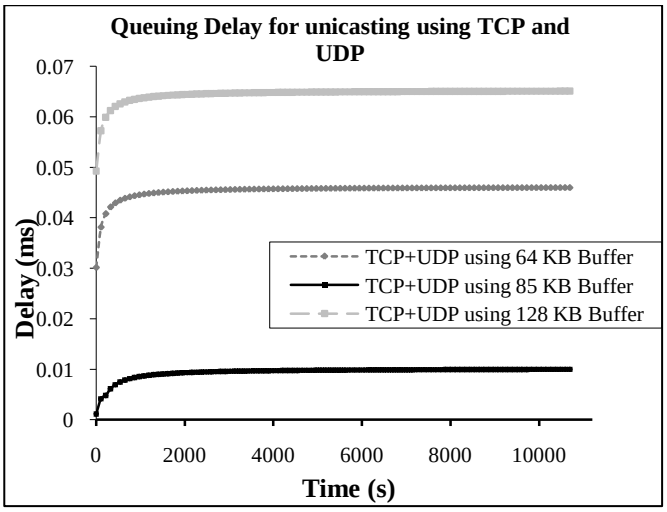

Fig.8b. Queuing Delay for Unicasting using TCP+UDP for different Buffer Sizes

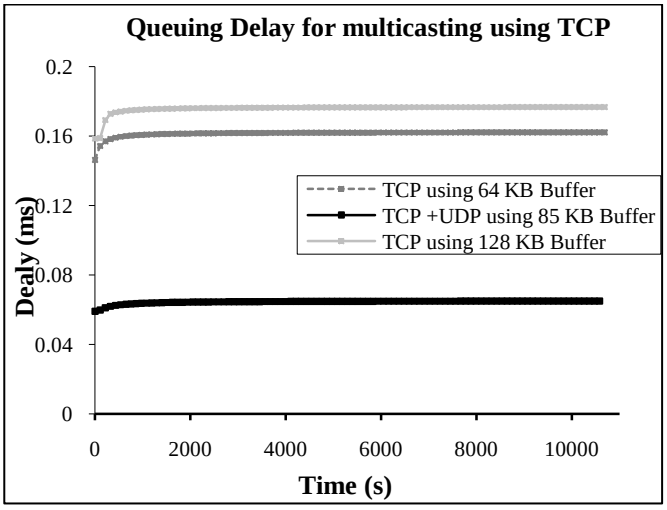

Fig.8c. Queuing Delay for Multicasting using TCP for different Buffer Sizes

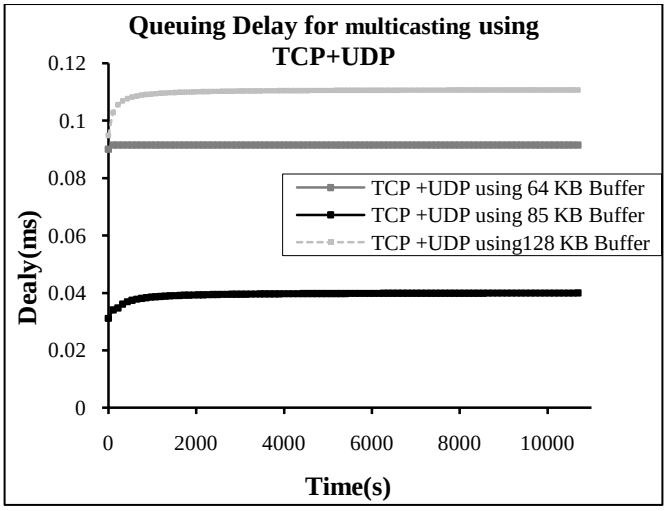

Fig.8d. Queuing Delay for Multicasting using TCP+UDP for different Buffer Sizes

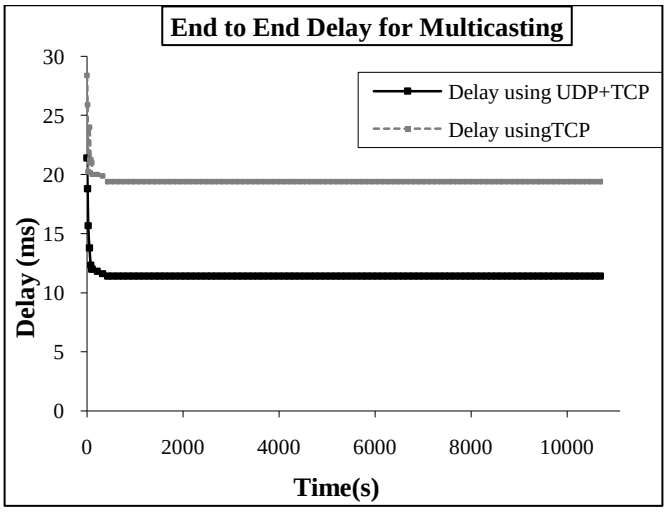

Fig.8e. Average value of End-to-end Delay for Multicasting as obtained from Scenario 3b and 4b for Buffer Size of 85 KB

7.2 THROUGHPUT AT THE CLIENT

Throughput was measured at the video client for the unicast environment and a maximum value of 207.22 Kbps was observed. This is due to the link speed of the bottleneck link and the bit rate of the video trace which was set at 256 Kbps. The maximum speed observed for the multicast environment was 200.34 Kbps. Fig. 9a to 9d depict the throughput at the client, for unicast and multicast environments using TCP and hybrid transport layer protocol approach.

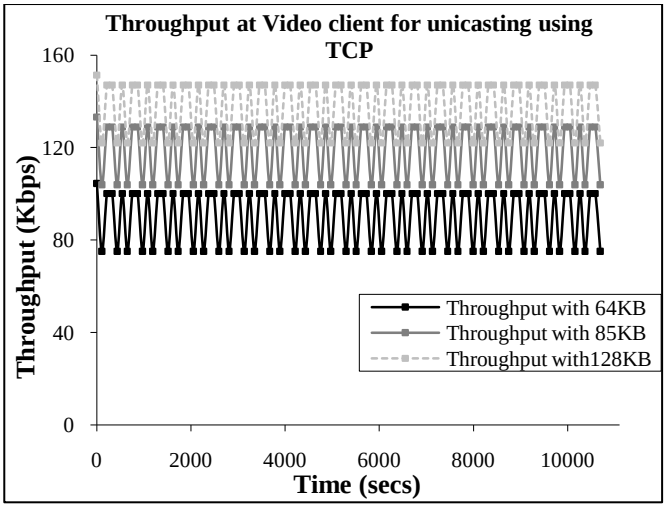

Fig.9a. Throughput at the Client for Unicasting using TCP

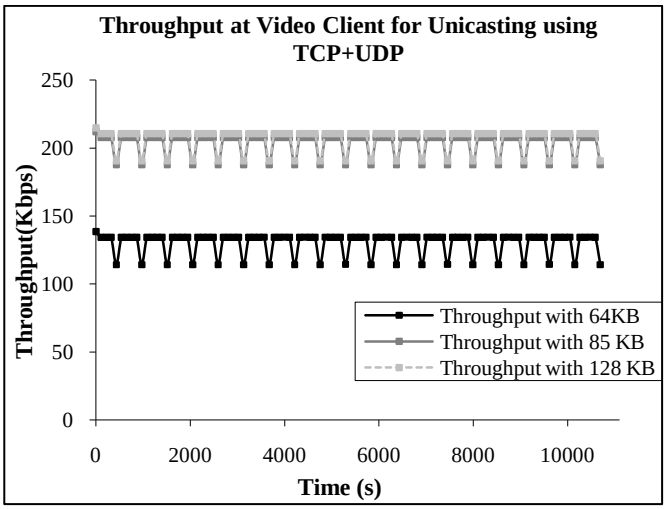

Fig.9b. Throughput at the client for unicasting using TCP +UDP

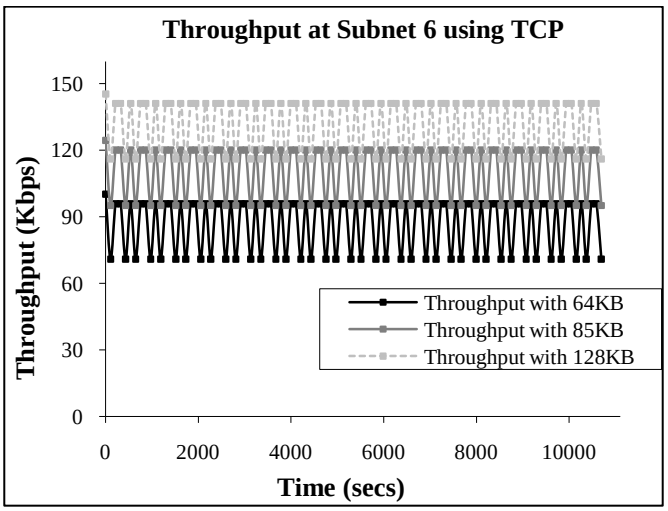

Fig.9c. Throughput at Subnet 6 for Multicasting using TCP

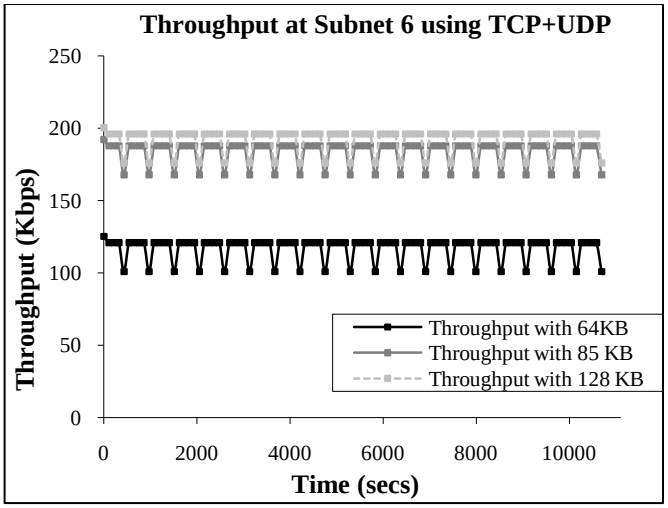

Fig.9d. Throughput at Subnet 6 for Multicasting using TCP +UDP

7.3 PACKET LOSS

This parameter was given the least priority as its effect on video quality is minimum. No packets were lost in unicasting environment, while the packets dropped for multicasting is shown in Fig. 10.

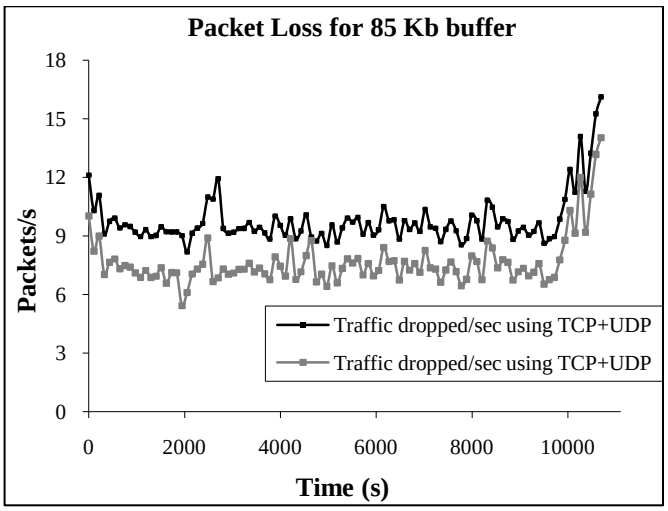

Fig.10. Traffic Dropped at Subnet 6 for Multicasting

7.4 LINK UTILISATION

Link utilization was maximum in the case of unicast routing for 85 KB buffer. The average link utilization for unicasting is 82.27% and 80.02 % for multicasting.

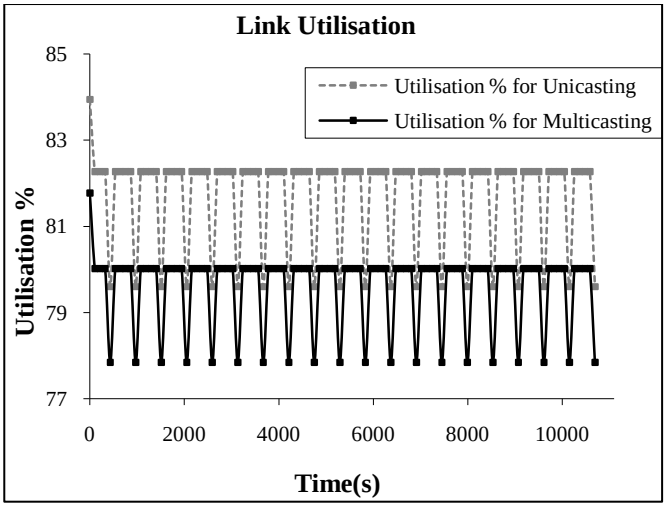

Fig.11. Link Utilization for 85 KB Buffer

7.5 BUFFER UTILISATION

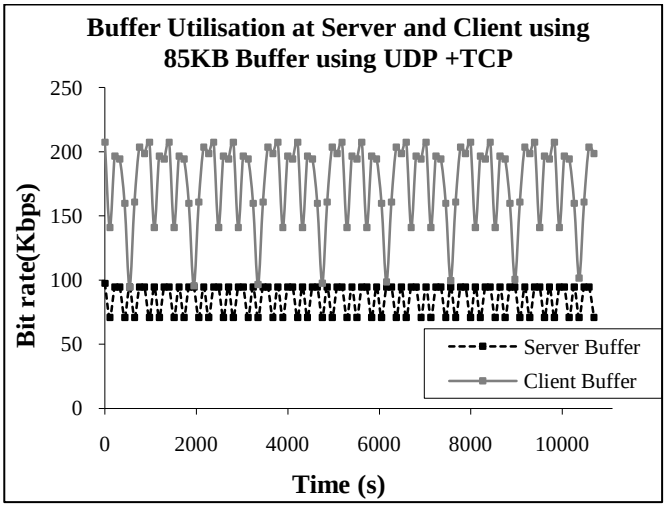

Fig.12. Buffer Utilization for 85 KB Buffer at Server and Client

7.6 DISCUSSION ON THE RESULT OBTAINED

This study explored the technical details and performance of transmitting a MPEG 4 encoded video trace under different environments using TCP and combination of TCP and UDP. It was observed that, the delay parameters were less and link utilization was high for the hybrid layer transport protocol scenarios. In both the cases packet loss was significant. Using TCP the traffic drop rate was better but the delay parameters were nit of acceptable level. Moreover throughput was slightly high in the case of hybrid layer transport protocol. It was also observed that the buffer at the server was only half utilized throughout all the scenarios. The entire simulation was done for a frame size of 1280 × 720 pixels which is the commonly used high definition value. Based on the prioritization assumed, and the results of simulation, we can conclude that the performance of video using hybrid layer transport protocol is better than performance using TCP. Hence the Buffer designed as per this work is optimum and will give the best QoS parameters for video traffic.

8. CONCLUSION

The results of various scenarios show that the design adopted by this novel approach (85 KB) for commonly used high definition video is appropriate. The QoS parameters of H.264 video trace under various environments using TCP and hybrid transport layer protocol approach for this buffer size were analysed. The scope of the work was limited as the environments were entirely simulated. Based on the results of simulation it is evident that the hybrid transport layer protocol approach is best suited for transmitting video traffic Future models could revisit these design parameters by including the SITL module of OPNET and create a real-sim-real environment where realtime video could be imported from a camera and the output can be viewed in another display unit. The difference in input quality and output quality of the video could be compared using PSNR values. Comparison could also be made with between UDP, TCP and hybrid transport layer protocol approach. Mean Opinion Score (MOS) and real time analysis could be considered in future work.

The current video traces do not account for audio content and background traffic. Incorporating audio data and background traffic would make the model more realistic. This study could act as a basic model upon which the above said future enhancements could be implemented.

REFERENCES

[1] Matilda.S, B.Palaniappan, March 2010, “Cross Layered Hybrid Transport Layer Protocol Approach To Enhance Network Utilisation For Video Traffic,” ICTACT Journal on Communication Technology, Vol.1, No.1, pp. 54-60.

[2] Yashar Ganjali Gavgani, March 2007, “Buffer Sizing in Internet Routers” Ph.D dissertation Stanford University, California.

[3] Konstantin Avrachenkov, Urtzi Ayesta, Alexei Piunovskiy 2005, “Optimal choice of the buffer size in the Internet routers”, Proceedings of IEEE Conference on Decision and Control, Also in European Control Conference ECC 2005 (CDC-ECC'05), Seville, Spain, pp.1143-1148.

[4] Ravi S. Prasad, Constantine Dovrolis Marina Thottan, 2007, “Router Buffer Sizing Revisited: The Role of the Output/Input Capacity Ratio” ACM proceedings of CoNEXT'07, New York, U.S.A.

[5] K.Avrachenkova, U.Ayesta, A.Piunovskiy, February 2010,“Convergence of trajectories and optimal buffer sizing for AIMD congestion control” ACM Transactions on Computer Communications, Vol. 33, No. 2, pp.149-159.

[6] Annanda Th. Rath, 2006, “A Hybrid Streaming Mechanism for Delay- Tolerant Multimedia Applications”, Masters Dissertation, IIT Bombay.

[7] Project kryonet by S.Nathan “TCP and UDP client/server library for Java” Internet Available: http://code.google.com/p/kryonet

[8] S.Krishnamachari, Mihaela van der Schaar, Sunghyun Choi, Xiaofeng Xu, April 2003, “Video Streaming over Wireless LANs: A Cross-layer Approach”, Proceedings of IEEE Packet Video 2003, Nantes, France.

[9] H.Schulzrinne, A.Rao, R.Lanphier, April 1998, "Real time Streaming protocol", RFC 2326, Internet Available: http://www.ietf.org/rfc/rfc02326.txt

[10] Matthew Syme and Philip Goldie, 2004, "Optimizing Network Performance with Content Switching: Server, Firewall and Cache Load Balancing – Understanding Application Layer Protocols", Prentice Hall, Part of the Radia Perlman Series in Computer Networking and Security series.

[11] "TCP and UDP Socket Interface" TCP/IP Manual, Vol.1, Chapter.3 -Internet Available:

http://www.rabbit.com/documentation/docs/manuals/tcpip/usersmanualv1/sockets.html

[12] Sachin Agarwal, April 2010, "Video Quality Analysis - Part 1" - Internet Available: http://sachinkagarwal/home/code-snippets/

[13] C. Villamizar, C. Song, November 1994, "High Performance TCP in the ANSNET", ACM SIGCOMM Computer Communication Review, Vol. 24, No. 5, pp.45–60.

[14] G. Appenzeller, I. Keslassy , N.McKeown, October 2004, "Sizing Router Buffers", ACM SIGCOMM '04, Portland, Oregon, September 2004. Also in Computer Communication Review, Vol. 34, No. 4, pp. 281–292.

[15] M.Enachescu, Y.Ganjali, A.Goel, N.McKeown, and T.Roughgarden, April 2006, "Routers with very small buffers" Proceedings of INFOCOM 2006, 25th IEEE International Conference on Computer Communications. pp.1-11.

[16] W.John , M.Dusi, K.C.Claffy, 2010, "Estimating Routing Symmetry on Single Links by Passive Flow measurements", IWCMC '10 Proceedings of the 6th International Wireless Communications and Mobile Computing Conference, Caen, France.

[17] Robert A. Van Valzah, Todd L. Montgomery, E.Bowden, July 2010, "Topics in High Performance Messaging", Internet Available: http://www.29west.com/docs/

[18] "OPNET version 14.5, Internet Available:

[19] http:// www.opnet.com

[20] F.Fitzek, P.Seeling, M.Reisslein, November 2002, "H.26L Pre-Standard Evaluation," Technical Report, ACTICOM–Mobile Networks, Germany.

[21] M. Reisslein, F. Fitzek, et. al., 2002, "Traffic and Quality Characterizaton of Scalable Encoded Video: A Large–Scale Trace–Based Study," Technical Report, Telecommunications Research Center, Arizona State University.