In-Network Scalable Video Adaption Using Big Packet Protocol

Stuart Clayman
University College London
London, UK
s.clayman@ucl.ac.uk

Müge Sayıt
Ege University
Izmir, Turkey
muge.sayit@ege.edu.tr

ABSTRACT

The essence of this work is to show how SVC Scalable Video can be adaptated in the network in an effective way, when the Big Packet Protocol (BPP) is used. This demo shows the advantages of BPP, which is a recently proposed transport protocol devised for real-time applications. We will show that in-network adaption can be provided using this new protocol. We show how a network node can change the packets during their transmission, but still present a very usable video stream to the client. The preliminary results show that BPP is a good alternative transport for video transmission.

CCS CONCEPTS

• **Information systems → Multimedia streaming**.

KEYWORDS

In-network Adaption, Scalable video, Big Packet Protocol

ACM Reference Format:
Stuart Clayman and Müge Sayıt. 2021. In-Network Scalable Video Adaption Using Big Packet Protocol. In *12th ACM Multimedia Systems Conference (MMSys 21), September 28–October 1, 2021, Istanbul, Turkey.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3458305.3478440

1 INTRODUCTION

This work presents a proof-of-concept for mapping SVC video into a packet structure compatible with BPP. Big Packet Protocol (BPP) is one of a number of new network protocols designed for the needs of future network architectures and new styles of applications. The main objective of BPP is to provide a framework which meets the requirements of high precision services, where those applications have specific service level guarantees.

A design goal of BPP is to define and implement application specific networking behavior, at the level of individual packet or flow, by utilizing functionality built into enhanced network devices [7]. An important aspect to consider is that a BPP-aware network node, given specific commands, can drop parts of the payload. To support this, BPP packets have nominated fields providing meta-information for signaling the commands to these network nodes. Also, the BPP payload, rather than being one group of bytes, is partitioned into a header and a set of data chunks. Depending on the contents of the header and the load of the network, some of these chunks can be dropped during network transmission.

With BPP, we do not see whole packets being dropped, rather parts of the packet payload, the chunks, are dropped. The adaptation of the packet size is done by observing the network conditions and by taking into account application metadata. The concept is to reduce the load on the network somewhat, by reducing the consumed bandwidth, yet keeping the flow of packets arriving at the receiver, so there is no stalling. It is designed for high bandwidth applications where retransmission times make resending too slow. So it is more at the UDP end of the spectrum than TCP like. BPP can be considered as a new type of transport layer protocol, which provides a partial reability.

Video transmission is commonly done either using UDP which sends discrete packets but is unreliable, or using TCP which presents data streams, that are reliable. There are advantages and disadvantages to each approach. When using UDP, there is a view of a network pipe that is packet based and presents loss at the receiver. The receiving application has to deal with packet loss in the network. The application has direct control over requests for resends, if they are needed. When using TCP, there is a view of a network pipe that is byte-stream based and has no loss but has delay / latency. The application only sees the stream of bytes, never the packets going over the network, and it has no control over the resend mechanism as this is done in the TCP stack in the operating system. TCP ensures that the packets arrive at the clients, but they can be delayed due to re-transmission, plus the congestion algorithms can limit real-time video interactions, with elongated buffer durations. The receiving application is responsible for dealing with the delay, and is usually done by implementing some buffering techniques.

Scalable Video Coding (SVC) enables video sequences with a number of qualities from a single encoded video file [8]. The frames of the video are encoded with different parameter settings, such that it produces enhanced quality alternatives of the video stream. Layered video coding takes advantage of the similarities between the encoded versions of the same frame, as well as between subsequent frames. However, this type of coding also introduces the dependencies between video layers. Video transmission using layered video has been shown to have a beneficial effects [10].

The majority of the video data that is sent over network is transferred with HTTP or with RTP over UDP. HTTP Adaptive Streaming (HAS) works well in many instances, and it has become one of the most popular and successful video streaming applications nowadays. HAS utilizes the current Internet architecture components, such as web caches, and the reliability of TCP. QUIC [1] has been suggested as another good approach for video transmission over UDP. RTP is a protocol whose structure is based on Application Layer Framing (ALF) [2]. RTP/RTCP protocols provides a packet structure that allows the communicating video streaming

ACM MM Sys, September 28 - October 1, 2021, Istanbul, Turkey

ACM ISBN 978-1-4503-8434-6/21/09…$15.00
https://doi.org/10.1145/3458305.3478440

entitites [5] to share information such as payload type, lost packets, timestamps and sequence numbers.

The protocols that run over TCP and UDP successfully manage the pros and cons of these transport layer protocols and hence meet the application requrements at a high level. As a new alternative transport layer protocol BPP can provide different kind of capabilities that might be useful in different aspects for video streaming applications. When BPP is combined with the ALF principle, its capability of in-network packet processing can be used for adapting quality within the network, hence pave the way for providing less latency and higher network utilization.

The first presention of using BPP with SVC was at ITU Network 2030 in 2019 [9]. BPP packet processing provides advantages to video streaming applications as encoded video consists of video frames, which can be packetized and sent through the network. Lost frames of layered video cause different levels of quality degradation, depending on the characteristics of the lost frames. Since it is possible, with BPP, to modify packets in the network during transmission, by deleting some of the video chunks from the packets, when considering the current network conditions and constraints, it can be beneficial for providing the highest perceived quality on the client side, as video receivers can still play the video even if some of these frames are lost during transmission.

2 VIDEO AND BPP

To make BPP effective for continuous media, BPP needs to be coupled with an encoder and decoder that can do multiple encodings for the same region. These encodings can be put into a packet, and if any of the chunks are dropped, then the receiver will still have data. By using layered video coding, the video file is encoded so that the encoded file contains one base and several enhancement layers. While the base layer provides the lowest video quality and does not require any other layer to be decodable; enhancement layers are dependent to the layers below to be able to be decodable. From a such a layered video, it is possible to extract the different qualities. Layered video coding can be considered compatible with the packet modification characteristic of BPP, allowing the deletion of some part of the packets during transmission. If a BPP packet arrives at a bottleneck during its journey and chunk deletion is necessary, enhancement layers can be removed from the packet. When this packet arrives at the client, the client is still able to play the video.

BPP supports a number of commands, and we utilize the *Packet Wash* command as this enables the elimination of some chunks, during the transmission of the packet [6]. This chunk reduction approach helps to prevent dropping of whole packets, particularly when the bandwidth is limited, as it reduces the size of the packets, thus allowing more packets to flow down a limited connection. In Packet Wash, the chunks which have a low-priority can be dropped if there is congestion, and those with higher priority are kept. Even though the packet payload is not the same as the data sent from the server, this is better than receiving no data or delayed data [6]. Thus, data arriving at the receiver will still provide some usable information, as opposed to a dropout. In our demo, we use the PacketWash command in the BPP packet transmission, as Packet Wash provides an efficient technique for managing streaming video.

2.1 Use of BPP for Video Transmission

Although the information to guide network nodes for selecting which chunk should be removed can be carried in the current BPP packet structure, it is necessary to add more BPP fields to succesfully transfer layered video. Because each packet carries chunks which are belong to different layers, the layers of each frame are partitioned into fragments, so new fields are added to the BPP packet structure to hold the information on fragmentation and source frame number. BPP packets include a Significance value and a Condition value. While the Significance value shows the importance of the chunks, the Condition defines the conditions related to chunk removal operation. With respect to the BPP field extensions specific to this study, the significance values are set by considering the impact of the chunks on QoE. The highest prioritized chunks are fragments from the base quality and temporal layers. The value for the Condition field is determined by the server, and for sending layered video, the server puts the average number of packets that will be transferred within a second. The network nodes decide whether chunks are to be removed, and if necessary, how many chunks should be removed when considering the available bandwidth and the value in Condition field.

2.2 SVC and BPP Experiments

We have successfully taken these concepts and have built a working proof-of-concept system that implements these ideas. The work clearly demonstrates that by using SVC coupled with BPP, and using the PacketWash mechanism in the network node, we are able to implement an *in-network video adaption scheme* which can change the video characteristics, as the data at the server is not always the same as the data at the client, but the video is still playable with good QoE attributes. Preliminary results have been presented in Netsoft 2021 [3], where we show the use of an SDN controller as the network node, to support layered video transmission. The SDN controller utilizes both real-time network condition information and video coding characteristics to manage the video transmission, with a specific view on a high QoE. In HPSR 2021 [4], we discuss the enhancements to BPP needed for SVC. Here we compare the performance of network transport protocols, rather than the application layer framing which is independent of the transport. For example, RTP can easily be carried in a BPP packet, but this work does not focus on those aspects. The results showed that BPP significantly outperforms UDP for PSNR and duration outages.

The experiments highlight that when using BPP and UDP, the clients in both cases continue to receive video layers until the end of the streaming session. For UDP, it was clear that the client could not play the video after a few seconds. The reason that the video could not be played, even when the client received some frames, was that all the main frames providing references to other frames were lost after a certain point, due to bandwdith limitations. Consequently, when having a UDP transport, the network resources used for the transmission of most of the layers arriving at the clients were mainly wasted. When using BPP for layered video transmission, the clients received a continuous stream of playable video and had low levels of outage, even with limited bandwidths. We observed that managing the network by jointly using SDN and BPP provides *higher effective bandwidth utilization*, by not having wasted packets,

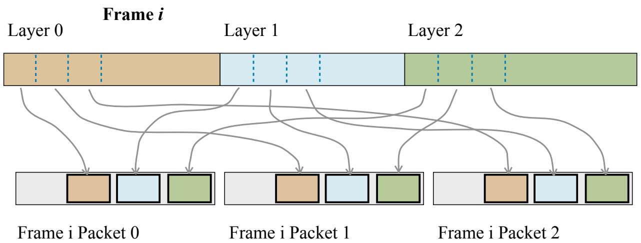


Figure 1: Mapping Layers of a Frame to Packets
Layers are mulitplexed across a number of packets

but by adjusting the quality during transmission and sending the higher layers only if there is enough capacity.

The mechanism by which video frames from the different layers are mapped and placed into BPP packets is shown in Fig. 1. This approach ensures that there is always at least one chunk that can be delivered, even if other chunks are removed in each packet. Hence, the network node can adapt the quality whenever it is needed. The structure of each packet has a BPP header, plus a number of chunks. The header keeps the size of each chunk, and it's offset in the packet, as well as each chunk's significance value, and some commands. The full structure of a BPP packet, with the definition of the main blocks, is defined in [7].

In these experiments, we measured the layers received by the client. The in-network quality adaptation provided by the use of BPP was measured for both fixed and changeable network bandwidths. Here the two scenarios are presented where the bandwidth is dynamically changed. We implemented two different types of tests for dynamic bandwidth changes, with ascending bandwidth, from 0.5 Mbps up to 1.5 Mbps, and descending bandwith values, from 1.5 Mbps down to 0.5 Mbps, adjusted over time. Figures 2, 4, 3, and 5 present the changes in the number of received layers in these experiments. As seen from the BPP graphs, when the network node detects the change in bandwidth, it removes some chunks within the packets so that the quality of the transferred video is adjusted under the contraint of limited capacity.

The distribution of the received frames, for BPP and UDP, is shown in Fig. 6a and Fig. 6b. Packets were categorised into 3 types: Valid Frames, Lost Frames, and Unsuable Frames, and the frames that are shown in the graphs represent the cumulative received layers, for each evaluated bandwidth. BPP has a very stable behaviour, as seen in Fig. 6a. As there is no frame loss with BPP, only chunk removal, there are no unusable frames with BPP. Therefore, BPP also provides higher effective bandwidth utilization, as it only sends the frames that will be played properly at the client. We have observed that UDP aggressively transfers all the layers, and the clients can still receive the highest quality layers, even if the bandwidth is limited. Unfortunately, this approach quickly causes congestion, which in turn, leads to a high levels of loss. As well as

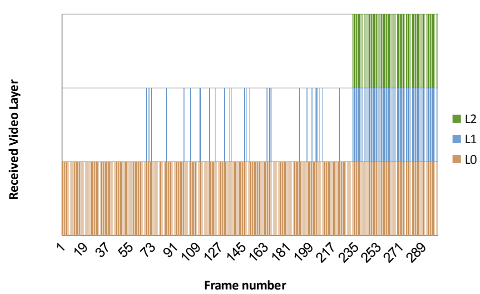


Figure 2: BPP Received Frames: Ascending
Layer 0 always received, Layers 1 and 2 received as bandwidth increases

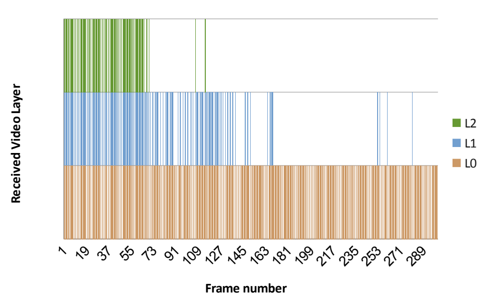


Figure 3: BPP Received Frames: Desccending
Layers 0, 1, 2 received until bandwidth decreases, Layer 0 always received

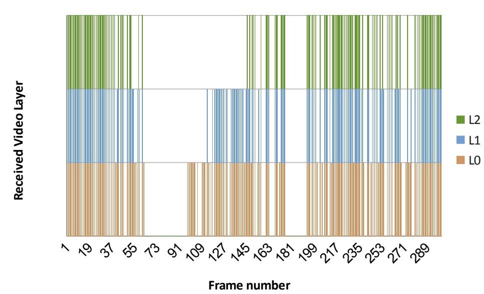


Figure 4: UDP Received Frames: Ascending
Large number of frames received, with big gaps and inconsistent playout

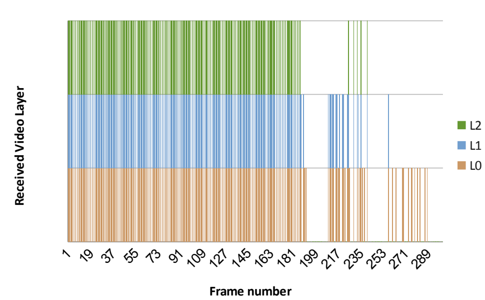
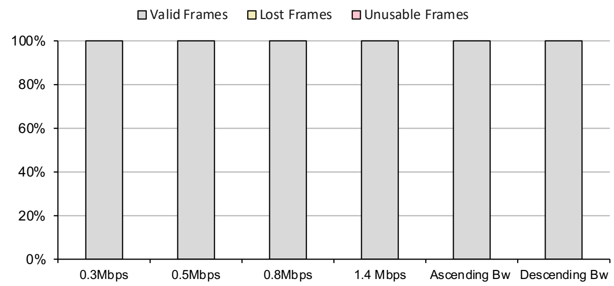
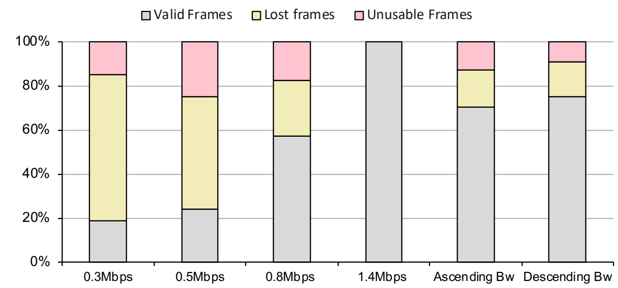


Figure 5: UDP Received Frames: Descending
First 190 frames of all layers received, then significant dropouts



(a) BPP



(b) UDP

Figure 6: Raw distribution of transferred frames

Bandwidth	BPP	UDP	TCP
0.5 Mbps	173	5470	14396
0.8 Mbps	205	2634	5697
1.5 Mbps	0	0	584
Ascending	0	916	2554
Descending	0	920	4691

(a) Duration of outages (in msec)
Smaller is better

Bandwidth	BPP	UDP	TCP
0.5 Mbps	40	29	44
0.8 Mbps	40	36	44
1.5 Mbps	44	44	44
Ascending	39	18	44
Descending	40	37	44

(b) Average PSNR values (in dB)
Larger is better

Bandwidth	BPP	UDP	TCP
0.5 Mbps	10	2.4	10
0.8 Mbps	10	4.2	10
1.5 Mbps	10	10	10
Ascending	10	1.7	10
Descending	10	6.7	10

(c) Video playout on the client side (in seconds)
10 secs is full video

Table 1: Comparing BPP vs UDP vs TCP QoE Metrics

loss with UDP, we observe many unusable frames. These unusable frames represent the frames received by the client that cannot be played properly as the referenced *I* frame has been lost. A further complexity is the loss of *P* frames, which can also affect the other frames. This effect is not considered here, nor shown in the graphs, but will be addressed in future. However, the lost *P* frame effect is less than that of losing *I* frames, in general.

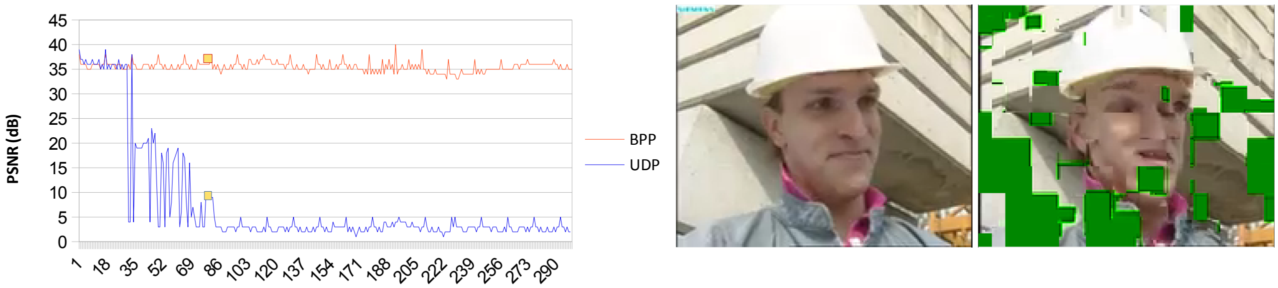
Some QoE metrics that were observed in the experiments from [3] are presented in Table 1. We streamed the *Foreman* video sequence whose base layer, first enhancement layer, and second enhancement layer bitrate equal 204 Kbps, 488 Kbps, and 1094 Kbps, respectively. The duration of the video is 10 secs. The results collected from the experiments that were performed with the fixed bandwidth values are given in the first three lines of the tables. We observed that the packet losses severely affected the received video quality with UDP. The video stopped playing before the session ends. When the bandwidth is limited, i.e. it equals 0.5 Mbps, the UDP client could play the video for only 2.4 secs (as seen in Table 1c) with an outage duration of 5.5 secs (as seen in Table 1a). The PSNR values for UDP decrease down to 18 dB. As well as results for BPP and UDP, these tables also include results for TCP. As expected, the TCP client always received the all layers. However, due to the retransmission of the lost packets when faced with limited bandwidth, the longest duration of outage is observed with TCP. It is clear that if client-side adaptation is used, the duration of outage would decrease. We implemented an additional test to evaluate this behaviour. In that test, the server sent the video with the medium quality, which is the quality that would be selected with respect to the available bandwidth value. Even in this situation, where the quality is selected according to the available bandwidth, the TCP client experienced 734 msec duration of outage. For the same scenario, the BPP client experienced 205 msec duration of outages and played the video with the same PSNR.

While the BPP client plays the video with a stable quality for all frames, the UDP client could only decode and play the video until its 81st frame. Fig. 7a shows the PSNR, as a function of time, with BPP in red and UDP in blue, when the bandwidth equals 0.5 Mbps. We see that PSNR for BPP is stable and around 35 dB, while for UDP, it drops suddenly after 81st frame is received. This is due to the buffer on the router being filled, and packets then being dropped. The screenshots which show the 81st frame, marked with a yellow square in Fig. 7a, is given in Fig. 7b and 7c. In these images, showing the same frames that BPP and UDP clients play are displayed. The effect of missing reference frames can clearly be seen from the screenshot of the UDP client.

We observe that using SVC and BPP aware network nodes provides *higher effective bandwidth utilization* and *in-network quality adaptation* by adjusting the video content during transmission, and sending the higher layers only if there is enough capacity.

3 DEMONSTRATION

This demonstration will directly present the elements of the proof-of-concept system, and highlight the in-network adaption capabilities that can be achieved when using SVC combined with BPP. We will also show the effects of using UDP. We compare BPP with



(a) Comparative PSNR results (Fixed bandwidth: 0.5 Mbps)

(b) Frame 81: The screenshot from the BPP client's video

(c) Frame 81: The screenshot from the UDP client's video

Figure 7: Perceptual Quality: BPP vs UDP

UDP, as it is also a packet-at-a-time mechanism, with no built-in retransmission or congestion control.

The topology and the system configuration that will be used in the scenarios is given in Fig. 8 for the BPP setup, and Fig. 9 for the UDP setup. In the demonstration, the server and the client will be connected through a virtualized network with soft routers. With the BPP Packet Processing Path, the network nodes will recognize the BPP header and implement the chunk removal policy by considering the information in the BPP header fields and the available bandwidth. We will show that the packet size changes in the face of congestion, as chunks can be dropped, but video data will still arrive at the client. With the UDP Packet Processing Path, the network nodes will drop packets in the face of congestion, and in that case the will be loss of video, and dropout at the client. In both cases, the server sends the video with the highest quality.

In the BPP scenario, the *BPP aware video process* at the server, in Fig. 8, takes an H264 SVC video stream and constructs the packets by taking some bits from each of the layers of a video frame, and multiplexing them into a sequence of packets, as highlighted in Fig. 1. For our demonstration, and for our experiments, we utilize SVC videos with a base layer (L0), and two enhancement layers (L1 and L2). As a consequence, each BPP packet contains 3 individual chunks, one for L0, L1, and L2. As the base layer (L0) is considered more important than the enhancement layers (L1 and L2), the significance value for those L0 chunks is set higher than than those chunks for L1 and L2. Also, the commands are configured so that the network node, can eliminate L1 and L2 chunks, but never remove L0. The client will reconstruct the video stream from the incoming packets, and dealing with the chunks for the layers in each of the packets. In the UDP scenario, *UDP aware video process* at the server, takes the same H264 video and fills the payload of the packets with a number of bytes from each of the video frames. Again, the client will reconstruct a video, but it will just have a sequence of bytes in the packet.

The demonstration will show the effects on the video quality on the client side, with various bandwidth values, some of which are fixed bandwidth and some of them are dynamic. When BPP is used, if the bandwidth is not adequate, the network nodes may shrink the packet size, as Fig. 8 illustrates, but the same situation causes packet loss when UDP is used. These effects will be shown in real-time, highlighting how the BPP-aware video process in the network node updates the packets, for the different bandwidths. We will show how the system dynamically adapts as the available bandwidth changes. More specifically, we will present the number of received layers and the PSNR of the received video for both protocols. On the client side, the successfully received layers will be also shown in real-time for both BPP and UDP transmission. In addition, the received videos will be shown after decoding is completed. We plan to stream several video with various characteristics, and to show the comparative performance for different bandwdith settings.

4 CONCLUSIONS

In this study, we have shown an approach for providing in-network quality adaption for video streaming systems and presented a system that performs scalable video streaming over BPP, while highlighting that SVC and BPP are compatible.

Video streaming systems have been successfully working for many years, by providing the highest possible QoE under the constraints of available network resources. The characteristics of the underlying transport layer protocols caused streaming systems to shape their policy, and the QoE is also affected by the transport layer protocols. Although TCP and UDP provide different advantages to the video transmission, their limitations might severely affect QoE. As a newly proposed transport protocol, which was designed to eliminate the limitations of current IP, BPP has the potential to provide good transport layer support with its own features, just as TCP has reliability and UDP has fast transmission.

This demo shows the advantages provided by a new transport layer protocol, as well as opening a platform for discussing: how the performance of video streaming applications over BPP can be increased, what additional mechanisms would be needed or in which type of applications what kind of BPP policy would be implemented. In this study, we aim to provide seamless streaming with minimum duration of outages possible under the available bandwidth. With a layered encoder, each packet contains all of the encodings from the frame, from the base layer, upwards. The BPP aware network node can drop chunks if necessary, but the packets will always have the base layer encoding, ensuring continuous video delivery. As the decoder applies any higher encodings onto

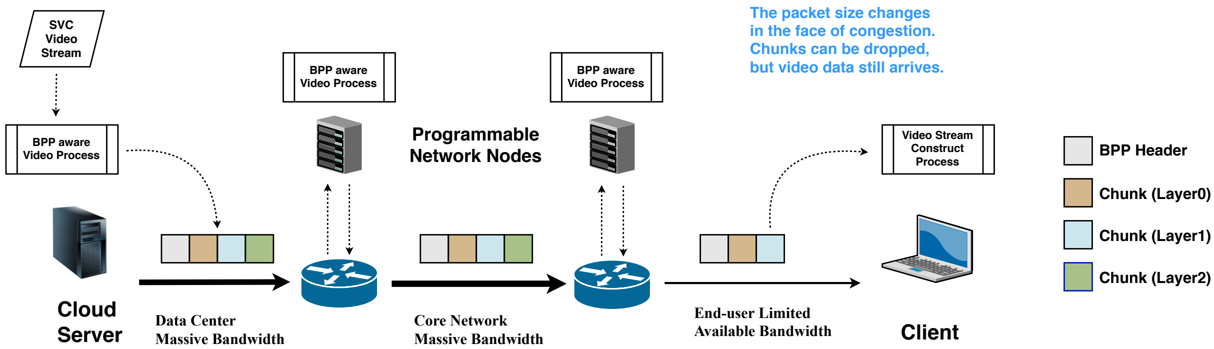


Figure 8: BPP Packet Processing Path

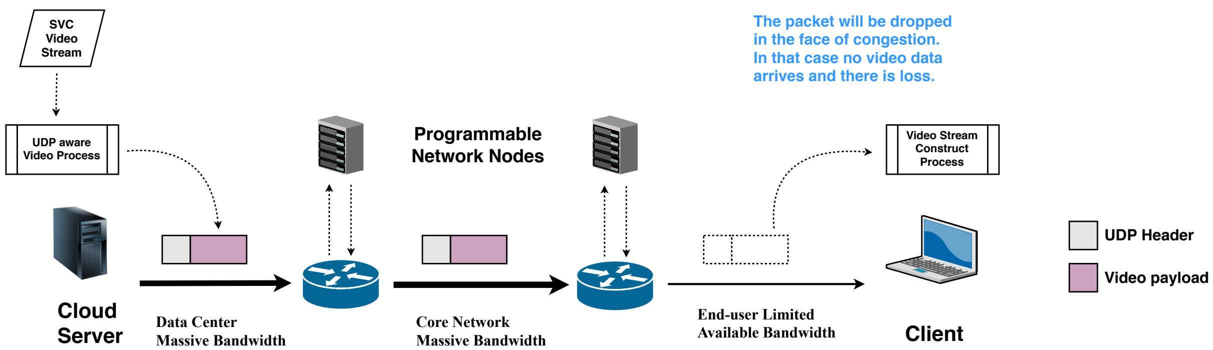


Figure 9: UDP Packet Processing Path

the base layer, the receiver always gets a packet with some usable video data.

The results have also shown that even when there are no approaches used on the client at the application layer, such as client-side quality adaptation, using BPP and SVC with a suitable chunk removal approach provides higher QoE. In addition, network resources were used efficiently, as network nodes transfered the layers that can be used by the client.

For a full working system we need to connect an SVC encoder directly to a BPP packet constructor in the server. At the client end we need a live decoder that can read BPP packets from the network, and recreate an SVC stream, which can be fed to a real-time decoder. We intend to look at the latest H.266/VVC encoders for this.

There is still room for improving the performance, as the network nodes can keep track of the previously deleted chunks, and if a chunk of a frame is removed, then all other chunks belonging to the same frame in the next packet can also be deleted. This is further work. As other enhancements, it is possible to develop different chunk removal policies and different packetization techniques for providing various service features.

We plan to develop a chunk removal approach by considering the affects of different layers on quality. We also plan to evaluate the use of RTP carried over a BPP transport, and compare the performance with RTP over UDP and HAS based systems. Finally, we will extend the QoE evaluations by using VMAF method.

REFERENCES

[1] Mike Bishop. 2021. *Hypertext Transfer Protocol Version 3 (HTTP/3)*. Internet-Draft draft-ietf-quic-http-34. Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/draft-ietf-quic-http-34

[2] David D. Clark and David L. Tennenhouse. 1990. Architectural Considerations for a New Generation of Protocols *(SIGCOMM '90)*. Association for Computing Machinery, New York, NY, USA, 200–208.

[3] Stuart Clayman, Mustafa Tuker, Halil Arasan, and Müge Sayıt. 2021. Managing Video Processing and Delivery using Big Packet Protocol with SDN Controllers. In *IEEE Conference on Network Softwarization – Netsoft*. Tokyo, Japan.

[4] Stuart Clayman, Mustafa Tuker, Halil Arasan, and Müge Sayıt. 2021. The Future of Media Streaming Systems: Transferring Video over New IP. In *IEEE 22nd Intl. Conf. on High Performance Switching and Routing (HPSR)*. Paris, France.

[5] Jon Crowcroft, Mark Handley, and Ian Wakeman. 1999. *Internetworking Multimedia*. CRC Press. 1–290 pages.

[6] Richard Li et al. 2019. A Framework for Qualitative Communications Using Big Packet Protocol. In *NEAT 2019: Proceedings of the ACM 2019 Workshop on Networking for Emerging Applications and Technologies*. 22–28.

[7] Richard Li, Alexander Clemm, Uma Chunduri, Lijun Dong, and Kiran Makhijani. 2018. A new framework and protocol for future networking applications. In *NEAT 2018 - Proceedings of the ACM 2018 Workshop on Networking for Emerging Applications and Technologies*.

[8] Heiko Schwarz, Detlev Marpe, and Thomas Wiegand. 2007. Overview of the Scalable Video Coding Extension of the H.264/AVC Standard. *IEEE Transactions on Circuits and Systems for Video Technology* 17, 9 (October 2007), 1103–1120.

[9] Stuart Clayman. 2019. The Inter-Dependence of Network Transport and Application Behaviour. In *ITU Network 2030*. Geneva. https://www.itu.int/en/ITU-T/Workshops-and-Seminars/2019101416/Documents/Stuart_Clayman_Presentation.pdf

[10] Su Yang, Zhang Wei, and Liu Ling. 2011. Multimedia Communication and Scalable Video Coding. In *Fourth Intl Conf on Intelligent Computation Technology and Automation*. Shenzhen, China, 616–619.