©ISTOCKPHOTO.COM/SCANRAIL

Si Yan, Peng Yang, Qiang Ye, Weihua Zhuang, Xuemin (Sherman) Shen, Xu Li, and Jaya Rao

Transmission Protocol Customization for Network Slicing

A Case Study of Video Streaming

In this article, we propose a software-defined networking (SDN)-based transmission protocol (SDTP) for future-generation networks. Different services are supported by dedicated network slices, each of which can be operated separately with a customized transmission protocol. In particular, we focus on video streaming services supported by in-network caching functions. By exploiting the flexibility of scalable video coding (SVC) and SDN control intelligence, we present an in-network bottleneck queue management strategy in conjunction with a novel selective caching policy (SCP) for congestion detection and mitigation. Additionally, an enhanced transmission (ET) scheme is devised to improve the video user experience

Digital Object Identifier 10.1109/MVT.2019.2936087
Date of current version: 14 October 2019

by opportunistically requesting cached video packets when network conditions permit. The proposed protocol can adapt to traffic dynamics and varying service requirements, and it is shown to effectively alleviate network congestion and provide a balanced user experience. Extensive simulation results are presented to validate the proposed protocol in terms of in-network queue stability and user-perceived video quality.

Background

Future-generation networks are expected to accommodate various applications with diversified quality-of-service (QoS) requirements. To this end, more and more network infrastructures, including base stations, edge switches, and middleboxes, are being deployed with significantly increased capital and operational expenditures. In contrast, the one-size-fits-all transmission control protocol (TCP) still dominates the transport layer. Without fine-grained packet differentiation, it is difficult to achieve service customization over TCP in a cost-effective manner. To improve transport-layer performance, different TCP variants (e.g., BBR [1] and CUBIC [2] with new congestion control) along with new transport protocols (e.g., QUIC [3]) have all been developed for different objectives. However, those protocols keep the end-to-end (E2E) semantics inherited from the TCP, which restricts the utilization of in-network information for decision making.

SDN has emerged as a promising architecture for flexible and agile network operation [4]. By decoupling control functions from substrate nodes, SDN enables informed network control and function reconfiguration on programmable switches. Because of the availability of global network information from control channels, fine-grained decision making can be realized to improve network performance. For instance, it is possible for an SDN controller to establish global optimal routing paths for different traffic flows, which facilitates load balancing and directly eliminates congestion. Moreover, the SDN paradigm can be exploited to improve transport-layer performance. Current TCP variants rely on acknowledgments (ACKs) and round-trip time (RTT) estimation for packet loss detection and congestion control. Such a method takes at least one RTT to react to network congestion. To improve responsiveness, active queue management (AQM) is proposed to notify the sender at the onset of congestion [5].

With SDN, real-time loss detection and fast congestion mitigation can be made possible based on overall network knowledge [6]. An SDN-based user datagram protocol framework is also proposed to reduce transmission signaling overhead [7]. Transmission control is no longer restricted to estimated network states at end hosts, as such information can be collected in the network for proactive decision making. In addition, traditional congestion control algorithms act on all data packets identically, e.g., AQM marks all incoming packets with the same probability.

SDN HAS EMERGED AS A PROMISING ARCHITECTURE FOR FLEXIBLE AND AGILE NETWORK OPERATION.

This is due to the dearth of application-level information for traffic differentiation. For services that are packet-loss tolerant (e.g., on-demand video streaming), certain packets can be dropped for congestion resolution with little impact on application-level performance. It is therefore critical to investigate how to properly react to congestion with differentiated QoS provisioning toward service-oriented data transmission.

In this article, we present a comprehensive SDTP to support differentiated data packet delivery. SDN offers such flexibility through in-network control and informed decision making. In particular, we investigate what in-network information and function modules can be used to support service-oriented transport-layer operations. Considering the prevalence of video services, we choose video streaming as an example to illustrate the design principles of SDTP, including its protocol functionalities and operation procedures. Specifically, a joint bottleneck queue management policy and SCP with ET are proposed to achieve fast in-network congestion detection and mitigation, while minimizing the impact on user experience. We leverage the SVC codec [8], by which raw video contents are encoded in various interdependent layers. The enhancement layers at the bottleneck node can be selectively cached in case of network congestion, without severe degradation of video playback experience. To further improve the received video quality, an ET policy is designed to obtain the in-network cached packets if the network condition improves.

System Model and Protocol Function Modules

Network Topology

Consider an embedded virtual network, where dedicated network resources (e.g., caching, processing, and transmission resources) are assigned to each service in the form of network slices [9], [10]. Different service slices that support the E2E packet delivery of different applications are well isolated, which facilitates service-oriented QoS provisioning [11]. Figure 1 shows the topology of two video slices and one mission-critical service slice embedded on a substrate network. For video slice 1, packets from the video server of slice 1 are first aggregated by edge switch $X1$ and are then delivered to edge switch $Y1$ along the routing path specified by a virtual network embedding algorithm. To ensure compatibility with current transport protocols between end hosts, we assume that the connections between end hosts and edge switches are running TCP. The proposed SDTP is operated over the embedded topology between edge switches of a certain slice. Edge

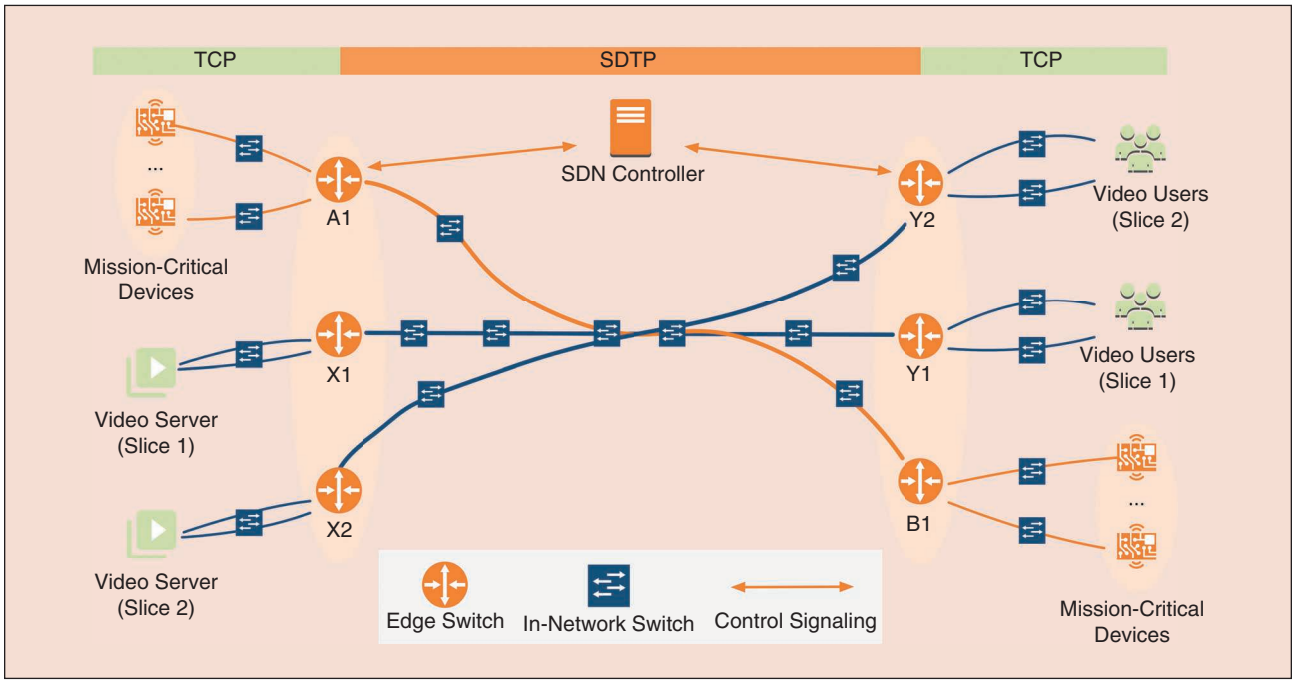


FIGURE 1 A network topology with multiple service slices.

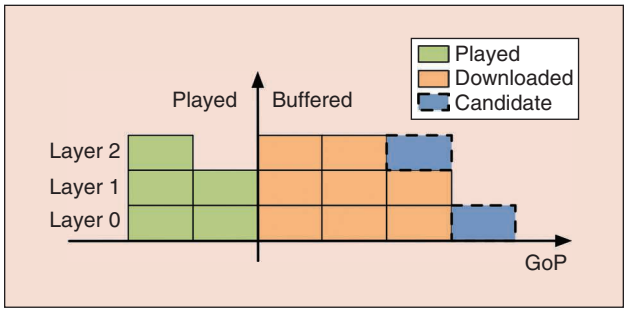


FIGURE 2 A snapshot of user buffer occupation of SVC video content.

switches are responsible for traffic aggregation and protocol translation.

Video Traffic Model

Consider on-demand video streaming services from a video server to video users. First, each video clip is segmented into a sequence of group of pictures (GoP). Each GoP is with a fixed playback time (typically a few seconds [12]) that can be decoded independently. Video contents are coded by an SVC encoder, which encodes one GoP into several interdependent layers, including one base layer (layer 0) and multiple enhancement layers (layer 1 and above). Figure 2 shows a snapshot of the user buffer occupation. Different GoP layers can be stored and streamed separately. Typically, the data volume of an enhancement layer is higher than that of a base layer. To decode a GoP from the user buffer, layer 0 is necessary, while enhancement layers are optional. However, enhancement layers can only be decoded in the presence of all the lower layers. For example, decoding layer 2 requires both layer 1 and layer 0. In the absence of a base layer, rebuffering events will happen at the destination node. The more enhancement layers there are, the higher the video quality.

Currently, dynamic adaptive streaming over HTTP (DASH) is the prevailing video streaming protocol [8]. It operates over TCP, which requires that all the packets be reliably delivered to users. In case of congestion, video packets experience frequent retransmissions, which leads to long E2E delay and frequent rebuffering events. The flexibility of the SVC codec offers the opportunity to balance video quality and rebuffering events. Users request video contents on a layer basis. Upon receipt of one layer, receivers decide the next requested layer (the candidate layers in Figure 2) based on the packets in the buffer and the estimated network throughput.

Protocol Functionalities

To support video services, SDTP incorporates the following functionalities: bottleneck queue management, selective caching, and ET. A new packet header format is designed to support data transmission with the protocol function modules and, at the same time, achieve backward compatibility.

A network bottleneck node (Figure 3) has a packet forwarding queue and a caching buffer, with the latter supported by in-network caching resources. The key issue is how to selectively put packets into the caching buffer in case of network congestion. Once the network condition improves, packets that help enhance user experience can be retrieved from the caching buffer for ET. In this way, network congestion can be alleviated with less packet dropping.

Bottleneck Queue Management

In conjunction with congestion control algorithms, existing AQM algorithms focus on generating congestion signals based on queue-size information. As shown in Figure 3, however, the queue dynamics on an embedded network node are influenced by packet arrival, packet forwarding, selective caching, and ET. Due to the in-network intelligence of SDN controllers, such complicated network information can be collected and analyzed for decision making, thus facilitating the fast and coordinated identification of bottleneck nodes of different traffic flows on various routing paths.

Selective Caching

The SVC codec enables flexible video decoding, and video contents can be successfully decoded, even in the absence of enhancement layer packets. Hence, higher-layer packets can be selectively cached in the network, without significantly degrading user experience. By exploiting the caching resources, instead of dropping packets when the network is congested, we designed an SCP that temporarily stores certain packets on network nodes, which provides a fast response to network congestion.

ET

To enable the continued transmission of cached packets once network conditions improve, we designed an ET module that processes ET requests. Considering the impact on the forwarding queue, not all ET requests will be satisfied. Hence, a fine-grained ET scheduling scheme is necessary to strike such a tradeoff.

Packet Forwarding and Bottleneck Queue Management

In this section, based on the embedded virtual network topology, we first describe the transmission process with a new packet header format, which enables packet differentiation and protocol compatibility, followed by description and analysis of the queue management module.

Video Data Transmission

Video contents are stored in the video server with different representations [13], including by the codec, encoding bit rate, resolution, and so on. A structured collection of video representations is called the *media presentation description* (*MPD*) file. Each video representation is further divided into consecutive segments, referred to as *encoded GoP layers* in the SVC codec. The receivers can choose from and switch to different representations based on the estimated data rate. In particular, after connection is established, the server sends the MPD file to the receiver. Then, based on the estimated data rate, the receiver chooses one representation and responds to the video server by sending an HTTP-GET message. Upon the receipt of one GoP layer, the receiver sends another HTTP-GET message to request the next. As shown in Figure 1, to be compatible with the current TCP, SDTP is applied between the edge switches. For packet transmissions between end nodes and edge switches, current TCP is employed. Therefore, the SDTP protocol can be implemented without requiring any changes to the end transceivers; however, the header format of TCP packets is not supported by switches that run SDTP. Hence, header conversion and reversion should be enabled at the edge switches. When SDN devices become prevalent, the proposed protocol can be applied to the entire network.

There are four types of SDTP packets for delivering video contents, i.e., the MPD data packet, the HTTP-GET request packet, the video data packet, and the ET request packet. Figure 4 shows the proposed SDTP header format and ET request packet encapsulation. After conversion, the source/destination address of the original IP header is replaced by the edge switch source/destination address.

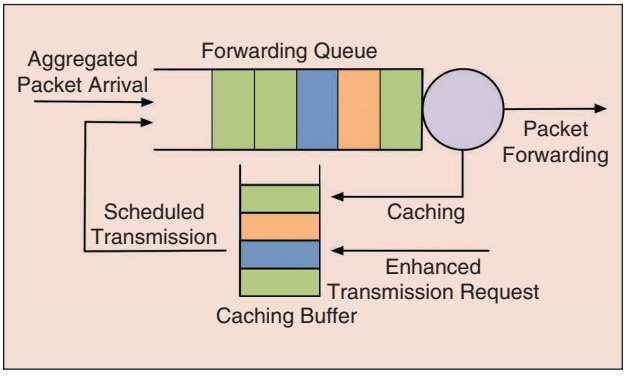


FIGURE 3 The dynamics of the bottleneck queue.

	1–8 b	9–16 b	17–24 b	25–32 b
1	Protocol	Total Length		Data Offset
2	Checksum		Flag	
3	Edge Switch Source Address			
4	Edge Switch Destination Address			
5	Edge Switch Source Port		Edge Switch Destination Port	
6	Slice-Level Sequence Number			
7–9	Connection ID			
10	(Requested) GoP Number			
11	(Requested) Layer Number			

FIGURE 4 The SDTP video data packet/ET request packet header format.

THE PROPOSED SDTP IS OPERATED OVER THE EMBEDDED TOPOLOGY BETWEEN EDGE SWITCHES OF A CERTAIN SLICE.

Because of the virtual network embedding process, the routing path of data packets is determined, and the network layer information can be significantly simplified in the proposed header. Furthermore, new fields (i.e., the gray cells in Figure 4) are added to support SDTP, including Protocol, Flag, and other optional fields. The Protocol field indicates the currently applied protocol, the Flag field differentiates among the four aforementioned types of packets, and the optional fields are reserved to support protocol functionality extensions.

In particular, each service slice has a unique combination of edge switch addresses and port numbers (as shown in the red dashed block), which can be used as a slice ID for service differentiation. For video data packets, the 20-byte optional field consists of a 12-byte connection ID, a 4-byte GoP number, and a 4-byte layer number. The connection ID contains the IP addresses and port numbers of the server and receivers. The GoP and layer numbers of a video data packet are extracted from the application layer payload by the sending edge switch and are then added to the optional fields to support selective caching at bottleneck nodes for congestion mitigation. Upon the receipt of ET requests, the corresponding requested enhancement layer information is extracted by the edge switches. This information is then added to the optional fields of the ET request packet header.

Bottleneck Queue Management

We focus on the bottleneck link along the routing path between edge switches, as the influence of nonbottleneck links on the E2E delay and throughput performance is insignificant. Relying on the control channel, the SDN controller can accurately identify the bottleneck link by periodically monitoring the instantaneous queue length on all of the switches. The objective of queue management is to stabilize the queue length against traffic burstiness for balanced E2E delay and throughput.

Queue Monitoring

Similar to conventional, random early-detection schemes [14], the proposed SDTP manages the bottleneck queue by the following iterative phases.

First, to filter out the impact of transient traffic burstiness, the average queue length $q_{\text{avg}}(t)$ at the bottleneck node is obtained by periodically sampling the instantaneous queue length $q(t)$. Specifically, let Δ be the sampling period. After the kth sampling, $q_{\text{avg}}[(k+1)\Delta]$ can be expressed as the sum of $(1-\alpha)q_{\text{avg}}[k\Delta]$ and $\alpha q[k\Delta]$, where $\alpha \in (0,1)$ is the weighting parameter for queue averaging. From sampling theory, the previously mentioned discrete-form relationship can be expressed by the differential equation [14]

$$\dot{q}_{\text{avg}}(t) = -Kq_{\text{avg}}(t) + Kq(t), \tag{1}$$

where $K = -(\ln(1-\alpha)/\Delta)$. By using a Laplace transform on both sides of this differential equation, the transfer function of average queue length is obtained as $1/(1+(s/K))$, which is a low-pass filter that tracks queue length and effectively filters out traffic burstiness. Specifically, s is the frequency domain variable, and K determines the system responsiveness, i.e., the higher the value of K, the faster it reacts to traffic fluctuations. However, an excessively high value of K may result in queue-length oscillations [14]. Tuning parameters Δ and α rely on network information collected by the SDN controller, including the number of connections, bottleneck capacity, and RTT, so as to drive the system into the stable region.

Secondly, the average queue length $q_{\text{avg}}(t)$ is compared with two thresholds, $\underline{q}$ and $\bar{q}$, to obtain the packet caching ratio $p(t)$. When $q_{\text{avg}}(t)$ is smaller than $\underline{q}$, $p(t)=0$, and all of the packets remain in the forwarding queue for transmission. If the value of $q_{\text{avg}}(t)$ is between $\underline{q}$ and $\bar{q}$, the packet caching ratio is $p(t) = (q_{\text{avg}}(t) - \underline{q})/(\bar{q} - \underline{q})$, and, once $q_{\text{avg}}(t)$ exceeds the maximum threshold $\bar{q}$, the packets inside the queue are cached with the highest ratio, i.e., $p_{\max}$.

Finally, the determined ratio, $p(t)$, is enforced in the selective caching phase. A portion, $p(t)$, of the packets inside the forwarding queue are temporarily cached at the bottleneck node based on the policy elaborated on in the following section. By the end of this period, the instantaneous queue length, $q(t)$, is monitored for the update of $q_{\text{avg}}(t)$ in the next period.

Essentially, periodical queue averaging gives a discrete-time system. During each sampling period Δ, video packets from different flows can be differentiated according to application-level GoPs and the layering information carried in the packet header, which facilitates selective packet caching. Based on sampled data system theory, a differential equation can exactly characterize the discrete system at the sample points without accumulating errors as sampling proceeds [14]. It enables us to characterize the discrete-time system using a continuous-time fluid model.

Queue Dynamics Analysis

The dynamics of the bottleneck queue length, as shown in Figure 3, include traffic arrival, traffic departure, packet caching, and scheduled ET. Classic fluid models can be used to characterize queue dynamics [14]. Because both ends are running current TCP, which employs the additive increase and multiplicative decrease strategy for rate control, the resultant sending rate dynamics have been well investigated (e.g., in [5] and [14]).

The bottleneck queue dynamics depend on both the sending window dynamics and the packet caching probability $p(t)$. In particular, $p(t)$ is chosen to serve the following queue management objectives: 1) maximizing network resource utilization, 2) regulating E2E latency, and 3) being robust to traffic burstiness. Intuitively, maintaining bottleneck queue length at a lower level will reduce E2E latency; however, a short queue length may result in buffer emptiness and, hence, the underutilization of link resources. Thus, $p(t)$ should be carefully determined by taking all factors into account.

Model Linearization and Queue Stabilization

The bottleneck queue has a nonlinear relationship with the sending rate, making it difficult to directly obtain the queue stability condition and system equilibrium. A feasible solution is to linearize the differential model around the equilibrium and then characterize the sending rate and queue dynamics in the frequency domain by applying a Laplace transform. Collectively, the source rate control, bottleneck queue dynamics, and queue management form a control loop, where the queue management module specifies how packet caching probability should vary according to queue length.

Selective Caching and ET

Given the caching ratio from the queue management module, we propose an SCP to alleviate network congestion, while considering user quality of experience (QoE). We also propose an ET strategy that consists of transmission scheduling and caching buffer management, for use after the network condition improves.

Selective Caching Policy

An SCP design takes into account the following conditions. First, the enhancement layer of the SVC codec can be decoded only in the presence of all the lower layers. Secondly, according to the DASH protocol, the receivers will not request the next GoP layer until all the packets of the current layer have been successfully downloaded [15]. Therefore, for each video session, there is always only one layer being streamed along the path. In addition, different GoP layers have different data volumes depending on their video contents and codec configurations. Specifically, the SCP includes the following strategies:

- Packets from the base layer of each GoP are not cached because the volume of a base layer is relatively smaller compared to that of the enhancement layers. Further, as they are essential to GoP playback, holding such packets at a caching node will lead directly to rebuffering events.
- To cache with a minimum number of affected video sessions, the packets are cached layer by layer. This is enabled by the GoP and layer information in the designed packet header, based on which packets from the same GoP layer can be identified and cached accordingly.
- The enhancement layers are first sorted in descending order according to their data volume. Then, packets from those layers are cached sequentially until the number of cached packets meets the caching ratio.

For the enhancement layers in certain GoPs, some packets may have already been forwarded to downstream nodes, while others are cached at the bottleneck node or queued at upstream nodes. In this case, all subsequent packets that belong to the selected layers are pushed directly into the caching buffer upon their arrival.

THE CONNECTION ID CONTAINS THE IP ADDRESSES AND PORT NUMBERS OF THE SERVER AND RECEIVERS.

ET

After selective caching, the enhancement layers in certain GoPs may be (partially) cached at nodes along the E2E path. Once the network condition improves, ET requests can be generated by receivers to obtain the cached packets. Thus, better QoE can be achieved at a lower cost compared with direct retransmission from the video server.

ET Request Generation

As shown in Figure 2, after successfully downloading one GoP layer, the receivers send an HTTP-GET request for the next GoP layer, which can be the enhancement layer of buffered GoPs or the base layer of the next GoP. For a partially received enhancement layer, if the estimated downloading time is less than the residual playback time of the corresponding GoP, an ET request is generated for the full receipt of that enhancement layer.

When an ET request arrives at the edge switch, its packet header is converted to the proposed format, as shown in Figure 4. Then, the request is routed toward the sending edge switch for several reasons. First, one GoP layer may be cached by multiple nodes; therefore, the ET request must be processed by all of the related nodes to ensure full receipt of the requested data. Second, the ET request can also be regarded as a caching release message, because the packets that belong to preceding GoPs will no longer be requested by the video user.

ET Scheduling

Upon receipt of an ET request, the caching node moves the corresponding packets (referred to as *ET packets*) from the buffer to the forwarding queue. The transmission of ET packets and normal packets will then be scheduled. It is crucial to optimize the allocation of bottleneck forwarding capacity, as it determines both the video

INTUITIVELY, MAINTAINING BOTTLENECK QUEUE LENGTH AT A LOWER LEVEL WILL REDUCE E2E LATENCY.

quality of buffered GoPs and the playback smoothness of future GoPs.

In general, normal packets are of higher priority because they contain base-layer information, while ET packets are exclusive to the enhancement layers. ET requests received in the previous time slot are processed on a first-come, first-served basis in the current slot. The amount of allocated ET transmission capacity depends on the instantaneous queue size $q(t)$ and threshold $\bar{q}$. If $q(t) > \bar{q}$, the bottleneck node will forward normal packets with full capacity, and no ET will be scheduled. If $q(t) \le \bar{q}$, $q(t)/\bar{q}$ of the forwarding capacity will be allocated for normal packets transmission and the remainder for ET packets.

Only the requests received in the previous time slot (but not earlier slots) are processed, so that the transmitted ET packets reach their destinations before the playback of corresponding GoPs with a higher probability. In addition, if the allocated forwarding capacity is insufficient to address all ET requests, the unattended ones will be discarded.

TABLE 1 *Simulation settings.*

Parameters	*Values*
Data rate of the links between the server and *Y*1	100 Mb/s
Data rate between *Y*1 and the receiver	30 Mb/s
Propagation delay between the server (receiver) and *X*1 (*Y*1)	20 ms
Propagation delay of the links from *X*1 to *Y*1	10 ms
Video data packet size	1,400 bytes
Queue length thresholds ($\bar{q}$, $\underline{q}$)	600, 100 packets
Maximum caching probability (p_{max})	0.7
Queue averaging weight (α)	0.1

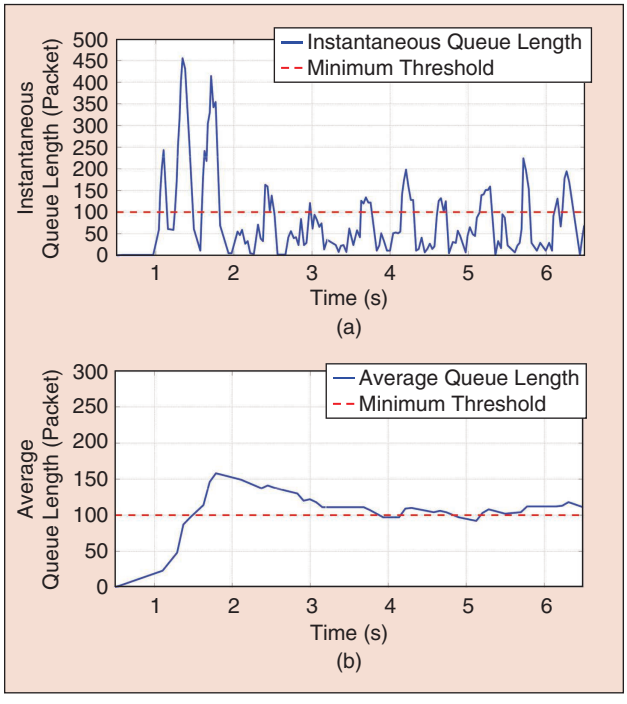


FIGURE 5 The bottleneck queue dynamics. (a) An instantaneous queue length and (b) an average queue length.

Caching Management and Source Rate Control

Because caching resources are limited on all of the nodes, packets inside the caching buffer must be constantly evicted to release the caching space. In particular, these rules are followed in caching release:

- When a cached GoP layer is successfully forwarded, the corresponding packets are eliminated.
- If an ET request is not addressed, the corresponding packets are eliminated.
- All the packets from the GoPs preceding the requested one are discarded.

If network congestion persists and the number of packets inside the buffer keeps increasing, in-network caching will be insufficient to effectively mitigate congestion. As a result, rate control is activated at the video server to reduce the number of in-flight packets. When the number of cached packets is larger than a certain threshold, the caching node sends a rate control message to the sending edge switch along the reverse routing path. Upon receipt of this message, the edge switch will hold one ACK for each video connection. The sender then detects the loss of an ACK and reduces the source sending rate.

Case Study

In this section, we present simulation results over the OMNeT++ platform (http://www.omnetpp.org/omnetpp) to demonstrate the advantages of the proposed SDTP. We consider a virtual embedded network topology (video slice 1 in Figure 1), where one video server is connected to five video receivers through two edge switches and a number of core network switches. An SDN controller is deployed for information collection and network configuration. General links in OMNeT++ are used to connect the SDN controller to all of the switches. We assume that all packet losses are caused by congestion. The transmission rate over the link between network switches is set as a uniformly distributed random variable within interval [45, 55] Mb/s to reflect the transmission rate fluctuation caused by background traffic. Other important system parameters for simulations are summarized in Table 1. The proposed SDTP is evaluated in terms of both bottleneck queue stability and user-perceived video QoE.

Bottleneck Queue Stability

We set the bottleneck forwarding capacity at $\mu = 3{,}500$ packet/s. As suggested in [14], the sampling duration should be comparable to the RTT; hence, it is set to $\Delta = 80$ ms.

Figure 5(a) and (b) depicts the instantaneous and moving-average queue length, respectively, with time. Due to features of the low-pass filter in (1), when $q(t)$ exceeds $\underline{q}$ for the first time (roughly 1.1 s), queue management is not triggered until approximately 1.5 s, when the average queue $q_{\text{avg}}(t)$ exceeds the minimum threshold. After triggering queue management, there is a time lag until the instantaneous queue length falls below the minimum threshold. As discussed in the "Bottleneck Queue Management" section, the system can be made more responsive by either increasing weight α or reducing sampling duration Δ. Figure 5(b) shows that the average queue length fluctuates slightly around the minimum threshold $\underline{q}$. Hence, by configuring $\underline{q}$, we can maintain the average bottleneck queue length at a desired level, which directly facilitates balancing between the average E2E queuing delay and link resource multiplexing.

User-Perceived QoE

Figure 6(a) compares the average E2E GoP downloading time for the proposed SDTP and the TCP. During the simulation, GoPs are encoded with a homogeneous bit rate, while layers 0, 1, and 2 of all the GoPs are comprised of 100, 200, and 300 packets, respectively. The TCP is configured with an explicit congestion notification (ECN), which is a congestion-detection measurement resulting from AQM. To simulate different levels of congestion, we vary the bottleneck link transmission rate from 40 Mb/s (severe congestion) to 100 Mb/s (congestion free). It can be seen that SDTP always outperforms TCP by providing users with a much lower GoP downloading time, especially during severe congestion. This is due to the fact that a server-side sending edge switch sends back ACKs to the video server with a shorter RTT (only one hop in our setting). Therefore, the video server increases the congestion window much faster than that of the TCP. Additionally, our proposed SDN-based AQM and selective caching schemes effectively resolve in-network congestion without increasing GoP rebuffering time and rebuffering frequency, so that the GoP downloading time is effectively reduced.

We also evaluate the displayed video quality of both the SDTP and TCP. Each GoP is encoded into three SVC layers. We evaluate the received video quality by collecting the layering information of played GoPs, considering the following widely employed factors [12]: video quality, video quality variations, and rebuffering time. The user-perceived QoE is considered to be the weighted sum of those metrics:

$$Q_K = \sum_{k=1}^{K} \text{GoP}_k^{\text{q}} - \omega_1 \sum_{k=1}^{K-1} |\text{GoP}_{k+1}^{\text{q}} - \text{GoP}_k^{\text{q}}| - \omega_2 \sum_{k=1}^{K} \text{GoP}_k^{\text{t}}, \quad (2)$$

AFTER TRIGGERING QUEUE MANAGEMENT, THERE IS A TIME LAG UNTIL THE INSTANTANEOUS QUEUE LENGTH FALLS BELOW THE MINIMUM THRESHOLD.

where GoP_k^{q} and GoP_k^{t} denote the quality and rebuffering time of the kth GoP in kilobits per seconds and seconds, respectively. By default, weights $\omega_1 = 1$ and $\omega_2 = 3{,}000$, which means that a 1-s rebuffer time leads to the same QoE degradation caused by reducing the GoP bit rate by 3,000 Kb/s [12]. We evaluate the QoE of both protocols considering the following three types of user preference: avoiding instability, avoiding rebuffering, and balanced [12], where (ω_1, ω_2) is configured as (1, 3,000), (1.8, 3,000), and (1, 6,000), respectively. As shown in Figure 6(b), the overall QoE achieved by SDTP is always higher than that of TCP for all types of users. Due to the SDTP queue management and the SCP, all of the base layers have been successfully delivered before the playback times. For the TCP, although all of the base layers are delivered to their destinations eventually, some cannot be downloaded before playback time due to network congestion and packet retransmissions. In contrast, fewer GoPs are played with enhancement layers by SDTP, but they provide good overall QoE tradeoff by smoothing video playback with minimal rebuffering events at the cost of nominal video quality degradation.

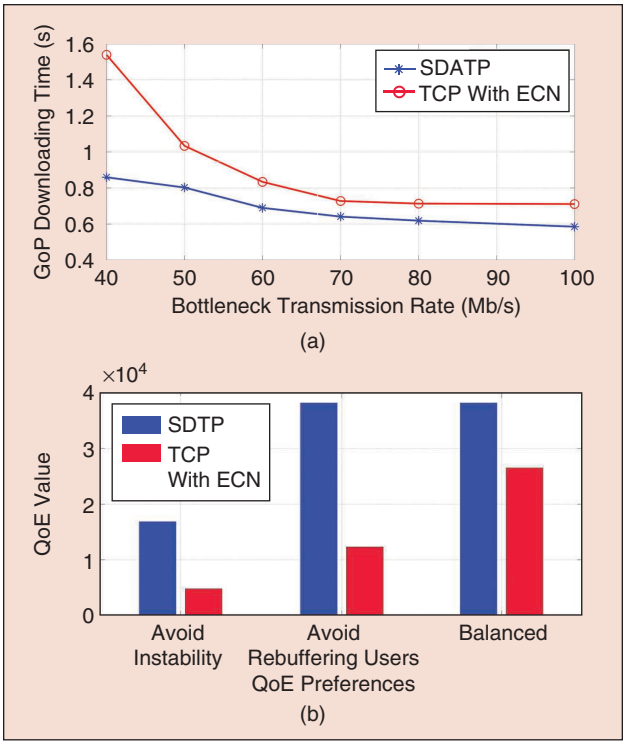


FIGURE 6 (a) A comparison of GoP downloading time. (b) A QoE comparison of different user preferences.

Conclusions

In this article, we presented a customized transport protocol for video services in future sliced networks. To improve user QoE, new protocol function modules were designed, including bottleneck queue management, selective caching, and ET to achieve early congestion detection and resolution. To support these protocol functionalities, a new packet header format was designed, with packet differentiation based on application-layer information. If the network condition improves, ET can be triggered to further improve video users' QoE. Simulation results were provided to demonstrate the advantages of the proposed SDTP over current benchmarks.

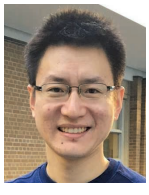
Acknowledgments

This work was supported by research grants from Huawei Technologies Canada and the Natural Sciences and Engineering Research Council of Canada. The authors thank Wenjing Chen for his help in conducting the simulations. Peng Yang is the corresponding author.

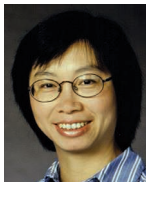
Author Information

Si Yan (s52yan@uwaterloo.ca) is currently pursuing his Ph.D. degree in the Department of Electrical and Computer Engineering, University of Waterloo, Ontario, Canada. His research interests include 5G networks, software-defined networking, network function virtualization, and network protocol design. He is a Student Member of the IEEE.

Peng Yang (p38yang@uwaterloo.ca) is a postdoctoral fellow in the Department of Electrical and Computer Engineering, University of Waterloo, Ontario, Canada. His research focuses on software-defined networking, network function virtualization, and mobile-edge computing. He is a Member of the IEEE.



Qiang Ye (q6ye@uwaterloo.ca) is a research associate in the Department of Electrical and Computer Engineering, University of Waterloo, Ontario, Canada. His research interests include artificial intelligence and machine learning for future networking, the Internet of Things, software-defined networking, network function virtualization, network slicing for 5G networks, virtual network function chain embedding, and end-to-end performance analysis. He is a Member of the IEEE.



Weihua Zhuang (wzhuang@uwaterloo.ca) is a professor and Tier I Canada Research Chair of wireless communication networks in the Department of Electrical and Computer Engineering, University of Waterloo, Ontario, Canada. Her research interests include distributed resource allocation, mobility management, and quality-of-service provisioning. She is a Fellow of the IEEE.

Xuemin (Sherman) Shen (sshen@uwaterloo.ca) is a professor in the Department of Electrical and Computer Engineering, University of Waterloo, Ontario, Canada. His research interests include network resource management, wireless network security, social networks, 5G, and vehicular ad hoc and sensor networks. He is a Fellow of the IEEE.

Xu Li (Xu.LiCA@huawei.com) is a senior principal researcher at Huawei Technologies Canada, Ottawa. His research interests are focused on 5G system design, along with more than 70 3rd Generation Partnership Project 5G standard proposals.

Jaya Rao (jaya.rao@huawei.com) is a senior research engineer at Huawei Technologies Canada, Ottawa. His current research interests include the design of the cellular Internet of Things, ultrareliable and low-latency communications, and vehicle-to-everything-based solutions in 5G New Radio.

References

[1] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "BBR: Congestion-based congestion control," *ACM Queue*, vol. 14, no. 5, pp. 20–53, 2016.

[2] S. Ha, I. Rhee, and L. Xu, "CUBIC: A new TCP-friendly high-speed TCP variant," *ACM SIGOPS Operating Syst. Rev.*, vol. 42, no. 5, pp. 64–74, 2008.

[3] A. Langley et al., "The QUIC transport protocol: Design and Internet-scale deployment," in *Proc. ACM SIGCOMM*, 2017, pp. 183–196.

[4] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[5] J. Ye, K.-C. Leung, V. O. K. Li, and S. H. Low, "Combating bufferbloat in multi-bottleneck networks: Equilibrium, stability, and algorithms," in *Proc. IEEE INFOCOM*, 2018, pp. 648–656.

[6] T. Hafeez, N. Ahmed, B. Ahmed, and A. W. Malik, "Detection and mitigation of congestion in SDN enabled data center networks: A survey," *IEEE Access*, vol. 6, pp. 1730–1740, 2017.

[7] M.-H. Wang, L.-W. Chen, P.-W. Chi, and C.-L. Lei, "SDUDP: A reliable UDP-Based transmission protocol over SDN," *IEEE Access*, vol. 5, pp. 5904–5916, Apr. 2017.

[8] X. Wang, J. Chen, A. Dutta, and M. Chiang, "Adaptive video streaming over whitespace: SVC for 3-tiered spectrum sharing," in *Proc. IEEE INFOCOM*, 2015, pp. 28–36.

[9] J. Li, W. Shi, Q. Ye, W. Zhuang, X. Shen, and X. Li, "Online joint VNF chain composition and embedding for 5G networks," in *Proc. IEEE GLOBECOM*, 2018, pp. 1–6. doi: 10.1109/GLOCOM.2018.8647700.

[10] N. Zhang et al., "Software defined networking enabled wireless network virtualization: Challenges and solutions," *IEEE Netw.*, vol. 31, no. 5, pp. 42–49, 2017.

[11] Q. Ye, W. Zhuang, X. Li, and J. Rao, "End-to-end delay modeling for embedded VNF chains in 5G core networks," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 692–704, 2019.

[12] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over HTTP," in *Proc. ACM SIGCOMM*, 2015, pp. 325–338.

[13] T. Stockhammer, "Dynamic adaptive streaming over HTTP—Standards and design principles," in *Proc. ACM MMSys*, 2011, pp. 133–144.

[14] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *Proc. ACM SIGCOMM*, 2000, pp. 151–160.

[15] E. Ozfatura, O. Ercetin, and H. Inaltekin, "Optimal network-assisted multiuser dash video streaming," *IEEE Trans. Broadcast.*, vol. 64, no. 2, pp. 247–265, 2018.

VT