Department of Communications and Networking

Video Streaming Transport: Measurements and Advances

Saba Ahsan

Aalto University

DOCTORAL DISSERTATIONS

Aalto University publication series
DOCTORAL DISSERTATIONS 15/2020

Video Streaming Transport: Measurements and Advances

Saba Ahsan

A doctoral dissertation completed for the degree of Doctor of Science (Technology) to be defended, with the permission of the Aalto University School of Electrical Engineering, at a public examination held at the lecture hall AS1 of the school on 21 February 2020 at 1300h.

Aalto University
School of Electrical Engineering
Department of Communications and Networking
Networking Technology

Supervising professor
Professor Jörg Ott, Aalto University, Finland

Preliminary examiners
Dr. Christian Timmerer, Alpen-Adria-Universität Klagenfurt, Austria
Dr. Thomas Schierl, Fraunhofer Heinrich-Hertz-Institute, Germany

Opponent
Professor Mostafa Ammar, Georgia Institute of Technology, USA

Aalto University publication series
DOCTORAL DISSERTATIONS 15/2020

ISBN 978-952-60-8927-0 (printed)
ISBN 978-952-60-8928-7 (pdf)
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)
http://urn.fi/URN:ISBN:978-952-60-8928-7

Unigrafia Oy
Helsinki 2020

Finland

Printed matter
4041-0619

Aalto University

Abstract

Aalto University, P.O. Box 11000, FI-00076 Aalto www.aalto.fi

Author
Saba Ahsan

Name of the doctoral dissertation
Video Streaming Transport: Measurements and Advances

Publisher School of Electrical Engineering

Unit Department of Communications and Networking

Series Aalto University publication series DOCTORAL DISSERTATIONS 15/2020

Field of research Networking Technology

Date of the defence 21 February 2020

Permission for public defence granted (date) 3 December 2019 **Language** English

☐ **Monograph** ☒ **Article dissertation** ☐ **Essay dissertation**

Abstract

Video streaming data forms the largest portion of the current Internet and is expected to rise even further. Providing and maintaining a high-quality experience for video is in the interest of both consumers and providers. However, the nature of network requirements for video traffic is different from other types of web traffic, and hence the goal requires a constant effort to advance video protocols in line with the changing Internet. This consists of not only developing new protocols and techniques to optimise the use of the Internet for video traffic but also developing meaningful video measurements that reflect user experience.

In this thesis, we propose extensions and mechanisms to existing Internet protocols that enhance video quality and experience. We present results from experimentation that show how slight modifications to current protocols can be done to significantly improve the video streaming experience: (1) by introducing multiple path awareness (e.g. 3G, WiFi, Ethernet) to RTP, and (2) by using out of order delivery over TCP to improve timeliness for adaptive streaming. We also present a framework for enhancing current edge network measurements to encompass video experience using active testing while keeping the tests computationally light and easily deployable for large-scale network measurements.

This thesis contributes to an evolving Internet, where video traffic has gained an equal if not higher importance to other web traffic. The large-scale video tests provide a mechanism where subscribers and Internet providers can better gauge performance from a video perspective, whereas, the work on adaptive video streaming over partially reliable and out of order transport paves the way for newer Internet protocols, which are now widely being recognised as a need of the hour in the heavily ossified and somewhat inflexible Internet.

Keywords Multipath RTP, Adaptive Video Streaming, DASH

ISBN (printed) 978-952-60-8927-0	**ISBN (pdf)** 978-952-60-8928-7
ISSN (printed) 1799-4934	**ISSN (pdf)** 1799-4942
Location of publisher Helsinki	**Location of printing** Helsinki **Year** 2020
Pages 150	**urn** http://urn.fi/URN:ISBN:978-952-60-8928-7

Preface

The research work for this thesis was carried out at Aalto University, School of Electrical Engineering, 2012-2018. The iconic design of Alvar Aalto's Otaniemi campus will forever symbolise this time of exploration that taught me more than the research questions this thesis looks to answer. Like all long journeys, this one comprises of faces and experiences that will be remembered for a lifetime.

First and foremost, I would like to thank my supervisor, Jörg Ott, for believing in my abilities and for the guidance and support he provided throughout my doctoral studies. I want to thank my co-authors, Varun Singh, Vaibhav Bajpai, Stephen McQuistin, Jürgen Schönwälder and Colin Perkins, for their valuable contributions to this research and for making every submission deadline and extended deadlines a little more exciting.

I want to thank my colleagues, IT support staff and secretaries at Comnet Department without whom this would have taken a lot longer than it did. I would like to thank Pasi Sarolahti for advising me throughout my studies. Special thanks to Zainab Saleem, Maliha Urooj Jada and Marcin Nagy for the insightful discussions that always managed to brighten a dull day.

Part of this research was funded by European Community's Seventh Framework Programme (FP7/2007-2013) grant no. 317647 (Leone) and EU's Horizon 2020 programme architectuRe for an Internet For Everybody (RIFE). My deepest gratitude to members of the project partner teams for their helpful comments and suggestions during the course of the projects.

I would like to thank the pre-examiners, Thomas Schierl and Christian Timmerer, for reviewing this work and for their constructive feedback.

I want to thank my mother for her unconditional love and support. I want to express my gratitude to my sisters, Baroosh, Mariam, Zainab and Tehreem, who have cheered me on every step of the way and have always been a source of inspiration. I am grateful to my parents-in-law, for all their love and prayers. I want to thank my friends in Espoo for making expatriate life easier. I also want to thank the Finnish weather for regularly providing one more reason to stay indoors and work the whole day.

Finally, I want to thank my children, Zeenia and Aazim, for the immense happiness they bring to my life even on the most stressful days and my husband,

Hussnain, for the unwavering support that made all of this possible. The three of them continue to fill my heart with love and pride, each in their own way.

This thesis is dedicated to Agha Amir Ahsan; father, friend, mentor and confidant, who is remembered and missed every day.

Espoo, January 4, 2020,

Saba Ahsan

Contents

List of Publications

This thesis consists of an overview and of the following publications which are referred to in the text by their Roman numerals.

I S Ahsan, V Singh, J Ott. Impact of Duration on Active Video Testing. *Proceedings of the 26th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, Klagenfurt, Austria, May 2016.

II S Ahsan, V Bajpai, J Ott, J Schönwälder. Measuring YouTube from Dual-stacked Hosts. In *International Conference on Passive and Active Network Measurement*, p. 249-261, March 2015.

III V Bajpai, S Ahsan, J Schönwälder, J Ott. Measuring YouTube Content Delivery over IPv6. *ACM SIGCOMM Computer Communication Review*, Volume 47 Issue 5, October 2017.

IV V Singh, S Ahsan, J Ott. MPRTP: Multipath Considerations for Real-time Media. In *Proceedings of the 4th ACM Multimedia Systems Conference*, Oslo, Norway, March 2013.

V S Ahsan, S McQuistin, C Perkins, J Ott. DASHing towards Hollywood . In *Proceedings of the 9th ACM Multimedia Systems Conference*, Amsterdam, Netherlands, June 2018.

Author's Contribution

Publication I: "Impact of Duration on Active Video Testing"

The author of this dissertation was the primary author of this paper and was responsible for the main idea, test development and configuration as well as statistical analysis.

Publication II: "Measuring YouTube from Dual-stacked Hosts"

The author of this dissertation was the primary author of this paper and was responsible for the main idea and test development. The statistical analysis and result compilation were equally shared by the authors.

Publication III: "Measuring YouTube Content Delivery over IPv6"

The author of this dissertation was one of the co-authors of this paper and was responsible for the test development and concepts used in the paper.

Publication IV: "MPRTP: Multipath Considerations for Real-time Media"

The author of this dissertation was one of the co-authors of this paper and was responsible for the implementation of an experimental framework, the design and implementation of Multipath algorithms as well as testing and compilation of results.

Publication V: "DASHing towards Hollywood "

The author of this dissertation was the primary author of this paper. She was responsible for implementing the DASH application over the underlying transport protocol, testing and analysis, and equally contributed to the ideas and concepts discussed within the scope of the paper.

List of Figures

List of Acronyms

API Application Programming Interface

AS Autonomous System(s)

B-frame Bidirectional predicted picture frame(s)

BBC British Broadcasting Corporation

BOLA Buffer Occupancy based Lyapunov Algorithm

CBR Constant bitrate

CDN Content-Delivery Network

CMAF Common Media Application Format

CSRC contributing source

DASH Dynamic Adaptive Streaming over HTTP

DCCP Datagram Congestion Control Protocol

DNS Domain Name System

ECN Explicit Congestion Notification

FCC Federal Communications Commission

FEC Forward Error Correction

FLV Flash Video

fps frames per second

FR Full Reference

GGC Google Global Cache

HD High Definition

HE Happy Eyeballs

HEVC High Efficiency Video Coding

HLS HTTP Live Streaming

HTTP Hypertext Transfer Protocol

I-frame Intra-coded picture frame(s)

IETF Internet Engineering Task Force

IPPM IP Performance Metrics

ISP Internet Service Provider(s)

IXP Internet Exchange Point

LMAP Large-Scale Measurement of Broadband Performance

MKV Matroska Multimedia Container

MOS Mean Opinion Score

MP4 MPEG-4 Part 14

MPRTP Multipath Real Time Protocol

MPRTCP Multipath Real Time Control Protocol

MPTCP Multipath TCP

NR No Reference

NREN National Research and Education Network

Ofcom Office of Communication

OTT Over the Top

P-frame Predicted picture frame(s)

PSNR Peak Signal to Noise Ratio

QoE Quality of Experience

RR Reduced Reference

RTC Real-Time Communication

RTP Real-time Transport Protocol

RTCP Real-time Transport Control Protocol

RTSP Real-time Streaming Protocol

SCTP Stream Control Transmission Protocol

SD Standard Definition

SIP Session Initiation Protocol

SNMP Simple Network Management Protocol

SS Smooth Streaming

SSIM Structural Similarity

SSRC synchronisation source

TCP Transmission Control Protocol

UDP User Datagram Protocol

UK United Kingdom

US United States

VBR Variable Bitrate

VMAF Video Multimethod Assessment Fusion

VoD Video on Demand

VoIP Voice over IP

1. Introduction

Video streaming, consisting of a range of Live and Video on Demand (VoD) services, is now the largest contributor to Internet traffic. Consumers have been gradually moving from traditional television to Over the Top (OTT) services for entertainment and news [12, 13]. Multimedia streaming is any multimedia content that is delivered to a client from a streaming server, which can be played out as it is delivered without waiting for the whole download to complete [5]. Traditionally, multimedia streaming involved streaming servers, such as Real-time Streaming Protocol (RTSP) servers, that managed connections to clients for the transport of media over the Real-time Transport Protocol (RTP). Unreliable and lightweight protocols such as User Datagram Protocol (UDP) were commonly used as the underlying transport protocol. Timeliness being a key factor in video streaming, use of reliable and latency prone protocols such as Transmission Control Protocol (TCP) was not a viable option until the increase in broadband connections to home users. Nowadays, video streaming is mostly done over Hypertext Transfer Protocol (HTTP). Adaptive streaming over HTTP is the de facto standard for media services as it conveniently uses the already deployed HTTP content distribution infrastructure [66]. It is worth noting that this consensus over a TCP based solution is also partly motivated by the ossification of the Internet and the difficulty in adopting new and, in some cases, more suitable protocols as well as the availability of high speed Internet that overcomes the reliability-induced latency in TCP [30, 60, 73].

The ubiquity of video streaming calls for better understanding of network performance in the context of video experience. Metrics that are not invasive of user privacy and yet provide important indicators about the health and workings of popular streaming services are a requirement for consumers, service providers and regulators. Furthermore, research towards improving transport protocols to better serve video streams is a necessary and constant process to keep up with the changing landscape of Internet technology.

1.1 Objectives and Scope

This thesis focuses on Internet video streaming; the delivery of video content from a server to a client over the Internet using incumbent and new transport protocols. The objective of the thesis is to improve streaming services using transport-level modifications and to advance network measurements to encompass video streaming performance. Our research area includes OTT video streaming that spans both traditional video streaming with dedicated streaming servers as well as HTTP based video streaming. Other types of video transport such as conversational video are considered out of scope, even though, some of the work is relevant to such use cases as well [82]. Network guarantees, video acceleration, and preferences imposed from the network operators are also considered out of scope. The thesis studies video in terms of its transport and delivery and does not expose coding algorithms any more than is necessary for the said functionality.

1.2 Summary of the Publications

This thesis is built upon five research publications based on original work by the author. In Publication I, we study the methodology of active testing of video streaming over HTTP. The impact of the duration of active testing from home gateways is discussed in reference to large-scale testing of YouTube. Publication II and Publication III present the results of a YouTube measurement study over dual stacked hosts using measurement probes connected to home gateways. The measurement points were scattered mainly in Europe, the United States, and Japan, which are also the forerunners in IPv6 deployment. The 34 months long dataset shows improvements in performance over IPv6 over time. It also shows the disproportion in the availability of Google Global Cache (GGC)s, which are more prevalent over IPv4.

The last two publications focus on studying the benefits of video streaming using new variants of traditional protocols with partial or no reliability. Publication IV investigates the advantages of using an unordered variant of TCP called TCP Hollywood for transporting Dynamic Adaptive Streaming over HTTP (DASH). TCP Hollywood is wire-compatible with TCP and outperforms TCP in high latency and high loss networks in our experiments. Publication V extends RTP to support multiple paths and shows through experiments that, in a client/server streaming scenario, MPRTP is able to compound the bandwidth of the available paths, offload traffic from congested paths using our scheduling algorithm, and fairly share bottlenecks with other MPRTP flows.

1.3 Contributions

The main contributions of this thesis are as follows:

- A YouTube test as an open source software for large scale active testing of YouTube from home gateways. The test was deployed as part of the SamKnows suite [1] by consumers and Internet providers for gauging network health for the YouTube service. We propose and implement test metrics that provide repeatable and actionable results that benefit service providers as well as consumers and provides recommendations for the minimum duration required for accurate results with such active tests based on YouTube's video content.

- A dataset and an analysis from a measurement study of YouTube over dual stacked hosts from nearly 100 probes connected to home gateways representing 66 different origin Autonomous System(s) (AS). The dataset consists of 34 months (Aug 2014-Jun 2017) of measurements including TCP connection establishment times to YouTube web and media servers as well as streaming related metrics such as startup delay and stall events. Our analysis reveals the disparity in the availability of geographically co-located media servers between the two address families.

- An open source DASH client and server written in C/C++ compatible with TCP Hollywood and a complete test suite for evaluation, released for the research community along with an analysis of scenarios where TCP Hollywood outperforms TCP. The study explores the concept of adaptive HTTP streaming over an unreliable or partially reliable protocol and the rate adaptation algorithms for such protocols.

- MPRTP, an extension for RTP that utilises multiple paths for transporting RTP traffic and a scheduling algorithm that distributes traffic on these paths based on their characteristics. The study contributed to an Internet Engineering Task Force (IETF) Internet draft.

Overall, this thesis provides recommendations for improving delivery of video streams over the Internet and quantifying its performance in a way that is comprehensible to users, scalable over large networks, and actionable in case of problems. The research is complemented by code and datasets not only for repeatability and verification but also to aid the research community in pursuing related future work.

[1] SamKnows is a global Internet measurements platform, www.samknows.com

1.4 Research Methodology

A variety of research methodologies were used to tackle the different research objectives in the thesis. Primarily, the design and abstraction methods described by Association for Computing Machinery (ACM) were used [20]. The design paradigm is used for engineering and constitutes these steps: identify requirements, define specifications, design, implement and, finally, test the system. This method was used for designing MPRTP and testing it in Publication IV. The abstraction paradigm is used for experimental science and consists of forming a hypothesis, constructing a model, experimentation, and collection of data and its analysis. It was used in Publications I, where we hypothesised and showed that too short active tests can yield inaccurately optimistic performance results for video and found a reasonable lower bound for video testing duration. It was again used in Publication V in which we showed that unordered TCP can improve DASH performance in high latency and high loss networks. Finally, Publications II and III are exploratory in nature employing data collection and analysis of existing systems.

The usefulness of any scientific study is deeply rooted in its reproducibility [8, 75]. Most of the software developed as part of this thesis, the details of the experimentation as well as measurement datasets have been made public for the benefit of the research community. Publication V has been awarded the Association for Computer Machinery (ACM) reproducibility badge for the artefacts (testing suite and code) that have been made publicly available. Furthermore, in the context of standardisation, research conducted as part of this thesis has made contributions to an IETF Internet draft [83].

1.5 Structure of the Thesis

This work contributes to video streaming transport; measuring the impact of video transport on user experience and refining transport techniques to improve performance. Chapter 2 provides the relevant background of the evolution of video transport protocols and techniques to the current state-of-the-art. It also describes the different methods used to measure video quality and experience.

In Chapter 3, we discuss large scale active measurements for video streaming performance that bring broadband performance metrics closer to actual video experience. The chapter discusses video-related network metrics and the results from Publication I on the recommended duration of active video testing. Chapter 4 presents the 34 month long measurement study conducted using these active measurements for studying the performance of YouTube over IPv6 in comparison to IPv4, as published in Publication II and III.

The next two chapters, covering Publication IV and V in order, deal with the evaluation of lossy video streams. In Chapter 5, we evaluate DASH over TCP Hollywood and discuss the performance benefits by removing head-of-

line blocking on a transport and application level as well as the impact of the introduction of packet loss in a technique that is primarily developed for reliable transport. In Chapter 6, we study video streaming in a multipath scenario using MPRTP.

Finally Chapter 7 concludes the thesis with a discussion on impact of this research and future directions for the work.

2. Background

At the turn of the century, faster Internet speeds and better compression techniques enabled the pervasiveness of video over the Internet [95]. In June of 1999, Apple released QuickTime 4, the first version to support video streaming and launched the free QuickTime Streaming Server. By 2002, Flash Player version 6 brought video to websites first supporting SWF and subsequently extending to Flash Video (FLV). This opened a new arena for web services with video sharing websites like Vimeo and YouTube entering the market in 2004 and 2005, respectively. While IPTV had been around since the mid nineties, the availability of high speed Internet to home users and the success of YouTube triggered a business case for OTT media services. In 2007, Netflix introduced VoD in addition to its online DVD rental. In the same year, British Broadcasting Corporation (BBC) launched the BBC iPlayer for users in United Kingdom (UK), providing streaming services for its television content, while Hulu began operation in United States (US) in the following year also offering OTT television programs. As of 2018, according to the Sandvine Global Internet Phenomena report, video comprises about 58% of all downstream traffic on the Internet.

During the past two decades, video has emerged as the leading source of Internet traffic, while the underlying transport protocols, compression techniques and video quality assessment mechanisms have seen a continuous and commendable progress [50, 19, 95, 18]. This chapter discusses the components, requirements and transport methods of video streaming to form the background for the remainder of the thesis. In Section 2.1, different compression techniques and the parts of a compressed video stream are explained. Transport requirements of a video stream are discussed in Section 2.2. Video streaming transport protocols and techniques used over the years and the scenarios and circumstances that lead to the success of some protocols and the decline of others are discussed in Section 2.3. The methods used for measuring video performance are described in Section 2.4. The chapter is summarised in Section 2.5.

2.1 Video Stream Format

Video streaming is bandwidth hungry and hence efficient compression goes a long way in improving its quality. Nowadays, due to higher resolutions, raw video files are often several Gigabytes in size. Video compression is used to minimise redundancy within the stream. Digital containers create the commonly seen media file formats such as MP4 (ISO base file format) or WebM (Matroska) by interleaving one or more compressed streams according to a specification. The containers add metadata that arrange and identify the different packaged streams as well as the various elements within each stream. RTP streams, described later in Section 2.3.2, don't require these containers as the metadata are carried within the packet headers. While the focus of this thesis is not on compression techniques or container formats, it is important to understand the features of a video stream to understand the nature and requirements of its transport.

2.1.1 Video Picture Types

The types of frames within a compressed video stream are characterised into three types: Intra-coded picture frame(s) (I-frame), Predicted picture frame(s) (P-frame), and Bidirectional predicted picture frame(s) (B-frame). I-frame are standalone compressed images and do not require any additional information in order to be decoded. Consequently, I-frames are larger than the other two types. P-frames depend on past frames, whereas the dependency of B-frames is bidirectional, which means they are interconnected to both past and future frames. The size and frequency of each type of frame within the stream is a function of the compression standard and the used parameters, as well as the content of the video.

2.1.2 Video Compression Standards

One of the most widely used video compression standard as of writing this thesis is the H.264 or MPEG-4 Part 10, Advanced Video Coding (MPEG-4 AVC), jointly developed and maintained by ITU-T Video Coding Experts Group and the ISO/IEC Moving Picture Experts Group [94]. The codec, which was first released in 2003, was designed to be a single solution for broadcast video, disc storage, conversational video as well as multimedia streaming. The increased coding efficiency combined with simplicity helped create an Internet video boom. H.264 has gone through several revisions since and is extensively used by media streaming websites including YouTube. In 2013, the first version of High Efficiency Video Coding (HEVC), also known as H.265, was released, which was developed as the successor to H.264; designed for HD and UHD content, it has shown bitrate savings of over 50% in comparison to H.264 [95].

The WebM project, lead by Google, has been working on producing royalty-

free media formats for the Internet since 2010. The coding algorithm VP8 [10] and its successor VP9 [58] are competitors for the royalty based H.264 and HEVC coding algorithms mentioned above. The VPx encoded stream contains keyframes (equivalent to I-frames) and interframes (equivalent to P-frames). Bidirectional frames are not defined in the specification but they can be emulated if desired. Studies have shown that the quality of the compressed video for VP8 and H.264 is comparable, however, the latter is as much as 350% faster in terms of encoding speed [25]. Encoding speeds of VP9 are also over 100 times higher than H.264 [35] but lower than HEVC. The VPx codecs offer a competitive alternative to the patented codecs; they are being used by video services for HTML5 content and supported by a majority of web browsers in the market.

Compression algorithms are mostly used with Variable Bitrate (VBR) encoding nowadays. VBR encoding adjusts the bitrate of the stream around a mean encoding bitrate according to the complexity of the video content, thus producing a stream that has an almost constant quality but a variable bitrate. Encoders using VBR may be configured to do multiple passes over the file to find the most efficient use of available bits. In contrast, Constant bitrate (CBR) encoding produces almost equal sized frames (keeping the bitrate constant) but the quality of the frames may fluctuate because of this restriction. Consequently, VBR encoders have a higher coding efficiency but are slower than CBR, especially when doing multiple passes. From a network traffic perspective, VBR streams are less predictable than CBR. Extensive work has been done to model VBR traffic to assist in performance evaluations, network simulations, and other benchmarking exercises [93]. An alternate approach to this problem is to use data from actual encoded streams such as the one provided by the dataset in [68] for H.264 and H.265 streams.

2.2 Reliability and Timeliness

Video has long been understood to be more tolerant towards losses (within acceptable parameters) than towards latency and timing issues. Video streams commonly suffer from two types of distortions: spatial and temporal. Spatial distortions, such as pixelation, are caused by missing or incomplete data. Temporal distortions, such as stalling and synchronisation problems, are caused by delayed data or clocks that are not synchronised. The level of distortion strongly depends on the particular bytes within the stream, specifically the nature of the frame that they are a part of. Since all frames are not created equal, all frames are not equally important and discarding a lost or delayed P or B frame usually results in minor spatial distortion. This is at times preferable to the temporal distortion caused by waiting for the said frame to arrive. On the other hand, the loss of an I-frame can produce spatial distortions that will affect all subsequent P and B frames that are dependent on it. Hence, despite its tolerance to losses, some level of reliability is still required for transporting video traffic.

Temporal distortions, such as stalling, jitter, and audio playout that is not in sync with video, are all undesirable in video streaming and have a significant impact on the user experience. Video clients have their own de-jitter buffers for storing or reordering data as per timing information. These de-jitter buffers, commonly known as playout buffers, are used to remove or minimise the effects of network and transport induced jitter and unordered delivery and can also buffer video in advance for smooth playout. The length of a playout buffer and level of buffering depends on several factors. For instance, short buffers may be used due to memory constraints on the client. Longer buffers may be used to ensure stall-free experience even in the presence of network impairments and limited bandwidth.

In modern VoD and even Live video streaming, the timing requirement is not as strict as real time media and video clients handle delays by stalling until the playout buffer is refilled and video is resumed instead of discarding the expired frames. This is in fact a consequence of the underlying TCP protocol, which is strictly reliable. As previously mentioned, these applications use pre-buffering to avoid buffer underruns once the video has started playing. Another important factor is the startup delay or click-to-play latency, loosely defined as the time it takes from when a user clicks to play a video till it starts playing. Pre-buffering contributes to startup delay and systems strive to find the right balance between the two to maximise user experience.

2.3 History of Video Streaming Protocols

Traditional transport protocols such as TCP and UDP were generally considered unsuitable for video streaming. TCP is connection-oriented, reliable, and strictly time-lined with congestion control for fairness and is, therefore, also inherently slow and carries a large overhead. As the protocol is ordered, it is prone to head-of-line blocking, illustrated in Figure 2.1; if a segment or part of it is lost or delayed, the transport layer does not deliver any data to the application layer until that segment has been recovered. In video streams, head-of-line blocking can compound the loss of a single segment to a loss of multiple subsequent ones if the delay caused by head-of-line blocking is large enough that the data queued at the transport layer expires in the mean time. UDP, on the other hand, is lightweight but has no reliability, ordering, and fairness mechanisms in place, again, falling short of meeting media transport requirements. Given the shortcomings of the two core transport protocols of the Internet suite, there have been numerous efforts to design a protocol that is suitable for its transport.

2.3.1 Transport Ossification

The two core protocols of the TCP/IP suite offer a simple socket Application Programming Interface (API) that has been adopted and implemented by mul-

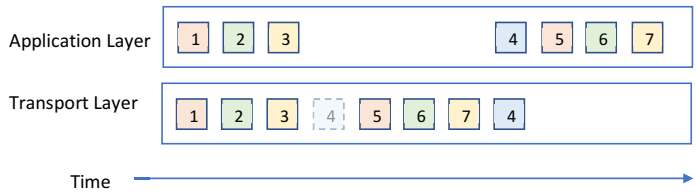


Figure 2.1. Head of Line Blocking in TCP and other ordered, reliable transport protocols may cause delayed delivery to application layer when a packet is delayed or lost.

tiple operating systems. Middleboxes are prevalent in both small and large networks performing a variety of functions ranging from security (e.g. firewalls, gateways) to optimisation (e.g. load balancers, proxies), at times in numbers at par with switches and routers [81]. The ubiquity of the Internet, however, has become a hindrance to innovation on the transport layer. The simplicity of the socket API provides little flexibility and extensibility to accommodate new features in transport protocols and the presence of middleboxes that operate on transport layer implies that upgrades are required to allow newer protocols to traverse the Internet. The adoption of a new API socket across multiple platforms with kernel level modifications is an expensive and time consuming exercise. The same is true for the upgrade of middleboxes, which are manufactured and managed by a range of vendors. Depending on the level of modifications needed, it requires a coordinated effort with interoperability testing and warrants a reasonable business case. Such a business case can only be provided by a widely used and validated protocol, which in turn cannot be achieved without implementing a new API and traversable middleboxes, thus creating a deadlock. This phenomenon is commonly known as transport layer ossification [62]. Notable protocols that failed in terms of deployability despite strong design are Stream Control Transmission Protocol (SCTP) and Datagram Congestion Control Protocol (DCCP). The protocols, briefly explained below, overcome some of the drawbacks of TCP and UDP respectively.

SCTP is a reliable message-oriented protocol that by design supports congestion avoidance and multihoming (support for hosts connected to more than one network) [89]. It eliminates the problem of head-of-line blocking in two ways: (1) the protocol supports multiple independently ordered streams and head-of-line blocking in one stream does not block any of the other streams, and (2) message ordering is optional, allowing applications to receive data as it arrives when needed with the caveat that the upper layer must handle unordered delivery. Furthermore, SCTP performs a four-way handshake unlike the three-way handshake of TCP. This makes SCTP resilient to SYN flood attacks, which are a threat for exposed TCP servers.

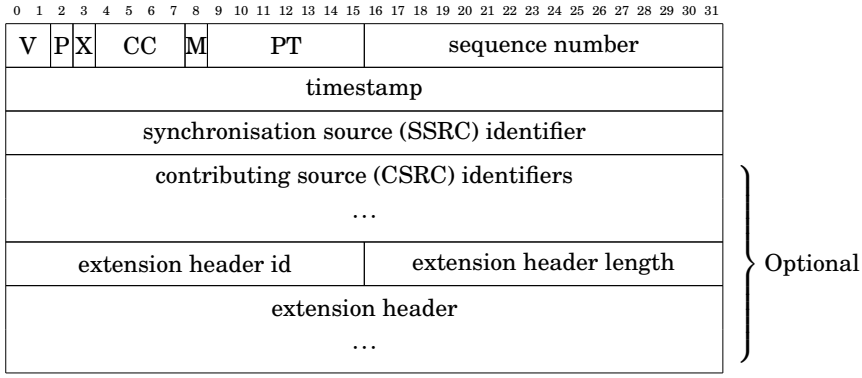


Figure 2.2. RTP Header format.

DCCP was developed to overcome the lack of fairness in UDP flows, while retaining the light-weighted unreliability [27]. UDP performs fairly well for short lived, request-response flows that produce minimal traffic load such as Simple Network Management Protocol (SNMP) and Domain Name System (DNS). It is also extensively used for real-time traffic such as multimedia streams due to its low overhead and low latency, which cannot afford the latency associated with TCP. However, these streams are neither short lived nor low traffic and the absence of congestion-control in UDP makes them unfair to the congestion-controlled TCP flows. DCCP is end-to-end congestion-controlled. It uses reliable connection setup, feature negotiation and teardown of connection, while providing unreliable data transfer. Packet loss and Explicit Congestion Notification (ECN) are communicated back to the sender using ACKs.

Despite their advantages, neither protocol has been able to create its mark in the Internet. SCTP, however, was developed for SS7 and the control plane and is part of the SIGTRAN protocols. In the next section, we discuss RTP, a protocol designed for transporting real-time data such as voice and video, which typically runs over UDP or TCP and is implemented at the application layer, consequently overcoming some deployment challenges related to ossification.

2.3.2 Real-time Transport Protocol

RTP is an end-to-end protocol for transmitting real-time data such as audio and video over unicast and multicast networks [76]. RTP runs over a transport protocol such as UDP or TCP (UDP is more commonly used) and is coupled with an out-of-band control protocol called Real-time Transport Control Protocol (RTCP). The RTP/RTCP suite has been in use for conversational media as well as multimedia streaming where low-latency timelined transport is needed [98]. More recently, it is used for real-time communication in WebRTC, a free and open-source project that enables web-browsers to support Real-Time Communication (RTC) such as Voice over IP (VoIP) and video conferencing using Java API. RTP

packets have sequence numbers for reordering at the application layer, however, the protocol does not provide any form of congestion control. The IETF working group, Audio/Video Transport Core Maintenance (avtcore), maintains the core specification and develops architectural guidelines for extensions, while the working group RTP Media Congestion Avoidance Techniques (rmcat) is tasked with developing a suitable congestion-control mechanism and steering it towards standardisation.

Figure 2.2 shows the RTP header. Each packet carries a monotonically increasing sequence number that is used for reordering and loss detection at the receiver. The timestamp indicates the playout time of the payload and increases at the media-specific sampling frequency. Unlike sequence numbers, RTP timestamps are not monotonic because packets are not always transmitted in the same order as they are sampled. Compressed video streams often have frames whose decoding timestamps precede their playout timestamps because of picture inter-dependencies and are transmitted as such. Furthermore, two or more consecutive packets may have the same timestamp, e.g., if their payloads are part of the same frame. Sequence numbers and timestamps are randomly initialised to protect against known-plaintext attacks on encryption and therefore care should be taken to make the value unpredictable.

The synchronisation source (SSRC) represents the media source and its value is chosen randomly by the sender. A single RTP session may contain multiple streams multiplexed together using different SSRC identifiers to distinguish them, for instance, an audio and video stream or streams from multiple viewports [49]. A mixer can fill a list of contributing source (CSRC) identifiers for the payload and the number of identifiers (maximum value is 15) is indicated in the field CSRC count (CC) if multiple sources contribute to the mixer signal. The Payload Type (PT) field is used by the application for identifying the RTP payload and is specified by the RTP profile [78]. The RTP header can be extended; the extension header is appended at the end of the RTP header and its presence is indicated by setting the extension bit (X) to 1. The extension header id is a fixed identifier that specifies the type of extension and the header length is the number of 32bit words that follow. Note that the extension header length excludes the extension id and length fields, making 0 a valid length. Further details about the fields can be found in RFC 3550 [76].

RTCP control packets are multiplexed with RTP packets using a separate port number and are primarily used for feedback and synchronisation. The most common types of RTCP packets are Sender Reports (SR), sent by RTP receiver that are also active senders and Receiver Reports (RR), sent by RTP receivers that are not active senders. RTCP reports are transmitted periodically and rate controlled to never exceed a fraction of the RTP session bandwidth in the interest of scalability in a multicast group.

RTP/RTCP are used in conjunction with a signalling protocol for session initiation, management and termination such as RTSP [77] or Session Initiation Protocol (SIP) [74]. Both protocols are text-based application layer protocols

used for setting up and managing media sessions.

2.3.3 Streaming over TCP

The pervasiveness of broadband and high-speed Internet may mitigate the effects of latency induced by TCP on video streaming [15]. Over the past decade, streaming over HTTP has become the de facto standard for VoD and Live streaming services. Services like YouTube, Vimeo, Netflix, Hulu, Amazon, and various others all rely on HTTP for streaming video to their viewers. The technique is referred to as progressive download and differs from the traditional definition of video streaming as the client downloads the video files from HTTP servers instead of fetching video through streaming servers as would be done for RTP, RTSP and SIP. HTTP streaming owes its success to a number of factors, which include the low cost of setting up and maintaining an HTTP server in comparison to video streaming servers, the lack of deployability challenges associated with HTTP and TCP, and the reusability of the CDN infrastructure [100, 66]. The entire process is client-driven; HTTP is stateless and each HTTP request/response is a single operation, whereas streaming protocols are stateful and session parameters are negotiated for each session [90]. Most video clients today use adaptive streaming over HTTP, which is a type of progressive download that allows adapting the quality of the stream according to the network conditions. Both techniques are discussed below.

Progressive Download
Progressive download works with media files that store the metadata about the streams and their seek points in the beginning of the file. A seek point is an I-frame, such that no preceding frames are needed in order to begin playing from that point in the file. Hence, as soon as the client receives this preamble it can begin playing the content as it arrives without waiting for the entire file to download, however, most clients pre-buffer a predefined amount of audio/video data before playing in order to ensure smooth playout that is free of stalling.

The presence of seek points allows the users to select a playout position from the timeline once the video starts playing, which the video client can then skip to by using the *Range* field in HTTP GET request and requesting the specific part of the file that corresponds to media at and after that play point. Progressive download creates bursts of TCP traffic unlike the steadily timed RTP streams that usually transmit data according to frame timestamps [4]. In order to prevent wasting network resources caused by downloading full videos that users may abandon in the middle, video clients define limits on buffering [70]. YouTube web clients download about 60 seconds of video in advance and only request more data when the playout proceeds. Other mechanisms to limit this waste of resources include *Trickle* which uses TCP congestion window size to rate limit YouTube streams and has been shown to reduce retransmissions by 50% in high speed networks [32].

Commonly used container formats for progressive download are FLV [40], Matroska Multimedia Container (MKV) [1], and MPEG-4 Part 14 (MP4) [3].

Adaptive Streaming over HTTP

Adaptive Streaming over HTTP (HAS) adds bitrate adaptation capabilities to progressive download. Several flavours exist including Apple's HTTP Live Streaming (HLS), Microsoft's Smooth Streaming (SS), and the industry standard DASH; the basic technique is the same [1, 12, 13]. This thesis mainly focuses on the DASH specification for adaptive streaming.

A HAS video is encoded using multiple encoding bitrates yielding different qualities, known as representations. Each representation is divided into chunks (known as a segment, as per the DASH specification) that are equally sized by duration and always begin with an I-frame. These aligned chunk boundaries allow clients to switch from one representation to another. The presence of the I-frame ensures that the chunks can be played independently as long as the metadata is available with the client for the relevant representation. HAS clients are specialised video players with adaptation capabilities, however, the servers are simple HTTP servers hosting HAS content. Clients can download a manifest file from the HAS server, which contains information about the different representations the audio/video streams are available in, encoding information such as bitrate and resolution in case of video, the duration of the chunks, the associated URLs and *init* segments. *Init* segments, which must be downloaded first, contain metadata about the streams while the media frames are packaged inside the subsequent chunks. The DASH specification recommends using ISO Base Media File Format (e.g. MP4) or MPEG-2 Transport Streams; the latter is loss tolerant. Note that we use the term chunk throughout this thesis also for DASH segments. This is not to be confused with the chunked encoding used in Common Media Application Format (CMAF), which is a new file format for HTTP streaming [41].

Like all traditional HTTP operations, HAS streams are also client-driven. The bitrate adaptation is controlled using adaptation algorithms embedded in video clients. The algorithms are designed to select the highest possible bitrate that can be safely downloaded in the current network conditions without causing any stalls in the playout. While a number of algorithms exist, many commercial clients using their own proprietary ones, the type of algorithms can be roughly divided into three categories. The simplest ones are throughput-based, which use observed throughput at the client to estimate an appropriate representation for the next chunk [53]. Others use the time to download a chunk as the main metric for estimation [11]. However, there is general consensus among researchers that buffer based algorithms that utilise the length of playout buffer (the duration of the video that has already been prebuffered at the client) as the primary indicator for quality estimation perform better [87, 44, 39]. Most clients would use a mix of indicators for the estimation and additional rules

[1] www.matroska.org, www.webmproject.org

such as chunk abandonment to ensure smooth playout. Chunk abandonment is when a client would abandon an active chunk download in a higher quality when it foresees that continuing may result in stall events and requests the same chunk at a lower bitrate. Some clients may download chunks that have already been downloaded in a lower quality again at a higher representation and replace them in the playout buffer for a better user experience. However, this practice may cause strain on the network resources as multiple copies of the same chunk will be downloaded [86].

2.4 Measuring Video Performance

Measuring video performance is an arduous task given the number of application and network level parameters that can influence the experience [97]. These include the underlying technology used for creating and distributing the video content such as encoders and encoding parameters, transport protocols, as well as the network quality and the content distribution network [17]. Other factors include size and type of display devices, the content, and intended use of the video. For instance, user expectations and subsequently quality requirements are higher for a feature film on a home studio in comparison to a short internet video on a handheld device. Similarly, clarity of sound is a higher priority for a newscast whereas sports transmissions require a higher quality video. The actual Quality of Experience (QoE) of the video may be further influenced by the past experience, preferences, and expectations of the users [46]. Subjective QoE methods are aggregated scores based on ratings assigned by real users such as Mean Opinion Score (MOS). The quality ratings can be a scale of 1 to 5, where 1 is bad and 5 is excellent; the ratings from a statistically significant group of people are aggregated to get a numerical quality indicator. Since they require human subjects, these methods are expensive and unscalable for large distribution networks and hence are often impractical for performance evaluation studies despite their advantages. Furthermore, it is difficult to isolate human factors such as content preference and personal experiences from the quality ratings [46, 65].

In this section, objective methods of calculating QoE and network metrics that are known to have a direct impact on Internet video experience are discussed. These metrics are benchmarked using subjective methods to assess their validity [79].

2.4.1 Objective QoE

Objective QoE metrics can be of three types depending on the information required to estimate them; Full Reference (FR), Reduced Reference (RR), and No Reference (NR) [26]. FR metrics compute QoE by comparing the output to the original stream. RR methods do not require original stream but still require

some information about it to compute the quality of the received stream, while NR models require no additional information from the sender at all.

The dependency of FR models on the availability of the original stream makes them unsuitable for live network performance evaluations, however, they are extensively used in experimental setups for evaluating the performance of video systems. Two FR metrics are used in this thesis; Peak Signal to Noise Ratio (PSNR) and Structural Similarity (SSIM). PSNR measures signal fidelity by comparing the maximum power of the original signal to the power of the received signal. SSIM measures perceptual quality by comparing the similarity between the original and the received frames. Another QoE metric is the Video Multimethod Assessment Fusion (VMAF) developed by Netflix. VMAF is a FR model that combines several weighted objective QoE metrics to calculate an overall score for the video quality. The weights for the component metrics are assigned using machine learning on a Netflix dataset, making it relevant for OTT VoD services similar to Netflix [52]. The VMAF metric, which was released in 2016, is used by Netflix to estimate the quality of their encoded streams to achieve better compression ratios with higher visual quality.

NR models do not require the original signal, making it easier to deploy them at different network points where access to the original signal is difficult or not possible. However, because of the lack of reference, NR models rely on a number of assumptions that make them fine tuned to detect only specific types of distortions (e.g. blockiness or blurriness) and susceptible to errors resulting from mistaking original content for distortions [56]; for example, interpreting a chess board as a block artefact [97].

2.4.2 Network and Application Metrics

Traditional network performance metrics such as throughput, packet loss, and jitter are insufficient to predict the perceived video quality at the user end. This is especially true for traditional video streaming solutions where video streams could be lossy. As described in Section 2.2, unlike data streams, not all parts of a video stream are equally significant. The effect of the loss of an I-frame is carried on to all subsequent frames dependent on it, whereas the effect of the loss of a P or B-frame, which has no other frames dependent on it, is limited to the duration of that frame or as little as 40ms in a stream with 25 frames per second (fps). Hence, the occurrence of packet loss alone is insufficient to estimate the effect on video quality.

Network and application metrics gain relevance in the context of HTTP streaming, which works over reliable transport and hence loss effects don't need to be accounted for. All impairments are temporal in nature caused by delays of one kind or another. Most HTTP streaming and DASH services rely on application level metrics such as startup delay, stall events, and streaming bitrate to estimate video performance. Studies have shown that these metrics provide a good estimation of the perceived quality at the user end [80, 72, 23, 88, 57]. The

metrics are discussed in more detail in Chapter 3.

2.5 Summary

This chapter provides a short history of video streaming over the Internet and an overview of the components and techniques involved including protocols, codecs, and quality metrics. Traditional video streaming was transported using protocols such as RTP supported by signalling protocols such as RTSP and SIP. The streams were carried over unreliable transport and were prone to spatial impairments due to packet loss. Objective QoE metrics, e.g., PSNR and SSIM, were popular amongst researchers to quantify these distortions. At the turn of the century more efficient compression algorithms such as H.264, higher Internet speeds, and support for Flash video in HTML triggered the adoption of HTTP as the de facto standard for video streaming over the Internet. Progressive download and subsequently adaptive streaming over HTTP rely on the strictly reliable TCP protocol. Therefore, spatial impairments in video due to packet loss are not an issue and video performance metrics such as startup delay, occurrence and duration of stall events, and the selection and stability of the stream bitrate are considered important indicators. In the next chapter, we present a testing environment for large-scale active measurements for Internet video streaming based on such performance metrics.

3. Measuring YouTube

YouTube is a video sharing service that has not only grown to become one of the most widely used video sharing websites, but has also greatly influenced video streaming since its launch in February 2005. The website Alexa ranks YouTube as the second most popular website in the world based on number of visitors and page views, second only to google.com. The success of YouTube, a website that uses video streaming over HTTP, proved that TCP and HTTP servers were a workable choice for video streaming despite the compromise in terms of latency of retransmissions and head-of-line blocking. Robert Kyncl, the previous Vice President of Content Acquisitions at Netflix, writes in his book *Streampunks: YouTube and the Rebels Remaking Media* that seeing YouTube's popularity prompted Netflix to adopt OTT web streaming over a hardware-based solution, a service that was launched in 2007.

The popularity of YouTube also makes the performance of its service of particular interest to Internet users and providers as well as the research community. Passive measurements are non-intrusive, generating no additional traffic into the network, and measure performance as it is perceived in a real scenario by real users in their own environment, e.g., watching a football match with friends on a home TV or binge watching a television series on a laptop. However, passive measurements have privacy and repeatability issues. Passively collected data is also harder to analyse and compare across measurement points because it is not uniform; varying amounts of cross traffic and user behaviour/preferences make it difficult to compile. The alternative is active measurements. Large-scale active measurements to gauge Internet broadband performance have gained momentum in the past few years and several benchmarking exercises and global Internet measurement platforms have emerged. Internet speed and latency continue to be popular metrics for benchmarking Internet performance but offer little insight into the performance of video streaming.

In this chapter, we present our work pertaining to performance metrics suitable for large-scale measurements for YouTube as a service. Our research focuses on active testing from the home gateway. In Section 3.1, we discuss scalable Internet performance measurements for broadband Internet. We describe various application level metrics for HTTP streaming in Section 3.2 and go on to

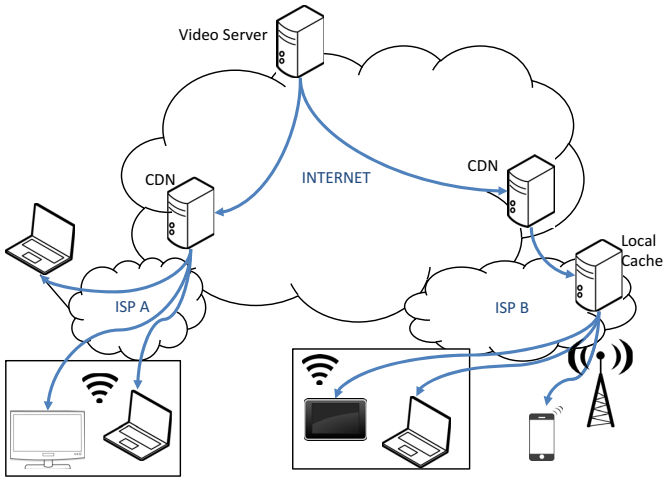


Figure 3.1. An overview of the Internet showing origin Video servers that host Live/On demand video. The video is distributed using CDNs, and in some cases cached at dedicated servers in the ISP for higher performance.

describe our large-scale active measurement framework for YouTube as a service in Section 3.3. In Section 3.4, we discuss suitable durations for active video testing and the factors that influence their choice. The chapter is concluded and summarised in Section 3.5.

3.1 Large-scale Measurement of Broadband Performance

The interest in measuring broadband performance is shared amongst users, operators, and regulators. Performance indicators have long been collected and analysed by Internet providers for benchmarking and troubleshooting. More recently third party measurement platforms (e.g. SamKnows, BISmark, RIPE Atlas, Netradar, etc.) have emerged that provide a more neutral perspective on Internet performance to consumers and regulators [7, 91, 85, 16]. Regulatory bodies such as Federal Communications Commission (FCC) in the US [92] and Office of Communication (Ofcom) in the UK have also been involved in gathering datasets to compare broadband performance across Internet providers and formulate better future policies [9]. An IETF working group for Large-Scale Measurement of Broadband Performance (LMAP) was concluded in 2015 providing informational guidelines for performance measurements from a large number of broadband access devices [6].

Measurement platforms rely on commonly used measurement tools and metrics to evaluate performance (e.g. iperf, traceroute, ping, etc.) or specifically designed tools for the specific platform. The IETF IP Performance Metrics (IPPM) working group updated the IP Performance Metrics Framework in RFC 7312,

which is a set of informational guidelines on testing, measurement methodologies and subsequently benchmarking of IP networks. The RFC proposes that metrics should be repeatable and actionable. Repeatability implies that if the test is repeated under similar conditions, the measurements collected should be equivalent. Actionable means that the metrics yield results that the users and providers can understand; they are able to identify problems and implement corrective measures if possible.

Performance indicators for video streaming are vital for benchmarking any Internet connection. Traditional metrics such as throughput and latency provide little insight about video streaming experience and there is a need to find application-level metrics that fill this gap. A typical video streaming service is distributed using an infrastructure similar to the one shown in Figure 3.1. The performance is best measured close to the endpoint since the service is affected by the distribution of video servers, CDNs and the health of the access network as well. Measurements at the endpoint are affected by th variety of end devices, home WiFi network issues and the consumer's usage patterns. Service-specific passive measurements are commonly used as video performance indicators at the endpoints, which more accurately capture the experience of real users[1] [59]. As already mentioned, this thesis focuses on active measurements from the home gateway. Active measurements at the home gateway are easier to manage and interpret in comparison to testing from the endpoints, as remote updates and result collection pose lower privacy concerns and device uniformity can be ensured if needed. Furthermore, the absence of user device and WiFi related impact factors make such measurements more meaningful for Internet providers for reporting and troubleshooting.

In the next section, we discuss the video performance metrics that relate network and application level impairments to user experience.

3.2 Application/Network Metrics for Video

In Chapter 2 we talked about performance metrics for video streaming over HTTP. In the absence of packet losses, spatial distortions such as pixelation or missing frames are no longer caused by network impairments. Furthermore, the uncertainty of the effect of a packet loss on the video experience without input from the resource-consuming decoding process, for instance whether the packet loss affects a single or multiple frames, is also alleviated. The premise for video QoE remains unchanged, which is that the actual user experience will be influenced by a number of compounding factors including network, application, hardware and human components. However, it is easier to measure the impact

[1]Most service providers collect performance metrics that are used for internal troubleshooting and monitoring and, in some cases, may be shared with users. For instance, Google provides Video Quality Report to its subscribers: https://www.google.com/get/videoqualityreport/

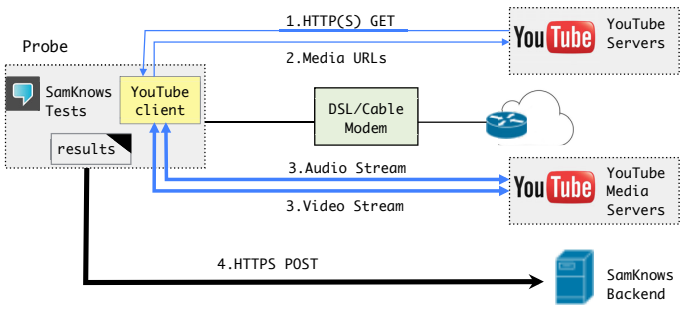


Figure 3.2. An overview of the YouTube test running on SamKnows probes for collecting performance metrics. The collected measurements are pushed to SamKnows data collection servers for aggregation and analysis.

of the network factors in HTTP video streaming with lower computational and memory requirements and have more actionable results.

Common application level metrics that are used to evaluate the performance of HTTP video streaming are listed below. While these metrics are affected by higher layer latencies, such as decoding and rendering, etc., in the context of Internet performance, we focus on the network related aspects only.

- Startup Delay is the click-to-play latency in video clients. For a smooth playback, video applications only begin playout once they have buffered a certain amount of video, known as prebuffering. High network latency results in a longer startup delay as well as slow loss recovery through retransmissions.

- Stalls occur in the event of a buffer underrun. If the throughput is lower than the stream bitrate, the playout buffer may become empty and cause the video to stall. Playout resumes once there is enough data in the buffer to play. Both the number and duration of stalls are important indicators for video experience.

- In adaptive streaming, the ability of the client to achieve a high download rate shows its adaptability to changing network environments, whereas the ability to avoid oscillation between two rates indicates its stability; both parameters have an impact on video experience. The number of bitrate switches and the average download bitrate may be used to quantify stability and adaptability respectively.

Research studies have shown the relation of all of the above-mentioned metrics with QoE [31, 38, 59, 43, 22, 45]. Generally, stall events have been recognised to have the highest impact on experience. Furthermore, users prefer longer startup delays for a smooth and high quality video experience, making prebuffering an important aspect of HTTP streaming.

3.3 Large-scale YouTube Measurements

To bring video streaming to Internet performance metrics, we developed a lightweight test to measure YouTube performance for the SamKnows Internet measurement platform[2], which is illustrated in Figure 3.2. As shown, the test runs on the SamKnows' whitebox, a measurement probe with an OpenWRT firmware, which is connected to the home gateway over a wired connection. The test emulates a YouTube client by requesting a YouTube video from its servers using HTTP. The server responds with a manifest of the available formats, bitrates and locations of the audio and video streams. The client selects an appropriate quality of audio and video stream to download based on previous throughput measurements conducted on the platform. It then downloads both audio and video and emulates a playout to calculate performance indicators such as startup delay and stall events. The test is designed to run on small, computationally limited probes and hence does not perform any decoding or rendering of the frames.

SamKnows provides its measurement services to regulators, operators, end-users and researchers across the globe. The YouTube test has been deployed as part of the SamKnows measurement platform in several live Internet networks, providing measurements that encompass video performance. The test is scheduled to run on the probe at the beginning of every hour during peak hours and every six hours after midnight. The probe also runs other performance tests including a speed test for throughput measurements. The speed test measures achievable throughput by downloading a 1GB file from the nearest MLab [3] server over an HTTP connection. Speedtest results are used to guide the YouTube test to only attempt video downloads for quality levels that do not exceed the achievable throughput for the probe. The results are sent to the back-end servers for storage, compilation, and analysis at the end of each scheduled run.

We chose to measure YouTube service because of its popularity and registration-free access. The test is repeatable and provides actionable results. For instance when used by Internet providers for their consumer base, ISPs can isolate degraded performance or failures to a particular consumer, a geographical area or a particular server. Time logs and origin and destination IP addresses can be further used to timely identify and address issues. Individual users can also use performance statistics to make more informed decisions when choosing ISPs.

3.3.1 Performance Metrics

Information regarding timestamps and frame boundaries is extracted from the media container to emulate a playout for the video and accurately measure startup delay and stall events. To limit storage requirements, the data is

[2] http://www.samknows.com
[3] http://www.measurementlab.net

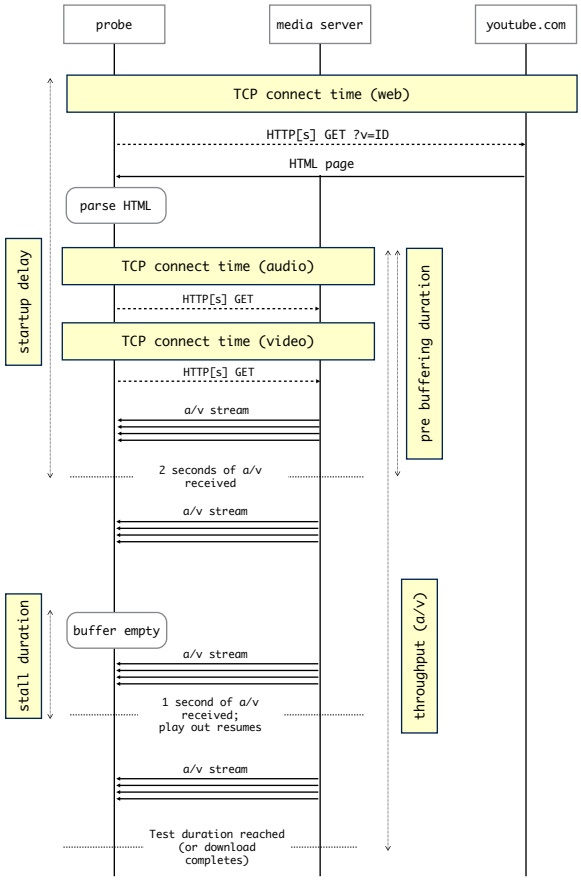


Figure 3.3. The sequence of actions within the YouTube test and the measurement periods of the different metrics that are collected. Throughput is measured separately for audio/video (a/v) streams as well as the overall throughput.

discarded after demuxing, however, the test maintains a playout timer. Startup delay is measured from the beginning of the test until 2 seconds of playable audio and video frames have been downloaded. At this point, the playout timer is started and stall events are recorded if the timer exceeds the playout time of the last received frame. In accordance with client behaviour, the stall ends when the client has rebuffered enough video to begin playout again. The test uses 1 second of rebuffered video as an indication to resume playout. Audio and video streams are synchronised using the timing information and content is considered playable only if both streams are available. This sequence of actions and the timing of the different metrics that are collected during the test are illustrated in Figure 3.3. As shown in the figure, the test also measures transport level metrics such as TCP connect times and throughput achieved for both audio and video streams.

Different video clients use different adaptation algorithms and these algorithms are evolving all the time. This makes the adaptability and stability of a video client, an application specific parameter. In order to remove this bias, we use the performance indicator *bitrate reliably streamed*: the highest stream bitrate that the test is able to download free of stall events. This is a deviation from adaptive clients, however, it makes the metric more useful and scalable for large-scale measurements.

3.3.2 Video Selection

YouTube offers different video streams in different formats and bitrates depending on the quality and nature of the original content and the preferences of the users. In order for the test results to be comparable across different probes, the selected video content needs to be the same. The YouTube API is used to fetch a list of globally popular videos, rejecting any video that is shorter than 60 seconds, is not available in Full HD, or is regionally restricted. This list is fetched by the probes from SamKnows' back-end servers on a daily basis. By using the most popular videos, the tests cover the content that is most watched and can keep up with the changing content and interests of the users over time. Hourly trends are better observed and compared if the same video is used throughout the day, especially because metrics such as throughput and video bitrate are influenced by the choice of the video and its available representations.

3.3.3 Throttling Download

Users often abandon videos in the middle instead of watching them to the end. Downloading the entire video is thus wasteful in such scenarios. To limit this waste of network resources, YouTube and other HTTP streaming clients maintain fixed length buffers. Once the buffer is full, the video download is paused until the playback proceeds and the buffer has space again. Our YouTube test uses a 50 seconds buffer, which is similar to the buffer lengths of Safari and

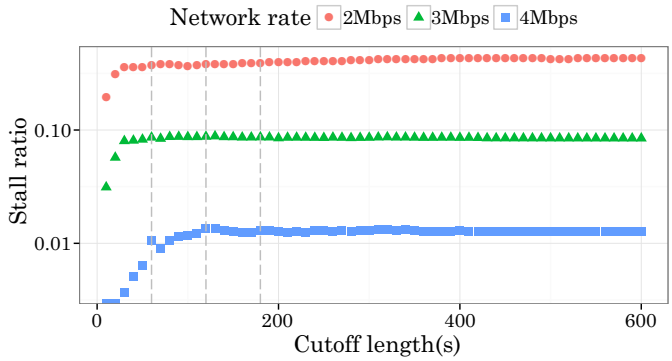


Figure 3.4. Results for the average stall ratio (number of stalls/duration of video) stabilise after 60s of running the YouTube test. The y axis uses a log10 scale to magnify the axis for the 4Mbps network.

Chrome players. Initially, the test requests byte range of 2.5MB until the buffer is full. It then requests 500kB byte ranges as and when the buffer empties.

3.4 Duration of Active Video Tests

Active video testing can be directly influenced by the presence of cross traffic on the home gateway and can also effect the experience of users who are using the network at the same time because of the bandwidth requirements for streaming multimedia. It is, therefore, important that the test duration is kept short. However, in order to gather meaningful results from a video streaming test, it needs to run long enough for the metrics to stabilise. In Publication I, we study a dataset of popular videos collected from YouTube to determine the minimum duration of an active video test.

Encoders try to keep the bitrate of the encoded stream constant on average but depending on the content, the instantaneous media bitrate is not constant. These fluctuations in bitrate can affect the download and, hence, it is important for the test to continue for long enough to compensate. Using clips of different durations from the videos in the dataset and comparing the instantaneous bitrate distributions of these clips with the entire video using the Kolmogorov-Smirnov (KS) goodness of fit test and the Autocorrelation Function (ACF) dissimilarity test, we observed that clips of 1 to 3 minutes matched their source video distributions reasonably well. The results were validated using our YouTube test in a simulated network environment, which showed that the extent of stalling in a video is more reliably measured when the tests ran for at least 1 minute and further improved for a cut-off of 3 minutes. Shorter test durations produced more optimistic measurements. Network characteristics were kept constant throughout the duration of the test as the study focused on the fluctuations in media bitrates. The aggregated results of multiple tests for 10 globally popular

YouTube videos of different genres and ranging in length from 15 seconds[4] to 95 minutes are shown in Figure 3.4. Network fluctuations can also impact test results, however, its affects are not limited to video streaming tests only and hence were considered out of scope in this study.

3.5 Summary

Large-scale active measurements for the monitoring and benchmarking of Internet have gained momentum and have garnered the interest of users, Internet providers, and regulators. In order to extend such measurements from the home gateway to encompass video streaming metrics that users can comprehend and network providers can act upon, we developed a YouTube test to measure one of the most widely used video streaming platform. We detailed in this chapter the design choices and the recommended duration of running active measurements using the test. The YouTube test that was developed as part of this thesis has been deployed as part of SamKnows' testing suite for Internet performance measurements. In the next chapter, we present the results of a dual-stacked performance study that was conducted using this test to compare YouTube performance over IPv4 and IPv6.

[4]For videos shorter than the cutoff time, the test simply ran to the end of the video.

4. Adoption of IPv6 in Streaming Services

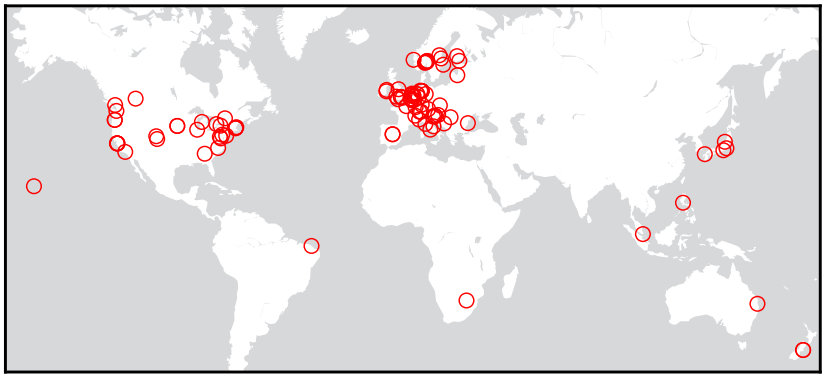
The World IPv6 Launch began in June 2012 with a number of large Internet service providers, device manufacturers and web service providers coming together to enable IPv6 support[1]. As ISPs start offering dualstack gateways to users, more and more users shift towards IPv6 for video streaming. Algorithms such as Happy Eyeballs (HE) [96] use connection establishment times to determine the suitability of an IPv6 destination to ensure that the upgrade towards IPv6 does not negatively impact user experience; however, the policy is limited to the connection establishment phase. While such steps are understandable at this point where the goal is to gradually move towards IPv6, it is worthwhile exploring what kind of streaming experience is offered to early adopters of IPv6 in comparison to the IPv4 users.

In Publication II and III, we present the results of a YouTube measurement study that compares performance of IPv4 vs. IPv6 on geographically distributed dual-stacked endpoints. The measurements were conducted using the YouTube test presented in Chapter 3. The study reveals the disparity in the distribution of Google Global Caches (GGC) that serve YouTube media streams for the two address families.

The measurement study, which lasted for over 34 months, was conducted using SamKnows probes connected to dual-stacked home gateways by volunteer users. The study began in September 2014; the preliminary results for a month long period from 21 probes were published in Publication II. By 2017, the number of probes increased to nearly 100 representing residential, National Research and Education Network (NREN), operator lab, business, and Internet Exchange Point (IXP) networks. The results of the extended study, which was concluded in June 2017, were published in Publication III. The geographical distribution of the probes as well as the number breakdown by network type and Internet registry is shown in Figure 4.1.

This chapter is organised as follows. A brief overview of the methodology is described in Section 4.1, note that the details of the YouTube test are already covered in the previous chapter. The HE algorithm and its effects in the light

[1] http://www.worldipv6launch.org



RESIDENTIAL	78
NREN / RESEARCH	10
BUSINESS / DATACENTER	08
OPERATOR LAB	04
IXP	01

RIPE	60
ARIN	29
APNIC	10
AFRINIC	01
LACNIC	01

Figure 4.1. Measurement trial of ~100 dual-stacked SamKnows probes as of Jun 2017. The separate tables represent the number of probes by network type (left) and by regional Internet registries (right). The metadata for each probe is available online: https://goo.gl/E2m22J

of the connection establishment times measured during the trial are discussed in Section 4.2. The YouTube related results are presented in Section 4.3 and the distribution of GGC in observed networks is discussed in Section 4.4. The chapter is summarised in Section 5.5.

4.1 Methodology

The study used the YouTube test presented in Chapter 3. SamKnows' back-end servers chose a video for testing from the most popular category based on its availability in Full High Definition (HD) and a minimum duration of 60 seconds. The probes retrieve the URL of the selected video every 12 hours. The test itself is repeated hourly during peak hours and attempts to download the video first over IPv4 and then over IPv6. After midnight the frequency of the test is reduced to once every 6 hours. The highest requested bitrate during the test was limited by the highest recorded throughput measured by the probes. In case of stalls, the test is designed to report failure and re-attempt audio/video download at a lower quality until it either has a stall-free run or it reaches the lowest quality in which case it continues to run for a minute and reports the number and duration of stall events. We used a playout buffer length of 40 seconds during the measurement trial.

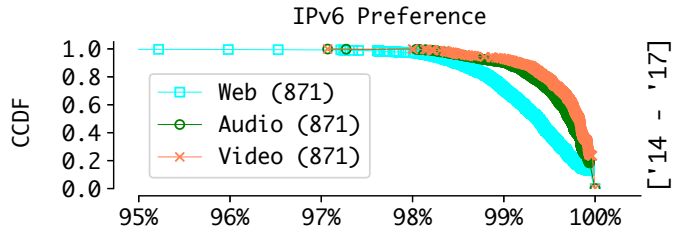


Figure 4.2. CCDF of TCP connection establishment preference over IPv6 observed during the trial. Applications with HE algorithm prefer a connection over IPv6 if the TCP connect times are no more than 300ms higher than TCP connect times over IPv4. At least 97% of the times, an IPv6 connection was preferred for all streams.

The study measured TCP connection establishment times to the YouTube web and media servers over both families in addition to video related metrics such as startup delay, highest achieved throughput, and stall durations where the test experiences stalls for even the lowest available quality of the video. The collected data is post-processed for aggregation and further analysis.

4.2 Happy Eyeballs and IPv6 Preference

As the Internet moves towards IPv6, the problem of broken or unavailable IPv6 connectivity can deteriorate performance of dual-stacked hosts. The destination address selection policy in RFC 6724 mandates `getaddrinfo()` to resolve DNS names in an order that supports an IPv6 upgrade. However, it is important that a switch to IPv6 does not come at the cost of user experience. The HE algorithm, which is implemented in most modern browsers like Google Chrome and Mozilla Firefox, is used to prevent this [96]. The algorithm initiates a connection to the first returned address, presumably an IPv6 address in a dual-stacked host, and if the connection does not complete within a short time, another connection to the first address belonging to the other address family is initiated. The application then uses whichever connection is established first and discards the other. Both previously mentioned browsers use 300ms as the allotted time for the first connection to complete before initiating a connection to the second address. This gives a 300ms advantage to IPv6; an acceptable delay to aid the upgrade without seriously compromising user experience.

We measured TCP connection establishment times for web, audio and video servers over both address families and observed that IPv6 connections were able to establish a connection within the 300ms advantage it is offered through HE. In 97% of the connections, IPv6 would be preferred over IPv4 for both audio and video streams according to the HE algorithm as shown in Figure 4.2. This is a desirable result for the IPv6 upgrade scenario. This reiterates the importance of IPv6 testing for video streaming, which could suffer from such a preference when the achievable throughput is not comparable for both address families.

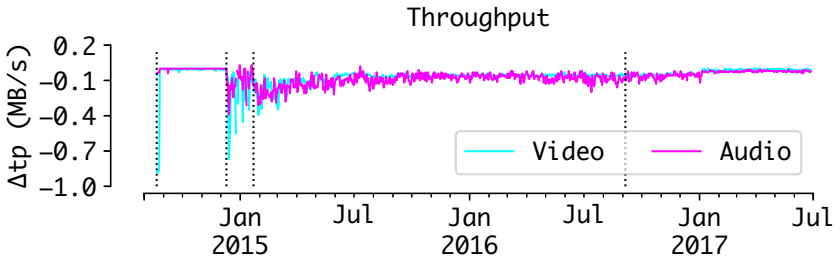


Figure 4.3. The difference in achieved throughput over IPv4 and IPv6 during the duration of the trial. IPv6 probes reported lower throughput, however, it improved over time.

Results also show that connect times to the media servers were slower over IPv6 63% of the times and the difference was over 10ms in 14% of the test instances.

4.3 YouTube Performance on Dual-stacked Hosts

We discussed in previous chapters that the primary indicators of video experience for Adaptive HTTP streaming are stall events and the stream bit rate. We use the mechanisms explained in Section 3.2 to measure *stalling* and *bitrate reliably streamed*, where the latter is the highest stream bit rate that the test was able to download without stall events. Since the available stream bit rates vary from video to video, the value of bit rate reliably streamed needs to be normalised for comparison across tests. We use the ratio of the bitrate reliably streamed to the highest available bit rate of the stream so that the observed value lies in the range of 0 to 1. We note that for nearly 95% test cases the ratio is 1 for IPv4, implying the highest available bit rate was downloaded without experiencing any stall events. The ratio is 94% for IPv6, which is only slightly lower and still comparable to IPv4. For 90% of the probes, the number of test results with stalling was under 1% for the entire measurement trial. Achievable throughput values towards media servers were found to improve for IPv6 over time but remained slightly lower in comparison to IPv4 as shown in Figure 4.3. The figure shows the difference in median throughput observed during the whole day for each address family, with positive scale denoting a higher IPv6 throughput and negative scale denoting a higher IPv4 throughput. Audio and video streams are shown separately.

4.4 Content Caches

To investigate the availability of content caches, we performed reverse DNS lookup on the IP addresses of the media servers that were used by the probes over both address families using keyword filtration to identify servers that were GGCs. GGCs are servers hosted by Internet operators within their own networks

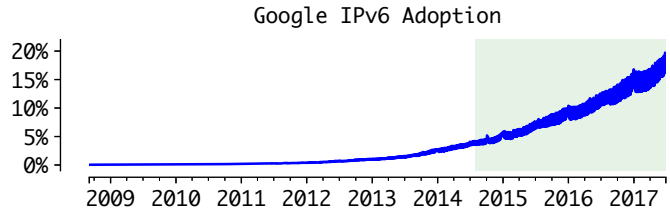


Figure 4.4. Percentage of IPv6 users accessing Google services [34]. The shaded area represents the duration (Aug 2014-Jun 2017) of this measurement study.

that serve as content caches for Google services, moving content geographically closer to the end user and reducing transit traffic. We observed that a GGC served content over IPv4 for 97% of the probes compared to 5% in case of IPv6. We noted before that despite slightly longer startup delays (~100ms) and lower throughput values, the aggregate results for YouTube experience (in terms of stalls and bitrate reliably streamed) did not favour IPv4 substantially enough for a migration towards IPv6 to be a cause of alarm for YouTube users for now. However, the absence of locally served media content over IPv6 is worrisome. Trends indicate that IPv6 users are increasing at a higher rate than before as indicated by the rising number of IPv6 Google users in Figure 4.4, which shows the percentage of Google users during the last decade. As IPv6 traffic increases, the absence of local GGCs to serve content can lead to an increased load on the transit network. Thus, the absence of dual-stacked GGCs needs to be addressed by operators that offer IPv6 services.

4.5 Summary

In this chapter, we discussed the results of a YouTube measurement study for dual-stacked probes. We examined the results in light of the HE algorithms and determined that the hosts located at the test home gateways would establish an IPv6 connection for their YouTube streams 97% of the time. A comparable achievable throughput for both families reinforces the notion that IPv6 users are not at a serious disadvantage, however, the disparity in the availability of GGCs on IPv6 does affect the performance for some probes. About 97% of the probes were served content from a GGC over IPv4 in contrast to a minimal 5% over IPv6. A timely switch to dual-stacked GGCs by ISPs can reduce connection latencies and ease transits by serving the content locally.

For the remainder of the thesis, we expand our focus from measuring video performance towards improving streaming techniques; this entails changes at both the application and protocol level. In the next chapter, we explore how adaptive streaming over unordered TCP can be used to reduce stall events, one of the primary causes of diminished user experience for video streaming.

5. Treading the Ossified Internet

We have discussed in previous chapters about the pervasiveness of Adaptive streaming over HTTP in the current Internet, which is widely used for VoD and Live streaming for a variety of content. Its success, as previously mentioned, is attributed to the well-established HTTP with its large content distribution infrastructure, reducing deployability time and costs. However, streaming over TCP has a higher latency due to its congestion control and strictly reliable design. High speed Internet combined with prebuffering and quality switching minimise the effects of this latency. Nonetheless, quality degradation and subsequent stall events are at times unavoidable in the presence of high packet loss or rapidly changing network conditions, significantly lowering user experience. We propose that some of the simplicity of the HTTP adaptive streaming solution can be traded for a lower stall ratio for such network conditions by using unordered delivery and removing the head-of-line blocking that is inherent to TCP.

In Publication V, we evaluated DASH streaming over TCP Hollywood; a TCP variant that provides partial reliability and out of order delivery to benefit time sensitive data. TCP Hollywood is wire compatible with TCP, alleviating any deployability issues due to Internet ossification. We used out-of-order delivery in TCP Hollywood and observed reduced stall events for DASH streams in high latency and high loss networks in comparison to TCP. In this chapter, we discuss the setup and results of our experiments as well as the implications of an unordered transport layer for DASH systems.

The rest of the chapter is organised as follows. In Section 5.1, the reader is familiarised with the design of TCP Hollywood. Steps needed to adapt DASH to TCP Hollywood are discussed in Section 5.2. Section 5.3 explains the modifications we made to the DASH adaptation algorithm to handle unordered delivery over TCP Hollywood. The results of our experiments are presented in Section 5.4 and the chapter is summarised in Section 5.5.

5.1 TCP Hollywood

In §2.3.1 we discussed deployability issues associated with new protocols due to transport layer ossification. Consequently, Internet applications are limited to the use of either UDP or TCP on the transport layer. While solutions such as RTP for real-time data exist, the protocol runs on top of UDP/TCP, implementing additional features on the application layer. Motivated by this problem, TCP Hollywood is a message-oriented transport layer protocol for low latency operation that preserves wire compatibility with TCP. It is, therefore, readily deployable in the current Internet as confirmed by studies carried out in the UK [54, 55]. We explain the major design components of the protocol below: message abstraction, multistreaming, inconsistent retransmission, and unordered delivery.

5.1.1 Message Abstraction

TCP Hollywood is implemented at the kernel level on top of the TCP layer. A send operation on a TCP Hollywood socket creates a single message, with leading and trailing markers for boundary detection. It is then encapsulated in a TCP segment for transmission. The message abstraction is required for partial reliability and unordered delivery. Congestion control and acknowledgments are the same as TCP. The leading and trailing markers preserve message integrity despite any resegmentation and segment coalescing during transmission and are used to reconstruct the message at the receiver end. A receive operation on a TCP Hollywood socket would read messages that have been received in their entirety; partially received messages are not handed to the application. The size of a message is determined by the application based on the use case.

5.1.2 Multistreaming

TCP Hollywood supports multistreaming, where multiple substreams can be multiplexed into a single transport-level connection using substream identifiers assigned by the sender. For example, senders may choose to use different substreams for multiplexing audio, video, and subtitles. In our own experiments, we used multistreaming for control and data traffic, i.e., separate substreams for HTTP request/response and DASH encoded chunks.

5.1.3 Inconsistent Retransmissions

TCP Hollywood uses inconsistent retransmissions to prioritise timeliness over reliability when needed. Senders assign an expiration time and dependability information to each message. For example, in case of video the expiration time is the playout time of the frame that the message is composed of and the dependability information is based on the inter- and intra-frame dependencies within the encoded stream. If the message is lost, the sender will only retransmit

it if the message itself or the messages dependent on it are still playable, i.e., their expiration time has not exceeded. In order to preserve wire compatibility with TCP, however, the sender must still send a retransmission when a segment is lost. This is done by replacing the contents of the retransmitted segment with new data while keeping the same TCP sequence number.

5.1.4 Unordered Delivery

TCP Hollywood receivers may use unordered delivery; complete messages are delivered to the application in the order of arrival. Sequencing and reordering of the data is handled at the application layer. This eliminates any latency introduced due to head-of-line blocking on the message level. However, as noted already, partially received messages are not delivered to the application and hence, careful thought must be given when deciding message boundaries and sizes. In the next section, we discuss the protocol in the context of DASH streams; message sizes, reordering, and adaptation algorithms are considered.

5.2 DASH over TCP Hollywood

Adaptive streaming is not generally designed for time constrained operation and finds most of its use cases in VoD and Live streaming, which are both tolerant of at least a few seconds of delay, the former more so. Nevertheless, it is currently the standard way for video streaming and has found use cases beyond conventional video streams with several video streaming services including YouTube offering 360-degree videos, and ongoing research on improving delivery of 360-degree videos using DASH [67][48]. Optimising DASH to deliver video at lower latency with more responsive adaptation is, hence, a need of the hour.

While TCP Hollywood's time saving features offer little benefit to conventional DASH videos (Standard Definition (SD), HD, etc.) on stable, high speed, and low latency networks, there is still a case for situations where latency or retransmissions are high or where the bandwidth is unstable leading to miscalculations by the adaptation algorithm in selection of suitable representations and consequently producing stall events. A partially reliable and unordered transport such as TCP Hollywood can prevent unnecessary stalling in contrast to the strictly reliable and ordered TCP.

Different aspects of creating a DASH system over TCP Hollywood and the setup used during our research as well as the rationale behind it are discussed below.

5.2.1 Partial Reliability and Content-Awareness

Partial reliability in TCP Hollywood requires a content-aware server. Assigning timestamps and dependability information to each message requires the server

to read the video stream prior to transmitting it. Some degree of synchronisation between the server and client is also required so the server can accurately estimate when a message is expired given client-side playout delay: the playout time of a frame is directly dependent on when the client begins playing out the frames. Pausing, rewinding, and other play functions can alter these timing even if such a synchronisation is done in the beginning.

DASH servers are simple HTTP servers; the process is client-driven and the server is content-agnostic. Client-side operation is a key feature of any HTTP adaptive streaming technique. For this reason, in our research we chose to preserve DASH and not use partial reliability in TCP Hollywood. Messages were sent without expiration time, which meant that all messages were always delivered to the client and there were no inconsistent retransmissions. Despite the fact that partial reliability is a prominent feature of TCP Hollywood, the aim of the study was not to analyse TCP Hollywood, but to reduce stall events in DASH streams by eliminating head-of-line blocking while still maintaining deployability. As we will show later, just the elimination of strict reliability helped achieve this goal. Note that inconsistent retransmissions in adaptive streaming with close to real-time latencies is something to be pursued as future research.

5.2.2 Unordered Delivery

DASH clients, like all video players, have playout buffers that are used for prebuffering received data. However, the clients expect reliable, ordered streams. In order to support unordered delivery, the function of the playout buffer has to be extended to encompass reordering. Messages need to carry sequence numbers and stream offsets so that the client can place them in the correct order. In order to eliminate head-of-line blocking completely, the client must also be able to discard a message that is delayed and continue playout with the data that it has received. For this reason, it is important to use a container format that is loss tolerant. The most commonly used DASH video format is MP4, which is not loss tolerant. Therefore, clients must use MPEG-TS as it tolerates missing bytes within the stream. MPEG-TS is less common but DASH compliant and is extensively used by Apple's HTTP Live Streaming (HLS).

In addition to the elimination of head-of-line blocking within the video stream, a DASH client's access to data as it arrives also allows adaptation algorithms to be more responsive to the streaming process. In our experiments, we found that the greatest advantage comes from the ability of the client to request future chunks while the current chunk was still being downloaded. Requesting future chunks is also possible with TCP using HTTP pipelining and bundle requests in HLS. However, both approaches request multiple chunks together based on the current adaptation algorithm estimations, which may become outdated on a rapidly changing connection. Our approach over TCP Hollywood requests only the next chunk while the current one is being downloaded, so the estimations

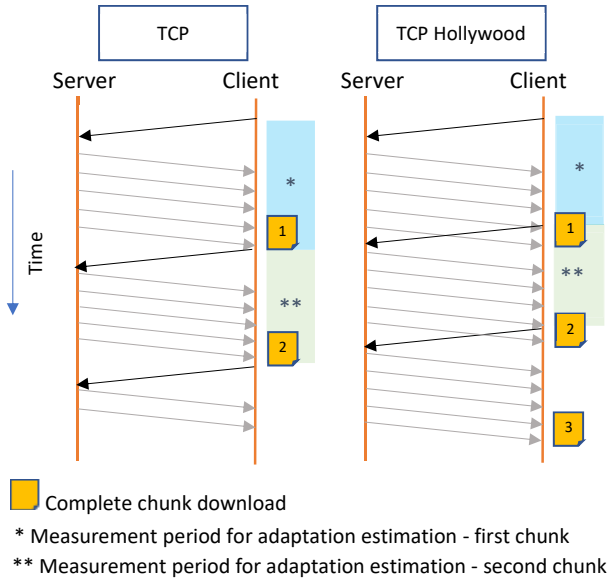


Figure 5.1. An illustration of chunk download with HTTP over TCP and over TCP Hollywood. The TCP Hollywood client requests the next chunk while the current chunk is still downloading. The shaded area represents the measurement period for each chunk.

are more accurately matched to current network conditions. This is illustrated in Figure 5.1 with a side-by-side comparison to TCP.

5.2.3 Message Boundaries

As already mentioned, message size is determined by the application. In the case of video, the boundaries can be aligned with frame boundaries with the caveat that at higher resolutions, frames can be several hundred or thousand kilobytes in size. A partially received frame is not handed to the application and may expire in the process, which is acceptable if the decoder is not capable

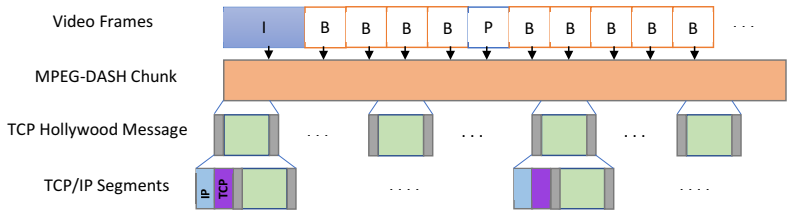


Figure 5.2. Illustration of the transport-layer encapsulation used for transporting DASH over TCP Hollywood

Algorithm 1 BOLA rate adaptation under TCP

```
1:  procedure DOWNLOADSTREAM
2:      n ← index of current chunk
3:      N ← Total number of chunks
4:      B ← Maximum Buffer Length
5:      b_now ← Current Buffer Length
6:      S_n^q ← Size of chunk with index n and quality q
7:      B_n^q ← Bytes received for chunk with index n and quality q
8:      while n < N do
9:          if b_now > B then
10:             Δt ← b_now − B
11:             WaitForDuration(Δt)
12:         end if
13:         q ← GetBolaQualityEstimate(b_now)
14:         DownloadChunk(q,n)
15:     end while
16: end procedure
```

of handling partially received frames. Applications that support error recovery from partial frames using Forward Error Correction (FEC) or other schemes such as the ones used for RTP [24], would benefit more from using smaller message sizes to minimise the effect of a lost or delayed TCP segment.

MPEG-TS containers also used FEC with a packet size of 188 bytes. Aligning TCP Hollywood messages with MPEG-TS packets is another option, but given the small packet size, overhead should be taken into account. Note that message boundaries based on the video stream frames or MPEG-TS packets require content awareness. In our own implementation, we used a fixed message size of 1400 bytes to preserve the content-agnostic nature of the server. A representation of TCP Hollywood messages in the context of chunked video used in DASH and other HTTP adaptive streaming is shown in Figure 5.2. An MPEG-DASH chunk begins with an I-frame, as shown, followed by other types of frames. TCP Hollywood messages (having leading and trailing boundary markers) are shown as fix-sized in this case, encapsulated as TCP/IP segments. In order to preserve chunk boundaries at the receiver end and ensure seamless switching, TCP Hollywood messages do not overlap with the chunk boundaries, i.e., no message contains data from more than one chunk. Note that a server does not need to be content-aware to ensure this behavior as DASH chunks are generally stored as separate files. The server simply needs to ensure that the TCP Hollywood messages are terminated at the end of the file even if they are less than 1400 bytes.

In the next section, we discuss adaptation algorithms and how existing algorithms can be modified to work with an unordered and unreliable transport layer.

Algorithm 2 BOLA rate adaptation under TCP Hollywood

```
1: procedure DOWNLOADSTREAM
2:     n ← index of current chunk
3:     N ← Total number of chunks
4:     B ← Maximum Buffer Length
5:     b_now ← Current Buffer Length
6:     S_n^q ← Size of chunk with index n and quality q
7:     B_n^q ← Bytes received for chunk with index n and quality q
8:     Rx_T ← Receive Threshold
9:     L ← Losses since last download
10:    T ← Duration of chunk
11:    while n < N do
12:        if b_now > B then
13:            Δt ← b_now − B
14:            WaitForDuration(Δt)
15:        end if
16:        if L > 0 and b_now > T then
17:            b_now ← b_now − T
18:        end if
19:        q ← GetBolaQualityEstimate(b_now)
20:        InitiateChunkDownload(q, n)
21:        WaitUntil(B_n^q < Rx_T * S_n^q)
22:    end while
23: end procedure
```

5.3 Adaptation Algorithms for Unreliable Transport

DASH clients must rely on an adaptation algorithm to measure and estimate the most suitable quality for the next chunk to be requested based on the available bit rates, past chunk qualities, and parameters such as length and fill of the playout buffer. Different categories of adaptation algorithms and their operation is discussed in Section 2.3.3. These algorithms generally neither expect nor account for missing data in the stream and, thus, have to be modified to do so in a TCP Hollywood DASH client.

As a first step towards developing better adaptation algorithms for such use cases, we modified the widely used adaptation algorithm, Buffer Occupancy based Lyapunov Algorithm (BOLA). BOLA is a buffer-based algorithm that performs well in a variety of network conditions [87, 44]. Algorithm 1 shows the download operation with BOLA in a TCP DASH client. Since the estimation process is retained as is for TCP Hollywood, we indicate it as a single call to the function $GetBolaQualityEstimate$. The download process consists of a loop over the total number of chunks, N, sequentially downloading them one by one until the end of the stream. The current chunk, denoted by the index n, is downloaded in the quality q, where q is estimated by the BOLA algorithm on the basis of the current buffer length b_{now}. If b_{now} exceeds the maximum buffer length, B, the download is stopped for the buffer to sufficiently empty to accommodate the next chunk.

We introduce two major differences in the TCP Hollywood client's download logic. Firstly, when a message is delayed enough to be discarded, it constitutes a loss and the player reduces the size of the effective buffer length, b_{now}, that

is passed to $GetBolaQualityEstimate$. This allows for a more conservative estimation. Secondly, in most TCP clients, the quality assessment of the path and estimation is made at the end of the chunk download, however, waiting for the entire chunk to arrive before proceeding creates head-of-line blocking on a chunk level. To eliminate it, we used a parameter receive threshold, Rx_T. The client continues to download the chunk n at quality q as long as the ratio of bytes received, B_n^q, to the size of the chunk, S_n^q, is less than Rx_T. In other words, Rx_T is the fraction of the current chunk download that must be done before quality estimation of the next chunk. Once Rx_T is reached, the download is placed in the background thread and the client estimates the quality for the next chunk on the current network metrics and immediately sends the HTTP request for the next chunk as well. Note that the server is still sequential, so the chunks will still be downloaded sequentially, however, the server does not have to wait for the client to send the next request as was previously shown in Figure 5.1. In our experiments, we used an Rx_T value of 90% which was chosen based on trials with different values. Algorithm 2 shows the modified version for TCP Hollywood.

5.4 Experimental Results

In Publication IV, we created a framework that experiments with the usability of TCP Hollywood as a transport layer alternative for DASH. We have discussed the different aspects of DASH over TCP Hollywood already in this chapter and the rationale behind the setup used in our own experiments. To summarise, we retain the client-driven operation of the DASH technique and use regular HTTP servers in order to preserve the simplicity of DASH. This implies that the inconsistent retransmissions that require the server to insert timing information to the messages were not used. Furthermore, since HTTP servers are not aware of the content being served, frame boundaries or packet boundaries of the underlying MPEG-TS format were not used to align TCP Hollywood message boundaries. Fixed size messages were used instead, which keep the server content-agnostic. Finally, using the modified adaptation algorithm, the TCP Hollywood client sent requests during the download of the current chunk eliminating the off period between chunk downloads. Finally, HTTP requests and responses were sent using a separate TCP Hollywood substream than the one used for DASH chunks.

We ran experiments in an emulated network environment with a single DASH client and server and compared performance over TCP Hollywood with TCP. Test scenarios included i) stable network conditions with varying degrees of loss and network latency and ii) varying network conditions with network parameters changing every 30 seconds. We found that TCP Hollywood was able to reduce stall events in high latency and high loss networks by eliminating the delays involved in loss recovery. Both SSIM and PSNR values showed that the level

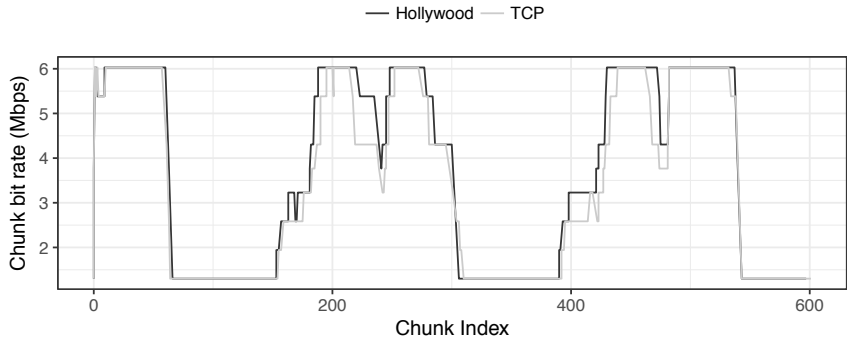


Figure 5.3. TCP Hollywood allows the adaptation algorithm to adapt more quickly to higher bit-rates and also to sustain them for longer periods in the presence of network degradation.

of visual distortion despite dropped bytes remained similar to DASH over TCP and startup delay values were similar for both protocols. TCP Hollywood client was also able to maintain a higher quality, however, in a few cases this meant more quality fluctuations. A timeline graph of a sample network where network characteristics changed every 30 seconds is shown in Figure 5.3. It can be seen that TCP Hollywood client can react faster to an increase in throughput and move to a higher bit rate stream and maintains the higher bit rate for longer in the presence of network degradation.

5.5 Summary

In this chapter, we presented our work on DASH over TCP Hollywood, a message-oriented, partially reliable protocol that is wire compatible with TCP to ensure deployability in an ossified Internet. We showed that stall events for DASH application can be reduced by the removal of head-of-line blocking 1) at the transport level by using TCP Hollywood and discarding delayed messages and 2) at the application level by modifying the download process of a DASH client. The former of these comes at the cost of visual impairments, however, our experiments produced similar objective QoE, SSIM and PSNR, for both protocols.

The research, which applies not only to TCP Hollywood but also to other unordered or unreliable/partially reliable protocols, has potential for future work in a number of directions. Visual impairments due to discarded/lost packets can be further minimised by preventing the clients from discarding I-frames. This can be done based on information from the demuxer/decoder or the knowledge that the beginning of a chunk is always an I-frame combined with rough estimation of the size of an I-frame may also significantly improve performance with very little complexity. Extensive research on adaptation algorithms is also required to fully utilise the advantages of TCP Hollywood in

adaptive streaming, similar to the one done for TCP based adaptive streaming.

In the next chapter, continuing the discussion on unreliable video streaming, we present a multipath solution for RTP to maximise throughput between a sender and a receiver.

6. Pooling Multiple Paths

Mobile and fixed network devices, nowadays, have access to heterogeneous networks such as WiFi, 3G, and/or LTE. An endpoint that is able to use these networks for a single connection for redundancy or bandwidth aggregation is known as a multihomed endpoint. When two or more disjoint paths exist, multihomed endpoints can pool the bandwidth resources of the available network interfaces for redundancy, load sharing, or increased throughput. Even partly disjoint paths can be utilised as long as they don't share a common bottleneck. To address the requirements of real-time traffic, we developed a multipath protocol, Multipath RTP (MPRTP), which is an extension of RTP for real time data. Notable protocols that support multihoming other than MPRTP include Multipath TCP (MPTCP) and SCTP. MPTCP [29] is an experimental IETF effort that adds multipath capability to TCP. SCTP, previously discussed in §2.3.1, supports both multihoming and multistreaming (multiplexing multiple streams over the same connection) with extensions for multipath transmission [2, 64]. Both protocols offer connection-oriented reliable delivery with congestion control.

In this chapter we outline the design of MPRTP with considerations for senders and receivers and present the results from our own implementation of MPRTP. An overview of the advantages and challenges of multipath transport solutions is provided in Section 6.1. Design features of MPRTP are discussed in Section 6.2, followed by scheduling algorithms for traffic distribution on available paths in Section 6.3. Multipath RTCP (MPRTCP) reporting and packet skew estimations for playout are discussed in Section 6.4. Finally, an evaluation of the results from our own experiments is provided in Section 6.5. The chapter is summarised in Section 6.6.

6.1 Advantages and Challenges of Multipath Transport

There are several advantages of multihoming including increased availability, reliability, load balancing, and throughput. Multipath transmission solutions from the research community range from link, network, transport, application

as well as cross-layer levels [69, 51, 36]. It is especially considered beneficial to voice and video calls, which will ordinarily be dropped in the event of network failure but can use path redundancy for keeping the connection active. In the context of video streaming, which is a bandwidth-hungry application, bandwidth aggregation can offer a significant advantage. Multiple paths can increase the achievable throughput and support higher quality streams. They increase availability and prevent sudden quality degradation by seamlessly switching over from one path to another in case of failure. Additional paths may also be used for carrying supplementary, redundant or backup data, such as for sending low quality chunks in DASH applications to prevent stalling.

Despite the benefits, it is important to note that the network characteristics of different paths can vary significantly. Consequently, packets on different paths experience varying degrees of latency and packet loss. This affects the extent of reordering and jitter at the receiver end. Mutually independent data such as different objects on a webpage would not suffer significantly as long as the paths offer an overall lower latency or higher throughput for the application when used together. However, careful scheduling is required when transmitting mutually dependent data such as packets from the same stream or video and audio streams that need to be synchronised at the receiver. Multipath schedulers determine when and what kind of data is to be sent on which path and can benefit from application layer support to ensure higher performance [21, 99]. Schedulers also need to ensure that the benefit of using multiple paths is not lost and the overall performance is not better for a single path in comparison to multiple ones.

Other concerns include energy efficiency as using multiple interfaces simultaneously also increases battery consumption [33]. Furthermore, there are monetary issues involved as many mobile users may prefer to restrict usage to only the free WiFi network, when available, to save cost on the billable mobile data usage. Multipath protocols, thus, need to take into account user preference as well as performance benefits while scheduling traffic across multiple paths [37].

6.2 Multipath RTP

As previously discussed in §2.3.2, RTP is designed for transporting real-time data and is coupled with the control protocol, RTCP. RTP headers carry monotonically increasing sequence numbers and timestamps that are used for reordering and synchronisation at the receiver. RTCP reports are sent periodically by receivers to monitor connection health and are rate limited to never exceed a fraction of the RTP traffic.

MPRTP is a backward compatible extension of RTP that enables bandwidth aggregation for multihomed endpoints. MPRTP endpoints use subflows to send and receive data over multiple paths, where each subflow is assigned to a partic-

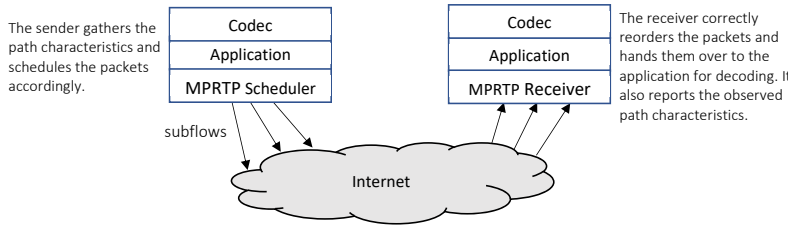


Figure 6.1. MPRTP is an RTP extension for resource pooling in the presence of multiple disjointed or partially disjointed paths between the sender and receiver. The figure shows client/server operation with unidirectional flow. The protocol is complemented with MPRTCP.

ular path. Senders distribute packets over the subflows, while MPRTP receivers reorder and recombine the packets from the subflows before giving to the application. MPRTCP reports from the receivers carry observed characteristics for each path back to the sender, which are used to monitor and characterise the paths and redistribute traffic accordingly. The operation of a MPRTP sender and receiver is illustrated in Figure 6.1.

RTP sequence numbers and timestamps remain unaffected by MPRTP. Fields specific to MPRTP are carried in the extension header. Each subflow has its own unique identifier known as the *subflow ID* and its own monotonically increasing *subflow-specific sequence numbers*. Each path carries its own MPRTCP reports, which together with *subflow-specific sequence numbers* allow endpoints to accurately measure path specific characteristics such as jitter, packet loss, and discards. MPRTCP reporting is restricted to a fraction of the media rate for the sake of scalability; no more than 2.5% of the media rate is recommended [61]. The limit applies to the combined MPRTCP reports from all paths and the reporting frequency on a particular path corresponds to the percentage of data that is sent/received over that path.

6.3 Scheduling Algorithm

MPRTP schedulers distribute the packets across the available paths and may be designed to maximise throughput, increase reliability or provide fallback. As part of our research, we developed a scheduling algorithm that aggregates bandwidth for increased throughput and reliability. In this section we outline the design considerations for such a scheduler.

At the start of the session, the sender does not have enough information about the paths. Assuming that the number of active paths is known, the sender can distribute the traffic equally over all paths. Alternatively, initial distribution may be based on location based historical data collected by the device during previous sessions, such as network performance of a home WiFi and 3G network.

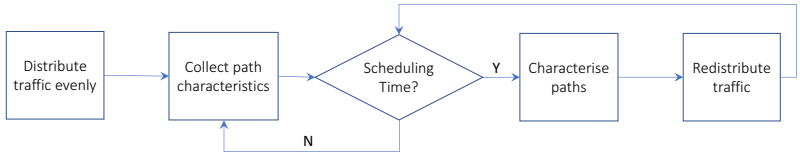


Figure 6.2. A flowchart of the operation of a MPRTP scheduler. Note that the scheduling interval varies throughout the session with quicker scheduling at startup and in the event of NACKs and path congestion.

The sender collects path characteristics from the MPRTCP reports. It must then characterise the paths according to the throughput, latency and losses that are observed on them. The traffic is then redistributed on the paths based on this characterisation. However, in order to avoid instability due to frequent traffic redistribution, the rescheduling frequency needs to be controlled. The operation of the scheduler is shown in Figure 6.2. A more detailed discussion on path characterisation, scheduling interval, and traffic distribution follows.

6.3.1 Scheduling Interval

When the initial distribution of traffic in a MPRTP sender is not based on ground truth (no prior insights about path characteristics are available), then the scheduling decision needs to be revisited sooner than otherwise. The scheduling interval (the time that the scheduler waits before it re-calibrates traffic distribution) should therefore be kept short in the beginning and gradually increased over time as the algorithm stabilises. The scheduling interval is again shortened if multiple paths become congested and a more reactive scheduler is needed to minimise losses. Similar to RTCP intervals, in order to prevent multiple senders from becoming synchronised, the value of the scheduling interval is randomised and must never be lower than the minimum MPRTCP interval and at least one MPRTCP report should have been received before traffic redistribution. In our implementation, we defined the scheduling interval $SchInt$ using the paramater $S_{interval}$ with the following equation.

$$SchInt = \gamma \times S_{interval}, \quad 0.5 \leq \gamma \leq 1.5$$

$$\gamma = 0.5 + rand(0.0, 1.0)$$

The factor γ is randomised by a factor between 0 and 1 and $S_{interval}$ is set to minimum RTCP interval at startup or in case of congestion and gradually increased to maximum RTCP interval.

6.3.2 Path Characterisation and Traffic Distribution

The sender estimates four path characteristics based on the MPRTCP reports: path latency (Round Trip Time), packets discarded by the receiver due to late

arrival, packets lost, and throughput. The bandwidth of paths with similar latencies is additive and, hence, MPRTP senders benefit most by scheduling data on paths with similar latencies in order to increase the aggregate throughput values. Based on the losses and discards the paths can then be categorised as one of the following :

- A path that reports no losses or discards is considered *uncongested*. These paths are preferred over others and should get the highest share of traffic. If possible, only uncongested paths should be used for I-frames, which have a higher impact on visual quality if lost in comparison to other types of frames.

- If discards and losses are observed in single or consecutive intervals, the path is characterised as *mildly congested*. Traffic share should be reduced for such a path to a fraction of the rate reported for it.

- If discards and losses are observed for three consecutive intervals, the path is considered *congested*. Again, the traffic share should be reduced to a fraction of the reported rate. A smaller fraction is used for congested paths in comparison to mildly congested ones.

- If random losses but no discards are observed on a path in successive intervals, it is considered *lossy*. Paths with higher loss rates should be avoided by the sender if the throughput requirements are met from other paths.

In the presence of reliable and low latency paths, the scheduler does not need to use lossy and congested paths as long as the throughput requirements are being met. However, in order to maintain seamless operation in case of network changes, a minimal amount of data should still be sent on congested routes to monitor path characteristics. This ensures that if the primary path deteriorates, the scheduler has the required information to redistribute the traffic. The reason for reducing traffic to a fraction of the reported rate is to help alleviate the congestion condition.

In real scenarios, the various paths would have different network latencies and while achieved throughput can be increased using multiple paths, the variability in latency also leads to higher levels of reordering and jitter values that need to be accounted for at the receiver end. In the next section, we discuss the receiver side operation for MPRTP.

6.4 MPRTP Receiver

The role of the MPRTP receiver in a multipath session is to recombine the incoming packets from different paths for playout and send per path performance feedback using MPRTCP reports. We look at each function in turn in this section.

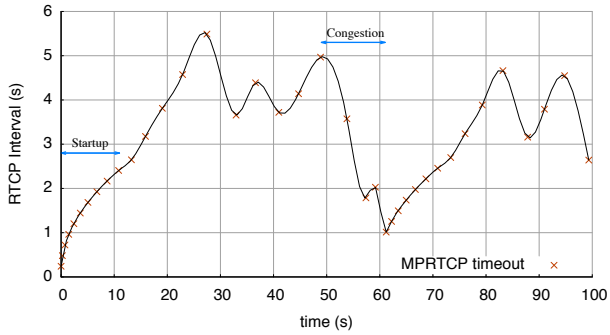


Figure 6.3. MPRTCP interval range is 5 ± 2.5 in stable conditions. from More frequent MPRTCP reports are sent in the beginning of a connection or in the event of congestion.

6.4.1 Multipath RTCP Reports

The feedback from the receiver forms the core of the MPRTP transfer as it is based on these reports that the sender will adjust traffic on each path to maximise performance. The scheduling interval, as discussed previously, also depends on MPRTCP reporting interval. It is, therefore, crucial that the reports are sent timely with accurate information for each subflow. Early feedback reports in case of severe congestion or at the start of the session should be used to timely correct scheduling rules at the sender [61]. However, the combined reporting frequency for all paths must not exceed 2.5% of the media rate in order to ensure scalability. Each path gets a share of this frequency based on its own receiving rate; paths with higher traffic share report more frequently and those with lower shares report less. Figure 6.3 shows the changes in MPRTCP interval in a multipath scenario. Startup and congestion at 60 seconds show more frequent reporting, which stabilises after about 10 seconds in each case.

6.4.2 Packet Skew and Playout

Another important aspect of the MPRTP receiver is the jitter and packet skew calculations. Packet skew is a consequence of the difference in timing clocks of the sender and receiver, while jitter is the variation in packet transport delay. From an implementation perspective, the values are calculated together by the receiver and used to compensate the playout delay at the receiver end, which we will refer to as skew here for simplicity. RTP implementations like `gstreamer` [1] use a windowed low point averaging technique [28] to prevent overestimation caused by temporary spike in skew due to congestion. However, in the case of MPRTP, low point averaging for all paths would lead to underestimations by disregarding paths with higher jitter and latency values.

We propose that for MPRTP receivers, skew is calculated on a per path basis

[1] An open source multimedia framework for streaming media: https://gstreamer.freedesktop.org

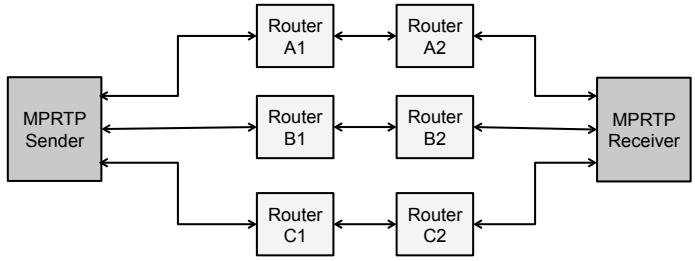


Figure 6.4. Evaluation Setup.

using a windowed low-point or other averaging technique. The per path skew can then be used to calculate an overall receiver playout delay also using a windowed technique but using the highest value instead of the lowest. The reason is that any temporarily affected high values are already excluded during the path level skew calculations. Using the highest value in the overall packet delay ensures that path skew for packets from all paths are compensated accurately and a single subflow with low skew does not negatively impact packets on other paths.

6.5 Evaluation

In Publication V, we presented the design principles of MPRTP and evaluated its performance for a video streaming scenario using a scheduling algorithm similar to the one explained above. The scheduler assigned equal traffic volumes to each path in the beginning to gather performance indicators. Once the paths were successfully characterised, the scheduler assigned minimal traffic on mildly congested and congested paths and distributed the remaining on the uncongested paths. Traffic distribution was adjusted by the scheduler regularly after the previously discussed scheduling interval elapsed. We show through testing that MPRTP performs better in comparison to single path RTP on any of the available paths.

The evaluation uses CBR video streaming using MPRTP over a virtual network. The endpoints are connected over three disjoint paths as shown in Figure 6.4 and tests are done under various emulated network conditions. Testing is done using static network conditions on paths as well as changing network conditions based on real 3G traces [84]. Experiments involving multiple 3G paths, shared bottlenecks between subflows of the same stream and between two separate MPRTP senders all reveal that the scheduler is able to balance load on the available paths and successfully shift load to alternate routes when a path becomes congested. Figure 6.5 shows the traffic distribution over time on two available paths, 3G and WLAN. WLAN is a 1Mbps link with 0.25% loss ratio while 3G is emulated [84] with a rapidly changing bandwidth and a loss rate of 1%. The graph shows that the traffic is distributed evenly as long as both paths have sufficient bandwidth for their traffic share, however, more load is shifted to

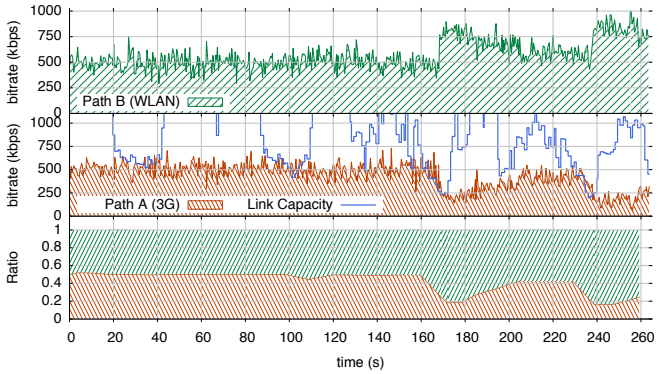


Figure 6.5. MPRTP traffic share across two paths in a test environment: a stable WiFi connection and a 3G connection whose capacity changes every second.

the WLAN when the 3G connection becomes constrained. Average PSNR values further showed that using multiple paths comes at acceptable costs to video quality.

Our MPRTP evaluation is limited to video streams with constant media rate and only provides a starting point. Future work with focus on conversational media and adaptive streaming is needed. Other future work consists of expanding mechanisms for congestion control and avoidance techniques, as defined by the charter of the RTP Media Congestion Avoiding Techniques (rmcat) IETF working group, to cover MPRTP. One such technique is using FEC-based rate control, which uses FEC packets for redundancy as well as to probe additional capacity on paths and subsequently increase media sending rate [47].

6.6 Summary

In this chapter, we presented MPRTP, an RTP extension that exploits multihomed endpoints to pool network resources in the interest of higher bandwidth and reliability. The extension operates between the RTP layer and the transport layer, distributing data from the application on multiple available paths while collecting path performance through MPRTCP reports from the receiver for each path and adapting the traffic distribution accordingly.

Multihoming and multipath protocols are expected to feature heavily in future 5G networks to keep up with the low latency, reliability and availability requirements of 5G [71]. Bandwidth requirements of future multimedia streams such as immersive video, augmented and virtual reality also need hosts with multiple connection to fully utilise the network capacity available to them. The research presented in this chapter is only one of several efforts to realise an efficient multipath solutions for the Internet. It addresses real time traffic, which is not directly covered by the MPTCP and SCTP multipath solutions. In

the next chapter, we conclude with a discussion on the implications, limitations and future work related to the research covered within this thesis.

7. Conclusion

The design of Internet and most of its protocols has been dominated by data streams. Video traffic has far exceeded web traffic in the current Internet and there is a need to approach future research with this in mind. Internet performance has been long evaluated using Internet speed, which is a useful but incomplete way of looking at Internet experience in the current dynamic where multimedia traffic dominates the web. Furthermore, the ossified Internet serves data traffic well enough but leaves a lot to be desired for video traffic. High speed Internet has made VoD and Live streaming over TCP an uncomplicated choice with good performance but its success should not limit new protocols to web data-like streams that are more beneficial to data than media.

In this thesis, we extended large-scale Internet measurements to include a YouTube streaming scenario so that Internet operators and users are able to perceive performance reports in the context of the actual services and experiences instead of speed numbers that offer little insight. YouTube and similar services are big contributors of Internet traffic and a test that quantifies its user experience already adds value to Internet performance indicators. However, the quality metrics need to be expanded further to encompass other applications such as conversational audio/video.

For a long time the Internet has been plagued by the problem of transport layer ossification, which has impeded development of more efficient transport protocols. More recently, Google has made way for innovation from within the company to resolve this through beta support of new protocols in their Chrome browser and enabling them on Google and YouTube servers. QUIC is one such solution, which introduces connection management, congestion control and multiplexing on top of UDP protocol [42]. The protocol is built to produce an HTTP over TCP+TLS like behaviour using UDP. QUIC is being developed for standardisation by the IETF QUIC Working Group in conjunction with the IETF HTTP working group. The third major revision for the HTTP protocol, HTTP/3 will be based on HTTP over QUIC. While QUIC reduces connection setup times and head-of-line blocking across multiple streams, it does not eliminate head-of-line blocking within the stream. Given the large amounts of media traffic in the Internet, it is important that as we find ways to improve transport design that

gives an equal consideration to media traffic as to other web traffic, if not more.

We experimented with two unreliable/partially reliable protocols that improved performance for video streaming. HTTP streaming is seen as the best solution in the current Internet, but traffic volumes are increasing at rapid rates. As video technology moves towards immersive 360-degree video, augmented and virtual reality, the demands on the network for delivering quality are increasing. It is important that we don't limit innovation to strictly reliable, content unaware transport solutions and build more intelligent solutions in addition to the simple ones to aid development. The experimentation with MPEG DASH over TCP Hollywood offers insights into DASH delivery over a partially reliable protocol. Ultra-low latency adaptive HTTP streaming using chunked transfer encoding that enables delivery of a media segment while it is being encoded has been able to produce startup delays under one second in a test environment [14]. The use of HAS for ultra-low latency offers an opportunity to use inconsistent retransmissions and partial reliability offered by TCP Hollywood to achieve even lower delay values as future work.

Protocols that can save in terms of unnecessary retransmissions through content awareness and reduce latency with unordered delivery offer great value to future networks. Both TCP Hollywood and MPRTP carry timing information, which offers some level of content awareness to the transport layer. MPRTP can further improve transport for bandwidth-hungry multimedia streams by pooling the bandwidth from multiple heterogenous paths, if they exist. While 5G promises higher-speeds, until such speeds are available, solutions like MPRTP can help with real-time emerging media technologies, such as, 360-degree conference calls. Working groups at 3GPP are already working towards 360-degree conference call support over mobile networks using RTP/RTCP.

This thesis aims to bring video traffic considerations to the design of Internet in general and not view it as a special case with limited use. It contributes by bringing video experience to large-scale Internet measurements and showing improvement in video streaming performance when using transport protocols that are designed to work with video. The rest of the chapter discusses specific practical implications and validity of our work.

7.1 Practical Implications

This thesis has contributed to an Internet draft for Multipath RTP and formed the basis of various studies that pool multiple paths for video streaming. The study on TCP Hollywood paves the way for unreliable HTTP streaming and can be extended to QUIC using multiple streams for unordered delivery. Our work on large-scale measurements has produced an active measurement test that is being deployed as part of the SamKnows testing suite through efforts from regulators, consumers and providers, giving rise to a new standard of performance metrics for the Internet. It has also lead to a study that identified

the lack of IPv6 media servers that serve content locally causing IPv6 YouTube traffic to flow over transit networks to fetch content directly from YouTube servers.

7.2 Reliability and Validity

All research carried out as part of this thesis has been done with great thought towards its usefulness to the research community. The relevant tests and frameworks, such as the YouTube test and the TCP Hollywood testing suite, have been open sourced for the public in the interest of repeatability. The dataset collected for the dualstack study of YouTube is also available online.

7.3 Limitations

The focus area of this research was video streaming, however, we see that the scope of the work that was done has applicability beyond that, which has not been explored. The active measurement-related results focused on YouTube videos and are, therefore, limited by their type of content as well as the effects of the YouTube distribution network. We also acknowledge that by making the active test for YouTube agnostic of the adaptation algorithm by not including one in the implementation, we are unable to capture the quality adaptation aspects, which are important for user experience.

The experiments with TCP Hollywood and MPTCP were conducted in emulated networks and real world testing with a broader scope is required to ascertain their applicability. Both evaluations did not consider the effects of congestion control and the TCP Hollywood study does not evaluate the effect of multiple streams using HTTP/2. Note that multistreaming in HTTP/2 eliminates application-level head-of-line blocking. This implies that while an HTTP/2 client may be able to download multiple chunks at the same time, it would observe stalls if the entire chunk is not downloaded in time to be played. Hence, even if only a few bytes are delayed within the chunk while the subsequent data has arrived, the strictly ordered and reliable TCP layer will not hand over this content to the application, unlike TCP Hollywood. The benefits of HTTP/2 and TCP Hollywood for ultra-low latency streaming are left for future work.

7.4 Visions for Future

We explored TCP Hollywood as part of this thesis to show that the removal of head of line blocking can benefit adaptive streaming. Initiatives such as QUIC and IPv6 deployment have the Internet gradually moving away from the ossification problem and we foresee that wire compatibility with TCP will

no longer feature as a design requirement for future protocols. However, TCP Hollywood does offer a solution that most newer protocols fail to address. It gives the application the freedom to enforce a varying degree of reliability and timeliness depending on the requirements.

In general, traffic on the Internet can be categorised into three groups depending on the level of reliability and timeliness they require. 1) Multimedia such as audio, video, online gaming, augmented/virtual reality and even images fall into a class of objects or streams that are tolerant to small degrees of loss and are more sensitive to time. Parts of such objects have a higher priority over others such as Field of View (FoV) [67] or I-frames and they have the tendency to lose relevance if delayed. 2) Web pages, style sheets and scripts on the other hand are loss intolerant and time sensitive in terms of experience. While the data may not lose relevance over time and latency requirements are not as short as those for online gaming or conversational video, increased delay significantly lowers user experience. 3) File downloads such as BitTorrent classify as traffic that is loss intolerant but time tolerant and hence often run as background services. Partial reliability with optional timelined data meets the requirements of all three categories. Whether these features come as core part of the protocol or through extensions [63], it is pertinent that past mistakes are not repeated and future infrastructures and networks are built with this flexibility in mind. Encrypted headers within QUIC protocol could prevent against ossification caused by middleboxes, however, deployability is also affected by infrastructure and API support, which is still possible.

The Internet is no longer a space dominated by a single type of traffic. As web moves towards HTTP/3, TCP is likely to be replaced with UDP as the common underlying protocol. This opens up space for a more application-oriented design. The vision we have for the future and the one that has fuelled this research is a flexible, cross-layer transport solution. Multihoming, partial reliability and application-oriented network measurements hold significance in the 5G and beyond 5G networks that would have multi-connected endpoints and strict requirements for throughput, latency and reliability. Our hope is that this work would benefit future research in these areas.

References

[1] 3GPP. Transparent end-to-end Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH). Technical Specification (TS) 26.247, 3rd Generation Partnership Project (3GPP), September 2018. Version 16.0.0.

[2] Ahmed Abd, Tarek Saadawi, Myung Lee, et al. LS-SCTP: a bandwidth aggregation technique for stream control transmission protocol. *Computer Communications*, 27(10):1012–1024, 2004.

[3] Saba Ahsan, Varun Singh, and Jörg Ott. Characterizing internet video for large-scale active measurements. *arXiv preprint arXiv:1408.5777*, 2014.

[4] Saamer Akhshabi, Lakshmi Anantakrishnan, Ali C. Begen, and Constantine Dovrolis. What happens when HTTP adaptive streaming players compete for bandwidth? In Baochun Li and Shervin Shirmohammadi, editors, *Network and Operating System Support for Digital Audio and Video Workshop, NOSSDAV '12, Toronto, ON, Canada, June 7-8, 2012*, pages 9–14. ACM, 2012.

[5] David Austerberry. *The Technology of Video and Audio Streaming*. Focal Press, 2nd edition, 2015.

[6] Marcelo Bagnulo, Trevor Burbridge, Sam Crawford, Philip Eardley, and Jürgen Schönwälder. A framework for large-scale measurements. In *2013 Future Network & Mobile Summit, Lisboa, Portugal, July 3-5, 2013*, pages 1–10, 2013.

[7] Vaibhav Bajpai, Steffie Jacob Eravuchira, and Jürgen Schönwälder. Lessons learned from using the RIPE atlas platform for measurement research. *Computer Communication Review*, 45(3):35–42, 2015.

[8] Vaibhav Bajpai, Mirja Kühlewind, Jörg Ott, Jürgen Schönwälder, Anna Sperotto, and Brian Trammell. Challenges with reproducibility. In *Proceedings of the Reproducibility Workshop*, pages 1–4. ACM, 2017.

[9] Vaibhav Bajpai and Jürgen Schönwälder. A survey on internet performance measurement platforms and related standardization efforts. *IEEE Communications Surveys & Tutorials*, 17(3):1313–1341, 2015.

[10] Jim Bankoski, Paul Wilkins, and Yaowu Xu. Technical overview of VP8, an open source video codec for the web. In *Multimedia and Expo (ICME), 2011 IEEE International Conference on*, pages 1–6. IEEE, 2011.

[11] Andrzej Beben, P Wiśniewski, J Mongay Batalla, and Piotr Krawiec. ABMA+: lightweight and efficient algorithm for http adaptive streaming. In *Proceedings of the 7th International Conference on Multimedia Systems*, pages 2:1–2:11. ACM, 2016.

[12] Ali C. Begen, Tankut Akgul, and Mark Baugher. Watching video over the web: Part 1: Streaming protocols. *IEEE Internet Computing*, 15(2):54–63, 2011.

[13] Ali C. Begen, Tankut Akgul, and Mark Baugher. Watching video over the web: Part 2: Applications, standardization, and open issues. *IEEE Internet Computing*, 15(3):59–63, 2011.

[14] Abdelhak Bentaleb, Christian Timmerer, Ali C. Begen, and Roger Zimmermann. Bandwidth prediction in low-latency chunked streaming. In Ali C. Begen, Thomas Schierl, and Sejin Oh, editors, *Proceedings of the 29th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video, NOSSDAV 2019, Amherst, MA, USA, June 21, 2019.*, pages 7–13. ACM, 2019.

[15] Eli Brosh, Salman Abdul Baset, Vishal Misra, Dan Rubenstein, and Henning Schulzrinne. The delay-friendliness of TCP for real-time traffic. *IEEE/ACM Trans. Netw.*, 18(5):1478–1491, 2010.

[16] Igor Canadi, Paul Barford, and Joel Sommers. Revisiting broadband performance. In John W. Byers, Jim Kurose, Ratul Mahajan, and Alex C. Snoeren, editors, *Proceedings of the 12th ACM SIGCOMM Internet Measurement Conference, IMC '12, Boston, MA, USA, November 14-16, 2012*, pages 273–286. ACM, 2012.

[17] Pedro Casas, Alessandro D'Alconzo, Pierdomenico Fiadino, Arian Bär, Alessandro Finamore, and Tanja Zseby. When youtube does not work - analysis of qoe-relevant degradation in google CDN traffic. *IEEE Trans. Network and Service Management*, 11(4):441–457, 2014.

[18] Yanjiao Chen, Kaishun Wu, and Qian Zhang. From qos to qoe: A tutorial on video quality assessment. *IEEE Communications Surveys and Tutorials*, 17(2):1126–1165, 2015.

[19] Cisco. Cisco visual networking index: Global mobile data traffic forecast update, 2012-2017, 2012.

[20] D. E. Comer, David Gries, Michael C. Mulder, Allen Tucker, A. Joe Turner, and Paul R. Young. Computing as a discipline. *Communications of the ACM*, 32(1):9–23, January 1989.

[21] Xavier Corbillon, Ramon Aparicio-Pardo, Nicolas Kuhn, Géraldine Texier, and Gwendal Simon. Cross-layer scheduler for video streaming over MPTCP. In *Proceedings of the 7th International Conference on Multimedia Systems*, pages 7:1–7:12. ACM, 2016.

[22] Johan De Vriendt, Danny De Vleeschauwer, and Dave C Robinson. QoE model for video delivered over an LTE network using HTTP adaptive streaming. *Bell Labs Technical Journal*, 18(4):45–62, 2014.

[23] Zhengfang Duanmu, Kai Zeng, Kede Ma, Abdul Rehman, and Zhou Wang. A quality-of-experience index for streaming video. *J. Sel. Topics Signal Processing*, 11(1):154–166, 2017.

[24] Martin Ellis, Dimitrios P. Pezaros, and Colin Perkins. Performance analysis of AL-FEC for rtp-based streaming video traffic to residential users. In *19th International Packet Video Workshop, PV 2012, Munich-Garching, Germany, May 10-11, 2012*, pages 1–6. IEEE, 2012.

[25] Christian Feller, Juergen Wuenschmann, Thorsten Roll, and Albrecht Rothermel. The VP8 video codec-overview and comparison to H. 264/AVC. In *Consumer Electronics-Berlin (ICCE-Berlin), 2011 IEEE International Conference on*, pages 57–61. IEEE, 2011.

[26] Markus Fiedler, Tobias Hossfeld, and Phuoc Tran-Gia. A generic quantitative relationship between quality of experience and quality of service. *Network, IEEE*, 24(2):36–41, 2010.

[27] Sally Floyd, Mark Handley, and Eddie Kohler. Problem statement for the datagram congestion control protocol (DCCP). *RFC*, 4336:1–22, 2006.

[28] Dominique Fober, Yann Orlarey, and Stéphane Letz. Real time clock skew estimation over network delays. In *Proceedings of the International Music Conference, ICMA*, 2005.

[29] Alan Ford, Costin Raiciu, Mark Handley, and Olivier Bonaventure. TCP extensions for multipath operation with multiple addresses. Technical report, 2013.

[30] Bryan Ford and Janardhan R. Iyengar. Breaking up the transport logjam. In Carey L. Williamson, David Andersen, and Steve D. Gribble, editors, *7th ACM Workshop on Hot Topics in Networks - HotNets-VII, Calgary, Alberta, Canada, October 6-7, 2008.*, pages 85–90. ACM SIGCOMM, 2008.

[31] Marie-Neige Garcia, Francesca De Simone, Samira Tavakoli, Nicolas Staelens, Sebastian Egger, Kjell Brunnström, and Alexander Raake. Quality of experience and HTTP adaptive streaming: A review of subjective studies. In *Sixth International Workshop on Quality of Multimedia Experience, QoMEX 2014, Singapore, September 18-20, 2014*, pages 141–146. IEEE, 2014.

[32] Monia Ghobadi, Yuchung Cheng, Ankur Jain, and Matt Mathis. Trickle: Rate limiting youtube video streaming. In *Usenix Annual Technical Conference*, pages 191–196, 2012.

[33] Yunmin Go, Oh Chan Kwon, and Hwangjun Song. An energy-efficient http adaptive video streaming with networking cost constraint over heterogeneous wireless networks. *IEEE Trans. Multimedia*, 17(9):1646–1657, 2015.

[34] Google. IPv6 Adoption Statistics, 2017.

[35] Dan Grois, Detlev Marpe, Amit Mulayoff, Benaya Itzhaky, and Ofer Hadar. Performance comparison of H.265/MPEG-HEVC, VP9, and H.264/MPEG-AVC encoders. In *Picture Coding Symposium (PCS), 2013*, pages 394–397. IEEE, 2013.

[36] Sana Habib, Junaid Qadir, Anwaar Ali, Durdana Habib, Ming Li, and Arjuna Sathiaseelan. The past, present, and future of transport-layer multipath. *J. Network and Computer Applications*, 75:236–258, 2016.

[37] Bo Han, Feng Qian, Lusheng Ji, and Vijay Gopalakrishnan. Mp-dash: Adaptive video streaming over preference-aware multipath. In *Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies*, pages 129–143. ACM, 2016.

[38] Tobias Hoßfeld, Michael Seufert, Matthias Hirth, Thomas Zinner, Phuoc Tran-Gia, and Raimund Schatz. Quantification of YouTube QoE via crowdsourcing. In *Multimedia (ISM), 2011 IEEE International Symposium on*, pages 494–499. IEEE, 2011.

[39] Te-Yuan Huang, Ramesh Johari, Nick McKeown, Matthew Trunnell, and Mark Watson. A buffer-based approach to rate adaptation: evidence from a large video streaming service. *Computer Communications Review*, 44(4):187–198, Oct 01 2014. Date revised - 2016-06-01; Last updated - 2016-06-02.

[40] Adobe Systems Incorporated. Adobe Flash Video File Format Speciation. Specification, August 2010. Version 10.1.

[41] Information technology — Multimedia application format (MPEG-A) — Part 19: Common media application format (CMAF) for segmented media. Standard, International Organization for Standardization, January 2018.

[42] Janardhan Iyengar and Martin Thomson. QUIC: A UDP-based multiplexed and secure transport. draft-ietf-quic-transport-15, 2018.

[43] Parikshit Juluri, Louis Plissonneau, and Deep Medhi. Pytomo: a tool for analyzing playback quality of YouTube videos. In *Proceedings of the 23rd International Teletraffic Congress*, pages 304–305. International Teletraffic Congress, 2011.

[44] Theodoros Karagkioules, Cyril Concolato, Dimitrios Tsilimantos, and Stefan Valentin. A comparative case study of HTTP adaptive streaming algorithms in mobile networks. In *Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 1–6. ACM, 2017.

[45] Michalis Katsarakis, Renata Cruz Teixeira, Maria Papadopouli, and Vassilis Christophides. Towards a causal analysis of video qoe from network and application qos. In Pedro Casas, Fabián E. Bustamante, Martín Varela, and David R. Choffnes, editors, *Proceedings of the 2016 workshop on QoE-based Analysis and Management of Data Communication Networks, Internet-QoE@SIGCOMM 2016, Florianopolis, Brazil, August 22-26, 2016*, pages 31–36. ACM, 2016.

[46] Philip T. Kortum and Marc Sullivan. The effect of content desirability on subjective video quality ratings. *Human Factors*, 52(1):105–118, 2010.

[47] Balázs Kreith, Varun Singh, and Jörg Ott. Fractal: Fec-based rate control for RTP. In Qiong Liu, Rainer Lienhart, Haohong Wang, Sheng-Wei "Kuan-Ta" Chen, Susanne Boll, Yi-Ping Phoebe Chen, Gerald Friedland, Jia Li, and Shuicheng Yan, editors, *Proceedings of the 2017 ACM on Multimedia Conference, MM 2017, Mountain View, CA, USA, October 23-27, 2017*, pages 1363–1371. ACM, 2017.

[48] Jean Le Feuvre and Cyril Concolato. Tiled-based adaptive streaming using mpeg-dash. In *Proceedings of the 7th International Conference on Multimedia Systems*, MMSys '16, pages 41:1–41:3, New York, NY, USA, 2016. ACM.

[49] Jonathan Lennox, Magnus Westerlund, Qin Wu, and Colin Perkins. Sending multiple RTP streams in a single RTP session. *RFC*, 8108:1–29, 2017.

[50] Baochun Li, Zhi Wang, Jiangchuan Liu, and Wenwu Zhu. Two decades of internet video streaming: A retrospective view. *TOMCCAP*, 9(1s):33:1–33:20, 2013.

[51] Ming Li, Andrey Lukyanenko, Zhonghong Ou, Antti Ylä-Jääski, Sasu Tarkoma, Matthieu Coudron, and Stefano Secci. Multipath transmission for the internet: A survey. *IEEE Communications Surveys and Tutorials*, 18(4):2887–2925, 2016.

[52] Zhi Li, Anne Aaron, Ioannis Katsavounidis, Anush Moorthy, and Megha Manohara. Toward a practical perceptual video quality metric. Online, Jun 2016.

[53] Zhi Li, Xiaoqing Zhu, Joshua Gahm, Rong Pan, Hao Hu, Ali C Begen, and David Oran. Probe and adapt: Rate adaptation for HTTP video streaming at scale. *IEEE Journal on Selected Areas in Communications*, 32(4):719–733, 2014.

[54] Stephen McQuistin, Colin Perkins, and Marwan Fayed. TCP goes to hollywood. In *Proceedings of the 26th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 5:1–5:6. ACM, 2016.

[55] Stephen McQuistin, Colin Perkins, and Marwan Fayed. TCP Hollywood: An unordered, time-lined, TCP for networked multimedia applications. In *IFIP Networking Conference (IFIP Networking) and Workshops, 2016*, pages 422–430. IEEE, 2016.

[56] Anish Mittal, Anush K Moorthy, Alan C Bovik, Chang Wen Chen, Periklis Chatzimisios, Tasos Dagiuklas, and Luigi Atzori. No-reference approaches to image and video quality assessment. *Multimedia Quality of Experience (QoE): Current Status and Future Requirements*, pages 136–154, 2015.

[57] Anush K. Moorthy, Lark Kwon Choi, Alan Conrad Bovik, and Gustavo de Veciana. Video quality assessment on mobile devices: Subjective, behavioral and objective studies. *J. Sel. Topics Signal Processing*, 6(6):652–671, 2012.

[58] Debargha Mukherjee, Jim Bankoski, Adrian Grange, Jingning Han, John Koleszar, Paul Wilkins, Yaowu Xu, and Ronald Bultje. The latest open-source video codec VP9-an overview and preliminary results. In *30th Picture Coding Symposium, PCS 2013, San Jose, CA, USA, December 8-11, 2013*, pages 390–393. IEEE, 2013.

[59] Hyunwoo Nam, Kyung-Hwa Kim, Doru Calin, and Henning Schulzrinne. YouSlow: a performance analysis tool for adaptive bitrate video streaming. In *Proceedings of the 2014 ACM conference on SIGCOMM*, pages 111–112. ACM, 2014.

[60] Michael F. Nowlan, Nabin Tiwari, Janardhan R. Iyengar, Syed Obaid Amin, and Bryan Ford. Fitting square pegs through round pipes: Unordered delivery wire-compatible with TCP and TLS. In Steven D. Gribble and Dina Katabi, editors, *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2012, San Jose, CA, USA, April 25-27, 2012*, pages 383–398. USENIX Association, 2012.

[61] J Ott, S Wenger, N Sato, C Burmeister, and J Rey. Extended RTP profile for real-time transport control protocol (RTCP)-based feedback (RTP/AVPF)". *RFC*, 4585:1–51, 2006.

[62] Giorgos Papastergiou, Gorry Fairhurst, David Ros, Anna Brunstrom, Karl-Johan Grinnemo, Per Hurtig, Naeem Khademi, Michael Tüxen, Michael Welzl, Dragana Damjanovic, et al. De-ossifying the internet transport layer: A survey and future perspectives. *IEEE Communications Surveys & Tutorials*, 19(1):619–639, 2017.

[63] Colin Perkins and Jörg Ott. Real-time audio-visual media transport over QUIC. In *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC, EPIQ@CoNEXT 2018, Heraklion, Greece, December 4, 2018*, pages 36–42. ACM, 2018.

[64] Federico Perotto, Claudio Casetti, and Giulio Galante. SCTP-based transport protocols for concurrent multipath transfer. In *Wireless Communications and Networking Conference, 2007. WCNC 2007. IEEE*, pages 2969–2974. IEEE, 2007.

[65] Margaret H. Pinson and Stephen Wolf. Comparing subjective video quality testing methodologies. In Touradj Ebrahimi and Thomas Sikora, editors, *Visual Communications and Image Processing 2003, Lugano, Switzerland, July 8, 2003*, volume 5150 of *Proceedings of SPIE*, pages 573–582. SPIE, 2003.

[66] Lucian Popa, Ali Ghodsi, and Ion Stoica. HTTP as the narrow waist of the future internet. In Geoffrey G. Xie, Robert Beverly, Robert Tappan Morris, and Bruce Davie, editors, *Proceedings of the 9th ACM Workshop on Hot Topics in Networks. HotNets 2010, Monterey, CA, USA - October 20 - 21, 2010*, page 6. ACM, 2010.

[67] Feng Qian, Lusheng Ji, Bo Han, and Vijay Gopalakrishnan. Optimizing 360 video delivery over cellular networks. In *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, pages 1–6. ACM, 2016.

[68] Jason J. Quinlan, Ahmed H. Zahran, and Cormac J. Sreenan. Datasets for AVC (H.264) and HEVC (H.265) evaluation of dynamic adaptive streaming over HTTP (DASH). In Christian Timmerer, editor, *Proceedings of the 7th International Conference on Multimedia Systems, MMSys 2016, Klagenfurt, Austria, May 10-13, 2016*, pages 51:1–51:6. ACM, 2016.

[69] Allen L Ramaboli, Olabisi E Falowo, and Anthony H Chan. Bandwidth aggregation in heterogeneous wireless networks: A survey of current approaches and issues. *Journal of Network and Computer Applications*, 35(6):1674–1690, 2012.

[70] Juan J Ramos-Muñoz, Jonathan Prados-Garzon, Pablo Ameigeiras, Jorge Navarro-Ortiz, and Juan M López-Soler. Characteristics of mobile youtube traffic. *IEEE Wireless Communications*, 21(1):18–25, 2014.

[71] Azad Ravanshid, Peter Rost, Diomidis S. Michalopoulos, Vinh V. Phan, Hajo Bakker, Danish Aziz, Shreya Tayade, Hans D. Schotten, Stan Wong, and Oliver Holland. Multi-connectivity functional architectures in 5g. In *IEEE International Conference on Communication, ICC 2015, London, United Kingdom, June 8-12, 2015, Workshop Proceedings*, pages 187–192. IEEE, 2016.

[72] Demóstenes Zegarra Rodríguez, Zhou Wang, Renata Lopes Rosa, and Graça Bressan. The impact of video-quality-level switching on user quality of experience in dynamic adaptive streaming over HTTP. *EURASIP J. Wireless Comm. and Networking*, 2014:216, 2014.

[73] J. Rosenberg. UDP and TCP as the New Waist of the Internet Hourglass. `https://tools.ietf.org/html/draft-rosenberg-internet-waist-hourglass-00`, February 2008. IETF Internet Draft.

[74] J Rosenberg, H Schulzrinne, G Camarillo, A Johnston, J Peterson, R Sparks, M Handley, and E Schooler. SIP: session initiation protocol. *RFC*, 3261:1–269, 2002.

[75] Quirin Scheitle, Matthias Wählisch, Oliver Gasser, Thomas C Schmidt, and Georg Carle. Towards an ecosystem for reproducible research in computer networking. In *Proceedings of the Reproducibility Workshop*, pages 5–8. ACM, 2017.

[76] H Schulzrinne, S Casner, R Frederick, and V Jacobson. RTP: A transport protocol for real-time applications. *RFC*, 3550, 2003.

[77] H Schulzrinne, A Rao, R Lanphier, M Westerlund, and M Stiemerling. Real-time streaming protocol version 2.0. *RFC*, 7826, 2016.

[78] Henning Schulzrinne and Stephen L. Casner. RTP profile for audio and video conferences with minimal control. *RFC*, 3551:1–44, 2003.

[79] Kalpana Seshadrinathan, Rajiv Soundararajan, Alan Conrad Bovik, and Lawrence K. Cormack. Study of subjective and objective quality assessment of video. *IEEE Trans. Image Processing*, 19(6):1427–1441, 2010.

[80] Michael Seufert, Sebastian Egger, Martin Slanina, Thomas Zinner, Tobias Hoßfeld, and Phuoc Tran-Gia. A survey on quality of experience of HTTP adaptive streaming. *IEEE Communications Surveys and Tutorials*, 17(1):469–492, 2015.

[81] Justine Sherry, Shaddi Hasan, Colin Scott, Arvind Krishnamurthy, Sylvia Ratnasamy, and Vyas Sekar. Making middleboxes someone else's problem: network processing as a cloud service. In Lars Eggert, Jörg Ott, Venkata N. Padmanabhan, and George Varghese, editors, *ACM SIGCOMM 2012 Conference, SIGCOMM '12, Helsinki, Finland - August 13 - 17, 2012*, pages 13–24. ACM, 2012.

[82] Varun Singh. *Protocols and Algorithms for Adaptive Multimedia Systems*. PhD thesis, Aalto University, Department of Communications and Networking, 2015.

[83] Varun Singh, T Karkkainen, J Ott, S Ahsan, and Lars Eggert. Multipath RTP. *IETF Internet draft*, 2011.

[84] Varun Singh, Jorg Ott, and Igor DD Curcio. Predictive buffering for streaming video in 3g networks. In *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–10. IEEE, 2012.

[85] Sebastian Sonntag, Jukka Manner, and Lennart Schulte. Netradar-measuring the wireless world. In *Modeling & Optimization in Mobile, Ad Hoc & Wireless Networks (WiOpt), 2013 11th International Symposium on*, pages 29–34. IEEE, 2013.

[86] Kevin Spiteri, Ramesh Sitaraman, and Daniel Sparacio. From theory to practice: improving bitrate adaptation in the dash reference player. In *Proceedings of the 9th ACM Multimedia Systems Conference*, pages 123–137. ACM, 2018.

[87] Kevin Spiteri, Rahul Urgaonkar, and Ramesh K. Sitaraman. BOLA: near-optimal bitrate adaptation for online videos. In *35th Annual IEEE International Conference on Computer Communications, INFOCOM 2016, San Francisco, CA, USA, April 10-14, 2016*, pages 1–9, 2016.

[88] Nicolas Staelens, Jonas De Meulenaere, Maxim Claeys, Glenn Van Wallendael, Wendy Van den Broeck, Jan De Cock, Rik Van de Walle, Piet Demeester, and Filip De Turck. Subjective quality assessment of longer duration video sequences delivered over HTTP adaptive streaming to tablet devices. *TBC*, 60(4):707–714, 2014.

[89] Randall Stewart. Stream control transmission protocol. Technical report, 2007.

[90] Thomas Stockhammer. Dynamic adaptive streaming over http–: standards and design principles. In *Proceedings of the second annual ACM conference on Multimedia systems*, pages 133–144. ACM, 2011.

[91] Srikanth Sundaresan, Sam Burnett, Nick Feamster, and Walter de Donato. Bismark: A testbed for deploying measurements and applications in broadband access networks. In Garth Gibson and Nickolai Zeldovich, editors, *2014 USENIX Annual Technical Conference, USENIX ATC '14, Philadelphia, PA, USA, June 19-20, 2014.*, pages 383–394. USENIX Association, 2014.

[92] Srikanth Sundaresan, Walter De Donato, Nick Feamster, Renata Teixeira, Sam Crawford, and Antonio Pescapè. Broadband internet performance: a view from the gateway. In *ACM SIGCOMM computer communication review*, volume 41, pages 134–145. ACM, 2011.

[93] Savera Tanwir and Harry G. Perros. A survey of VBR video traffic models. *IEEE Communications Surveys and Tutorials*, 15(4):1778–1802, 2013.

[94] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the h. 264/avc video coding standard. *IEEE Transactions on circuits and systems for video technology*, 13(7):560–576, 2003.

[95] Mathias Wien. *High Efficiency Video Coding: Coding Tools and Specification*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[96] Dan Wing and Andrew Yourtchenko. Happy eyeballs: Success with dual-stack hosts. Technical report, 2012.

[97] Stefan Winkler and Praveen Mohandas. The evolution of video quality measurement: from psnr to hybrid metrics. *Broadcasting, IEEE Transactions on*, 54(3):660–668, 2008.

[98] Dapeng Wu, Yiwei Thomas Hou, Wenwu Zhu, Ya-Qin Zhang, and Jon M. Peha. Streaming video over the internet: approaches and directions. *IEEE Trans. Circuits Syst. Video Techn.*, 11(3):282–300, 2001.

[99] Jiyan Wu, Chau Yuen, Ming Wang, and Jun-Liang Chen. Content-aware concurrent multipath transfer for high-definition video streaming over heterogeneous wireless networks. *IEEE Transactions on Parallel & Distributed Systems*, (1):1–1, 2016.

[100] Alex Zambelli. IIS smooth streaming technical overview. http://www.iis.net/learn/media/on-demand-smooth-streaming/smooth-streaming-technical-overview, March 2009.

At the turn of this century, faster Internet speeds and better compression techniques enabled the pervasiveness of video over the Internet. Video streaming is now the largest contributor of traffic in the Internet and is expected to rise even further. This thesis contributes to an evolving Internet, where video traffic has gained an equal if not higher importance to other types of web traffic. The study looks at video streaming from a user perspective to provide network performance indicators beyond bandwidth values, which are meaningful for both Internet providers and subscribers for gauging video experience. Furthermore, it includes work on novel and more optimized techniques for video delivery paving the way for newer Internet protocols in the heavily ossified and somewhat inflexible Internet. As we move towards immersive video and virtual reality, bandwidth requirements for video delivery and user expectations continue to rise and these developments are a need of the hour.

Aalto-DD 15/2020

ISBN 978-952-60-8927-0 (printed)
ISBN 978-952-60-8928-7 (pdf)
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)

Aalto University
School of Electrical Engineering
Department of Communications and Networking
www.aalto.fi

BUSINESS + ECONOMY

ART + DESIGN + ARCHITECTURE

SCIENCE + TECHNOLOGY

CROSSOVER

DOCTORAL DISSERTATIONS