Experimental Performance Analysis of Multipath TCP

John Simon Wejin

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Eastern Mediterranean University
November 2017
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Assoc. Prof. Dr. Ali Hakan Ulusoy
Acting Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Computer Engineering.

Prof. Dr. H. Işık Aybay
Chair, Department of Computer Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Computer Engineering.

Prof. Dr. Doğu Arifler
Co-Supervisor

Prof. Dr. Hasan Amca
Supervisor

Examining Committee

1. Prof. Dr. Hasan Amca
2. Prof. Dr. Doğu Arifler
3. Assoc. Prof. Dr. Gürcü Öz
4. Assoc. Prof. Dr. Ali Hakan Ulusoy
5. Asst. Prof. Dr. Devrim Seral

ABSTRACT

Multiple wireless interfaces in modern devices today have given great promise in enhancing multimedia service delivery over wireless networks. However, the IP coupled nature of the conventional TCP/IP protocol inhibits the simultaneous use of these interfaces. Multipath TCP (MPTCP), a protocol undergoing IETF standardization has been developed to simultaneously use multiple interfaces for delivery of services over the Internet. Unlike other earlier studies that use simulations, stationary scenario, and Advanced Video Codec-Dynamic Adaptive Streaming over HTTP (AVC-DASH) to study MPTCP, this thesis uses mobile scenario, real measurements, and Ultra High Definition, High Efficiency Video Codec (UHD HEVC-DASH) to evaluate the performance of MPTCP.

The findings show that MPTCP gives a higher performance in terms of throughput, shorter download time, and increased bandwidth compared to Single Path TCP (SPTCP). The results from delivery of multimedia services using UHD HEVC- DASH streaming, though specific instead of general, reveal that under balanced and unbalanced network paths, MPTCP offers good Quality of Experience (QoE) compared to SPTCP. However, with variability in latency, and packet loss between paths, MPTCP underperforms compared to SPTCP in terms of video buffering in the unbalanced network case, and high packet retransmission rate in either balanced or unbalanced network paths.

Keywords: DASH, MPTCP, HEVC, QoE, Throughput, RTT, UHD

ÖZ

Günümüzde Modern haberleşme cihazları üzerindeki çoklu kablosuz ara-yüzler kablosuz ağlar üzerinden multimedya hizmetlerini geliştirme yönünde önemli vaatler sunmaktadırlar. Bununla birlikte, geleneksel TCP / IP protokolünün IP'ye bağlı doğası, bu ara-yüzlerin eşzamanlı olarak kullanımını engellemektedir.

IETF standardizasyonuna tabi olan bir protokol olan Çoklu Yol TCP'si (MPTCP), Internet üzerinden servislerin sunumu için eşzamanlı olarak birden fazla arabirim kullanması için geliştirilmiştir. MPTCP'yi incelemek için simülasyonlar, sabit senaryo ve HTTP üzerinden Dinamik Adaptif Akış (AVC-DASH) kullanan diğer eski çalışmaların aksine, bu tez mobil senaryo, gerçek zamanlı ölçüm ve Ultra Yüksek Çözünürlüklü Yüksek Çözünürlüklü Video Codec kullanıyor (UHD HEVCDASH).

Bulgular, MPTCP'nin, Tek Yollu TCP (SPTCP) ile karşılaştırıldığında, verimlilik, daha kısa indirme süreleri ve artan bant genişliği açısından daha yüksek bir performans verdiğini göstermektedir. UHD HEVC-DASH akışını kullanarak multimedya hizmetlerinin sunumundan elde edilen sonuçlar, genel yerine spesifik olmakla birlikte, dengeli ve dengesiz ağ yolunda MPTCP'nin SPTCP'ye kıyasla İyi Kalite Deneyimi (QoE) sunduğunu ortaya koymaktadır. Bununla birlikte, gecikme ve yollar arasındaki paket kaybı değişkenliği ile MPTCP, dengesiz ağ durumunda video arabelleğe alma ve dengeli veya dengesiz ağ yollarındaki yüksek paket yeniden iletim hızı açısından SPTCP'ye nazaran daha düşük performans sergilemektedir.

Anahtar Kelimeler: DASH, MPTCP, HEVC, QoS, Çıktı, RTT, UHD

This thesis is dedicated to my entire family,

For their unwavering love and understanding

ACKNOWLEDGEMENT

I would like to thank God Almighty, for His grace, provision, protection, and favour throughout the period of this study. He has been faithful to me through life. Lord, I'm grateful.

I would remain grateful to my Supervisor and Co-Supervisor Professors Hasan Amca and Doğu Arifler for their untiring support, academic guidance, and fatherly advice.

My thanks also go to Hilmi Kansu, the CEO Comtech, Cyprus for initiating this research. Your contribution towards cutting edge technology in Cyprus is highly appreciated.

I would like to thank the entire staff of Computer Engineering for their academic impact and excellent service offered to me throughout my studies in North Cyprus. To my colleagues at the department, thanks for the assistance and encouragement.

My gratitude also goes to the staff of EMU library, especially Sevim and Beyza, for their trust and loving hearts. To my friends Don Atsa'am, Henry, Emma, Seun, Humphrey, Felix, John Olaifa, and Hamid. Thanks for your heartfelt friendship.

To the Vice Chancellor and his team at Taraba State University, I remain grateful for the opportunity given me to undergo this program.

Finally, though not the all, I remain exceedingly grateful to my entire family, who despite being out of their side, have never let me out of their hearts. Mum, Dad, Uncles, Aunties, Cousins, and Nephews. Your support, love, encouragement, and calls really kept me going. I'm so grateful.

TABLE OF CONTENTS

ABSTRACT iii

ÖZ iv

ACKNOWLEDGEMENT vi

LIST OF TABLES xi

LIST OF FIGURES xii

LIST OF ABBREVIATIONS xiii

1 INTRODUCTION 1

1.1 The Internet 1

1.2 Connectivity in Modern Devices 1

1.3 Solutions to Multipath and Implementation Approaches 3

1.3.1 Application Layer 3

1.3.2 Layer 3 Approach 3

1.3.3 Data Link Layer 4

1.3.4 Transport Layer 4

1.4 Dynamic Adaptive Streaming over HTTP 5

1.4.1 Quality of Experience Factors for DASH 6

1.5 Problem Statement 7

2 FUNDAMENTALS OF SPTCP AND MPTCP 9

2.1 Overview of Transmission Control Protocol 9

2.2 TCP Operational Plane 9

2.3 TCP Data Plane 11

2.4 Multipath TCP 12

2.5 Goals of MPTCP 13

2.5.1 Functional Goals 14
2.5.2 Compatibility Goals 14
2.6 Architecture of MPTCP 15
2.7 How MPTCP Operates 16
2.7.1 Connection Establishment 16
2.7.2 Data Transfer 17
2.7.3 Connection Teardown 18
2.8 MPTCP Modes 19
2.8.1 Full-mesh Mode 19
2.8.2 Backup Mode 19
2.8.3 Single–Path Mode 20
2.9 Effects of Middleboxes 20
2.9.1 MPTCP Congestion Control 20
2.10 Security Issues in MPTCP 21
2.11 Related Protocols 21
3 METHODOLOGY 25
3.1 Measurement Methodology 25
3.2 Downlink Test 26
3.3 Video Streaming 27
3.4 Tools for Evaluation 27
3.4.1 Wireshark 27
3.4.2 TCPdump 27
3.4.3 Iperf 27
4 EXPERIMENTS AND RESULTS 29
4.1 Throughput Test 29

4.2 Bandwidth 31
4.3 Download Times 33
4.4 Video Streaming 34
4.5 Video Streaming Results 35
4.5.1 Startup Delay 35
4.5.2 Stalling 37
4.5.3 Download Rate 39
4.5.4 System and Media Player Challenge 40
5 CONCLUSION AND FUTURE WORK 41
5.1 Conclusion 41
5.2 Future Work 42
REFERENCES 43
APPENDICES 49
Appendix A: Iperf Results for Bandwidth Measurement 50
Appendix B: WGET Download Output 54
Appendix C: HEVC-DASH Dataset 61
Appendix D: Media Presentation Description 63
Appendix E: VLC Streaming Statistics 65

LIST OF TABLES

Table 1. Download time from WGET output 34

LIST OF FIGURES

Figure 1. Multipath interface in smartphones 2

Figure 2. DASH streaming mechanism [9] 6

Figure 3. TCP 3-way handshake 10

Figure 4. TCP 4-way handshake 11

Figure 5. Fast retransmission [15] 12

Figure 6. TCP multipath scenario 13

Figure 7. (a) Conventional internet architecture (b) Enterprise reality [19] 14

Figure 8. TNG and MPTCP stacks [21] 15

Figure 9. Connection setup for (a) MPTCP (b) SPTCP 16

Figure 10. Header-like format of MPTCP data sequence number mapping 17

Figure 11. Connection teardown (a) Graceful release (b) FAST_Close. 19

Figure 12. Design of implementation setup for video and download times. 26

Figure 13. Flow chart of method used. 28

Figure 14. Setup for throughput evaluation 29

Figure 15. Viewing TCPdump capture using Wireshark. 30

Figure 16. Download throughput (a) MPTCP (b) SPTCP 31

Figure 17. Bandwidth setup 32

Figure 18. Bandwidth comparison 33

Figure 19. Download time comparison of MPTCP and SPTCP 34

Figure 20. Buffering comparison 36

Figure 21. Packet retransmission rate in (a) SPTCP (b) MPTCP balanced (c) MPTCP unbalanced 38

Figure 22. Download rate in balanced network 40

LIST OF ABBREVIATIONS

3G	Third Generations
4G	Fourth Generations
ACK	Acknowledgement
AP	Access Points
CMT-SCTP	Content Multipath –Stream Transmission Protocol
DAR-SCTP	Dynamic Address Reconfigurable-Stream Transmission Control Protocol
DASH	Dynamic Adaptive Streaming over HTTP
DSN	Data Sequence Number
HEVC	High-Efficiency Video Coding
IEEE	Institute of Electrical and Electronic Engineering
IETF	Internet Engineering Task Force
IP	Internet Protocol
LACP	Link Aggregation Control Protocol
LAN	Local Area Network
MPD	Media Presentation Description
MPTCP	Multipath Transmission Control Protocol
NAT	Network Access Translators
OLIA	Opportunistic Linked Increase Algorithm
QoE	Quality of Experience
RAN	Radio Access Network
RTT	Round Trip Time
SCTP	Stream Transmission Control Protocol

SPB	Shortest Path Bridging
SPTCP	Single-Path Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TNG	Transport Next Generation
UDP	User Datagram Protocol
UHD	Ultra High Definition
WAN	Wide Areas Network
WiFi	Wireless Fidelity
WMN	Wireless Mesh Network

Chapter 1

INTRODUCTION

1.1 The Internet

The Internet is a global interconnection of networks to enable communication and sharing of resources, ranging from hardware, software, or services between devices. These interconnections between networks are realized via a structured protocol stack known as the Transmission Control Protocol/Internet Protocol (TCP/IP). The physical layer transmits information as streams of electrical pulses, while the succeeding layer 2 connects Local Area Networks (LANs) over switching hub and Wi-Fi networks. The diverse networks of LANs connected to each other at the network layer are identified using distinct IP address.

Packets sent by a device on the network are redirected with the help of network routers, using destination address indicated in the source host packet. While layer 3 of the TCP/IP protocol stack ensures an end-to-end communication, the transport layer handles reliable or best effort process-to-process delivery of data stream between end hosts. Of the protocols present at the Transport layer, the TCP protocol is often preferred for data transmission, and as such remains the most notable at this layer.

1.2 Connectivity in Modern Devices

When the Internet started, it was driven by lightweight data transmission, and the single interface available in devices of those years was sufficient to handle communication. However, as technology evolves and improves, modern-day end hosts

have been developed with multiple interfaces for connection. Laptops connect to the Internet via Ethernet and Wi-Fi using wired and wireless interfaces. Tablets and smartphones are capable of connecting to the Internet either through the 3rd Generation (3G)/4th Generation (4G) network provided by the mobile operators or via a Wi-Fi access-point available.

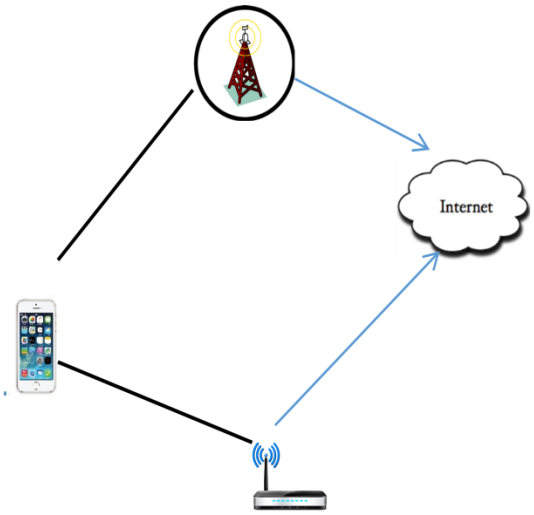


Figure 1. Multipath interface in smartphones

Though interfaces have evolved in modern devices as shown in Figure 1, the underlying protocols that facilitate communication and transfer of information between devices on the network were designed for a single interface. This characteristic of the earlier Internet protocols inhibits simultaneous utilization of the multiple interfaces built in modern day devices. Thus, equipping the underlying network protocols with multipath capability will not only enable pooling of accessible resources on different paths [1] and balancing of network traffic [2], but will also improve continuity in service provision in the presence of connection failure.

1.3 Solutions to Multipath and Implementation Approaches

Different researchers have proposed protocols and argued different TCP/IP layer implementation approaches to bridging the gap between the single–path nature of the TCP/IP protocols and the multiple interfaces available in modern devices. Some are of the notion that the best layer such a protocol will be of maximum benefit is the network layer. Those that look at maximizing throughput alone have implemented such a technique at the application layer. Others are of the opinion that the best layer will be the transport layer since it is more transparent to the network layer, independent of the application layer and does not need modification of the present Internet architecture, or any of the TCP/IP layers to be changed. The following subsections analyse these approaches.

1.3.1 Application Layer

An application layer solution to a multipath protocol was presented in Bit torrent [3]. This implementation maximizes throughput by employing a peer–to–peer protocol (P2P). It divides a file into different chunks and downloads them from different peers. Bit torrent feature of resource pooling technique operates like uncoupled multipath congestion control, thus runs self-standing congestion control on each path, with paths having different end-points. This method demonstrates the possibility of a multipath application and use of path diversity with P2P protocol. However, the drawback of this type of solution is a restriction of other users of the network, leading to unfair competition for available resource and difficulty in end-to-end multipath ability [4].

1.3.2 Layer 3 Approach

To get the maximum merit of multipath transmission at this layer, various approaches were suggested by different researchers. This method appears logical and simple to implement. Interfaces on devices can now be joined to separate IP address pairs.

Example is a laptop transmitting packets via Ethernet adapter or wireless interface [4]. A mobile IP multipath technique that enables a device to change its IP address without TCP re-establishing its connection and a site multihoming method that shifts traffic from one IP address to another have been proposed in [5]. In both techniques, the changes in IP address is unaware to the transport layer, thus enabling transport level connections to remain uninterrupted. In such network solutions, each path has a different network characteristic which when transparent to TCP protocol, results in uncoordinated management of resources. This leads to low performance and poor experience by users [5].

1.3.3 Data Link Layer

A data link layer multipath has been implemented in [6, 7]. Implementing multipath at the link layer uses the technique of combining the capacities of interfaces to a common switch to pool its network bandwidth. The use of same IP address on these interfaces makes the top layers unaware of the path combination. In [6] the use of Link Aggregation Control Protocol (LACP) defined in IEEE 802.1aq and IEEE 802.1Q Ethernet Bridging standard were used to aggregate switch ports in order to utilize multiple connections. In [7], the dual idea of multi-channel link (responsible for packet scheduling) and multipath routing (responsible for path selection) has been proposed as a means of increasing throughput in Wireless Mesh Networks (WMNs). Though multipath characteristic at this layer gives better throughput between switches, a change in the configuration of the switch makes end host incapable of using path aggregation to efficiently utilize the advantages offered by multiple interfaces [5].

1.3.4 Transport Layer

The solution at this layer doesn't develop a new protocol, but rather extends TCP to have multipath capability. As such, it is highly compatible with the current Internet

and needs no modification of any layer or device. From the information (e.g. latency, capacity etc.) obtained, this solution offers a good reaction to network congestion and therefore gives a better approach to congestion management. The TCP-like behavior of this implementation makes it pass through middleboxes easily. In this case, for a successful implementation of multipath protocol at this layer, it only requires that devices on the network run same protocol. No upgrade of a device in the network is needed [7].

1.4 Dynamic Adaptive Streaming over HTTP

Streaming video over the Internet has increased recently due to the different possible ways of video encoding and delivery. One way of delivery of video content is DASH. It is an adaptive bitrate streaming (ABS) standard used in delivering media content based on HTTP [8]. DASH can support streaming of live video like video conferencing, and on-demand video delivery.

In Video-On-Demand (VOD) media delivery as shown in Figure 2, the server stores two types of files: the video segments and the Media Presentation Description (MPD). The MPD is an eXtensible Markup Language (XML) file that holds information on the segments' location, bit rate and resolutions. The client initiates a connection by requesting the MPD from the server. Segments are then delivered through HTTP GET request, decoded and played at the client side either by a plugin (when using a browser) or a media player as used in this thesis.

Based on the adaptive technique used, the client switches between segments of different bitrate and resolutions [8]. With High Efficiency Video Coding (HEVC) gaining acceptance globally as the next generation video compression technique and

undergoing evaluation by researchers, this thesis employs the use of DASH streaming of video files compressed by HEVC (HEVC- DASH) to evaluate the streaming performance of MPTCP.

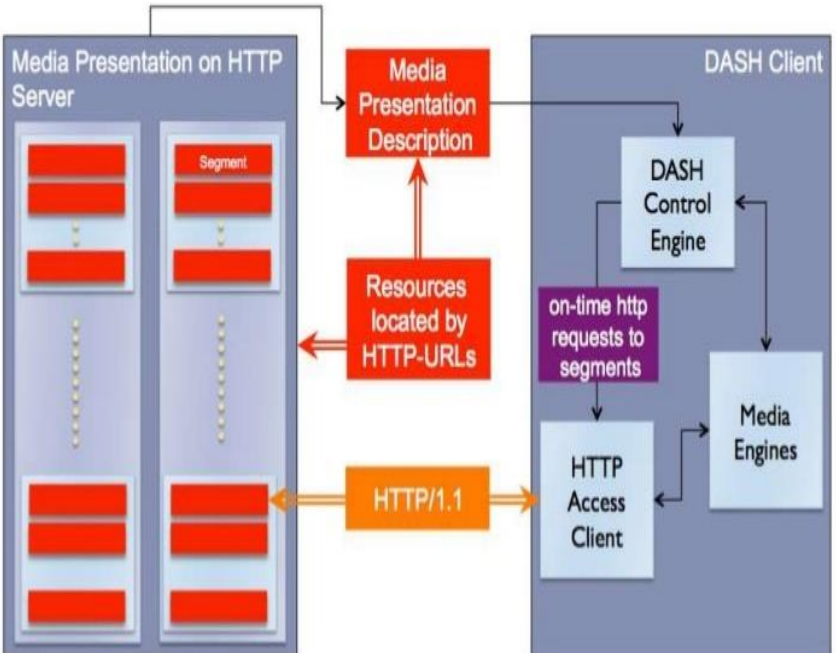


Figure 2. DASH streaming mechanism [9]

1.4.1 Quality of Experience Factors for DASH

Quality of Experience (QoE) has been defined as a subjective method of evaluating how well the underlying network satisfies users' requirement. These features are functions of a given level of users' perception, and the interaction with the service provided [10]. The QoE parameters affecting DASH streaming are startup delay, buffering/stalling, throughput and quality switch.

Start-up delay is the time between the request of the first segment and start of the actual playback. This might comprise the client and server processing time, network timing, and the first buffering time. In general, this parameter depends on the condition or case under study [10].

Stalling is characterized by playback stoppage or freezes during streaming. These freezes occur as a result of low buffering or buffer underruns. The playback is resumed after the media has re-buffered. It is clearly understood that wireless network experiences variable bandwidth, thus in DASH streaming, switches in quality bitrate are usually experienced. This is usually adopted in DASH streaming so as to avoid buffer underrun [10].

The throughput or the download rate is a parameter measured in bits/seconds. This signifies the rate at which video segments are downloaded from the video server. A higher value usually means better QoE and a lower value has a negative impact on the QoE [10]. To evaluate these parameters, we used MPTCP path characteristics like network delay, packet loss, and network bandwidth as influence factors. Rather than using Mean Opinion Score (MOS) which is expensive and time-consuming to obtain the effect of those factors on QoE, this study used the statistics obtained from the streaming media player.

1.5 Problem Statement

The demand for access to digital information by people and organizations has substantially increased in recent years [11]. This is evident by the high global Internet and mobile traffic generated by various applications and services. As the data overloads on the networks continue to rise, mobile operators are put under pressure to alleviate such situations. To cope with this increase, upgrade of the cellular Radio Access Network (RAN), mobile core infrastructures, and increase in spectrum coverage need to be done. From the economic viewpoint, such steps are capital intensive [12].

Furthermore, suppressing network speed while pegging data usage as in [13] gives poor Quality of Experience (QoE). Different techniques at various TCP/IP layers as outlined earlier have been implemented, with multipath giving more promising capabilities in improving QoE as well as leveraging the mobile and the Internet from data overload. The well-known multipath solution is Multipath TCP (MPTCP), which enables unchanged application to transmit data along multiple paths.

Though comparative studies of MPTCP in relation to congestion control, energy consumption, data offloading and video transmission have been conducted, the behaviour of a mobile user equipped with MPTCP when connected to multiple base stations (access points) remains largely unexplored. In this thesis, the use of real measurements and DASH will be employed in order to analyse the effect of MPTCP on the throughput of a communication device when connected to multiple Access Points (APs) through multiple interfaces.

The rest of the thesis is structured as follows: Chapter 2 reviews the theoretical and fundamentals of Single-Path TCP (SPTCP) and MPTCP. Chapter 3 explains the methodology used in this thesis while Chapter 4 discusses the findings from the experiments conducted. The conclusion and direction for future work are presented in Chapter 5.

Chapter 2

FUNDAMENTALS OF SPTCP AND MPTCP

2.1 Overview of Transmission Control Protocol

TCP is a protocol which functions at the layer 4 of the TCP/IP protocol stack. It is byte-oriented and ensures reliable ordered delivery of bytes sent from a socket of an end-host to a socket of another end-host on the network. The majority of applications over the Internet rely on TCP's services. This protocol together with User Datagram Protocol (UDP) account for the traffic on the Internet, thus the choice of extending TCP to support multipath capability as a way of balancing and pooling resources by splitting traffic along multiple interfaces becomes an option. Therefore, an understanding of TCP's functionality and operation gives a better comprehension of the MPTCP. The following subsection explains the operation of TCP protocol.

2.2 TCP Operational Plane

TCP divides an application packet into smaller chunks and delivers them via IP for an end to end delivery. Service identification during communication is made possible through the use of port numbers (source and destination) in the transport header of TCP segment. Unique identification of data stream is achieved by the combined use of port numbers and IP addresses.

To establish a session using TCP, a device follows a 3-way handshake stage as demonstrated in Figure 3. This allows parties in the connection to mark the beginning of a data stream. Firstly, the client program requests for an active open connection by

sending a Synchronization (SYN) segment which serves as a synchronization for sequence number to the server. In response to the client's SYN segment, an SYN/ACK segment with a window size (rwnd) indicating the reception of the client's SYN segment and its initial sequence number is sent by the server. The connection is established after the server receives the client's ACK segment which indicates the window size of the client and the reception of the server's SYN segment [5].

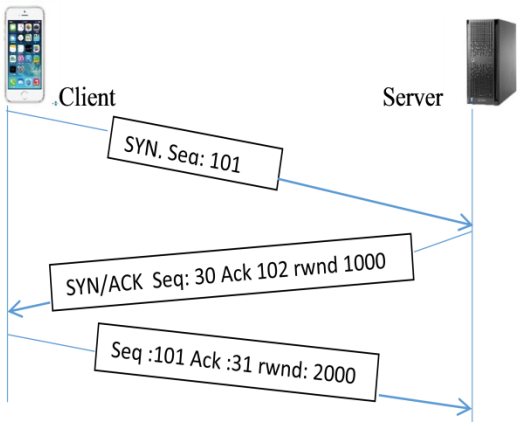


Figure 3. TCP 3-way handshake

While connection establishment indicates the start of a TCP session, the connection teardown phase indicates the end. Modern implementation today provides two alternatives**:** the TCP- handshake and the graceful release with half closure [14] as shown in Figure 4. In the graceful release, a client can opt to close an active connection while still receiving data by sending a FIN segment. The FIN ensures that prior data has been reliably and correctly received.

The server signals back to the client with a FIN acknowledgment indicating correct reception of data up to the sequence number in the FIN segment. Data from

the server to the client can still flow. This type of connection termination usually occurs when the server needs all data for processing, e.g. in sorting [14].

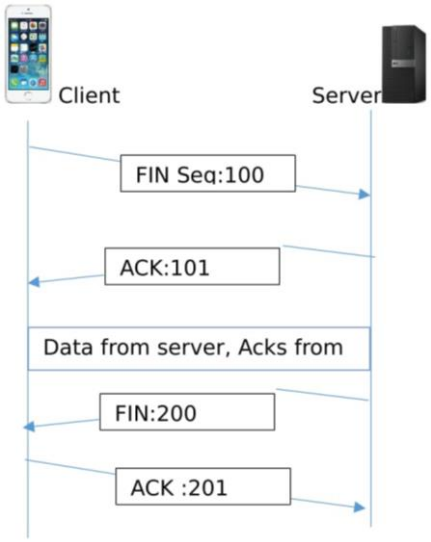


Figure 4. TCP 4-way handshake

2.3 TCP Data Plane

As stated earlier, TCP is a reliable and ordered-delivery protocol. This ordered delivery feature of TCP is achieved through numbering of each byte of data sent. Beginning with the initial sequence number from the connection establishment phase, TCP increments the sequence number for every byte delivered via an application's socket. In this way, a destination host on the network can get incoming data in the correct order. Reliability is maintained by sending acknowledgments for every data segment received, with each specifying the next sequence number expected.

When loss of packet occurs during a connection, double acknowledgments to the source help it determine the packet that was lost. To recover lost segments, the sender counts the duplicate acknowledgments. Detecting three duplicate acknowledgments as

shown in Figure 5, makes the sender presume packet loss on the network and as such, retransmits lost packet. This scheme is known as fast-retransmission [5].

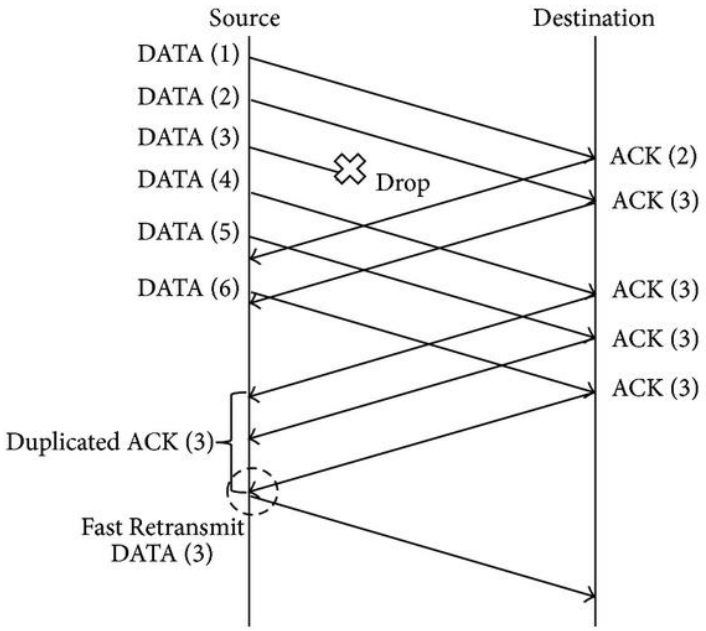


Figure 5. Fast retransmission [15]

To prevent congestion over the network, TCP congestion control mechanism is employed. It does this by automatically adjusting the rate at which an end-host TCP connection sends packets to the network. Reasons for congestion on the network may be of various natures. The common ones are those that indicate packet loss (loss-based) as in [16], and delay-based as suggested in [17].

2.4 Multipath TCP

Most of the applications over the Internet today employ TCP protocol to reliably transfer data from one end node to another. The traditional TCP byte-stream as stated in the previous chapter is strongly coupled to a single interface when transferring data. Thus, to utilise the multiple interfaces in modern day devices, the need for a transport protocol with multipath feature becomes necessary [18]. MPTCP is a protocol undergoing IETF standardization.

MPTCP extends TCP by splitting a single TCP flow into subflows, allowing concurrent transfer of data via multiple interfaces. Unlike TCP which suffers a connection failure when the interface changes as shown in Figure 6, MPTCP offers significant benefits like connection continuity in the presence of link failure, decreasing network overload, and good user experience. For MPTCP to function and be recognized by existing Internet applications as typical TCP, reliability and in-order byte transmission service must be incorporated. This requires that it includes a connection setup-phase for data signaling and an acknowledgment system that tracks in-order delivery of data.

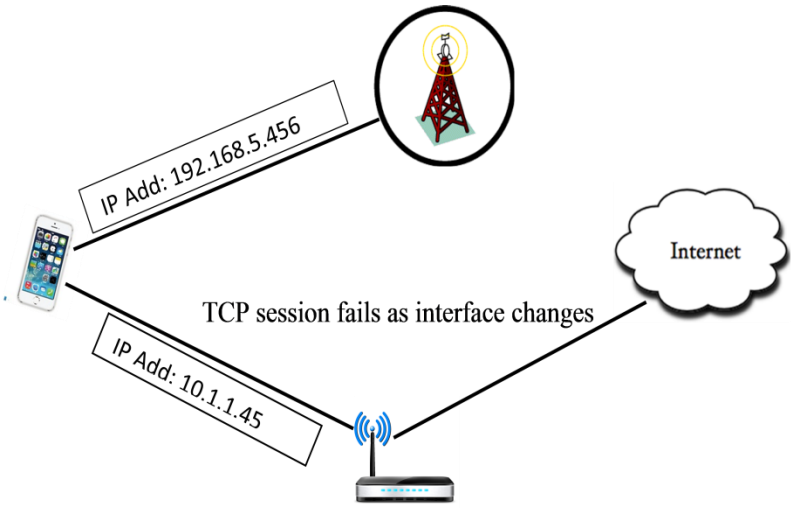


Figure 6. TCP multipath scenario

2.5 Goals of MPTCP

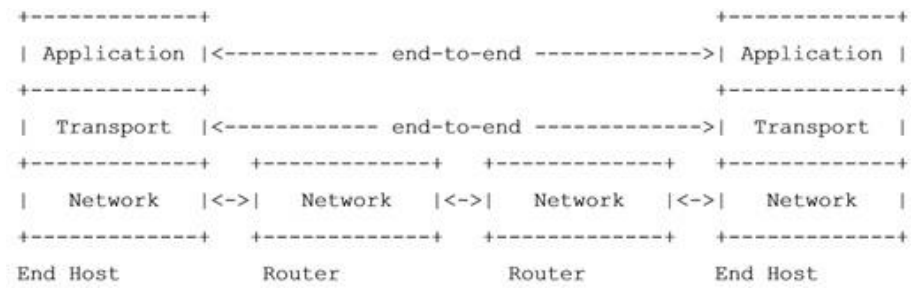
The aim of MPTCP is to concurrently pool resources available, giving an appearance of a single TCP resource to an application at the users' end. The goals of MPTCP as enumerated in [19] can be categorized into:

2.5.1 Functional Goals

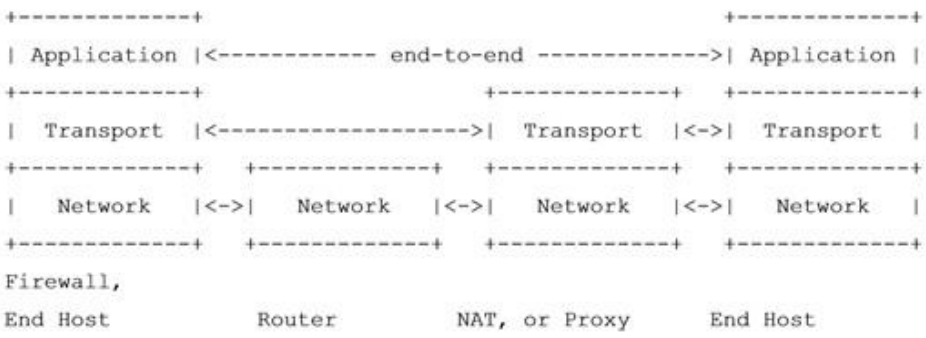
This means that MPTCP should be able to maximize the available interfaces to give greater throughput. It should also ensure connection continuity as session changes from one interface to another.

2.5.2 Compatibility Goals

In order to be deployed successfully on the existing Internet structure, MPTCP should achieve certain compatibility issues. From the application point of view, MPTCP should be able to run applications where TCP runs, with no modification to the application. The Internet is usually viewed as a stack of layers (OSI or TCP/IP models), with devices such as hops, switches, and routers, operating at various layers. This architecture shows how packets move from one end node to another. However as discovered in [20], the real enterprise networks have middleboxes that interfere with the transfer of data as shown in Figure 7.



(a)



(b)

Figure 7. (a) Conventional internet architecture (b) Enterprise reality [19]

Thus, considering the reality on the Internet, for MPTCP to be network compatible, it should be able to traverse through middleboxes without been interfered, and be able to work over IPv4 and IPv6 respectively.

2.6 Architecture of MPTCP

The structure of MPTCP follows the structure of the proposed Transport – Next–Generation (TNG) protocol which divides the transport layer into application and network function layers respectively [21]. In the structure of MPTCP, the TCP–like semantic layer gives application compatibility, while the network compatibility is provided by the subflow TCP component [22]. Figure 8 gives a vivid description.

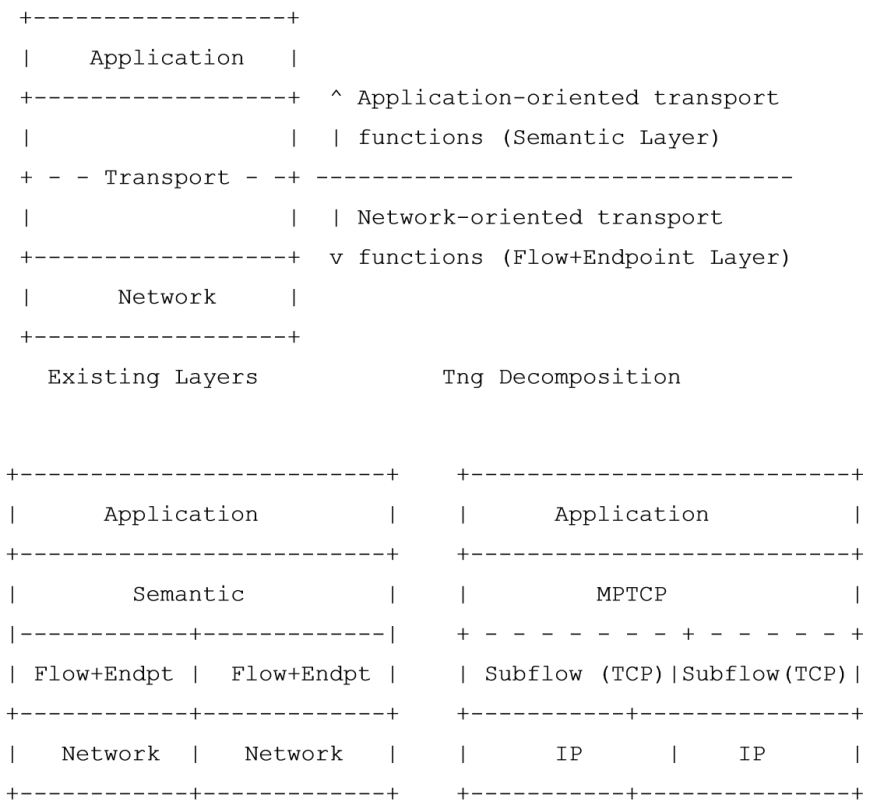


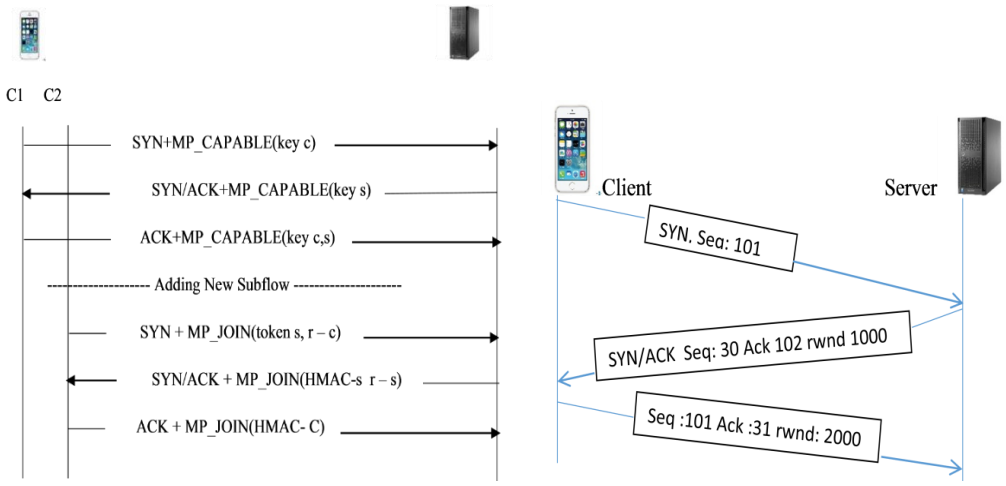
Figure 8. TNG and MPTCP stacks [21]

2.7 How MPTCP Operates

MPTCP has three major operational phases, connection establishment, data transfer, and connection tear down, respectively [22].

2.7.1 Connection Establishment

This begins with path detection since MPTCP has the capability of utilizing multiple interfaces. As shown in Figure 9, the connection set up is similar to the 3–way handshake in TCP, except for the added MP_CAPABLE, ADD_ADDRESS, and MP_JOIN options present in MPTCP. The MP_CAPABLE option is added to the SYN, SYN/ACK, and ACK segments respectively. MP_CAPABLE option indicates that the end host supports MPTCP, and is ready to use it for data transfer. The ACK+ MP_CAPABLE confirm to the other communication end that there is no interference by middleboxes. The ADD_ADDRESS segment is used to advertise a client after the connection has been established. The address exchange is used to open a new subflow which is linked to the master subflow by the SYN segment with MP_JOIN.



(a) (b)

Figure 9. Connection setup for (a) MPTCP (b) SPTCP

2.7.2 Data Transfer

After a successful establishment of a connection between client and server, data is ready for transmission. MPTCP uses two fundamental principles in ensuring that data is transferred reliably and in order. Firstly, every subflow is regarded as a conventional TCP connection with its sequence number (usually 32 bits). This enables MPTCP segment to pass through any middleboxes present along the transmission path. Secondly, it uses the DSN_MAP (which maps the data sequence number with the subflow sequence number) and DSN_ACK (a 64bits number used in data acknowledgement) sequence numbers as shown in Figure 10, to ensure retransmission of data from different subflow of same Data Sequence Number (DSN), and reordering of out of sequence segment. The F, m, M, a, A fields in Figure 10 serve as flags for setting various operations as indicated in the length field, while the kind field is a TCP option number employed to signal MPTCP operations. The sub-type is used to identify MPTCP options e.g. MP_CAPPABLE. Segment loss at the server is detected by the gap in the 32-bit sequence number and the regular retransmission of TCP process is initiated to recover lost segment. If a subflow fails, MPTCP detects it and retransmits on an active subflow.

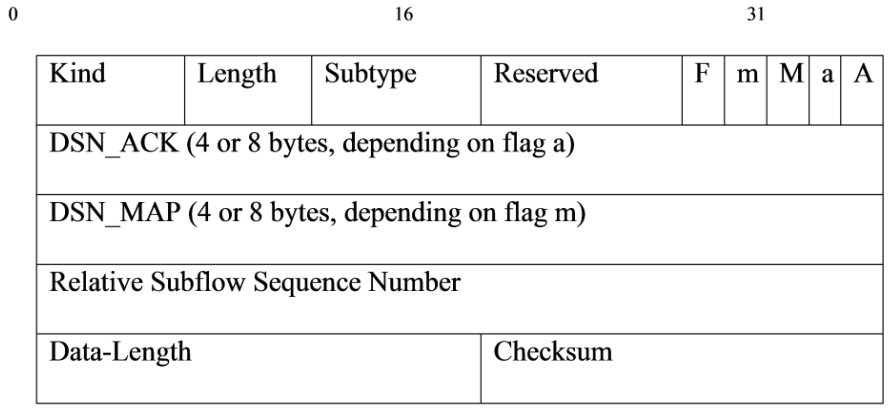
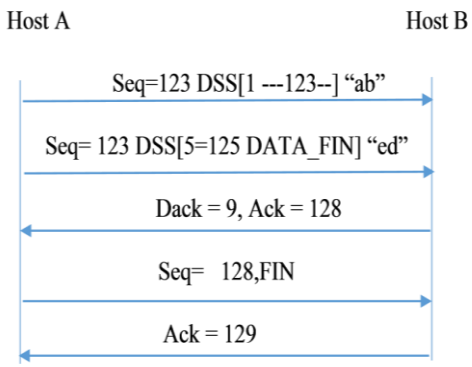


Figure 10. Header-like format of MPTCP data sequence number mapping

2.7.3 Connection Teardown

To close or tear down an established connection, MPTCP uses two different techniques. These are the graceful release and the FAST_CLOSE option as depicted in Figures 11 (a) and (b). In the graceful release, the client indicates to the server that there is no more data to send by adding a DATA_FIN segment as part of the Data Sequence Signal (DSS). The connection becomes closed when both ends acknowledge the DATA_FIN option. In a situation in which there is a need for a quick closure of connection e.g. in servers, MPTCP uses FAST_CLOSE option. This contains keys that were exchanged at the beginning of the connection to authenticate the closure and termination of the connection The Reset (RST) is necessary for this technique so as to kill the state in any middle boxes.



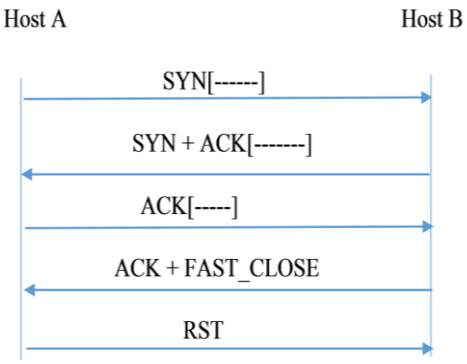


Figure 11. Connection teardown (a) Graceful release (b) FAST_Close.

2.8 MPTCP Modes

The MPTCP was developed so as to utilize the multiple interfaces in modern devices to increase throughput and connection continuity when connection fails. To achieve this, the protocol operates in three major handover modes [23]. These are:

2.8.1 Full-mesh Mode

This mode makes use of all the sub-flows created between the communication ends. Though the protocol gives maximum throughput when in this mode, stripped data on a specific sub-flow is delayed at the time of handoff in a congested situation, leading to glitches and poor service continuity [23].

2.8.2 Backup Mode

Unlike the full-mesh mode which uses all subflows created over the interfaces, this mode uses subset out of the created subflows for data transfer. Packets are transferred on different sub-flows only if the master sub-flow fails. This phenomenon is not ideal for handover scenarios because waiting for a connection to fail during congestion has an adverse effect on the continuous provision of service or connection [23].

2.8.3 Single–Path Mode

This mode is similar to the backup mode. The difference is that in this mode, a sub-flow is created at any moment [24].

2.9 Effects of Middleboxes

Middleboxes are an intermediary mechanism that performs functions other than the forwarding of packets between hosts on the network. They can change, examine, interfere with communication sessions and even block packets from being transmitted over the network. Examples are firewalls, Network Access Translators (NAT), Wide Area Network (WAN) optimizers, and load balancers [25].

The introduction of these mechanisms interferes with the end–to –end design of the Internet structure. Thus, for successful deployment of MPTCP on the Internet, it must be able to pass through middleboxes with minimal or no interference. To prevent the interference of the control and data plane by middleboxes, MPTCP uses the technique of falling back to regular TCP communication flow.

2.9.1 MPTCP Congestion Control

Congestion is a phenomenon that occurs when the network carries data that exceeds its normal capacity. To handle congestion, MPTCP uses coupled congestion control mechanism. Congestion is measured on each subflow. As congestion increases the traffic is balanced by moving it to the path experiencing lesser congestion. By default, all sub-flows are used and traffic is kept on each. To give fairness MPTCP limits the congestion window to that of a normal TCP connection, making it grow [26, 27]. As in [28], other congestion mechanized used in MPTCP are

- Opportunistic Linked Increase Algorithm (OLIA)
- Linked Increase Algorithm (LIA)

- Balanced Linked Adaptation Congestion Control Algorithm (BALIA)
- Delay-based Congestion Control for MPTCP (wVegas)

2.10 Security Issues in MPTCP

As earlier stated in this chapter, MPTCP was designed to extend TCP so as to simultaneously use the multiple interfaces in modern devices to increase throughput and connection resilience. To achieve this, various features and techniques such as DSN, Subflow Sequence Number (SSN), etc. were incorporated into MPTCP design. From the security viewpoint, these added features expose MPTCP to more attacks compared to SPTCP. Though features like Hash Message Authentication Code (HMAC) and token seek to secure MPTCP connections, an attacker can use either off-path or time-shift approaches to change the starting TCP handshake [29].

Some of the security threats to MPTCP are Denial of Service (DoS) on MP_JOIN option, keys eavesdrop, SYN/ACK attack, and ADD_ADDR attack [29]. Possible proposed solutions to these threats are detailed in [30]. Recently, the combined use of authentication and encryption techniques as demonstrated in [31] are gaining attention as a method of making MPTCP more secure.

2.11 Related Protocols

Researchers have suggested many multipath protocols at the transport layer. One that is closely related to MPTCP is Stream Control Transmission Protocol (SCTP). This is a message-oriented, and reliable transport layer protocol initially proposed for signaling between two SCTP end devices. One of the paramount significance of this protocol is its capability to support multiple IP addresses on which the streams created per association can be transmitted, which provides multi-homing and substantial resource pooling at this layer.

Though multi-homing is possible with SCTP, it reduces vertical handover, supported on most network devices, and therefore difficult to be deployed on the current Internet architecture [32]. To overcome some of the problems of SCTP, its variant such as Dynamic Address Reconfiguration (DAR-SCTP) in [33], and Content Multipath Transfer (CMT-SCTP) as in [34, 35], were developed to make SCTP support vertical handover and to provide higher association throughput. Still, with the enhanced capability provided by SCTP variants, protocols at the application level need to be modified so as to use SCTP.

To evaluate the performance of MPTCP with TCP and other related protocols, many researchers have used various approaches and methods. In [36], the performance of MPTCP in the wild wireless scenarios was investigated. The focus was to determine the effect of MPTCP on applications' performance in relation to Round Trip Time (RTT), loss rate, and throughput. Their findings which use cellular and Wi-Fi setup show that MPTCP gives better data transfer when file sizes are larger, and lowers the variance in latency during download of files. This thesis extends their research by considering the effect of MPTCP in streaming video services in a mobile environment using wireless network. In [37], a proposed packet scheduling algorithm for MPTCP in wireless networks using simulation with Network Simulator 3 (NS3) was employed to analyze the efficiency of MPTCP. Though this finding offer insight into the behaviour of MPTCP in wireless situation, it didn't evaluate MPTCP performance in a mobile environment using real measurements as used in this thesis.

Unlike other studies which use Ethernet and wireless interfaces to study MPTCP, in [38] purely multihomed wireless interfaces with a dedicated link were used to evaluate MPTCP performance. The focus of the paper was to examine how MPTCP coupled

congestion control functions with receive buffer size and segment size when a communication device was connected to various Wi-Fi standards in multi-homed networks. Conclusions from the research reveal that the coupled congestion control in MPTCP increases the share of the traffic over Wi-Fi as receiver buffer size increases. This increase as discovered in the paper can only be larger than that on the dedicated links if the data rate on the Wi-Fi exceeds that on the dedicated link. Though the paper and this thesis employ the use of wireless networks to evaluate MPTCP, the studies could not assert whether the throughput of MPTCP outperforms that of TCP which this thesis investigates.

The authors in [39] look at whether MPTCP improves throughput when connected to multiple Wi-Fi APs in a mobile environment. The focus was to comprehend how MPTCP and Wi-Fi Media Access Control (MAC) of various wireless standards interact. Association to multiple APs on the same channel and on different channels were studied. Results show that job balancing is frequently carried out by the Wi-Fi-MAC or by the MPTCP congestion control when in hidden terminal mode. It concludes that there is a situation in which associating to multiple APs can affect MPTCP performance.

To evaluate the performance of MPTCP in transmitting multimedia services over the Internet, James et al in [40] investigated the overall QoE when MPTCP was used with DASH for video streaming on the Internet. A LAN network and a Wireless network were used to emulate multipath for evaluation. The finding focuses on the effect of MPTCP paths bandwidth in streaming DASH videos. The research concludes that the variability in bandwidth on the paths plays a key role in the streaming quality offered by MPTCP. This thesis extends their research by studying the effect of delay and

packet loss on network paths using UHD HEVC-DASH dataset in a mobile environment. HEVC-DASH was used so as to evaluate the bandwidth need of the new codec, with priority given to the relationship between QoS and QoE.

In this chapter, fundamental concepts of TCP and MPTCP have been explained. Furthermore, earlier methods used by researchers to evaluate MPTCP performance have been reviewed. The next chapter describes the method employed in this thesis to evaluate MPTCP performance in mobile environment.

Chapter 3

METHODOLOGY

3.1 Measurement Methodology

The setup used in this thesis to evaluate video services, bandwidth, and downlink test is shown in Figure 12. The client is a Toshiba Satellite C-50B laptop with Intel dual-core processor, running at 2.16 GHz. The server is an HP 2000 with Intel Celeron dual-core processor running at 2.10 GHz. Both client and server run Ubuntu 16.04, with Linux kernel implementation of MPTCP v0.92 installed and enabled. To access files and play video segments using HTTP, HTTP Apache server was installed and configured on the server. The client was equipped with TCPdump/Wireshark for packet capture and analysis, with VLC media player installed and used as DASH client. The server stores files of various sizes and video segments which are requested by the client.

Default MPTCP configurations were used in all experiments (the operation mode was set to function in full mesh and congestion control set to OLIA). The server connects to the Internet via LAN and wireless interfaces, while the client connects to the Internet via built-in and external wireless interfaces. The default configuration of Ubuntu's TCP functionality was chosen to create a uniform underlying environment. This allowed full supervision of MPTCP with minimal impact on the computational power of the devices.

The mobility pattern emulated in this thesis is a simple movement from the server as client streams or downloads files from the server. As earlier stated, the client connects to the Internet via built-in and external Wi-Fi adapter. The external adapter is an LB-Link 802.11N adapter that supports roaming. It was on this interface that stable mobility and connection to multiple APs (3 APs were used in this thesis) was enabled. At the beginning, both interfaces are connected to different APs. The client then moves away from the server so as to create a scenario that produces a weak signal between the server and client, thus forcing the device to roam.

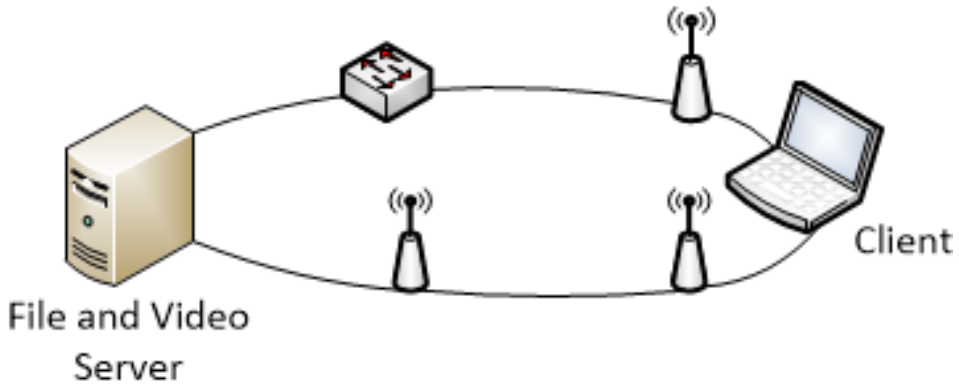


Figure 12. Design of implementation setup for video and download times.

Various link characteristics in terms of available bandwidth and latency were considered. This was done so as to find out how MPTCP will adapt to various network conditions and if the overall goodput could be degraded in comparison to the best consistent path using regular TCP. As a reference to the MPTCP test, the same test scenario was performed using single-path technology with regular TCP.

3.2 Downlink Test

To measure the downlink performance of MPTCP, the client downloads files of various sizes from the server. The download starts with the client in close proximity to the server. The client then moves away from the server as the download progresses.

This method enables the effect of mobility to be investigated, and how using MPTCP will affect the overall throughput of a device as it moves and associates to multiple APs.

3.3 Video Streaming

Recently, HTTP adaptive streaming has attracted so many researchers globally. This thesis investigates streaming HEVC-DASH using MPTCP. The server as shown in Figure 12 holds the HEVC-DASH public dataset as in [41] and the corresponding Media Presentation Description (MPD). The client requests for sets of the video sequence from the server using VLC player. Streaming statistics for MPTCP and SPTCP were obtained via Wireshark and VLC media player logged file.

3.4 Tools for Evaluation

This section gives a description of software tools that were employed for evaluating results.

3.4.1 Wireshark

This is a network packet capturing and analysis software with a Graphical User Interface (GUI). It is basically used for investigating network protocols, troubleshooting of the network at the micro level, and for educational purposes [42].

3.4.2 TCPdump

This is a command-line based packet analyser which captures packets transmitted over the network through various interfaces [43]. It mostly works on Mac and Linux. This was used in this thesis to capture packets on the client, for further analyses with Wireshark.

3.4.3 Iperf

Iperf is a tool for measuring the highest achievable bandwidth on IP networks. It supports different parameters that are tuned based on the test being carried out e.g.

timing, buffers, and protocols [44]. For each test, it gives the bandwidth, loss, and tuned parameters. Iperf was installed on both client and server (as required) and started as client and server respectively.

The flowchart of the method used in the thesis is shown in Figure 13.

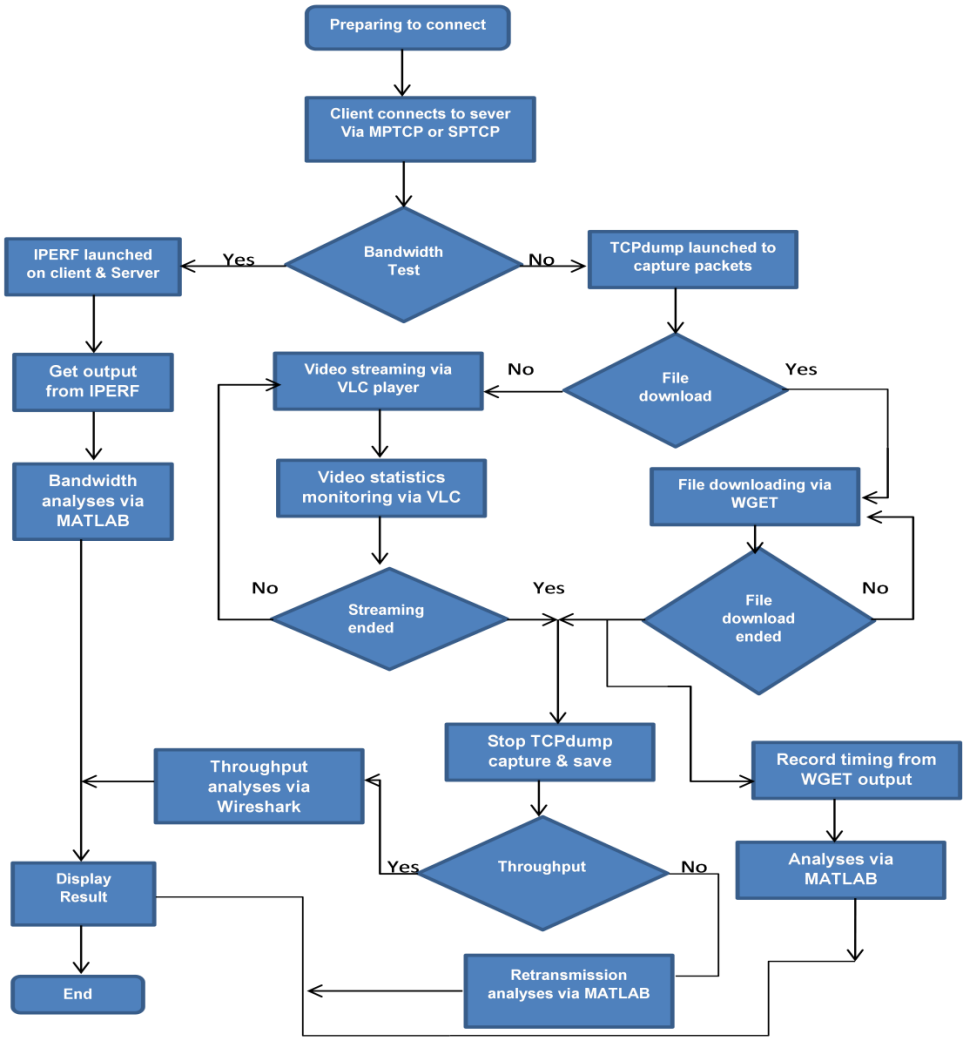


Figure 13. Flow chart of method used.

The methods this thesis uses to evaluate MPTCP have been explained in this chapter. Detailed description of experiments and results obtained using these methods are presented in the next chapter.

Chapter 4

EXPERIMENTS AND RESULTS

In this chapter, various experiments conducted on various setups and results are presented.

4.1 Throughput Test

Network throughput has been defined as the rate of successfully transmitted messages over the network. To test for the throughput, a file of size 121Mbytes was downloaded from MPTCP enabled Web Server at multipath-tcp.org as shown in Figure 14

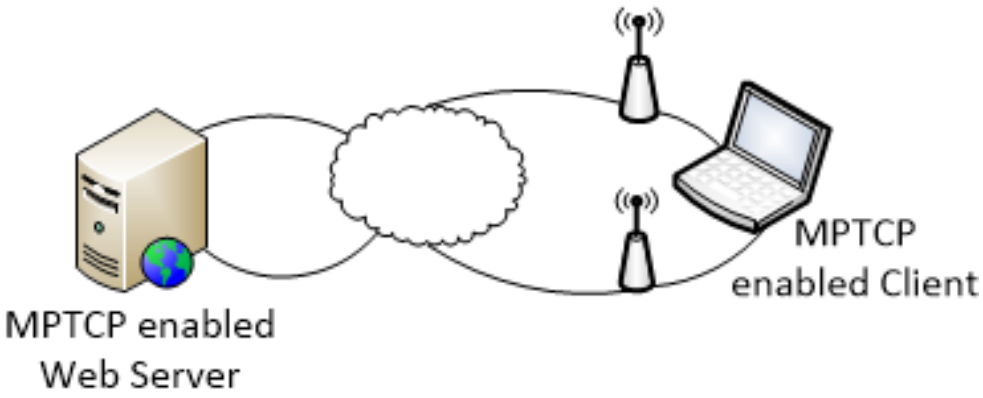


Figure 14. Setup for throughput evaluation

APs to which the client connects while downloading run at the speed of 54 Mbits/s with signal strength quality ranging from 42.9% to 57%. To capture packet traces TCPdump was installed on the Client. Wireshark, as shown in Figure 15, was used to view captured packets.

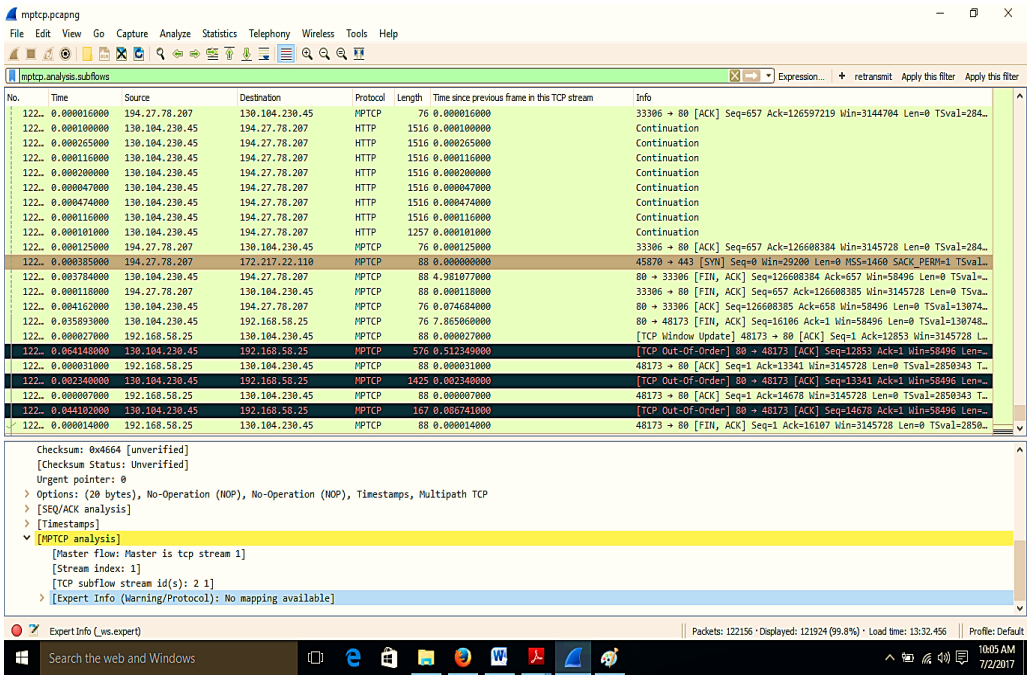
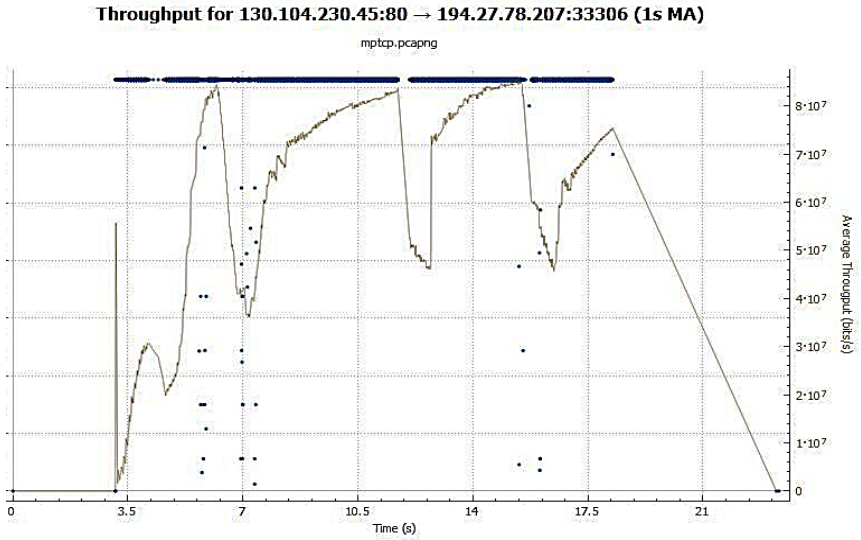
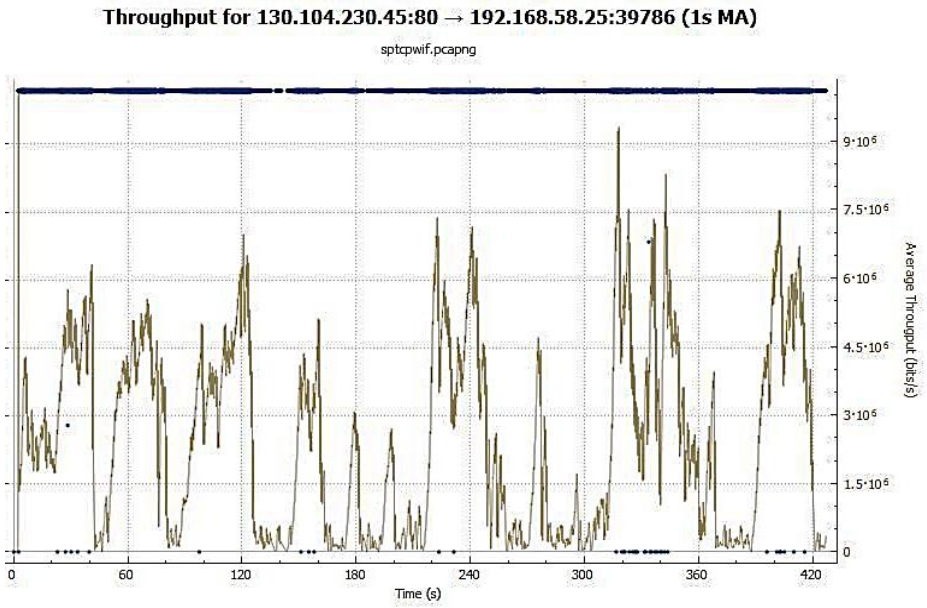


Figure 15. Viewing TCPdump capture using Wireshark.

To analyze the throughput, Wireshark's statistical features were used, and the results in Figures 16 (a) and (b) were obtained for MPTCP and SPTCP respectively.



(a)



(b)

Figure 16. Download throughput (a) MPTCP (b) SPTCP

Looking at the figures separately, it is understood that MPTCP gives slightly higher average throughput value of 5×10^7 bits**/s** compared to SPTCP which has 3×10^6 bits**/s** as a device downloads a file while connected to different APs than SPTCP. It is also comprehended that MPTCP maintains various levels of average throughput for a longer time than SPTCP. This is evident from the frequent drops in throughput experienced when using SPTCP.

4.2 Bandwidth

Unlike throughput which is the amount of data transmitted within a period of time, bandwidth measures the maximum number of bits per second on a link. To measure the bandwidth, IPERF, a bandwidth measuring tool was used.

IPERF was started as a Server on Server side, and as a Client on the Client. The server was connected to the Computer Engineering department's internal LAN network with speed of 100 Mbits/s and the University's WiFi network with speed at 54 Mbits/s.

Default settings of MPTCP with OLIA set as congestion control method was set on both client and server. Data was sent using IPERF from the client to the server as illustrated in Figure 17 for 10s. Default TCP window size of 45.0 Kbytes was maintained. To compare the bandwidth offered by MPTCP with SPTCP, the same procedure was done, with MPTCP disabled, and TCP congestion control changed to Reno. To see variability in results, the test was conducted 10 times.

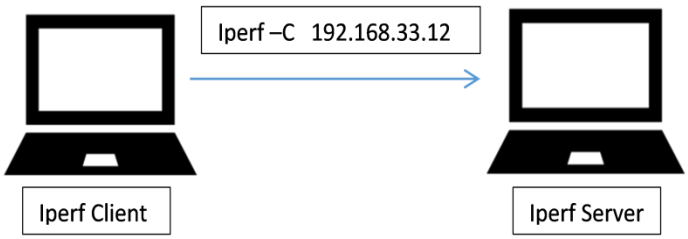


Figure 17. Bandwidth setup

Data recorded from the bandwidth test is given in Appendix A. Figure 18 shows the bandwidth comparison as analysed using MATLAB. Observations from the figure show that MPTCP gives average bandwidth value of 102.27 Mbits/s with a standard deviation of 9.78 compared to SPTCP with 23Mbits/s and 1.78 respectively.

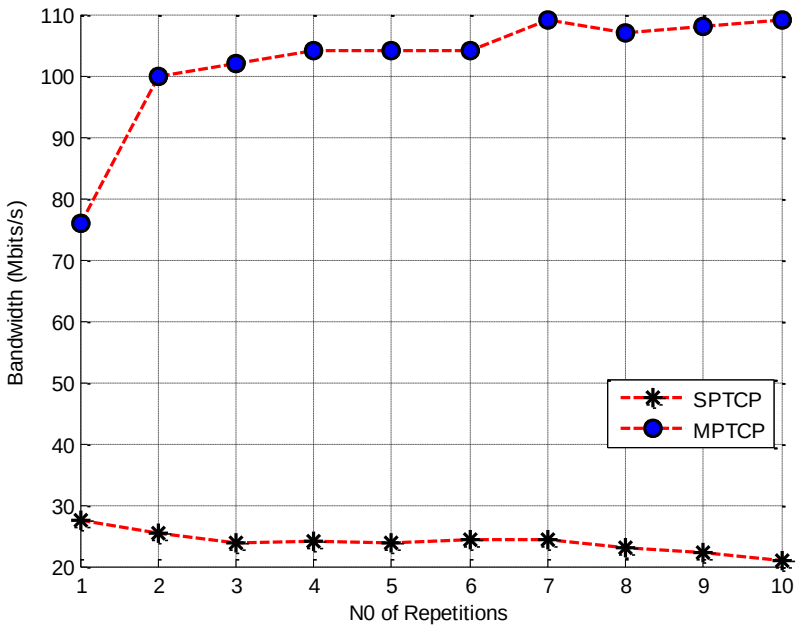


Figure 18. Bandwidth comparison

4.3 Download Times

To measure downlink characteristics, files of various sizes were downloaded from the server using the *WGET* command. Apache2 was installed on the proxy server and configured so as to enable the use of the *WGET* command. Since most mobile users are more interested in how fast their download is, the time it takes to download a file rather than download speed or bandwidth was measured. MPTCP and SPTCP configurations were as in the bandwidth setup. The download time from the WGET output and its MATLAB analysis are shown in Table 1 and Figure 19 respectively.

Observing the results show that file sizes of 0.007MB and 0.012MB show no difference in download time between MPTCP and SPTCP. This is because data at this point is transmitted on a single path. As a result of received data during slow start, MPTCP slightly performs better than SPTCP for file sizes ranging from 0.027MB to 0.27MB. As the file sizes increase from 20MB to 714MB, significant differences in

download time between MPTCP and SPTCP are observed. This is because of the efficient bandwidth aggregation of over the paths as the files are downloaded.

Table 1. Download time from WGET output

File Size(MB)	MPTCP Download Time(s)	STCP Download Time(s)
0.007	Negligible	Negligible
0.012	Negligible	Negligible
0.027	0.001	0.007
0.037	0.001	0.01
0.076	0.004	0.03
0.27	0.02	0.09
20	2	6.7
31	3.1	10
129	13	50
132	14	59
714	69	276

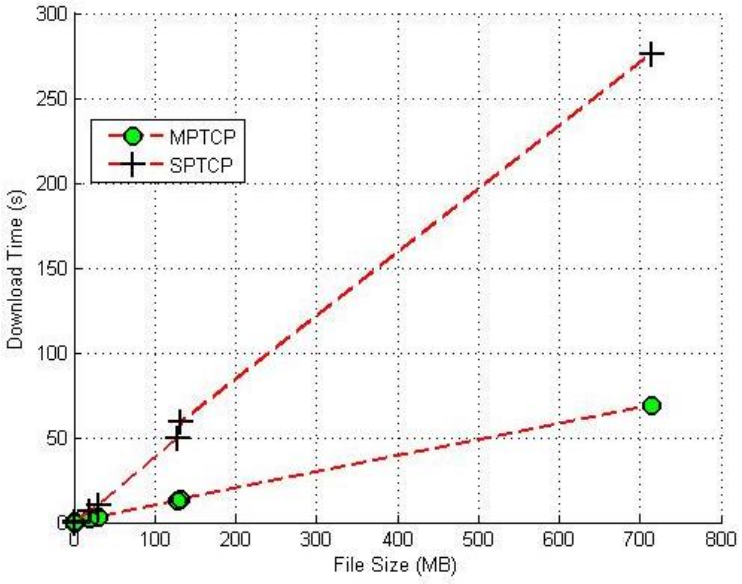


Figure 19. Download time comparison of MPTCP and SPTCP

4.4 Video Streaming

The tesbted for the video streaming is a client and server scenario as in earlier setups. The video files and the corresponding manifest files are stored on the server while the

client is equipped with VLC media player configured to log streaming status to a file for performance analysis. To make the client access the video files via HTTP, Apache HTTP server was installed and configured on the server. The dataset used for analysing streaming with MPTCP was the public UHD HEVC-DASH dataset in [41]. This was chosen so as to evaluate the performance of MPTCP with HEVC codec which is yet to be studied.

Video codec used was H.265 and media encapsulation format was MP4. The segments are of lengths 2s, 4s, 6s, 10s, and 20s, with bit rate ranging from 2 Mbps to 20 Mbps. A detailed description of the dataset is given in Appendix C. The 6-second segments, video files with 720p, 1080p, and 2160p resolutions were used.

Wondershaper and *tc netem* traffic shaping tools were employed to create balanced and unbalanced network conditions. The balanced network has same QoS (bandwidth, network delay, and packet loss) values on both paths and the unbalanced network with different QoS values. The metric considered for evaluation are download rate of segments, startup delay, and stalling.

4.5 Video Streaming Results

4.5.1 Startup Delay

As many adverts are predominantly presented in multimedia format over the Internet today, startup delay in video streaming becomes critical. In this, thesis network latency as a QoS parameter was used to evaluate initial delay when streaming with MPTCP. The following paths characteristics was created using *tc netem*

(a) MPTCP at balanced network

Path 1:10ms

Path 2: 10ms

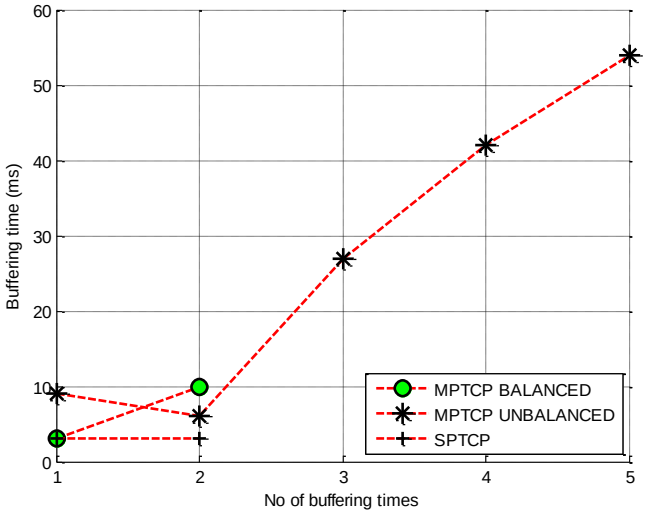
(b) MPTCP at unbalanced network

Path 1: 15ms

Path 2: 5ms

(c) SPTCP: 10ms

It can be deduced from the graph shown in Figure 20 that SPTCP and MPTCP in the balanced network have similar buffering time, while MPTCP in the unbalanced network performs worst. As packets are being transmitted over different subflows, the latency difference in the unbalanced case makes packets on the subflow with lower latency to wait for packets on the path with higher latency. This occurrence makes packets arrive out of order at the client side. This phenomenon makes the DASH client in the unbalanced case take a longer time with more buffering stalls, as reflected in Figure 20.



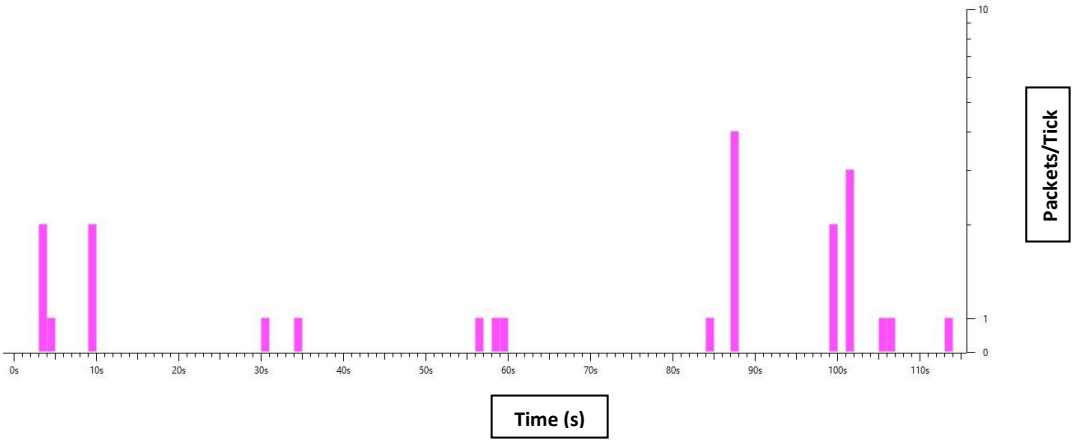
MPTCP balanced: Avg = 6.5ms MPTCP unbalanced: Avg = 27ms SPTCP: Avg = 3ms

Figure 20. Buffering comparison

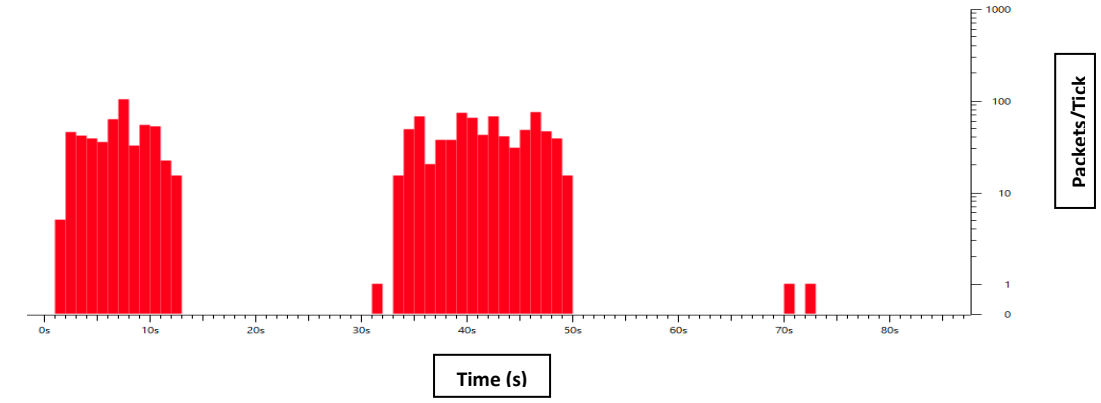
4.5.2 Stalling

This phenomenon in streaming is characterized by freezes or stoppage of playback during streaming. To evaluate this, packet loss rate in the network as a QoS parameter was used, and retransmission errors employed to evaluate its effects on video quality. This was chosen because of the reliable feature of TCP protocol which usually requests for retransmission of lost segments. As in the startup delay analyses, two network categories were considered, and *tc netem* was used to set packet loss values of 10% on both path for MPTCP in the balanced network, 15%/5% for MPTCP in the unbalanced network and 20% for SPTCP. Tcpdump was employed to capture packets on the interfaces, and Wireshark used to analyse retransmission.

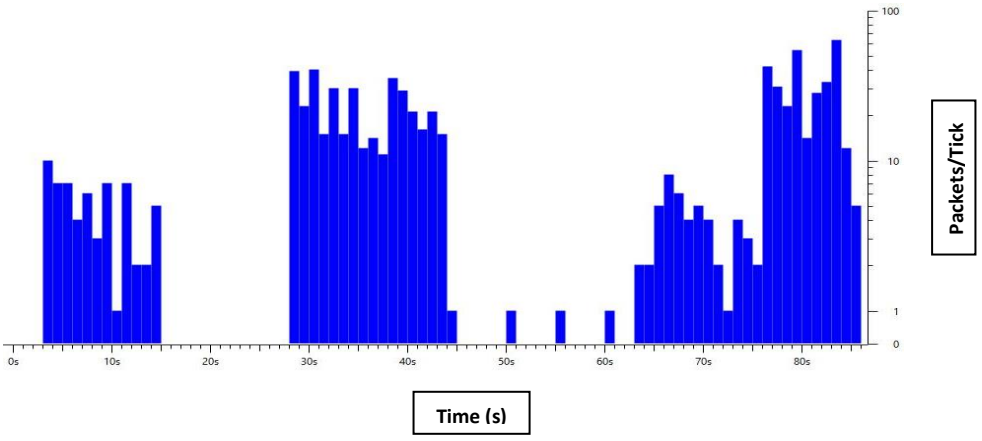
From the results for SPTCP and MPTCP in balanced and unbalanced cases as shown in Figure 21 (a), (b) and (c), it is observed that SPTCP with average retransmission rate of 1 packet per 1s tick interval, outperforms MPTCP in balanced and unbalanced cases with average retransmission rate of 14 packets per tick (14 packets/tick) and 9 packets per tick (9 packets/tick) respectively. Due to the effect of different path characteristics in MPTCP, high out-of-order data delivery is experienced at the receiver' end. This behaviour overwhelms the receiver buffer at the MPTCP level, leading to more packet drops and high packet retransmission rate observed when using MPTCP in either balanced or unbalanced cases.



(a)



(b)



(c)

Figure 21. Packet retransmission rate in (a) SPTCP (b) MPTCP balanced (c) MPTCP unbalanced

4.5.3 Download Rate

Bandwidth is an essential factor to consider when providing video content to consumers. It is very critical in DASH streaming in which the quality of video depends on the available network bandwidth. To study how bandwidth in MPTCP affects streaming the following paths characteristics for the balanced and unbalanced network conditions were created using *Wondershaper*, with paths considered as lossless paths (packet loss rate set to zero).

(a) MPTCP at balanced network
Path 1:15Mbps

Path 2: 15Mbps

(b) MPTCP at unbalanced network
Path 1: 25Mbps

Path 2: 5Mbps

(c) SPTCP: 30Mbps

The experiment for each network settings was repeated five (5) times and the average download rate for each network scenario as obtained from the media player was calculated. The graph for the segment vs average download rate for each network is shown in Figure 22.

From the graph, it is observed that MPTCP either in the balanced or unbalanced network condition, performs better than SPTCP. This is due to the advantage of path diversity in MPTCP. However, the unbalanced network conditions, when compared to MPTCP in the balanced scenario, show that when variability exists in bandwidth between paths, MPTCP performance in streaming is reduced.

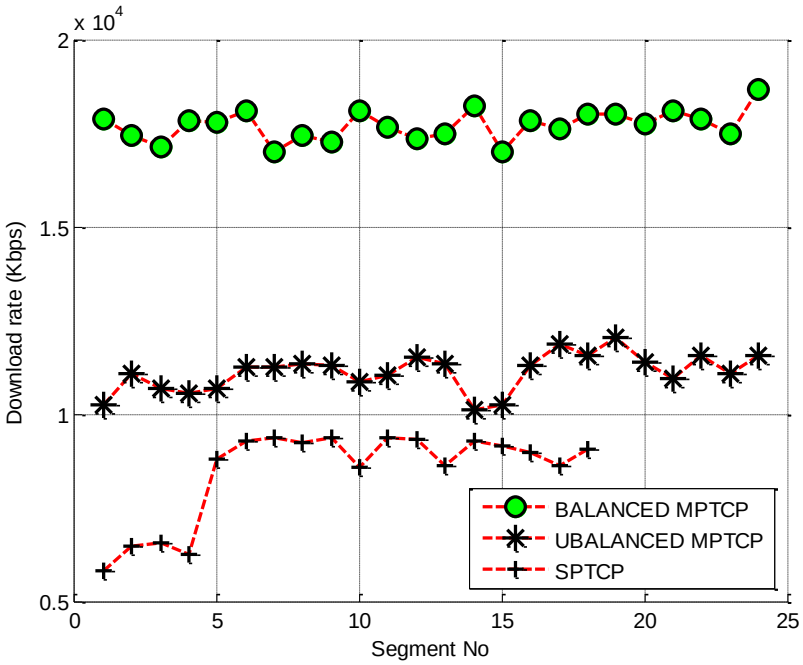


Figure 22. Download rate comparison

4.5.4 System and Media Player Challenge

During the streaming of UHD HEVC DASH, it was observed that though experimental results from the logged streaming file show sufficient relationship between QoS and QoE. The resolution of the system and the computational capability of the system has an effect on the quality of streaming. It was observed that the media player keeps dropping frames as a result of screen resolution not large enough to retain video resolution or computer being slow. Thus, a high computational-power computer, with a high quality graphic card could give better results.

Chapter 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

The demand for access to digital information by people and organizations has substantially increased in recent years. This is evident by the high global Internet and mobile traffic generated by various applications and services. To reduce the high increase of data traffic on the Internet and the demand for bandwidth by users, various techniques and technologies have been suggested and introduced by many researchers.

This research evaluates MPTCP performance in leveraging traffic and increasing bandwidth for transmitting different services in a mobile scenario. The state of the modern day Internet and the various multipath approaches in providing multipath solutions in relations to the multiple interfaces of present day devices have been presented. Theoretical and fundamental concepts of MPTCP, SPTCP, and DASH have also been explained.

Testbeds for comparing MPTCP and SPTCP performance in delivering data and multimedia services were setup, and experiments to evaluate MPTCP performance in mobile environment were conducted. From the results, it was shown that MPTCP can give better throughput and higher network bandwidth for delivery of data services in mobile environment. Furthermore, findings from the download time comparison show

that significant improvement is provided by MPTCP only when the files sizes get larger (between 31MB to 714MB in this thesis).

Though MPTCP gives higher throughput, the results from the delivery of multimedia services using UHD HEVC- DASH streaming, though specific instead of general, reveals a scenario in which MPTCP gives better quality and situations under which SPTCP performance is better. It was discovered that from the QoS viewpoint MPTCP under low packet loss rate, gives better streaming quality, by providing higher download rate when in balanced and unbalanced networks bandwidth conditions. However, path variability was found to affect MPTCP performance as the device roams from one APs to another, resulting in lower download rate in the unbalanced case. The video buffering delay results reveal that latency variation between paths greatly affects video buffering when streaming with MPTCP.

The contribution of this thesis is to study the delivery of different services on a device using MPTCP in a mobile environment and how various network conditions affect its performance. With the technology of femtocells and millimeter wave gaining attention, the findings from this study are expected to give network and wireless engineers the possibility of a MPTCP solution to some design constraint.

5.2 Future Work

Exploring the delivering of real-time services over MPTCP in a mobile scenario is a good future direction for research. Comparing the effect of varying path characteristics on performance of Multipath with other codec technique such as AVC, SVC, and interaction of DASH players are good further study areas.

REFERENCES

[1] J. Qadir, A. Sathiaseelan, and J. Crowcroft, “Resource Pooling For Wireless Networks: Solutions For The Developing World”. Arxiv preprint, arxiv 1602.07808, 2016.

[2] V. Olteanu and C. Raicin “Datacenter Scale Load Balancing for Multipath Transport”, In *Proc. 2016 Workshop. Hot Topics in Middleboxes and Network Function Virtualization*, NY, USA, pp 20–25, 2016.

[3] B. Cohen, “Incentives Build Robustness in BitTorrent”, In *Workshop. Economics. P2P System*, CA, USA, 2003.

[4] C. A. Barreia “Multipath Protocol”, MS Thesis, Dept of Telecom Eng, Technico Uni, Lisboa, 2014.

[5] C. Paasch “Improving Multipath TCP” Ph.D. Dissertation, Universit´e Catholique de Louvain, Belgium, 2014.

[6] D. Allan, P. Ashwood-Smith, N. Bragg, et al “Shortest Path Bridging: Efficient Control of Larger Ethernet Networks”, *IEEE Communications Magazine*, 48 (10), pp 128–135, 2010.

[7] WH. Tam and YC. Tseng, “Joint multi-channel link layer and multi-path routing design for wireless mesh networks”, In *26th IEEE Int. Conf. Computer Communications*, pp 2081–2089, 2007.

[8] Peschke and L.Victoria, “Combining DASH with MPTCP for Video Streaming”, MSC Thesis, UCL, Belgium, 2017.

[9] Mobile Video Delivery [Online]. Available. http://www.tvtechnology.com/cable-satellite-iptv/0149/mobile-video-delivery

[10] C. Timmerer , M. Maiero, B. Rainer, et al, “Quality of Experience of Adaptive HTTP Streaming in Real World Environment”, In *ICACCI2016,* India, pp 21-24, 2016.

[11] H. Sinky, B. Hamadaoui, and M. Guizani “Proactive Multipath TCP for Seamless Handoff In Heterogeneous Wireless Access Networks”, *IEEE Trans. Wireless Comm*, vol 15, (7), pp. 4754 – 4764, 2016.

[12] F. Rebecchi, M. Dias de Amorim, V. Conan, et al, “Data offloading techniques in cellular networks: A survey”, *IEEE Comm. Survey & Tutorials,* vol 17, (2), pp 580–603, 2015.

[13] S. Curtis, “Can you survive on 4g alone?” [Online]. Available: http://www.telegraph.co.uk/technology/internet/10272292/Can-you-survive-on-4G- alone.html.

[14] B. A Forouzan “Transmission Control Protocol”, In *TCP/IP*, 4th ed. New York, USA: McGraw-Hill, 2010, ch 15, sec 15.4, pp 442- 448.

[15] "Javis in Action" [Online]. https://www.isi.edu/nsnam/directed_research/dr_wanida/dr/javisinactionfastretransmitframe.html

[16] A. Afanasyev, N. Tilley, P. Reiher, et al, "Host-to-Host Congestion Control for TCP", *IEEE Comm Surveys & Tutorials*, vol 12(3) pp 304–342, 2010.

[17] L. Brakmo and L. Peterson. "TCP Vegas: End to End Congestion Avoidance on a Global Internet", *IEEE Journal o Selected Areas in Comm*, vol 13, (8), pp 1465–1480, 1995.

[18] O. Bonaventure, M. Handley, and C. Raiciu: "An overview of Multipath TCP", *USENIX*, vol 37, (5), 2012.

[19] M. Handley, C. Raiciu, A. Ford, et al, "Architectural Guidelines for Multipath TCP Development", [Online]. Available: http://tools.ietf.org/html/rfc6182

[20] J. Sherry, S. Hasan, C. Scott, et al, "Making Middleboxes Someone Else's Problem: Network Processing as a Cloud Service," In *Proc. ACM SIGCOMM*, pp 13–24, 2012.

[21] B. Ford, and J. Iyengar, "Breaking Up the Transport Logjam, *ACM HotNets*, 2008.

[22] S. Ro, and D.N Van, "Performance Evaluation of MPTCP over a Shared Bottleneck Link" *IJCCE*, vol 5, (3), 2016.

[23] C. Paasch, F. Duchene, G. Detal, et al, “Exploring mobile/wifi Handover with multipath TCP”. In *Proc. ACM SIGCOMM Workshop. Cellular Networks*, 2012.

[24] A. Gokhan, “Exploring Mobile/WiFi Handover with Multipath TCP”, BS, Dept of EE Eng, EMU, Famagusta, 2015.

[25] S. Brim, and B.E Carpenter, “Middleboxes: Taxonomy and Issues”, RFC 3234, [Online]. Available. https://tools.ietf.org/html/rfc3234

[26] C. Paasch and O. Bonaventure. “Multipath TCP”. [Online] Available: http://doi.acm.org/10.1145/2578508.2591369

[27] C. Raiciu, O. Bonaventure, and J. Iyengar, “Recent Advances in Reliable Transport Protocols”, *In SIGCOMM ebook on Recent Advances in Networking,* 2003.

[28] “Configure MPTCP” [Online]. Available: http://www.multipath-tcp.org.

[29] F. Demaria, “Security Evaluation of MPTCP”, MS Thesis, KTH Uni, Stockholm, Sweden, 2016.

[30] M. Bagnulo, F. Cont, O. Bonaventure, et al “Analysis of Residual Threats and Possible Fixes for Multipath TCP (MPTCP)”, RFC-7430, 2015.

[31] M. Jadin, G. Tihon, O. Pereira, and O. Bonaventure, “Securing Multipath TCP: Design and Implementation”, In *IEEE INFOCOM*, 2017.

[32] V. Jacquot, “Test Bed for Multipath TCP ”, MS Thesis, Dept of Comm & Net, AALTO Uni, Espoo, Finland, 2012.

[33] R. Stewart, M. Belinchon, Q. Xie, et al, “Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration,” RFC 5061, *IETF*, 2007. [Online]. Available: http://tools.ietf.org/rfc/rfc5061

[34] M. Riegel and M. Tuexen, “Mobile SCTP,” *Working Draft, IETF Secretariat*, Internet-draft draft-riegel-tuexen-mobile-sctp-09, 2007. [Online]. Available: http://tools.ietf.org/id/draft-riegel-tuexen-mobile-sctp-09.txt

[35] A. Ezzouhairi, A. Quintero, and S. Pierre, “A New SCTP Mobility Scheme Supporting Vertical Handover”, In *Proc IEEE WiMob, Int. Conf*, pp 205–211, 2006.

[36] YC. Chen, R. Khalili, J. Gibbens, et al, “A Measurement Based Study of Multipath TCP Performance over wireless networks” In *ACM SIGCOMM IMC*, 2013.

[37] B.H. Oh and J. Lee “Constrained – Based Proactive Scheduling for Multipath TCP in Wireless Networks” *Computer Networks*, vol 91, pp 548-563, 2015.

[38] A. Abdrabou and M. Prakash “Experimental Performance Study of Multipath TCP over Heterogeneous Wireless Networks” In *IEEE LCN*, pp 172 – 175, 2016.

[39] A. Croitoru, D. Niculescu, and C. Raiciu. “Towards WiFi Mobility without Fast Handover. In *Proc. USENIX NSDI*, pp 219 – 234, 2015.

[40] C. James, R. Jana, am. Wang, et al “Is Multipath TCP (MPTCP) beneficial for video streaming over DASH” In *MASCOTS 2016, IEEE 24th Int. Symp* pp 331 – 336, 2016.

[41] J. Le Feuvre, J. M Thiesse, M. Parmentier, et al, “Ultra High Definition HEVC Dash Dataset”, In *MMSys’14, 5th ACM Multimedia Systems Conf*, pp 7-12, 2014.

[42] “About Wireshark”, [Online]. Available: http://www.wireshark.org

[43] “TCPDUMP”. [Online]. Available: http://tcpdump.org

[44] “What is Iperf/Iperf3”. [Online]. Available: http://www.iperf.fr

APPENDICES

Appendix A: Iperf Results for Bandwidth Measurement

SPTCP Bandwidth

```
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34274 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  33.0 MBytes  27.4 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34276 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  30.6 MBytes  25.5 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34278 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  28.9 MBytes  23.9 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34280 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  29.0 MBytes  24.2 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34282 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  28.6 MBytes  23.8 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34284 connected with 192.168.37.110 port 5001
```

```
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  29.2 MBytes  24.3 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34286 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  29.4 MBytes  24.4 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34288 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  27.6 MBytes  22.9 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34290 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  26.8 MBytes  22.3 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34292 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  25.0 MBytes  20.8 Mbits
```

MPTCP Bandwidth Test

```
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34294 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  90.4 MBytes  75.8 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
```

```
[  3] local 192.168.32.175 port 34296 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   119 MBytes  99.9 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34298 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   122 MBytes   102 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34300 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   125 MBytes   104 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34302 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   124 MBytes   104 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34304 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   124 MBytes   104 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34306 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   130 MBytes   109 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34308 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
```

```
[  3]  0.0-10.0 sec   128 MBytes   107 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34310 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   129 MBytes   108 Mbits/sec
john@john-SATELLITE-C50-B:~$ sudo iperf -c 192.168.37.110
------------------------------------------------------------
Client connecting to 192.168.37.110, TCP port 5001
TCP window size: 45.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.32.175 port 34312 connected with 192.168.37.110 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec   131 MBytes   109 Mbits/sec
```

Appendix B: WGET Download Output

MPTCP

```
john@john-SATELLITE-C50-B:~$ clear
john@john-SATELLITE-C50-B:~$ wget
http://192.168.37.110/mptcpwire/Sully.2016.720p.BluRay.x264-
%5bYTS.AG%5d.mp4
--2017-08-17 11:13:48--
http://192.168.37.110/mptcpwire/Sully.2016.720p.BluRay.x264-
%5bYTS.AG%5d.mp4
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 748210276 (714M) [video/mp4]
Saving to: ‘Sully.2016.720p.BluRay.x264-[YTS.AG].mp4’

Sully.2016.720p.Blu 100%[==================>] 713,55M  10,1MB/s    in
69s

2017-08-17 11:14:58 (10,3 MB/s) - ‘Sully.2016.720p.BluRay.x264-[YTS.AG].mp4’
saved [748210276/748210276]
```

```
john@john-SATELLITE-C50-B:~$ wget
http://192.168.37.110/mptcpwire/Gaussian%20noise%20and%20Gaussian%20filter
%20implementation%20using%20Matlab.mp4
--2017-08-17 11:16:41--
http://192.168.37.110/mptcpwire/Gaussian%20noise%20and%20Gaussian%20filter
%20implementation%20using%20Matlab.mp4
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 32027425 (31M) [video/mp4]
Saving to: ‘Gaussian noise and Gaussian filter implementation using Matlab.mp4’

Gaussian noise and  100%[==================>]  30,54M  9,79MB/s    in
3,1s

2017-08-17 11:16:45 (9,79 MB/s) - ‘Gaussian noise and Gaussian filter
implementation using Matlab.mp4’ saved [32027425/32027425]
```

```
_
john@john-SATELLITE-C50-B:~$
http://192.168.37.110/mptcpwire/OneDriveSetup.exe
bash: http://192.168.37.110/mptcpwire/OneDriveSetup.exe: No such file or directory
john@john-SATELLITE-C50-B:~$ wget
http://192.168.37.110/mptcpwire/OneDriveSetup.exe
--2017-08-17 11:17:44--  http://192.168.37.110/mptcpwire/OneDriveSetup.exe
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 20738752 (20M) [application/x-msdos-program]
Saving to: ‘OneDriveSetup.exe’
```

OneDriveSetup.exe 100%[==================>] 19,78M 9,69MB/s in 2,0s

2017-08-17 11:17:46 (9,69 MB/s) - ‘OneDriveSetup.exe’ saved [20738752/20738752]

---

john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/mptcp.pcapng
--2017-08-17 11:18:32-- http://192.168.37.110/mptcpwire/mptcp.pcapng
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 135160332 (129M)
Saving to: ‘mptcp.pcapng’

mptcp.pcapng 100%[==================>] 128,90M 9,83MB/s in 13s

2017-08-17 11:18:46 (9,69 MB/s) - ‘mptcp.pcapng’ saved [135160332/135160332]

john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/sptcpwif.pcapng
--2017-08-17 11:19:39-- http://192.168.37.110/mptcpwire/sptcpwif.pcapng
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 138779683 (132M)
Saving to: ‘sptcpwif.pcapng’

sptcpwif.pcapng 100%[==================>] 132,35M 9,24MB/s in 14s

2017-08-17 11:19:53 (9,33 MB/s) - ‘sptcpwif.pcapng’ saved [138779683/138779683]

_
john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/md5sum.txt
--2017-08-17 11:21:17-- http://192.168.37.110/mptcpwire/md5sum.txt
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 21452 (21K) [text/plain]
Saving to: ‘md5sum.txt’

md5sum.txt 100%[==================>] 20,95K --.-KB/s in 0s

2017-08-17 11:21:17 (86,7 MB/s) - ‘md5sum.txt’ saved [21452/21452]

_
john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/filevideo.odt
--2017-08-17 11:22:07-- http://192.168.37.110/mptcpwire/filevideo.odt
Connecting to 192.168.37.110:80... connected.

HTTP request sent, awaiting response... 200 OK
Length: 28315 (28K) [application/vnd.oasis.opendocument.text]
Saving to: ‘filevideo.odt’

filevideo.odt 100%[==========>] 27,65K --.-KB/s in 0,001s

2017-08-17 11:22:08 (6,07 MB/s) - ‘filevideo.odt’ saved [28315/28315]

---

john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/dff.jpg
--2017-08-17 11:22:57-- http://192.168.37.110/mptcpwire/dff.jpg
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7574 (7,4K) [image/jpeg]
Saving to: ‘dff.jpg’

dff.jpg 100%[==========>] 7,40K --.-KB/s in 0s

2017-08-17 11:22:57 (102 MB/s) - ‘dff.jpg’ saved [7574/7574]

---

john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/Flare.jpg
--2017-08-17 11:23:54-- http://192.168.37.110/mptcpwire/Flare.jpg
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 38480 (38K) [image/jpeg]
Saving to: ‘Flare.jpg’

Flare.jpg 100%[==========>] 37,58K --.-KB/s in 0.001s

2017-08-17 11:23:54 (119 MB/s) - ‘Flare.jpg’ saved [38480/38480]

---

john@john-SATELLITE-C50-B:~$ wget
http://192.168.37.110/mptcpwire/Exams.pdf
--2017-08-17 11:24:40-- http://192.168.37.110/mptcpwire/Exams.pdf
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 285193 (279K) [application/pdf]
Saving to: ‘Exams.pdf’

Exams.pdf 100%[==========>] 278,51K --.-KB/s in 0,02s

2017-08-17 11:24:40 (13,8 MB/s) - ‘Exams.pdf’ saved [285193/285193]

---

john@john-SATELLITE-C50-B:~$ wget
http://192.168.37.110/mptcpwire/CYEOab6U0AA-HR9.jpg
--2017-08-17 11:25:33-- http://192.168.37.110/mptcpwire/CYEOab6U0AA-
HR9.jpg
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 80810 (79K) [image/jpeg]
Saving to: ‘CYEOab6U0AA-HR9.jpg’

CYEOab6U0AA-HR9.jpg 100%[==================>] 78,92K --.-KB/s in 0,001s

2017-08-17 11:25:33 (53,6 MB/s) - ‘CYEOab6U0AA-HR9.jpg’ saved [80810/80810]

SPTCP Download

john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/Sully.2016.720p.BluRay.x264-%5bYTS.AG%5d.mp4
--2017-08-17 11:34:56-- http://192.168.37.110/mptcpwire/Sully.2016.720p.BluRay.x264-%5bYTS.AG%5d.mp4
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 748210276 (714M) [video/mp4]
Saving to: ‘Sully.2016.720p.BluRay.x264-[YTS.AG].mp4.1’

Sully.2016.720p.Blu 100%[==================>] 713,55M 2,75MB/s in 4m 36s

2017-08-17 11:39:32 (2,59 MB/s) - ‘Sully.2016.720p.BluRay.x264-[YTS.AG].mp4.1’ saved [748210276/748210276]

---

john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/Gaussian%20noise%20and%20Gaussian%20filter%20implementation%20using%20Matlab.mp4
--2017-08-17 11:40:17-- http://192.168.37.110/mptcpwire/Gaussian%20noise%20and%20Gaussian%20filter%20implementation%20using%20Matlab.mp4
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 32027425 (31M) [video/mp4]
Saving to: ‘Gaussian noise and Gaussian filter implementation using Matlab.mp4.1’

Gaussian noise and 100%[==================>] 30,54M 2,93MB/s in 10s

2017-08-17 11:40:27 (2,93 MB/s) - ‘Gaussian noise and Gaussian filter implementation using Matlab.mp4.1’ saved [32027425/32027425]

_
john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/OneDriveSetup.exe
--2017-08-17 11:41:08-- http://192.168.37.110/mptcpwire/OneDriveSetup.exe
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 20738752 (20M) [application/x-msdos-program]
Saving to: ‘OneDriveSetup.exe.1’

OneDriveSetup.exe.1 100%[==================>] 19,78M 2,98MB/s in 6,7s

2017-08-17 11:41:14 (2,94 MB/s) - ‘OneDriveSetup.exe.1’ saved [20738752/20738752]

---

john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/mptcp.pcapng
--2017-08-17 11:41:54-- http://192.168.37.110/mptcpwire/mptcp.pcapng
Connecting to 192.168.37.110:80... ^C
john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/mptcp.pcapng
--2017-08-17 11:43:34-- http://192.168.37.110/mptcpwire/mptcp.pcapng
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 135160332 (129M)
Saving to: ‘mptcp.pcapng.1’

mptcp.pcapng.1 100%[==================>] 128,90M 2,79MB/s in 50s

2017-08-17 11:44:24 (2,59 MB/s) - ‘mptcp.pcapng.1’ saved [135160332/135160332]

---

john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/sptcpwif.pcapng
--2017-08-17 11:45:06-- http://192.168.37.110/mptcpwire/sptcpwif.pcapng
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 138779683 (132M)
Saving to: ‘sptcpwif.pcapng.1’

sptcpwif.pcapng.1 100%[==================>] 132,35M 2,87MB/s in 59s

2017-08-17 11:46:06 (2,24 MB/s) - ‘sptcpwif.pcapng.1’ saved [138779683/138779683]

_
john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/md5sum.txt
--2017-08-17 11:46:35-- http://192.168.37.110/mptcpwire/md5sum.txt
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 21452 (21K) [text/plain]
Saving to: ‘md5sum.txt.1’

md5sum.txt.1 100%[==================>] 20,95K --.-KB/s in 0,006s

2017-08-17 11:46:35 (3,51 MB/s) - ‘md5sum.txt.1’ saved [21452/21452]

```
john@john-SATELLITE-C50-B:~$ wget
http://192.168.37.110/mptcpwire/filevideo.odt
--2017-08-17 11:47:02--  http://192.168.37.110/mptcpwire/filevideo.odt
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 28315 (28K) [application/vnd.oasis.opendocument.text]
Saving to: ‘filevideo.odt.1’

filevideo.odt.1     100%[===================>]  27,65K  --.-KB/s    in 0,007s

2017-08-17 11:47:02 (3,95 MB/s) - ‘filevideo.odt.1’ saved [28315/28315]
```

```
john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/dff.jpg
--2017-08-17 11:47:35--  http://192.168.37.110/mptcpwire/dff.jpg
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7574 (7,4K) [image/jpeg]
Saving to: ‘dff.jpg.1’

dff.jpg.1           100%[===================>]   7,40K  --.-KB/s    in 0s

2017-08-17 11:47:35 (200 MB/s) - ‘dff.jpg.1’ saved [7574/7574]
```

```
john@john-SATELLITE-C50-B:~$ wget http://192.168.37.110/mptcpwire/Flare.jpg
--2017-08-17 11:48:12--  http://192.168.37.110/mptcpwire/Flare.jpg
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 38480 (38K) [image/jpeg]
Saving to: ‘Flare.jpg.1’

Flare.jpg.1         100%[===================>]  37,58K  --.-KB/s    in 0,01s

2017-08-17 11:48:12 (2,66 MB/s) - ‘Flare.jpg.1’ saved [38480/38480]
```

```
john@john-SATELLITE-C50-B:~$ wget
http://192.168.37.110/mptcpwire/Exams.pdf
--2017-08-17 11:48:39--  http://192.168.37.110/mptcpwire/Exams.pdf
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 285193 (279K) [application/pdf]
Saving to: ‘Exams.pdf.1’

Exams.pdf.1         100%[===================>] 278,51K  --.-KB/s    in 0,09s

2017-08-17 11:48:39 (2,87 MB/s) - ‘Exams.pdf.1’ saved [285193/285193]
```

```
_
john@john-SATELLITE-C50-B:~$ wget
http://192.168.37.110/mptcpwire/CYEOab6U0AA-HR9.jpg
```

--2017-08-17 11:49:11-- http://192.168.37.110/mptcpwire/CYEOab6U0AA-HR9.jpg
Connecting to 192.168.37.110:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 80810 (79K) [image/jpeg]
Saving to: ‘CYEOab6U0AA-HR9.jpg.1’

CYEOab6U0AA-HR9.jpg 100%[==========>] 78,92K --.-KB/s in 0,03s

2017-08-17 11:49:11 (2,85 MB/s) - ‘CYEOab6U0AA-HR9.jpg.1’ saved [80810/80810]

Appendix C: HEVC-DASH Dataset

The sequences provided in this dataset are a professional edit of several sequences shot during the 4Ever project. The edited sequence is an UHDTV 3840x2160 progressive video at 60 Hz and lasts 8536 frames, which corresponds to 2 minutes, 22 seconds and 226 milliseconds (16 frames). The sequence has been spatially down-sampled to generate HD (1280x720p60) and Full HD (1920x1080p60) sequences, which in turn have been temporally down-sampled at 30 Hz. The UHDTV sequence has not been temporally down-sampled, as most subjective viewing tests conducted by the 4Ever project on 4K materials at 30 Hz were not advocating for this.

The provided DASH sequences provide HEVC encoding ranging from 720p30 @ 2Mbps up to 2160p60 @ 20 Mbps, with one 1080p60 and one 2160p60 in 10 bits.

Resolution	**Frame Rate**	**Bit Depth**	**Bit rate**	**Dash ID Representation**
720p	30Hz	8	2Mbps	V1
720p	30Hz	8	3Mbps	V2
720p	60Hz	8	3Mbps	V3
720p	60Hz	8	4Mbps	V4
1080p	30Hz	8	4Mbps	V5
1080p	30Hz	8	6Mbps	V6
1080p	60Hz	8	6Mbps	V7
1080p	60Hz	8	8Mbps	V8
1080p	60Hz	10	8Mbps	V9
2160p	60Hz	8	12Mbps	V10
2160p	60Hz	8	15Mbps	V11
2160p	60Hz	8	20Mbps	V12
2160p	60Hz	10	20Mbps	V13

They are packaged using segment length of 2s, 4s, 6s, 10s, and 20s, and cover live, on-demand and Main profiles.

The 60 Hz links only describe 720p and 1080p content at 60 Hz, in Main profile (8 bpp).

The 4K links only describe 2160p content at 60 Hz, with bitrates ranging from 4 Mbps to 8 Mbps, in Main profile (8 bpp).

The 10-bit links only describe 1080p and 2160p content at 60 Hz in Main10 profile (10 bpp).

Appendix D: Media Presentation Description

6seconds 4k

```
<?xml version="1.0"?>
<!-- MPD file Generated with GPAC version 0.5.2-DEV-rev40-gd978dd3-master on 2015-02-10T09:19:58Z-->
<MPD xmlns="urn:mpeg:dash:schema:mpd:2011" minBufferTime="PT1.500000S" type="static" mediaPresentationDuration="PT0H2M31.85S" profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">
 <ProgramInformation moreInformationURL="http://gpac.sourceforge.net">
  <Title>dashevc-ondemand-6s-4k.mpd generated by GPAC</Title>
  <Copyright>Provided by 4Ever Project; Distributed under Creative Common (CC BY NC ND); Generated by GPAC MP4Box - GPAC version 0.5.2-DEV-rev40-gd978dd3-master</Copyright>
 </ProgramInformation>

 <Period duration="PT0H2M31.85S">
  <AdaptationSet segmentAlignment="true" maxWidth="3840" maxHeight="2160" maxFrameRate="60" par="16:9" lang="und" subsegmentStartsWithSAP="1">
   <Representation id="v9" mimeType="video/mp4" codecs="hvc1.1.6.L153.90" width="3840" height="2160" frameRate="60" sar="1:1" startWithSAP="1" bandwidth="10598412">
    <BaseURL>dashevc-ondemand-6s-v9.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="1010-1329">
     <Initialization range="0-1009"/>
    </SegmentBase>
   </Representation>
   <Representation id="v10" mimeType="video/mp4" codecs="hvc1.1.6.L153.90" width="3840" height="2160" frameRate="60" sar="1:1" startWithSAP="1" bandwidth="13539214">
    <BaseURL>dashevc-ondemand-6s-v10.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="1010-1329">
     <Initialization range="0-1009"/>
    </SegmentBase>
   </Representation>
   <Representation id="v11" mimeType="video/mp4" codecs="hvc1.1.6.L153.90" width="3840" height="2160" frameRate="60" sar="1:1" startWithSAP="1" bandwidth="18363720">
    <BaseURL>dashevc-ondemand-6s-v11.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="1010-1329">
     <Initialization range="0-1009"/>
    </SegmentBase>
   </Representation>
  </AdaptationSet>
  <AdaptationSet segmentAlignment="true" lang="fr" subsegmentStartsWithSAP="1">
   <Representation id="a1" mimeType="audio/mp4" codecs="mp4a.40.1" audioSamplingRate="24000" startWithSAP="1" bandwidth="129216">
```

```
    <AudioChannelConfiguration
schemeIdUri="urn:mpeg:dash:23003:3:audio_channel_configuration:2011"
value="2"/>
    <BaseURL>dashevc-ondemand-6s-a1.mp4</BaseURL>
    <SegmentBase indexRangeExact="true" indexRange="820-1163">
      <Initialization range="0-819"/>
    </SegmentBase>
   </Representation>
  </AdaptationSet>
 </Period>
</MPD>
```

Appendix E: VLC Streaming Statistics

a. Buffering statistics in balanced network

MPTCP(at latency 10ms/10ms)			SPTCP(at latency 10ms)	
No. of Buffering Times	**Initial Buffering(ms)**		**No. of Buffering Times**	**Initial Buffering(ms)**
1	3		1	3
2	10		2	3
-	-		-	-
-	-		-	-

b. Buffering statistics in unbalanced network

MPTCP(at latency 15ms/5ms)			SPTCP(at latency 10ms)	
No. of Buffering Times	**Initial Buffering(ms)**		**No. of Buffering Times**	**Initial Buffering(ms)**
1	9		1	3
2	6		2	3
3	27		-	-
4	42		-	-
5	54		-	-

c. Video segments download rate statistics

MPTCP BALANCE NETWORK																									
	V9n1	v9n2	v9n3	v9n4	v9n5	v9n6	v9n7	v9n8	v9n9	v9n10	v9n11	v9n12	v9n13	v9n14	v9n15	v9n16	v9n17	v9n18	v9n19	v9n20	v9n21	v9n22	v9n23	v9n24
Trial 1(15/15)	15473	18176	17592	17443	16839	18087	18486	18340	18074	18233	18324	18398	18022	18123	17457	16332	17734	16863	18424	16158	17717	16639	14636	18925
Trial 1(15/15)	18913	17075	16790	18511	18550	17361	13412	16380	18405	18077	17242	18329	17491	18355	18461	17649	18448	18170	17133	18521	18377	18396	17828	18925
Trial 3(15/15)	17357	17927	18493	18530	17845	18555	17689	16318	13801	18230	18480	17742	18133	18511	17135	18517	17980	18470	17705	18286	18212	18056	18363	18318
Trial 4(15/15)	18871	18016	18235	18531	17354	18264	17222	18238	18346	17563	16403	13841	15390	18478	14820	18529	16746	17932	18471	17156	18393	18159	18075	19089
Trial4(15/15)	18741	15882	14471	16158	18314	18151	18215	17880	17755	18271	17871	18506	18376	17572	17023	18177	17140	18606	18281	18591	17812	18178	18485	17954
AVERAGE	17871	17415.2	17116.2	17834.6	17780.4	18083.6	17004.8	17431.2	17276.2	18074.8	17664	17363.2	17482.4	18207.8	16979.2	17840.8	17609.6	18008.2	18002.8	17742.4	18102.2	17885.6	17477.4	18642.2
STDEV	1488.058	957.6214	1618.423	1047.665	698.0754	442.0032	2066.522	1002.809	1959.682	295.615	853.3039	1991.165	1213.588	386.6648	1333.45	916.1278	674.8917	691.7537	574.2937	1057.831	318.0923	707.7784	1608.815	484.3312
MPTCP UNBALANCE NETWORK																								
Trial 1(25/5))	9810	11466	10122	10970	10896	11378	10420	11571	11745	12552	12041	11810	11990	8409	9536	10554	12485	12006	12435	10237	10157	11771	10874	12185
Trial 1(25/5)	9783	10987	10812	9201	9576	10713	11623	10925	10904	10833	11084	11174	11970	10322	11380	11413	11220	12316	12489	13120	12196	12406	11544	6718
Trial 3(25/5)	10804	11855	11201	10419	10603	10402	10154	10507	10465	10911	10917	12471	12021	10442	9162	12432	13019	11907	12957	12019	12040	12048	11425	13696
Trial 4(25/5)	10227	9609	10379	11933	12200	12274	12587	12005	12097	10234	11111	11225	11261	10617	10750	10632	11114	10191	10729	10577	9527	11175	10907	12711
Trial 5(25/5)	10550	11463	10779	10219	10216	11538	11566	11696	11241	9684	9915	10832	9385	10711	10475	11439	11587	11340	11602	10996	10782	10470	10587	12531
AVERAGE	10234.8	11076	10658.6	10548.4	10698.2	11261	11270	11340.8	11290.4	10842.8	11013.6	11502.4	11325.4	10100.2	10260.6	11294	11885	11552	12042.4	11389.8	10940.4	11574	11067.4	11568.2
STDEV	449.4382	875.8881	417.8341	1004.44	974.0946	733.9571	989.2763	610.0067	650.0568	1076.983	756.0171	645.569	1130.305	957.3843	904.2266	760.9327	832.38	838.6152	881.6251	1176.239	1164.276	763.8269	402.7844	2768.817
SPTCP																								
Trial 1(30)	9363	9271	9439	8698	7685	9093	9434	9176	9444	9123	9425	9125	9441	8967	9444	8057	9408	9061						
Trial 1(30)	754	2720	3656	1857																				
Trial 3(30)	8372	8995	9421	8463	9234	9361	9162	9441	9341	9419	9294	9441	8217	9443	8680	9446	7126							
Trial 4(30)	1518	2476	771	4652																				
Trial 5(30)	9136	8985	9437	7495	9445	9355	9439	9090	9322	7220	9365	9438	8177	9448	9267	9441	9359							
AVERAGE	5828.6	6489.4	6544.8	6233	8788	9269.667	9345	9235.667	9369	8587.333	9361.333	9334.667	8611.667	9286	9130.333	8981.333	8631	9061						
STDEV	4307.917	3555.246	4083.371	2927.913	961.0343	153.0272	158.5024	182.949	65.64297	1193.358	65.57693	181.5829	718.5021	276.2734	399.9154	800.5001	1303.598							