KHAZAR UNIVERSITY

Faculty: Engineering and Applied Sciences

Department: Computer Science

Specialty: Computer Engineering

MA THESIS

Theme: Multipath RTP for video

Master Student: Vidadi Aliyev

Supervisor: Ph.D. Javad Mehri-Tekmeh

BAKU - 2015

KHAZAR UNIVERSITY

ENGINEERING AND APPLIED SCIENCES

COMPUTER SCIENCE DEPARTMENT

ABSTRACT

OF DISSERTATION FOR MASTER'S DEGREE

THEME

Multipath RTP for Video

Master student: Vidadi Aliyev

Supervisor: Ph.D. Javad Mehri-Tekmeh

Abstract

This is a dissertation about Multipath RTP for video streaming. The dissertation introduces readers with a new technology – Multipath TCP (MPTCP). Using MPTCP we can save more energy, e.g. we can save money and provide people with the new, non-congested connection. MPTCP idea originally came at the end of the XXI century. But it has become actual and developed since 2012. Traditional TCP only can send traffic to the destination with one way. If a TCP connection gets down it becomes annoying to make the connection again. Routers and switches can solve this issue, but computers or mobile devices can’t be capable to transmit packets with a multipath way. That kind of problems made actual multipath TCP. Later, MPRTP idea became actual. There are many implementations of MPRTP. But they are very mixed with different hardware and software applications. In this dissertation we analyse Multipath RTP video streaming (MPRTP). The main goal of the dissertation is to provide readers with the theory of how to implement easily MPRTP for video streaming. This dissertation gives wide theoretical explanation about how to use a MPRTP for video streaming.

Executive Summary

Multipath RTP for video

Subject matter: This report provides extensive detail about new extension of Transmission Control Protocol, (TCP) “Multipath RTP”, while covering background information and to provide readers theoretically how to implement Multipath RTP to video stream.

Introduction: When mobility of devices was not widespread, everybody was connecting computers to internet using telephone line, such as dial up, ADSL or etc. As mobile devices, laptops and tablets emerged; they brought a couple of technical problems with them while using wireless connection or cell connection to connect to internet. The problems were that, as traditional TCP uses single path, having mobility brought dropped connections, low bandwidth and delay with it. Multipath extension opened a new page in network connection with using more than one path at the same time in order to prevent those limits mentioned above. With multipath extension, a mobile device while moving between WI-FI access points or while moving in and out of access point coverage area, will not have dropped connections or high latency as multipath will use both available connections at the same time.

Methodology: This dissertation aimed theory of how to implement Multipath RTP for video streaming, which is one of those data types that is sensitive to latency, delays and drops. The main methodology that helps multipath extension to work is its smart congestion control, which periodically checks congestion window of each path and intelligently decides which path to send more traffic, while in normal conditions trying to balance data transmission between paths. Because the dissertation is a theoretical aspect of viewing, in order to make it clear for readers, as an example, we will take 2 WI-FI access points as connection path, while in order to make it more sophisticated we can take one access point and one 3G internet connection.

Findings: Dissertation specific research and this dissertation figures out significant benefits of multipath extension. There are more than one and some of them are: as multipath extension uses more than one path, user now benefits from other path as well, which wasn't available in single path transmission. This results in an increased bandwidth, non-dropping connections, high robustness and low latency. Proper evaluation proves the advantageous behavior of multipath extension, which will be an important connectivity function of future devices.

Conclusion: Although multipath extension brings vital advantages with it, it is still under improvement as it comes with side effects and disadvantages depending on network scenarios and behaviors, which is a reason why it is not widespread in today's devices.

Table of Contents

List of Figures

Acknowledgement

I would like to take this opportunity to convey my thanks to the following people or group:

- My Supervisor – For always being positive, supporting the ideas, giving right direction and helping me in every possible way.
- My Family – For always being in my side, supporting and believing to me in hard days.

1. Introduction

We are living in such a fast evolving world that we can meet hundreds of new technologies and more evolved versions of those technologies after a small period of time. Transmission Control Protocol is one of those technologies that has evolved year by year and become the one, which we use today. TCP provides reliable transmission of data within connected networks. Latest version of TCP provides more robust, redundant, and secure transmission of packets. Todays advanced version of TCP is Multipath TCP. The main aim of this dissertation is to adapt Multipath technology with RTP for real-time video streaming. The idea of implementing RTP with multiple paths comes from multipath TCP. Unlike the traditional single way transmission, the multipath TCP uses several paths (several Access Points or other available connection methods) to transmit the data.

Multipath RTP for video can solve several problems that the single path transmission will suffer while having video-conferencing or video streaming. Multipath RTP has more bandwidths concurrently, and it speeds up the transmission while supporting low delay and loss rate. Multipath RTP can automatically divide packets between other APs, as if one of the AP doesn't work, or manage other paths to increase the bit rate, therefore the streaming connection will not be interrupted. Furthermore it can balance the workload for different connection, based on its state of quality and congestion window.

At this point, it is worth to note that in multipath extension all paths have its own congestion window. For example if one path is more congested than other, more packets are directed through less congested connection and less data through poorer connection (more congested link). The main idea when designing the MPRTP is the need of carefully balancing the robustness, congestion and computation overhead and delay. The dissertation topic is MultiPath RTP for video. Main purpose of this dissertation is to show the theory of implementing MultiPath RTP for video streaming in order to get more effective, reliable connection with low loss rate and low delay with having non-dropping connections while being mobile between access points. To give a brief overview, overall design of this dissertation is to send captured video stream from one host to another host while using 2 available Access Points. As a result of this report, and conclusions there is a clear outcome that, Multipath RTP will play a key role in future devices for having reliable, effective and robust connection with low delay and loss rate. The overall aim of this dissertation is to show significance of transmission protocols new extension.

The dissertation consists of 5 parts. First part is an introduction about the new – MPTCP extension. It is about the need of MPTCP and what made it actual for now. Second part is about the definitions and acronyms that used in the dissertation. There are background information and the briefly explanation of MPTCP mechanism. Third part is about MPRTP video streaming example; its main components, what is needed for implementation, about architecture of this streaming and etc. Fourth part is about the system design. What is needed to take into the consideration while implementing this theory into real. Fifth part is about the implementation side, what problems can emerge during the implementation and how can we solve these problems.

2. Definitions and main aspects of dissertation

2.1 Definitions

Before starting with detailed dissertation information, it would be valuable to introduce and mention important aspects, facts and some background information first, which are related to this dissertation “Multi Path RTP for video”.

2.1.1 TCP

TCP - Acronym of Transmission Control Protocol, is one of the core protocols in TCP/IP networks [1]. TCP helps two hosts to establish a connection and exchange data. TCP provides reliable delivery of packets. TCP connection uses single-path delivery of packets in networking. It mostly based on an Open Systems Interconnection model (OSI model) of core networking standard. OSI is a model for designing a network architecture. There are 7 layers of OSI model. Fourth layer of OSI model is a transport layer. TCP is one of the protocols in a transport layer. It is a connection-oriented protocol that provides users with reliable delivery of packets that has been sent. Look at the figure 1:

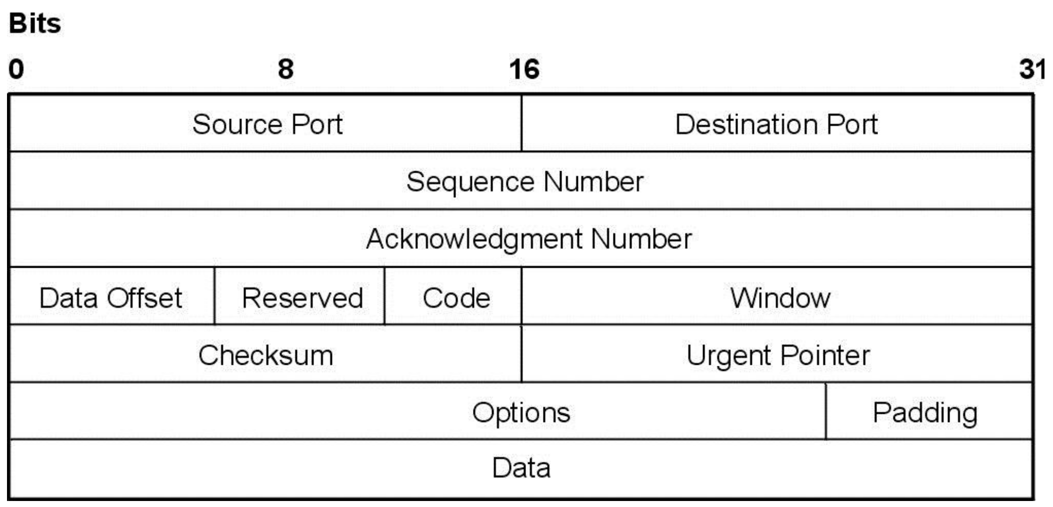


Figure 1. TCP Packet Header

The main parts of TCP header is a sequence number and an acknowledgement number. The Sequence number specifies the sequence number of the first byte of data in this segment. The Acknowledgement number identifies the position of the highest byte received. If data is not received within a given time range, TCP will retransmit the packet. TCP uses a single-path to transmit packets. The need of to do it in a multi-way changes TCP to multipath TCP. It will help networking to become more reliable and faster at the same time [1].

Lets give a brief overview for TCP connection in the following example:

TCP makes a connection by three-way handshake.

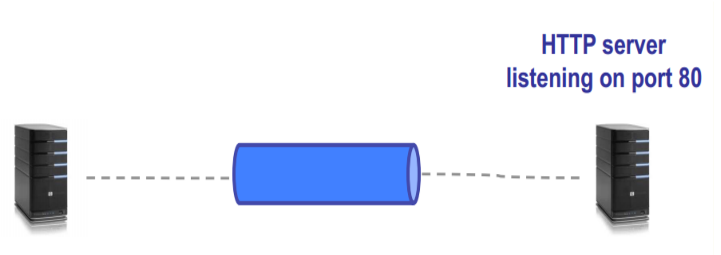


Figure 2. TCP Connection Setup.

First, source port sends a SYN message in order to be sure that a connection can be established;

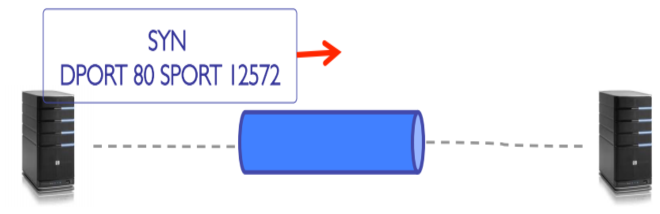


Figure 3. TCP Connection Setup.

Then, the destination port responds to this message with SYN/ACK message;

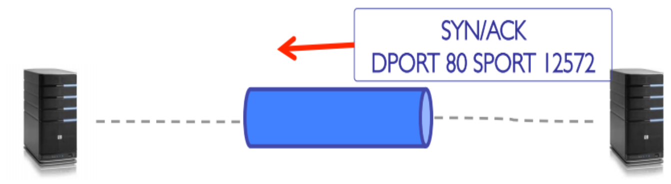


Figure 4. TCP Connection Setup.

Then, the source port sends ACK message again that connection is established, and at last two devices started to make conversation:

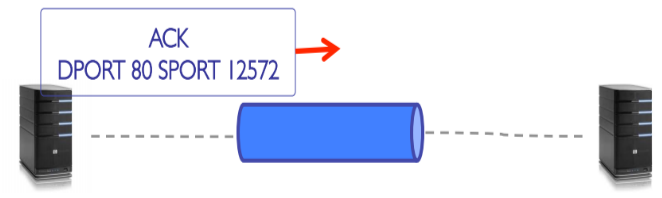


Figure 5. TCP Connection Setup.

That is the exploration how TCP makes a connection.

After the handshake, the client and the server can communicate with each other. The sequence number is used to delineate the data in the different segments, reorder them, and detect losses [1]. The TCP header also contains a number that acknowledges received data by telling the sender which is the next byte expected by the receiver. Various techniques are used by TCP to retransmit the lost segments.

If one of the hosts wants to close the connection, it can send reset (RST) packet, which will terminate the connection, but usually they use FIN packets in order to close the connection. FIN packets should be acknowledged by both directions.

2.1.2 RTP

RTP – Abbreviation of Real-Time Transport Protocol. This is an Internet protocol for real-time data transmission such as videoconference or audio streaming. Unlike TCP, RTP does not guarantee good reliability or very high robustness. Generally, as in our implementation, RTP runs over UDP protocol, while it also supports other transmission protocols. Advantage of RTP is its flexibility and low latency. Originally specified in Internet Engineering Task Force (İETF) Request for Comments (RFC) 1889, RTP was designed by IETF Audio-video working team group participants. It is used mainly in internet telephony applications. RTP does not guarantee itself audio-video streaming, because it is commonly a part of the UDP protocol; it does, however, provide the wherewithal to manage the data as it arrives to best effect. [2]

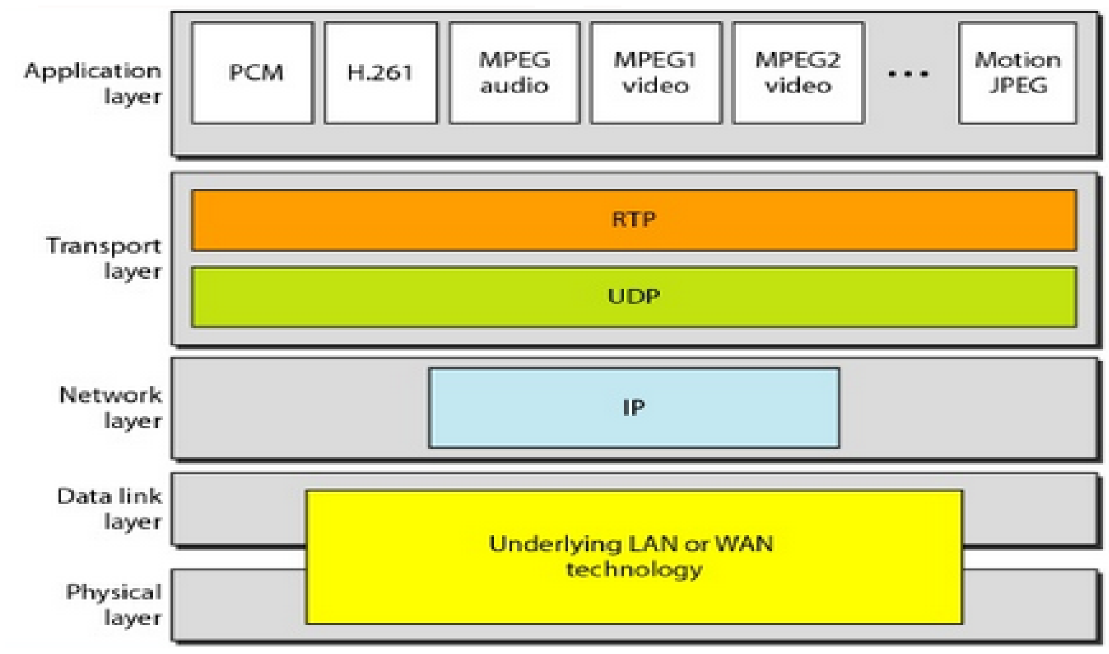


Figure 6. RTP in OSI layer.

RTP gives the responsibility for recovering lost segments and resequencing of the packets for the application layer. There are a couple of benefits in doing so. The application may accept less than perfect delivery and with video or speech there usually is no time for retransmission. Also the sender may provide, instead of retransmission, new or updated data that tries to fix the consequences of the original loss. What RTP then provides, are: 1. Payload type identification 2. Source identification 3. Sequence numbering 4. Timestamping which are required by many multimedia applications. The accompanying RTP Control Protocol (RTCP) provides feedback of the quality of the data delivery and information about session participants. An RTP session usually is composed of an RTP port number (UDP port), an RTCP port number (consecutive UDP port) and the participant's IP address [3].

2.1.3 FEC – Forward Error Correction

• FEC – Stands for Forward error correction, which is a popular method used to improve data reliability. It does this with sending redundant data, which is called an error correction code. With FEC, receiver does not need reverse channel to request retransmission of lost data, as it will have capability of correcting errors using redundant data. FEC adds redundancy to transmitted information using a predetermined algorithm. The redundant bits are complex functions of the original information bits. Bits are sent multiple times, because an error may appear in any of the samples transmitted. FEC codes generally detect the last set of bits to determine the decoding of a small handful of bits. [4]

• Multipath – Technique for using more than one path to the destination for providing more robustness, maximizing resource usage and increasing redundancy. It is the combination of words *multiple* and *path*.

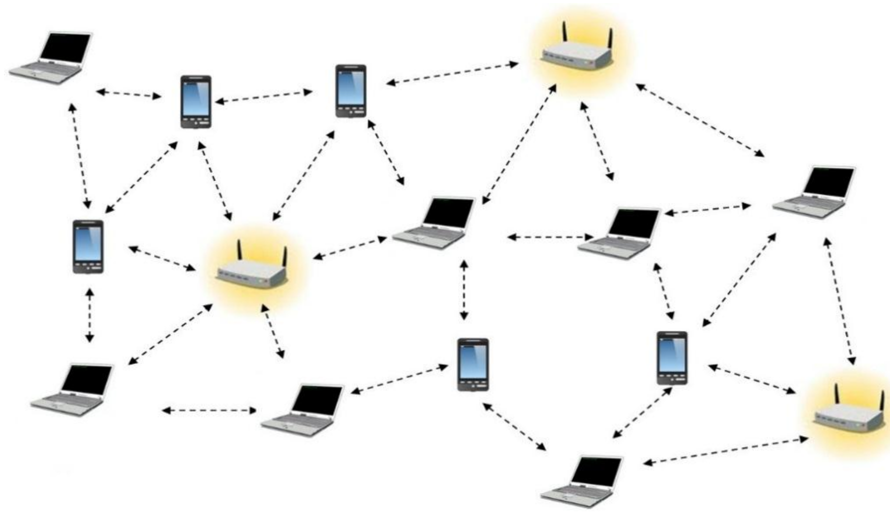


Figure 7. Multipath connections.

• AP – Stands for Access Point, a hardware device that performs as a communication hub for users of a wireless device. It may connect to a router or it may be internal component of a router itself. When we say AP, we mostly mean Wireless Access Points. It mostly used by a lot of people from whole the world. Wireless AP is a WI-FI router, ADSL or DSL modem and etc.

• 3G – Stands for Third Generation wireless technology. It is designed and being developed by International Communication Union (ITU) company since 1980s. With this technology people can get internet wirelessly via mobile telephony communication providers.

2.2 Background information and description of main aspects

When looking back to the first days of networking and Internet everything seemed simple. There were no any far away host concerns, mobility difficulties and other hassles. TCP/IP was working perfectly between two end hosts with providing reliable and robust connection. During those times, TCP was built with the concept of single path connection between two end points. As technology emerged, laptops with mobility characteristic and smartphones came out, this became an issue. The image below describes the issue with traditional single path TCP in a simple way. Given that there is a smartphone with 3G and WI-FI connection technology, it will use only one of these two connection possibilities.

If smartphone is within AP coverage area and WI-FI is turned on, it will automatically select WI-FI as an only connection method. In short, with traditional single path TCP, devices will never select two connection paths at the same time.

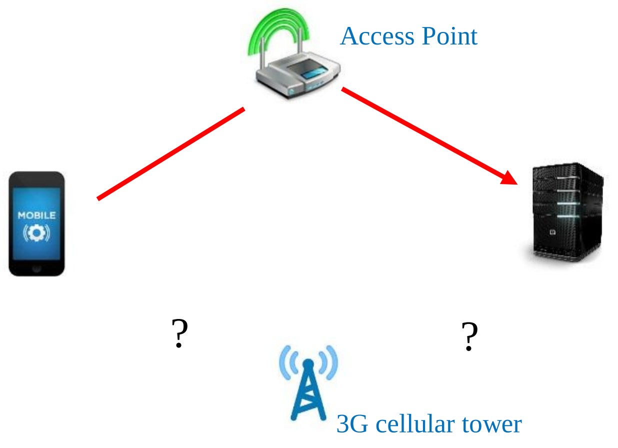


Figure 8. Single path connection .

The main issue is the loss of connection while switching between 2 connection options, no matter if there are 3G and WI-FI, or 2 WI-FI connections nearby. What Multi Path TCP (or RTP, does not matter) tries to solve is exactly this issue. The image below represents same scenario with Multi Path TCP extension supported by smartphone.

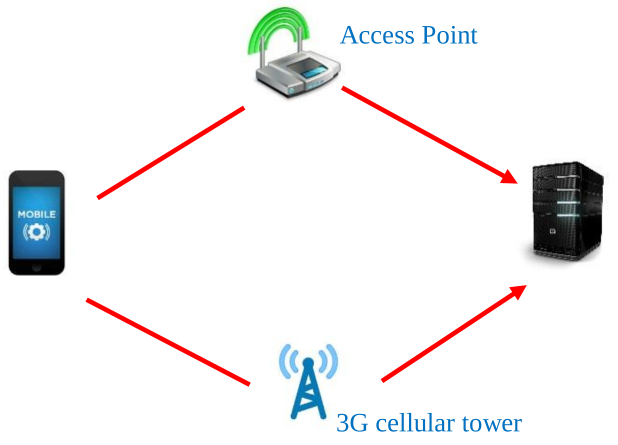


Figure 9. Multi path connection .

Multi Path extension tends to use both of available transport paths to endpoint together, which helps to avoid loss of connection and to provide more robust and effective connection with low loss rate. This is huge performance improvement and benefit for users as both two paths can carry data parallel and cleaver algorithm can decide which path to select depending on the congestion status of particular path [5]. One good example of Multi Path connection effectiveness is that, given that user is having a videoconference at home while using WI-FI connection, although user leaves home, he/she wishes to continue videoconference without any interruption and impact. With Multi Path extension, mobile device begins sending more packets using cellular Internet. Though Multi Path extension tends to solve issues of single path connection, it has some disadvantages and difficulties, which are still the topic of research. Some of the advantages and disadvantages of Multi Path extension are:

Advantages:

- MPTCP increases the bandwidth
- MPTCP delivers effective redundancy
- Overcomes traditional limits of single path TCP.

Of course these advantages brings some side effects with them. Such as:

Disadvantages:

- Deployable protocol design for internet is hard
- Policy considerations and functioning difficulties
- Potential security risks
- Not all networking technologies support new extension
- Servers should also have Multipath stack available within them to provide support

2.2.1 MPTCP

As said earlier, the networks are really becoming multipath. Mobile devices that have multiple wireless interfaces, with different coverage area and different energy consumption. Data centers, they have got many servers with many different access points. Between any 2 servers we have many paths that you can use to communicate. Finally, network providers have multi-home connections that with them they have better redundancy, better performance and so far. They become multipath but we still use single-path TCP connection. And the reason they use TCP connection is that they offers reliable, byte-oriented connection. But the trouble with TCP connection that is they use single-path connection. It is designed to bind the connection to two sources, two IP addresses and when one of the addresses changed the connection goes down. For example, I use 3G connection in the street, I want to switch to the WI-FI when I get home. The trouble is that at that time of transformation the connection goes down. The solution of this problem is multipath TCP. Multipath TCP is an evolution of TCP that allows us to use multiple paths within the single transport connection [6]. Multipath TCP has been first proposed in 1995 by Huitema (p-TCP, m-TCP, …) and then by many scientists. The most common difference in that version of MPTCP is their deployable character. New technology is suitable with recent network devices. MPTCP is tested and implemented in 2012 by a group of scientists. The source code of MPTCP connection can be found in the internet. (The Linux-Kernel version).

Let’s give some examples; Everybody knows that if we are on the street we use 3G connection to access the internet. Nowadays almost all people have a WI-FI connection in their home. Let’s say we came home from the street and we listen to internet radio. In order to use WI-FI at home we have to switch on WI-FI connections, which kills 3G connection automatically. This means we will lose connectivity for a period of time. It is because of TCP. TCP uses single-path transmission to provide connection-oriented (Secure) way of transmission. What if we want to switch to WI-FI and at the same not to lose 3g connection?

Another example:

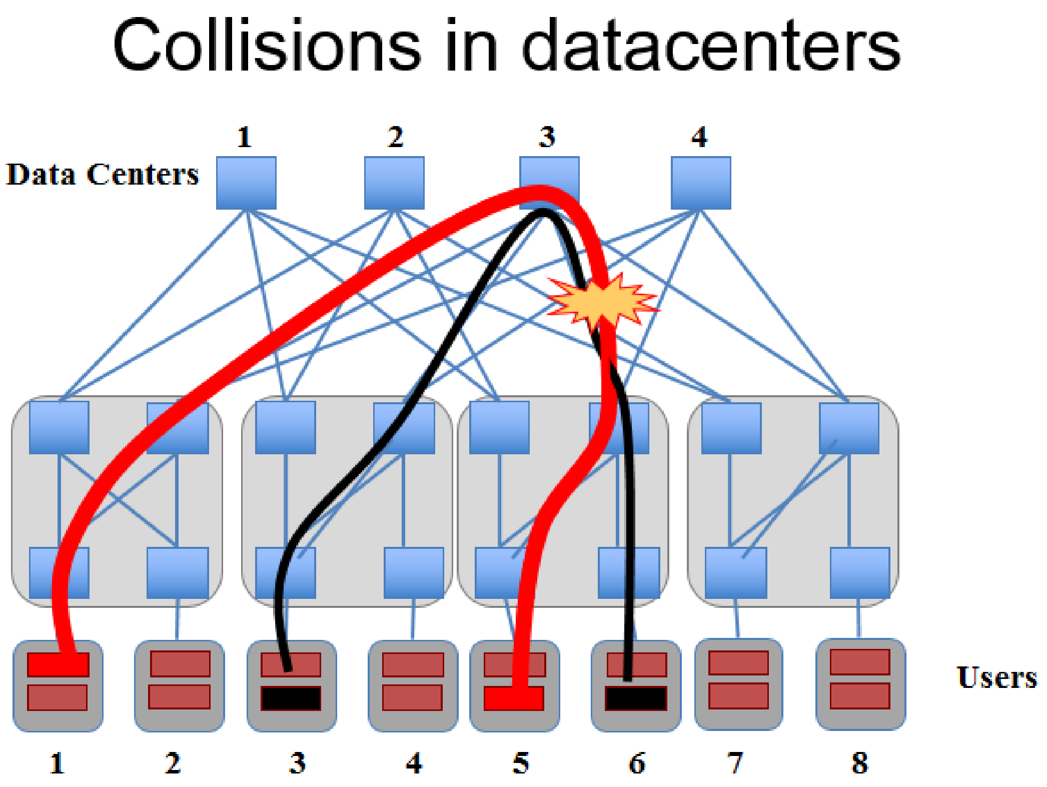


Figure 10. Collisions in Datacenters .

Let’s say user 1 wanted to connect with user 5 through datacenter 3 line randomly. At the same time user 3 wants to connect with user 6 randomly. Practice shows that each host that wants to communicate will choose its path randomly. There will be collision and one of the connection will cause delay in network. It is also because of single-path TCP.

This type of problems led to emerging of new tchnology called the multipath TCP.

2.2.2 MPTCP working technique

To make things clearer, this section briefly describes how Multi Path TCP works. Traditional TCP works with connection establishment using three-way handshake method. Note that, MPTCP is also implemented in Layer 4 (Transport layer). MPTCP connection establishment looks similar to traditional three-way handshake with some additional requirements.

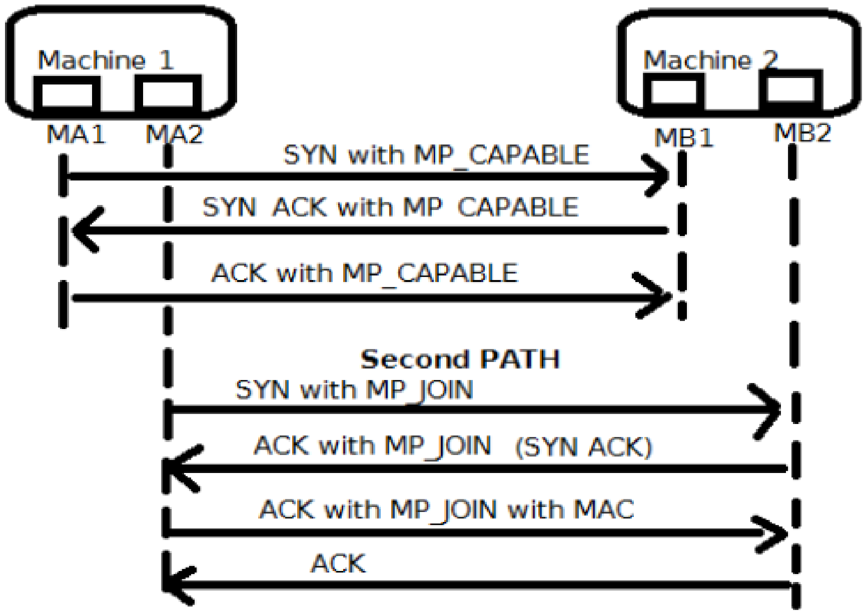


Figure 11. MPTCP connection establishment

Referring to the figure 11, two machines are used to setup Multi Path TCP connection with each other. Each of these two end points has two interfaces.

MA1= interface 1 on machine 1

MA2= Interface 2 on machine 1

MB1= interface 1 on machine 2

MB2= interface 2 on machine 2

Step 1: As a traditional three way handshake mechanism, first step is to send SYN request to destination host. Difference is that, now it has additional MP_CAPABLE flag with it to let destination host know about its desire and ability of supporting new extension.

Step 2: Also in step 2, normal SYN ACK message comes as a reply. If second machine supports MPTCP it will also add additional MP_CAPABLE flag, if not, the message will be traditional SYN ACK message.

Step 3: Finalizing ACK message is sent from machine 1 having MP_CAPABLE flag with it.

As this connection establishment was for one path, this establishment happens again for second interfaces of devices. The difference between first and second establishment will be MP_JOIN flag, which will be used in all steps in order to help for joining the new connection to previously established one. [7]

Look at the figure 12:

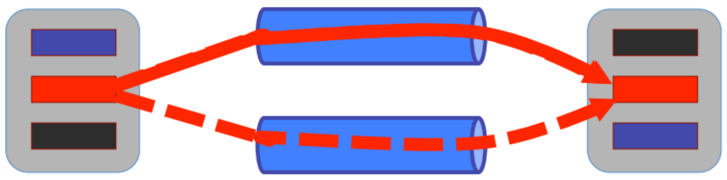


Figure 12. MPTCP Connection Setup.

As demonstrated at the picture, MPTCP uses multipath to establish the connection.

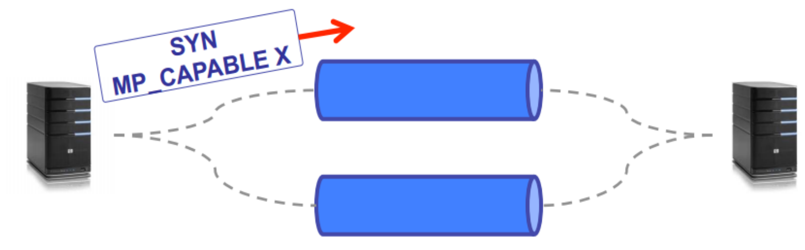


Figure 13. MPTCP Connection Setup.

With the SYN message, source port sends MP_Capable message.

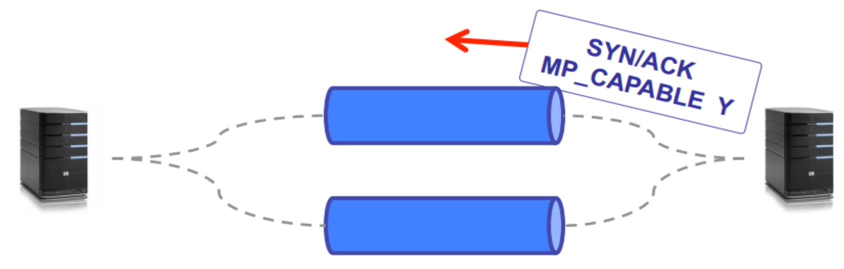


Figure 14. MPTCP Connection Setup.

As the responding, destination port sends ACK with MP_Capable message, too.

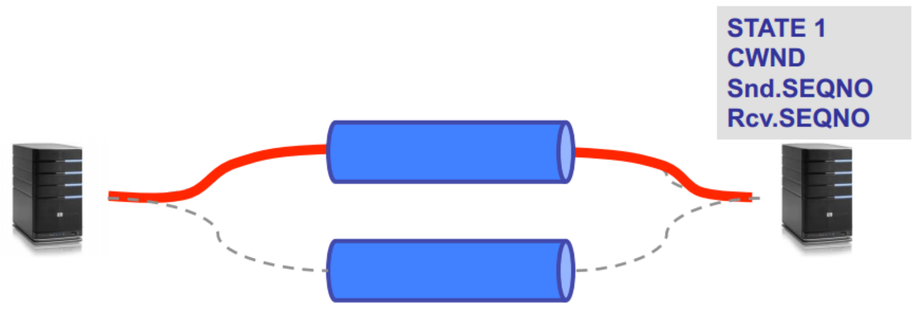


Figure 15. MPTCP Connection Setup.

As the first path is established, the second path will send MP_join message with the SYN message:

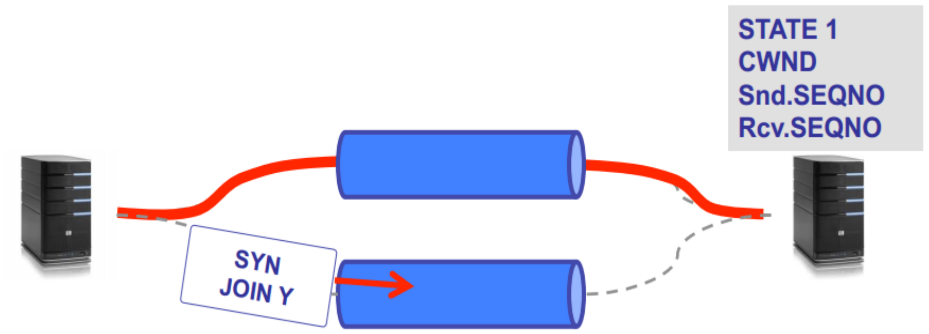


Figure 16. MPTCP Connection Setup.

Finally multipath connection will be established after destination port sends ACK with MP_Join flag:

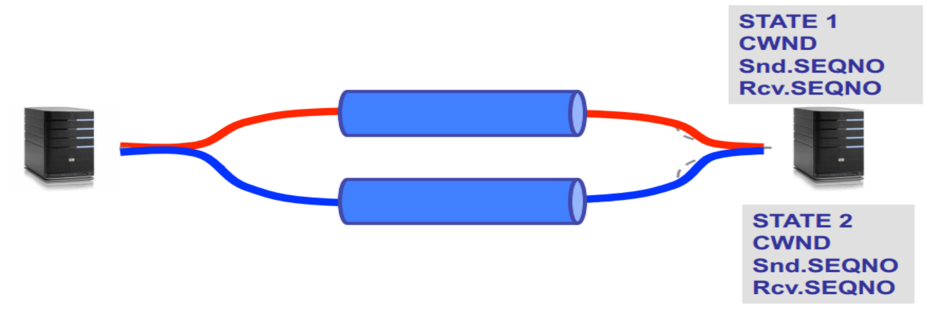


Figure 17. MPTCP Connection Setup.

Technology industry almost started adopting Multi Path extension to their new hardware and software products such as: Last version of Linux kernel as well as Apple operating system iOS 7 supports Multi Path TCP extension.

3. Multi Path RTP

Till now we have talked about Multi Path TCP and its operation. Considering that the dissertation is Multi Path RTP for video, it is worth to briefly explain dissertation. What does dissertation ask is to give a direction of how to implement Multi Path RTP for video streaming. Considering that, TCP tries to provide robustness, low packet loss and reliability, it is not suitable to use TCP protocol with services such as video streaming, video and audio conferences and these kinds of services, which require low delay. For these reasons we are focusing more on RTP version of multipath extension, while deploying RTP over UDP.

MPRTP is an extension to RTP that allows splitting a single RTP stream into multiple subflows that are transmitted over multiple paths. This allows pooling the capacity of multiple internet paths so that media with the higher bit rate can be delivered and system becomes more robust against path variations or disruptions. From the application side of view, the available bandwidth between the two sides increases or becomes more stable. [8]

3.1 Main Components

In order to make the theory real for implementation we will need some hardware devices, too. As an example we take Raspberry Pi and Pi cam module. With them most of technical difficulties and environmental hassles become more and more simpler. Below, we have listed main components of the dissertation.

Hardware:

- a Raspberry Pi single board computer
- a Power supply - a 5V micro USB power supply to power the Raspberry Pi.
- a Raspberry Pi Camera module
- a SD Card - An 8GB class 4 SD card – preinstalled with NOOBS
- a Monitor for connecting Raspberry Pi - Any HDMI/DVI monitor or TV 8
- 2 WI-FI dongles
- a Mouse
- a Keyboard
- 2 computers

Software:

- Raspbian OS
- Text editor for coding
- MPlayer for receiving and showing streamed video

3.2 System Architecture

Figure 18 shows how overall component structure would be with traditional single path TCP. As it is clear, if internet connection gets congested or drops, buffer gets loaded quickly and packets begin dropping. This is not serious problem anymore with adaptation of Multipath extension; as if one path gets congested data is sent through other available path. Next, we will see the overall view of the dissertation in a visual description.

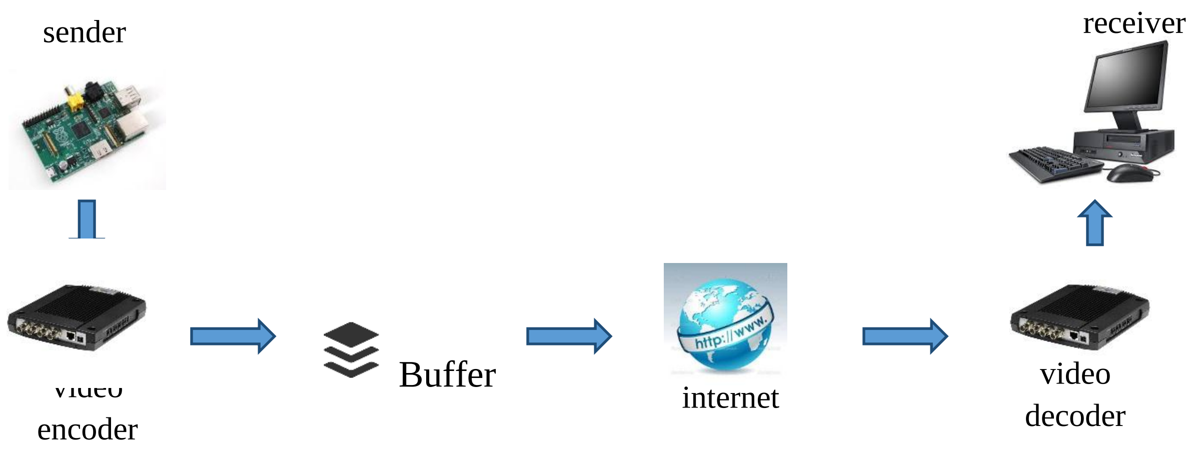


Figure 18. Overall component architecture with single path TCP .

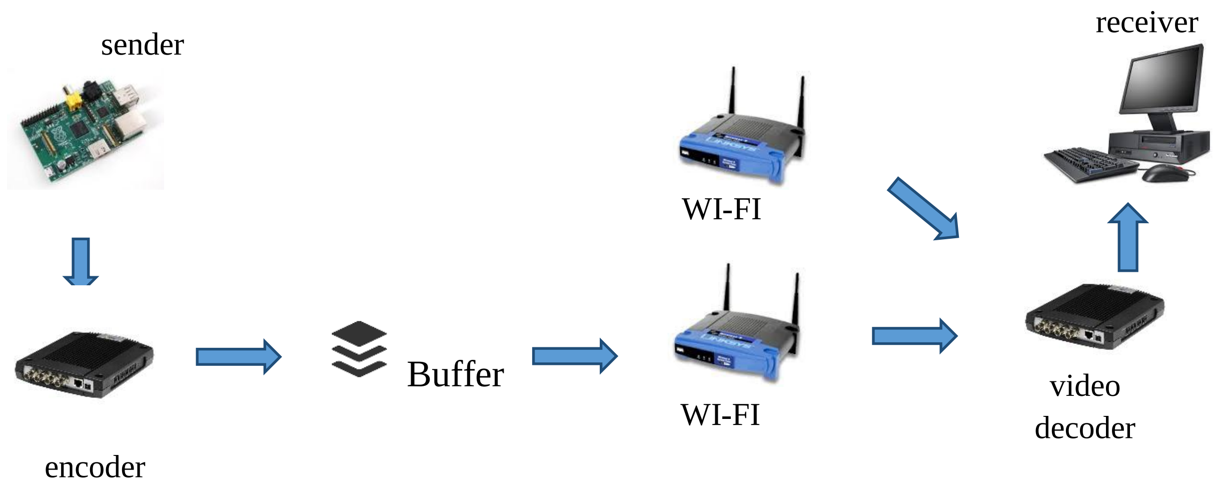


Figure 19. Overall component architecture for Multi Path RTP dissertation .

At this point, it is worth to note development phases of the dissertation. Later we will mention that how dissertation is divided into working plans. Dividing work plan into phases and having working result at the end of each phase perfectly fits incremental software development methodology, which is shortly mentioned later.

Phase 1 - Setup the environment, install operating system, setup network connection and Pi cam, send H.264 stream over a TCP connection and display it with Mplayer remotely.

Phase 2 - Implement video streaming using RTP protocol over UDP connection for low latency, and then implement FEC and Congestion control.

Phase 3 - Design and implement Multi Path version of Phase 2 (Using 2 WI-FI dongles)

3.3 Goals

There will be some essential goals while implementing multipath extension. As some of them were also indicated in multipath readings and papers, those goals are:

Goal 1. Each path should have its own congestion control window

Goal 2. Depending on the faith of sent packet, congestion control should adjust itself. (Increasing or decreasing the window size)

Goal 3. Depending on the congestion control window size of given path, Multi Path extension should send data via its least-congested paths.

Goal 4. Given that Multi Path extension is implemented, it should have as much as bottleneck link capacity.

Goal 5. Environment should adapt quickly when there is a change of congestion in one of available paths.

Goal 6. Improvement of parameters such as; low packet loss, low delay, high robustness, efficiency and etc.

4. Specification and design

4.1 System Design

4.1.1 Multipath Transmission

This part will explain how the system applies two multipath transmissions. In the dissertation, two WI-FI dongles have been plagued into Raspberry Pi to make the transmission with multipath. Because WI-FI dongles have to be controlled in the coding in order to send the data packets respectively, two sockets have been created and each of them binds to one different APs.

In this dissertation, we have to modify the source code of Raspberry Pi, which was open source and based on C [9]. In C there are two ways to bind one socket to one AP. The first one is to use bind () function in C to make one socket to bind one specific IP, and another one is So_bindtodevice () function, which can directly bind the socket to the specific device. So_bindtodevice () function belongs to Asio library.

We have to test both functions for the multipath transmission. First, letting them send the packets respectively and without interrupting them, then both of the functions work well. Both of them can make each socket send the data that distributed to that socket. For example, the socket1 will never ever send the packets that socket 2 should send.

However, to test which one would be more suitable for multipath transmission, during the transmission, one of the APs get disconnected, and let the author of this dissertation make it become active and connected again. We have to test it because one or both WI-FI may get disconnected from the AP, and later it can be connected to the AP again. To control the two APs, it is essential to make sure after the reconnection the socket is still bind to the AP that it is used to.

Because in the test, the senders APs does not have the static IP, as using bind () function in c to bind one socket to one AP by IP address, it will not work after the reconnection, simply because the router may give the WI-FI an internal IP address, which will be different from the previous one. Consequently the socket that is supposed to bind to the AP cannot continue to bind to the AP, after the reconnection [10].

In the case that the socket is directly bind to the device; the function in c will bind the socket and device together by using the name of device e.g, “WLAN01”. When one WI-FI dongle gets disconnected from the AP, and reconnected to the same AP again or to another AP, the internal IP address may be changed again. However, the name of the device will not change, therefore, even in the case of WI-FI disconnection, the socket is still able to keep the link between the socket and WI-FI dongle.

Based on the testing of the both functions, it is worth to bind the socket to device directly, without using binding to the IP address assuming that decision will be more suitable to multipath transmission.

4.1.2 Multi Path RTP Layer

Multi Path RTP (MPRTP) should be an extension for RTP protocol and MPRTP should mainly play another two roles, which is not included and supported in RTP:

- Scheduling the payload for each path

• Maintaining the multiple paths.

Regarding these two roles and implementation aspect, MPRTP should be deployed on the application layer. This makes MPRTP an easier and flexible solution for real time video streaming [11]. (No change in kernel level implementations or network level infrastructures)

Application			
MPRTP			
RTP	RTP	…	RTP
UDP	UDP	…	UDP
IP	IP	…	IP
Physical	Physical	…	Physical

Table 1. MPRTP Layer

The MPRTP should be responsible to manage the available paths, such as new available paths or existing paths failing. Keeping recording of each path's characteristics (loss rate) and using it for scheduling the packets payload among these paths is one of MPRTP's main jobs. Though multiple paths will be involved in the communicating between sender and receiver, both of them will treat the communicating as an End-to-End session. The MPRTP layer is responsible for managing the available paths. It is aware of new paths becoming available or old ones failing, and maintains path characteristics gathered through MPRTCP reports.

It also initiates and records the results for connectivity checks to see the health of a particular path between peers. MPRTP senders split a single RTP stream over the multiple paths available, a process referred to as packet scheduling. The packets on any particular path constitute a subflow. MPRTP provides unique identifiers and packet sequencing for each subflow [12]. The MPRTP layer acts common to all the subflows and ensures that the MPRTP session appears as a single RTP session to legacy applications. In receivers, the MPRTP layer recombines and reorders the packets coming from the multiple subflows to form a single stream for the application again. An example case with three subflows is illustrated in Figure 21.

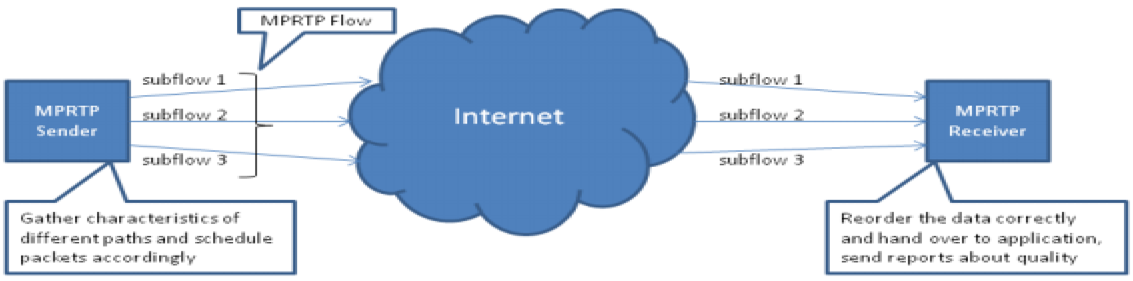

MPRTP Flow
MPRTP Sender
subflow 1
subflow 2
subflow 3
Internet
subflow 1
subflow 2
subflow 3
MPRTP Receiver
Gather characteristics of different paths and schedule packets accordingly
Reorder the data correctly and hand over to application, send reports about quality

Figure 20. MPRTP Connection establishment.

4.1.3 MPRTP Message Formats

MPRTP uses RTP/RTCP header extensions for sending MPRTP-specific information to peers. As already discussed, these extensions are simply ignored by peers that do not have MPRTP capabilities. In this section, we describe the basic header extensions used by MPRTP during a session. Details of connectivity checks are not included in the scope of the thesis.

4.1.4 MPRTP Subflow Header

The MPRTP RTP extension header can be a subflow header or a connectivity check. If required, other message types can also be included in the protocol specification. A subflow header is shown highlighted in Figure 22. The fields are explained as follows:

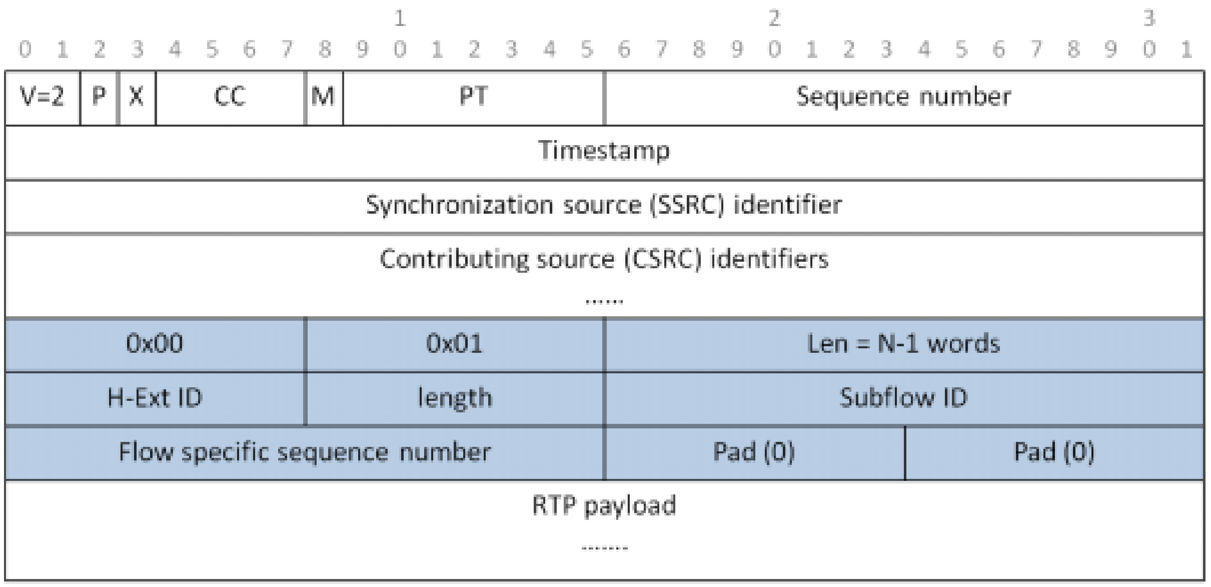


Figure 21. MPRTP Subflow Header

- H-Ext ID : Indicates the type of MPRTP message and has the value 0x00 in the case of subflow header. If this field has the value 0x01, it means the message is a connectivity check.
- Length : Indicates the number of bytes in the extension header excluding H-Ext ID and the length field itself. It has value 0x06 in case of subflow headers.
- Subflow ID : Each subflow has a unique identifier which is carried in this field.
- Flow specific sequence number : A strictly monotonically increasing sequence number assigned to each packet in the subflow.

4.1.5 MPRTCP Sender and Receiver Reports

The reports sent on a particular path will only contain subflow-specific information and hence are referred to as Subflow-specific SR (SSR), Subflow-specific RR (SRR). This ensures that the sender and receiver have information about the health and performance of each path and not just an overall value. A different approach would be to concatenate reports of various paths into a single packet and sending on all or any one single path. Concatenating reports of various subflows into a single packet would lead to larger packets, which if lost, would result in greater information loss in comparison to smaller packets of flow-specific reports. Furthermore, for RTT measurements, we have to send reports on all available paths [13]. Since a subflow's report is only relevant as long as the path is active, it is acceptable to only send it along the same path rather than on any other path. Subflow-specific extension reports, if any, are appended to the SRR. The fields within SRR and SSR are the same as RTCP RR and SR respectively, making them backward compatible as well as easy to implement. The message format of a MPRTCP report is shown in Figure 22.

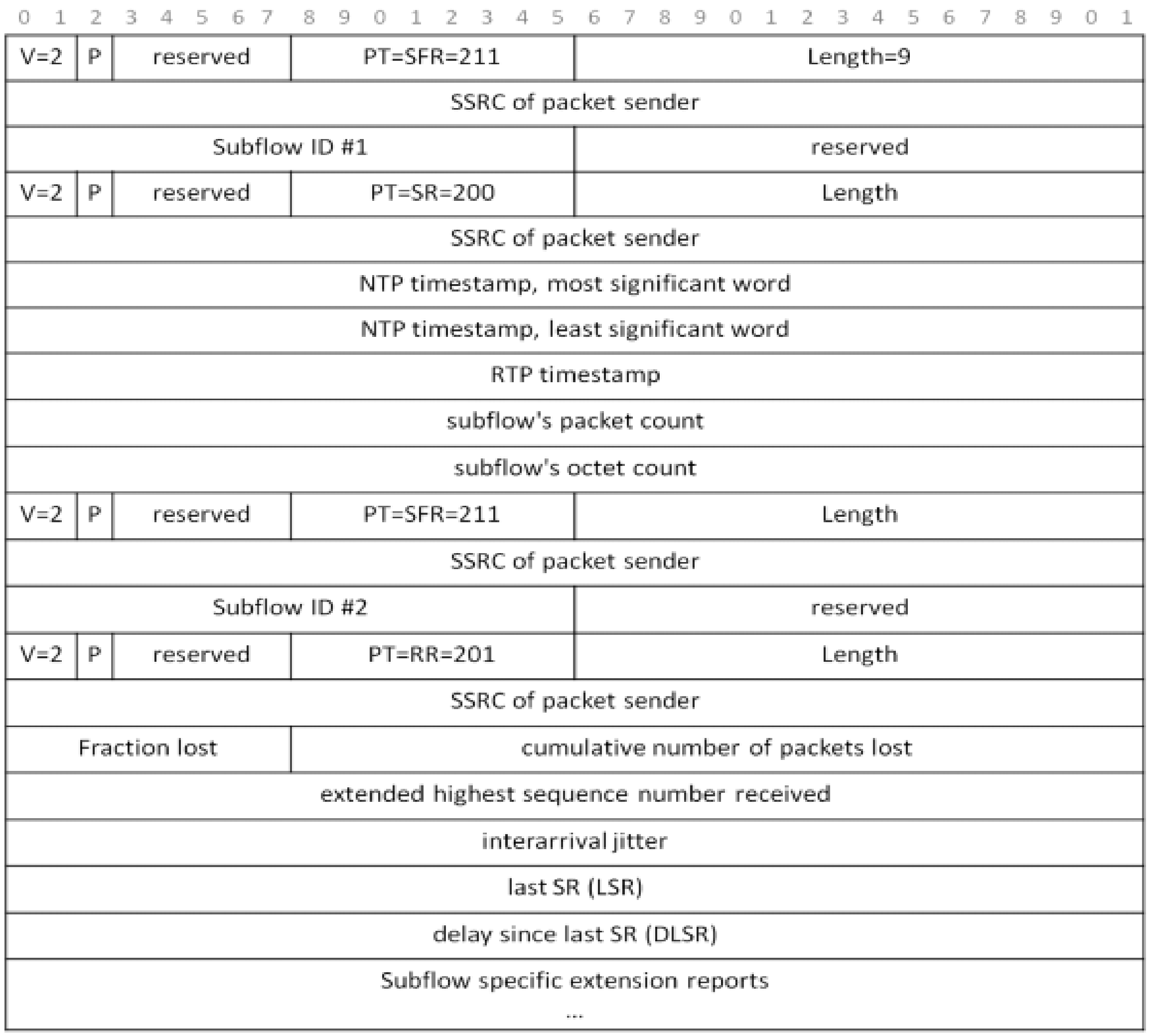


Figure 22. MPRTCP sender and receiver ports

MPRTP is a protocol designed to introduce multipath capability to real time communication with the aim to improve quality of service. It aims on achieving higher throughput by resource pooling if multiple paths are available between two endpoints. It should also provide a higher level of resilience and lower packet losses in comparison to RTP under similar conditions. MPRTP is designed to be network compatible as well as backward compatible with RTP applications.

An MPRTP sender is capable of splitting a single RTP stream over the available paths, while an MPRTP receiver reorders and recombines it before handing it over to the application. Packets travelling over different paths are more prone to out-of-order delivery and higher values of jitter than those travelling over a single path, and hence MPRTP receivers may require higher buffering delays for smooth playout than RTP receivers. Possible use cases of MPRTP include high bitrate streaming scenarios and voice/video calls. MPRTP can provide higher throughput for the former case and redundancy through fallback for the latter.

4.1.6 H.264 codec

There are two important aspects need to be considered regarding the video size and the transmission over the network. Because the raw video file/streaming size is really huge, it’s not acceptable especially for live video streaming [14]. In order to achieve transmitting efficiency, video compression is used to compromise with video quality and hardware resources. In order to provide performing a better video display on the receiver side, it is important to enhance the compression rates while ensuring the smallest possible loss of video quality.

There are many video compression standards: MPEG-1, MPEG-2, MPEG-4, H.264 (also known as MPEG-4 part 10) and so on. MPEG-2 is the current standard for High Definition Television transmission. But with the significant bandwidth saving and the flexibility of selecting from a number of reference frames for motion estimation for a given predicted frame [15], H.264 is supposed to replace the use of MPEG-2 video compression. Furthermore, H.264 can produce a perceptually equivalent quality video at about half the bit rate comparing to MPEG-2 [16]. Therefore, this dissertation will choose H.264 as the video compression/decompression standard.

4.1.7 Forward Error Correction

Packet losses due to transmitting errors and/or overdue packet delivery caused by congestion and/or the disorder of packets arriving should be kept low. In a wireless network connection scenario, it’s more frequently that wireless links are disconnected and new AP will be established because of the mobility. Furthermore, a wireless link usually suffers more from high transmission error rate due to shadowing, fading, path loss and interference with other nearby transmitting AP. (even it’s possible from its own different WI-FI dongles). Two traditional error control techniques can be adapted: Forward Error Correction (FEC) [17] and Automatic Repeat reQuest (ARQ) [18] . Since human senses can tolerate a certain degree of losses in video, a glitch (losing a single packet) is totally acceptable to the human eyes. Considering that, retransmission of the lost packets would cause a lot of unacceptable delay like what happens in MPTCP, it is perfectly suitable for this dissertation to adapt FEC rather than ARQ to deal with packet loss problem.

Although there already exists an RTP Payload Format for Generic Forward Error Correction [19], the transmission consists of either two separate streams: media stream and FEC stream or FEC stream only. In order to achieve congestion control in this dissertation, it’s better to transmitting packets mixed with media and FEC packets in one stream. Therefore, we will design our own RTP-FEC rules to achieve the goals.

4.1.8 Design Overview

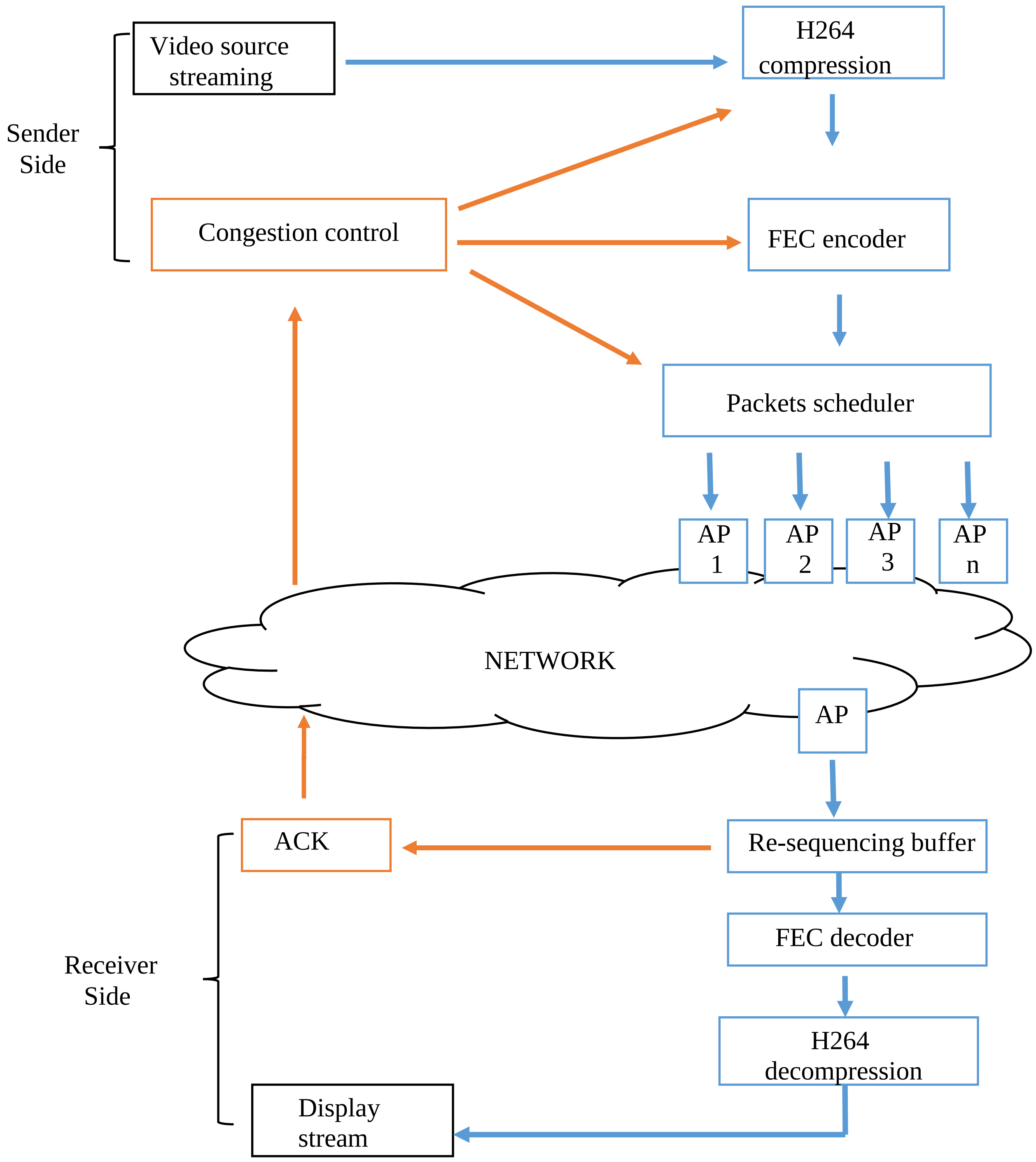


Figure 23. End to End communicating flow.

The figure 23 shows the overall flow-streaming diagram between sender side and receiver side. The detailed information can be explained as following:

Sender side:

- The raw video streaming will be encoded by H.264 compression first.

- Dividing the NAL (Network Abstraction Layer) units into payloads (based on the maximum transmission unit) and setting up the standard RTP headers for them to generate standard RTP packets.

- FEC (Forward Error Correction) encoder will take in the standard RTP packets then return original RTP packets plus several corresponding redundant FEC packets (for recovery purpose). Set up MPRTP "header" for these packets.

- Packets scheduler will dispatch these packets depending on the each path's loss rate. (Lowest loss rate should transmit more packets, even dispatching by default)

- Congestion control will adjust the compressing bitrate of H.264 compression, numbers of redundant FEC packets of FEC encoder that will be sent and packets scheduler's dispatching.

Receiver side:

- Re-sequencing buffer will cache the packets received. A sorting algorithm will be periodically invoked to re-sequence the packets based on the MPRTP's "header". If a group of packets satisfies FEC decoder's requirements, they will be removed the MPRTP's "header" and be popped out to FEC decoder.

- FEC decoder will take these packets and decode them back to original RTP packets.

- Since most popular video players (e.g. MPlayer, VLC player) support RTP payload format for H.264 video, there is no extra work for H.264 decompression.

- ACK will periodically send an acknowledgment, which contains information such as smallest/largest MPRTP sequence number and numbers of received packets corresponding to each sender side's path, to the sender side. So that the congestion control in the sender side can do corresponding actions in response to ACK.

4.2 Packet Acknowledgement

Multiple path transmission is different from the single path. In the traditional single path, the receiver will always know how many packets the sender sends, since there is only one access point. The receiver can easily know which packets have been lost during the transmission based on the sequence number. However, when it comes to the multipath it becomes different and complex, where in the case of multipath transmission, there is more than one sender, based on the traditional sequence number; the receiver will not be able to recognize the lost packet. A new ACK system therefore has been designed to fit the multipath transmission.

Since two access points have been used, the sender must obtain feedback (ACKs) for each of those two access points. The receiver does not know how many packets totally sent by each of the APs [20]. Based on the condition of the network, two APs will not permanently send equivalent amount of payload. With the case of having one AP with extremely good condition and another AP with bad Internet connection, two APs cannot be used to carry same amount of data. Most of packets will transmit through the

AP with good connection, and only a small amount of packets will be sent over the bad connection. As a result, it is impossible for the receiver to know how many packets arrived from each of the APs.

4.2.1 Receiver side

To solve the problem as I mentioned at the end of chapter 4.2, the receiver will not count how many packets are lost; instead the sender will handle this task. In the receiver side; the receiver records the total number of packets that have been obtained by the receiver, and it records the smallest/starting sequence number, and the biggest sequence number. To send the feedback of the network condition to the sender, the ACKs contain starting sequence number and the ending sequence number and how many packets have been received, because it is multipath transmission, the ACKs mention which AP that packets come from.

For example in the case of two APs, which means two senders exist, the receiver total obtained the packets with sequence number 0-9 (10 packets), and 7 packets (1,2,3,4,6,8,9) are from AP1, other 3 packets (0,5,7) from AP2. In the ACK for AP1 the starting sequence number is 1, and the biggest sequence number is 9, and number of received packets is 9. In the ACK to AP2 the starting number is 0, and 7. The ACK are sent back to the sender every 20 seconds.

4.2.2 Sender side

As it has been mentioned above, the receiver does not calculate the lost packets. The work for figuring out the loss rate is done on the sender side. In the sender side, it keeps one integer array for each of APs. The array records the sequence number of packets sent from that AP. For example, when the 10 packets totally are sent from two APs, packets 1,2,3,4,6,8,9 from AP1, the integer array will have 7 elements, which are 1,2,3,4,6,8,9, respectively. Consequently the array for AP2 has 3 elements, and the three are 0, 5, and 7.

Because the array can clearly record how many packets are sent from the AP, the sender can know the condition of the network based the information from the ACKs.

For example, assuming among the 7 packets sent from AP1 there are two packets (3, 8) lost, then the ACK to AP1 should be:

Starting sequence number: 1

Ending sequence number: 9

Number of received packets: 5.

After successfully receiving the ACK by the sender, the sender will scan the array to find out the starting sequence number which is 1 and then it will attempt to find the ending sequence number 9, while looking for the ending sequence number, sender counts how many packets are sent between sequence number 1 and sequence number 9. Then by comparing actual number of sent packets, the sender can know accurate loss rate for AP1.

To ensure the calculation will not be wrong, after finishing calculation on loss rate, the sender clean the used sequence number from the array. When the AP1 receive the ACK for packets 1-3, the sender will clean the first three elements from the array, and make the forth element become the first (int [0]), and second come to the second (int [1]) and so on. It undertakes this because it can make sure the calculation will not be wrong even in the case the packets with starting sequence number or ending sequence number is lost.

When packets 1,2,3 are lost, the receiver’s ACK back to AP1 will be:

S sequence number: 4

E sequence number: 9

Number of received packets: 4

Although there are no lost packets from 4 to 9, the sender can still know the packets from 1-3 are lost. When the sender is attempting to find the sequence number from the array, it will scan the array from the first element, and the count will include those sequence number before the smallest sequence number in the ACK. In this case, when the sender find out there are some elements in the array before the starting sequence number 4, then the sender knows those packets are lost.

4.3 Congestion Control and ACK

Congestion control in the sender side will react based on the information respond by ACK message from receiver side. Receiver side sends ACK message periodically. One possible ACK message format is RTP Control Protocol [21].

- The congestion control will start from sending maximum redundant FEC packets (as the same amount of RTP packets).

- If there is no packet loss or very few packet loss (minimum loss rate threshold), congestion control will increase the bitrate of the H.264 encoder and keep sending the maximum redundant FEC packets.

- If there is lots of packet loss (maximum loss rate threshold), congestion control will decrease the number of redundant FEC packets to reduce packet loss rate.

- If decreasing the number of redundant FEC packets can't reduce packet loss rate, congestion control has to decrease the bitrate of the H.264 encoder and check if the loss rate can be improved (to achieve minimum loss rate threshold).

If the streaming is stabilized, the congestion control still need to periodically increase the redundant FEC packets to see if it's possible to increase the bitrate of the H.264 encoder.

5. Implementation

5.1 Video Stream Processing

Since our dissertation will be deployed in the Raspberry Pi Operating System, all setting up and configuration related issues should be implemented in Raspberry Pi OS. There already exists an originally embedded program called “raspivid” for capturing video with the camera module. In order to both achieving goals and saving coding time, modifying the source code of “raspivid” program and to leverage it to continue on other coding parts (e.g. MPRTP, FEC) would be very nice. The C source code is the best choise for implementing to manipulate at a lower level and get more controls. The C source code of this program largely depends on a library called: MMAL (Multi-Media Abstraction Layer API) [22]. MMAL is a framework, which is used to provide a host-side, simple and relatively low-level interface to multimedia components running on VideoCore. It also provides a component interface so that new components can be easily created and integrated into the framework. The MMAL API is based on the concept of components, ports and buffer headers. Clients create MMAL components, which expose ports for each individual elementary stream of data they support (e.g. audio/video). Components expose input ports to receive data from the client, and expose output ports to return data to the client. Data sent to or received from the component needs to be attached to a buffer header. Buffer headers are necessary because they contain buffer specific ancillary data, which is necessary for the component and client to do their processing (e.g. timestamps). MMAL lets clients create multi-media components (video encoders, video decoders, camera, and so on) using a common API [23]. Clients exchange data with components using buffer headers. A buffer header has a pointer to the payload data and optional metadata. Buffer headers are sent to and received from ports that are provided by components.

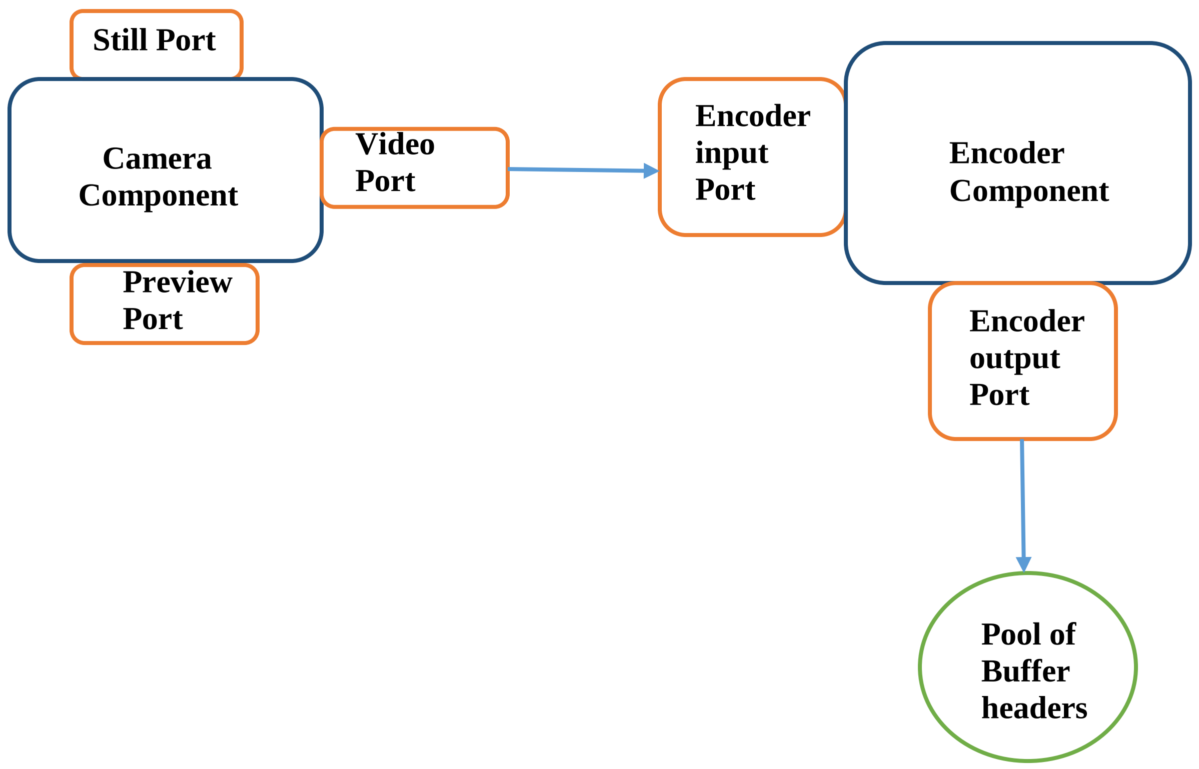


Figure 24. Video Stream Processing.

First attempt is to change the bitrate of the video encoding on the fly smoothly without any glitch. This seems not possible to achieve. The reason is that after encoder component's creation and related parameters commitment (e.g. bitrate), you can't change the commit parameters during its running. The only way to change the parameters afterwards is to disable the ports, destroy the connections between components' ports and re-create the encoder component and set up the ports and connections again. So a glitch might be inevitable but luckily the glitch time is very short, no longer than a wink time.

```
mmal_port_disable(ports);

mmal_connection_destroy(encoder_connection);

mmal_component_disable(encoder_component);

mmal_component_destroy(encoder_component);

…
```

5.2 MPRTP implementing

After getting the raw H.264 byte stream, next step is to encapsulate the Network Abstraction Layer (NAL) unit into the RTP packet. Contributing Sources (CSRC) was not included, which does not seem to be necessary. Three important information need to be mentioned are:

1. Sequence number is a monotonically increasing value assigned to the RTP packets for the purpose of reordering data and also to detect losses.

2. Timestamp indicates the “sampling instant of the first octet in the RTP data packet”. It is used by the receiver to play back the received voice or video. In this case, it will be increasing 3000 for each packet (Sample rate 90000 Hz divided by 30 f/s video).

3. Synchronization Source (SSRC) is a randomly chosen identifier for the source of the stream. SSRC identifiers must be unique within a single RTP session. One example of first 32 bits can be as following:

```
   1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 1 0 0 1 1 0 1 1 1 0 0 1 0

 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Payload                            |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 25. RTP Header.

MPRTP “header” actually is not in front of the packet; it’s attached in the tail of each packet. It is sort of “hacking” way to bypass the difficulty of distinguishing the differences of original RTP packet or FEC packet.

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           IP Header                           |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           UDP Header                          |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           RTP Packet                          |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| payload length |  sequence number with FEC  |     other       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 26. RTP packet.

```
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                  IP Header                  |
 |                    ....                  |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                  UDP Header                  |
 |                    ....                  |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                   FEC data                  |
 |                    ....                  |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |  FEC data  |  sequence number with FEC  |   other    |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 27. FEC packet.

1. Payload length (1 byte) indicates the RTP packet's payload length in bytes. Some packets especially the ending packets payload is shorter, in order to keep all packets' payload in the same length, these packets' payload will padding with 0.

2. Sequence number with FEC (2 bytes) is a monotonically increasing value assigned to the MPRTP packets for the purpose of reordering data, distinguishing original RTP packet or FEC Packet (e.g. least significant decimal number from 0 to 4 inclusive is RTP packet, least significant decimal number from 5 to 9 inclusive is FEC packet) and also to detect losses.

3. Other (1 byte) is left for other extension or testing purpose (e.g. in order to distinguishing the packet coming from which Path, it is necessary to assign different path number in this Other field).

IP header costs 20 bytes. UDP header costs 8 bytes. RTP header costs 12 bytes. Since nearly all IP over Ethernet implementations use the Ethernet V2 frame format, the Maximum Transmission Unit (MTU) is 1500 bytes [24]. It's better to fully use the MTU to maximize the payloads and minimize the cost for the headers (e.g. If RTP payload is 40 bytes, assuming there is no loss, 50% throughput will be wasted for the headers). We can fix the packet structure. Lets say, 12 bytes for RTP header, 1456 bytes for payload and 4 bytes for MPRTP header. For a FEC packet: 1469 bytes for FEC packet, 3 bytes for MPRTP header.

5.3 FEC implementing

The central idea of the FEC is the sender encodes the messages in a redundant way by using an error-correcting code so that the receiver can decode the messages with the redundancy in case some packets lost.

The simplest FEC can be as following:

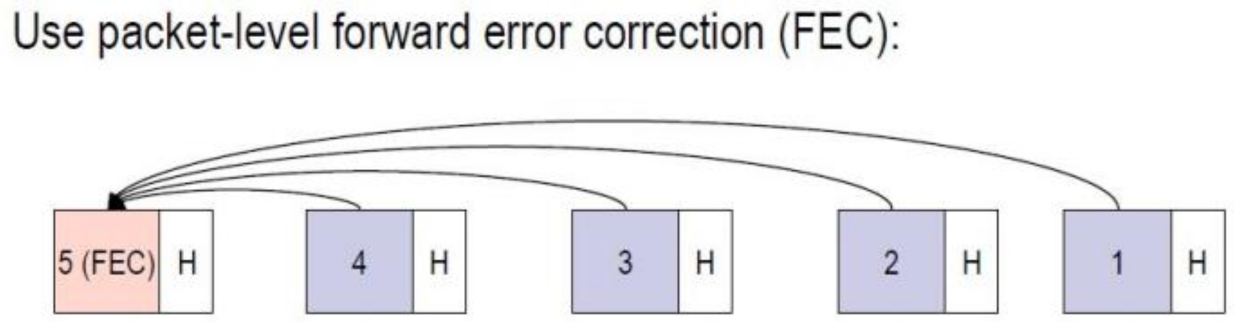


Figure 28. FEC.

Packet 5 is the XOR of packets 1 to 4. If packet 2 is lost, the original message can be reconstructed from 1 XOR 3 XOR 4 XOR 5. But this FEC approach can’t help to dissertation, which can only deal with one packet loss situation. We need a more powerful FEC, which can handle multiple packets loss situation.

Vandermonde matrices based FEC is a good idea [25], which contains an implementation of an encoder/decoder for an erasure code based on Vandermonde matrices computed over GF (2^m), m=2..16, in our case, m=3.

The encoded data is computed as:

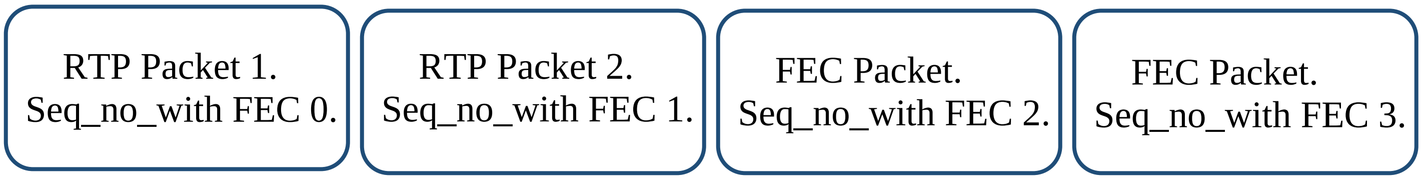
$y = Ex$

where x is a k-vector/array with source data, y is an n-vector/array with the redundant info, and E is an n*k matrix derived from a Vandermonde matrix. The code is systematic.

Assuming error-correcting code new_code (2, 4), two original RTP packets:

RTP Packet 1
RTP Packet 2

After fec_encode ():



For the fec_decode(), as long as any 2 out of 4 packets arrive, the original two RTP Packet 1 and 2 can be recovered. Vandermonde matrices based FEC is a very powerful and robust FEC approach.

Conclusion

This dissertation introduced wide information about new extension of traditional single path TCP protocol, which has main characteristic of using more than one path at the same time. Dissertation continued giving background information about both Multi Path TCP and RTP, why new extension emerged, how they are related to our dissertation and etc., following main specifications, design aspects and so on. Main part of this dissertation was the theory of realistic implementation of Multi Path RTP extension in order to signify its effectiveness. As this report gave wide information regarding development works and all sub-tasks for the implementation, a main purpose was to demonstrate significant advantage of Multi Path and prove how effective is it with achieving low loss, low delay, non-dropping connections and more robustness, which are very important indicators for today's network infrastructure. As we know, in wireless networks there are many high error rates, and that is the reason why people are not interested in mobile video communication. Multi-Path retransmission considerations gives a huge advantage to solve this problem. We gave an example of how to use multipath RTP with the helping of Raspberry PI modul single-board computer and its video module. We gave clarity to the problems that can be while transmitting the video in a multipath way. Specification and Design section verifies the importance of smart design decisions about acknowledgement methods, FEC and etc., realistic implementation can demonstrate how those modifications into traditional transmission architecture can get a result of having advanced and smart connectivity solutions.

Future work

Considering all the beneficial aspects of this new extension, it is believed that it will be an important part of all hardware and software products in a near future. Especially, from mobility aspects, it will be strongly appealing feature as it helps mobile devices use their own built-in multiple available network connections in order to have non-dropping connections with wide available bandwidth.

Another alluring feature of Multi Path extension that engineers are constantly working on to improve is that it is possible to use this extension in order to provide strong load-balancing within network infrastructure.

Though it comes with additional cost and some disadvantages, it is believed that Multi Path TCP research group and all the developers involved in it will achieve more and more advanced version of extension, which will gradually have intelligent congestion control and security measurements. One of the focus areas of developers for Multipath extension is to overcome the hurdle of stack implementation. This is one of the challenges that make it hard for Multi path extension to be universal, which requires Multipath stack to be not only in a client side, but also in the server side.

On the other hand, it is also task of near future to clarify the restriction of middlewares such as firewalls and Network Address Translation services, which gets confused because of different nature of SYN ACK methodology of Multi Path extension.

Although, Multi Path extension comes with handy benefits, it is vital to have highly effective and intelligent path control mechanism in order to differentiate types of data and change behavior regarding data type. It would bring undesirable results without proper intelligent mechanism for couple of scenarios. For example; User might use his/her smartphone, which supports taking advantages of multiple path at the same time, such as using 4G and WI-FI together. Imagining that user watches YouTube videos and Multi Path extension just divides traffic as 50/50 between 4G-cell connection and WI-FI, without investigating data type, this would give shocking phone bills for users spending big amount of data traffic on their 4G connections.

List of Literature

[1] Network Sourcery, Inc., Network Sourcery, 2008. [Online]. Available: http://www.networksorcery.com/enp/protocol/tcp.htm. [Accessed 15 January 2015].

[2] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," January 1996. [Online]. Available: https://www.ietf.org/rfc/rfc1889.txt. [Accessed 17 January 2015].

[3] A. Durresi and R. Jain, "RTP, RTCP, and RTSP - Internet Protocols for Real-Time Multimedia Communication," in *The Industrial Information Technology Handbook*, CRC Press, 2005, pp. 187-198.

[4] K. Azadet and M. Yu, "Forward Error Correction (FEC) techniques for optical communications," in *Lucent Technologies*, Montreal, 1999.

[5] S. Ahsan, V. Singh and O. Jörg, "MPRTP: multipath considerations for real-time media," in *Association for Computing Machinery*, New York, 2013.

[6] D. Wischik, C. Raiciu, A. Greenhalgh and M. Handley, "Design, implementation and evaluation of congestion control for multipath TCP," Usenix NSDI, 2014.

[7] C. Raiciu, M. Handley, S. Barre and J. Iyengar, "Architectural Guidelines for Multipath TCP Development," March 2011. [Online]. Available: https://tools.ietf.org/html/rfc6182. [Accessed 30 January 2015].

[8] C. Paasch and C. Raiciu, "Multipath TCP lecture," GoogleTechTalks, London, 2012.

[9] J. Hughes, "Raspberry Pi camera source code in C," 28 February 2013. [Online].

Available:https://github.com/raspberrypi/userland/blob/master/host_applications

/linux/apps/raspicam/RaspiVid.c#L128.. [Accessed 05 February 2015].

[10] A. Ford, C. Raiciu and M. Handley, "TCP Extensions for Multipath Operation with Multiple Addresses," 08 March 2010. [Online]. Available: https://tools.ietf.org/html/draft-ford-mptcp-multiaddressed-03. [Accessed 12 February 2015].

[11] J. Rosenberg and H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction," December 1999. [Online]. Available: https://tools.ietf.org/html/rfc2733. [Accessed 15 February 2015].

[12] D. Jurca and P. Frossard, "Video packet selection and scheduling for multipath streaming," *Multimedia, IEEE Transactions on,* vol. 9, no. 3, pp. 629-641, 2007.

[13] S.-H. Choi and M. Handley, "Designing TCP-Friendly Window-based Congestion," PFLDNeT, Tokyo, 2009.

[14] G. Cheung, "Near-optimal multipath streaming of h.264 using reference frame selection," in *Multiple Reference Motion Compensation*, Hanover, Publishers Inc., 2009, pp. 653-656.

[15] T. Friedman, A. Clark and R. Caseres, "RTP Control Protocol Extended Reports (RTCP XR)," November 2003. [Online]. Available: https://tools.ietf.org/html/rfc3611. [Accessed 03 March 2015].

[16] Y.-K. Wang, R. Even, T. Kristensen and T. R. Jesup, "RTP Payload Format for H.264 Video," May 2011. [Online]. Available: https://tools.ietf.org/html/rfc6184. [Accessed 14 March 2015].

[17] J. B. Cain and G. C. Clark, Error-Correction Coding for Digital Communications, Washington: Springer Science & Business Media, 1981.

[18] G. Fairhurst and L. Wood, "Advice to link designers on link Automatic Repeat reQuest (ARQ)," August 2002. [Online]. Available: https://tools.ietf.org/html/rfc3366. [Accessed 19 March 2015].

[19] Z.-G. Li and Z.-Y. Zhang, "Real-time streaming and robust streaming H.264/AVC video," in *Image and Graphics*, Shanghai, 2004.

[20] C. Hornig, "A Standard for the Transmission of IP Datagrams over Ethernet Networks," April 1984. [Online]. Available: https://tools.ietf.org/html/rfc894. [Accessed 28 March 2015].

[21] M. Fiore, "An adaptive transport protocol for balanced multihoming of real-time traffic," in *Global Telecommunications Conference*, Turin, 2005.

[22] "Multi-Media Abstraction Layer API," 2012. [Online]. Available: https://github.com/raspberrypi/userland/blob/master/interface/mmal/mmal.h. [Accessed 04 April 2015].

[23] H. Zhang, S. Zhang, F. Song, F. Ramos and J. Crowcroft, "An estimator of forward and backward delay for multipath transport," University of Cambridge, Cambridge, 2009.

[24] Y. Hasegawa, I. Yamaguchi, T. Hama, H. Shimonishi and T. Murase, "Improved data distribution for multipath TCP communication," in *Global Telecommunications Conference*, Tokyo, 2005.

[25] L. Rizzo, "Erasure codes based on Vandermonde matrices," March 2012. [Online]. Available: https://github.com/randombit/fecpp. [Accessed 19 April 2015].

`