ANALYZING THE IMPACT OF STACK-PROTOCOLS, ENCAPSULATION METHODS AND VIDEO CODECS ON THE PERFORMANCE OF LIVE VIDEO STREAMING

MCS

By

Afaq Iqbal

Submitted to the faculty of Computer Software Engineering Department, Military College of Signals, National University of Sciences and Technology, Islamabad in partial fulfillment of the requirements for the degree of MS in Computer Software Engineering

July 2013

ABSTRACT

Streaming of multimedia data is getting huge popularity in different fields of information communication and entertainment. This is possible because of the increase in available bandwidth of the network, increase in computing power of the computer processors and memory, and also because of the advancement in algorithms. Streaming of live video over the internet is dependent on diverse factors. These factors have direct impact on the quality of the transmitted video. The underlying protocol, encapsulation method and the choice of video codec are the main factors involved which affects the transmission of video over a network. Various studies have been performed to analyze the impact of these factors individually on the quality of video transmission. However, their joint impact on video transmission is not done hitherto.

In our thesis we characterize the performance of video streaming setup, while considering the effect of protocol, encapsulation method and the underlying video codec on delay and percent frame loss as the key investigating parameters. The results obtained while considering diverse video streaming setups are provided in the results section of this thesis.

DECLARATION

No portion of the work presented in this dissertation has been submitted in support of another award or qualification either at this institution or elsewhere.

DEDICATION

All praise and thanks to almighty Allah, the most gracious and the most merciful, Master of the Day of Judgment. Guide us with courage and right path, path of those to whom you have bestowed your blessings.

This research work is dedicated to my teachers and family, for their never ending love, faith and motivation.

ACKNOWLEDGEMENT

First of all, all praises to Almighty Allah for His Blessing and Mercy. Indeed, His Favor is the most important source for the completion of the present thesis.

I would like to pay my gratitude to my kind and supportive supervisor Brig. Dr. Fahim Arif for his time, guidance and support related to present work. In fact this thesis is an outcome of his concern that led me to the completion of this research work. He has been very cooperative throughout the thesis work.

Moreover, I am also thankful to Dr. Hammad Afzal, Lecturer Bilal Rauf, Dr. Kamran Ghani and Dr. Nasru Minullah whose guidance led me to complete this thesis. I further pay my gratitude to my family and friends for their prayers and support throughout my thesis work.

The last but not the least, I dedicate this thesis to the very kind and affectionate my parents whose commitment, devotion and prayers made me possible to complete this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ASF	Advanced Streaming Format (Former) / Advanced Systems Format (Later)
AVI	Audio Video Interleaved
CIF	Common Intermediate Format
GNU	General Public License
GOP	Group of Picture
HDTV	High Definition Television
HTTP	Hyper Text Transport Protocol
ITU-T	International Telecommunication Union-Telecommunication
JVT	Joint Video Team
LAN	Local Area Network
M-JPEG	Motion-Joint Photographic Experts Group
MMS	Microsoft Media Server
MOS	Mean Opinion Score
MPEG	Moving Pictures Experts Group
PS	Program Stream
PSNR	Peak Signal-to-Noise Ratio
QoE	Quality of Experience
QoS	Quality of Service
QCIF	Quarter Common Intermediate Format
RTP	Real-time Transport Protocol
RTSP	Real Time Streaming Protocol

SDTV	Standard Definition Television
SMPTE	Society of Motion Picture and Television Engineers
TCP	Transmission Control Protocol
TS	Transport Stream
UDP	User Datagram Protocol
VLC	Video LANClient
VCEG	Video Coding Expert Group
WMV	Windows Media Video

Chapter 1

INTRODUCTION

1.1 Introduction

Today users prefer information not only in traditional plain text but also in multimedia images and videos streaming. Websites such as Facebook, YouTube, tudou and DailyMotion are popular websites where people share rich multimedia information. Furthermore, online web television is also getting its popularity. Hence, live video streaming is becoming a vital part in today's Internet applications [1] [2].

Due to the overwhelming presence of live video streaming on internet it is very important that they should meet the performance standards. Since video streaming depends on several factors, research is desired to be carried out through literature review and experiments for the performance of video streaming setup, while keeping in view the effect of protocol, encapsulation methods and the underlying video codec on delay and percent frame loss as the key investigating parameters.

The rest of chapter gives a brief overview of the whole thesis document, discussing the background, indentifying the problem statement, describing the research questions and the research methodology. Before ending the chapter an outline to the whole thesis document is also given.

1.2 Background

Heterogeneous computer networks exist to connect different types of computers for sharing of data and communication of information. These computers can be on

different locations that may be within a city or two different parts of the world. In the old days data carried through these networks was only in textual form. But with the passage of time and advancement in technology, the transmission of multimedia data became more attractive choice and data like animation, pictures, voice and video became more popular on internet. Now these multimedia networking applications like internet telecommunication, internet protocol TV (IPTV) and video conferencing are used in health, education, e-commerce and many other sectors [3].

These multimedia communication scenarios involve two modes of media transmission -the download mode and the streaming mode. In the **download mode**, a user downloads the entire video file and then plays back the video file, while in **streaming mode**, the video content need not be downloaded in full, but is being played out while parts of the content are being received and decoded [4-6]. Streaming video over the Internet has been experiencing dramatic growth due to the increase in bandwidth and computing power [7].

Video streaming depends on several factors and these factors may be network dependant or independent [8]. The network dependant factors are those factors which are dependent on the network (e.g. bandwidth, throughput, loss etc) while those factors which are not dependent on the network (e.g. compression, decompression, encapsulation etc) are independent factors. Some of these factors that were used in our thesis are delay, streaming server, protocols, codec, encapsulation methods, resolution, bit rate, frame rate and media caching. A brief overview of these factors is given with the literature review in the next chapter.

This thesis also discusses different experiments we performed for studying and analyzing the effect of protocols, encapsulations and codecs on the performance of

live video streaming. Our experimental setup is composed of various software and hardware tools in order to carryout various experiments for proposed setup. Data were captured in those experiments and was analyzed. Based on the results obtained, analysis and conclusions were drawn at the end of thesis.

1.3 Problem Statement

As there is increasing demands of video streaming over a network that is watching the video live without downloading in every field of life. The performance of this live video depends on many factors which are network reliant or sovereign. Delay and frame drops are the important factors which affect the performance of video streaming and are dependent collectively on other factors, mainly protocol, encapsulation and compression algorithm. Research was needed to test and analyze the effect of these factors (protocol, encapsulation, codec) on delay and percent frame loss experimentally.

1.4 Research Questions

The analysis and study of the effect of protocol, encapsulation and codec on delay and frame loss gives better view of the performance of video streaming on a network. For this analysis research has been done to answer to the given below research questions.

- What is the effect of protocol, Encapsulation and codec on delay individually and collectively while streaming?
- What is the effect of protocol, Encapsulation and codec on frame drops while streaming?

To answer these research questions various experiments were performed in the designed experimental setup. Data was obtained in these experiments and then results were drawn based on the obtained data. Our thesis is helpful for the developers to get maximum performance in video streaming and also for the researchers to work on other factors independently and collectively to improve the performance of live video streaming to a much greater extent.

1.5 Research Methodology

The research methodology that we used in our thesis consists of both the literature review and experimental study. The literature review gives us information about the video streaming, protocols for video streaming, codecs, encapsulation methods and other factors like delay, streaming severs, frame rate, network cache etc. This literature study was also helpful in designing the experimental setup, on which different experiments were performed for data collection. The collected data was transformed to bar charts for comparisons and analysis. These comparisons and analysis results were used to answer the research questions.

1.6 Our Contribution

In our research study, we primarily investigated experimentally the effect of protocol, encapsulation and codec on delay by creating our own experimental setup. Data and results obtained from these primary experiments were transformed into a conference paper, "*Performance Evaluation of Stack-Protocols, Encapsulation Methods and Video Codecs for Live Video Streaming*", and published in IEEE conference ICoICT (International Conference on Information and Communication Technology) 2013, held on 20^{th} – 22^{nd} March, 2013, at Bandung, Indonesia.

1.7 Thesis Outline

In the next chapter, review of different dependent and independent factors for streaming is given. Moreover, the selection of video, software and their use is given, which would be helpful in executing the experiments. Chapter 3 explains the designing of the experimental tools and setup for performing the experiments of the thesis questions. In chapter 4, different experiments are explained that were performed on the experimental setups, designed for the thesis questions. In chapter 5, the results obtained from the experiments are explained. Also, evaluation of the research questions is done based on the obtained results. The last chapter 6 concluded the thesis report and gives the future directions.

Chapter 2

BACKGROUND

2.1 Introduction

Streaming is one of the multimedia transmission modes on the network. It depends on many factors to stream a video on the network from one place to another place. Some of these factors are protocols, encapsulation methods, codecs, streaming servers, network cache, resolutions, bit rate and frame rate of the transmitted media and have their affect on the streaming of live video. These factors affect the video performance by introducing delay in the live stream and degrading the quality of the video.

This chapter gives a brief overview of these factors. This literature review is helpful in understanding these factors which are later used in designing the experimental setup and conducting experiments. Besides these factors overview to other tools, software and hardware (Online Stopwatch [9], Camtasia Studio v.2.0.1 [10], VLC v 2.0.3 of VideoLAN [11]), that are used in our experiments is also given in this chapter.

2.2 Brief Overview of Streaming

Streaming is a method for transferring data that is being processed as an uninterrupted stream. Streaming became possible because of the increase in bandwidth, computers processing power, and advancement in audio and video compression algorithms [12].

In streaming a user creates a constant connection with the server and initiates the playback. The software used by user downloads several seconds of data from the

server and stores it into a buffer in the system. The media (audio/ video data) is played when this buffer is filled. Media from the server is consistently sent to the buffer until it is downloaded and played completely. To a user it seems like he is watching it live [12].

With the help of streaming users can watch media even if they have no space available to download the file completely. Streaming can be used in video chat, video messaging and video conferencing applications. In this way it is more secure because the media is not saved on the client's system [12].

2.2.1 Streaming Process

There are three essential components for streaming; a video server, an encoder, and a player. A video server is required to create streaming media and distribute it on demand. The encoder compresses the media for streaming over the network, and any system that does this encoding must have fast processing power and large memory. The player is the component that the user needs to watch the stream [12].

Sequentially, first, a raw video is taken as an input video, which is then passed through a compression algorithm. The compression algorithm encodes this raw video. After encoding the encoded video is then encapsulated with the encapsulation algorithm. Once the video is encapsulated it can then be transmitted via network.

To watch the streamed video on the client side it is collected from the network in encapsulated form. The encapsulated video is de-encapsulated to obtain it in encoded form. Then this encoded video is decoded to watch it as an output video. The whole process is shown in the Figure 2-1.

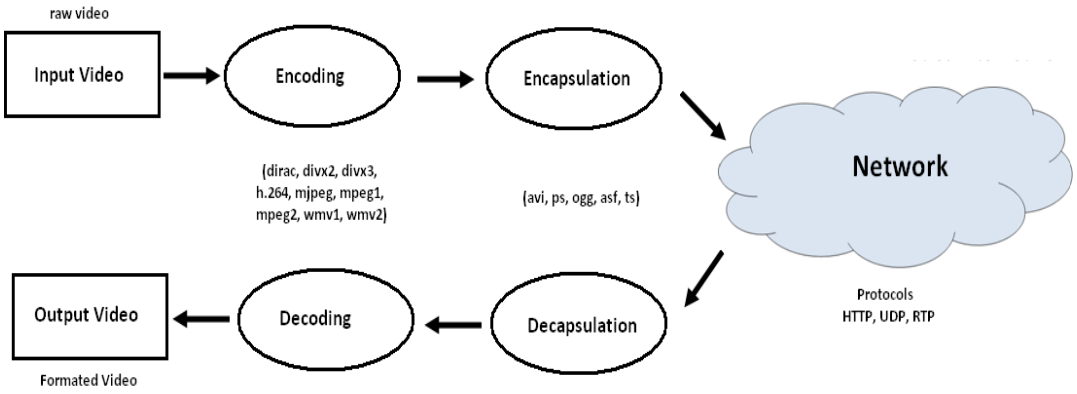

Figure 2-1 Streaming Sequential Diagram

In our thesis, we generated two scenarios based on this system diagram. These scenarios elaborate this system diagram in a detailed manner, which are then used to perform various experiments of steaming videos. These scenarios are explained in chapter 3.

2.3 Video Codecs

Codec is a compression algorithm used to compress the size of the media stream. There are many audio and video codec e.g. dirac, divx2, divx3, h.264, mjpeg, mpeg1, mpeg2, wmv1, wmv2 etc [11, 13].

2.3.1 Dirac

Dirac [14, 15] is an open source codec, developed by the British Broadcast Corp. (BBC), more stable between QCIF (176x144) and HDTV (1920x1080) while less stable at low bitrates [16-18]. Dirac GOP (Group of Picture) arrangement uses I-frame, L1 and L2 frames and uses an internal Quality (QM) for indication of image

quality within a frame [16]. Its performance is much better than MPEG-2 but a little less than H.264 [17]. It is not popular in market [15, 16].

2.3.2 H.264

H.264/MPEG-4 PART 10 AVC is a product of JVT (Joint Video Team) [18] and was the mutual project between MPEG (Moving Pictures Expert Group) [19] and VCEG (Video Coding Expert Group). H.264 compresses videos into a manageable size that can be used for streaming. H.264 is an expensive and popular codec and unlike Dirac, much stable at low bitrates [15-17].

2.3.3 Mjpeg

MJPEG is a product of Joint Photographic Experts Group. Motion-Jpeg is one of the commonly used codecs for videoconferencing. Mjpeg is a low cost coding and do not use the inter-frame compression [20]. In M-jpeg each frame is encoded independently as a JPEG image therefore yields higher Bit-rate [20, 21]. It is best for videos with a lot of stuff and actions [21].

2.3.4 Mpeg-1

MPEG-1 [22] was the first project of MPEG, designed for coding progressive pictures, and published in 1993 as ISO/IEC 11172. MPEG-1 is optimized for a non-interlaced video signals and has three standard parts – audio compression, video compression and a multiplexing system for audio and video synchronization. It mainly encodes videos at bit-rates up to about 1.5 Mbps and has almost clear stereo audio quality at 192 kbps/channel [23].

2.3.5 Mpeg-2

MPEG-2 is an international standard also known as ISO/IEC 13818 [24]. MPEG-2 is optimized for interlaced pictures and is able to encode SDTV at Bit Rates from about 4-9 Mbps and HDTV at 15-25 Mbps. MPEG-2 is more efficient than MPEG-1 for the interlaced video signals coded at Bitrates more than 3 Mbps. MPEG-2 audio is backwards compatible with MPEG-1 audio and also has expand the scheme to multi-channel surround sound coding [23].

2.3.6 Divx

DivX is the product of DivX, LLC, and a subsidiary of Rovi Corporation. It adds some extra features such as Xsub subtitles or chapter titles to the AVI container. DivX can be used on web-pages as well as for streaming purpose [25].

2.3.7 WMV

WMV is the proprietary format of Microsoft and is based on ASF. It is used for streaming files as well as progressive downloads [12]. WMV also know as VC-1 has been standardized by the Society of Motion Picture and Television Engineers (SMPTE) [19]. It was designed to achieve high quality compressed video from very low to very high bit rates. Comparing with MPEG-2, WMV achieve better quality [26].

2.4 Container or Encapsulation Schemes

A container contains one or many media streams that are encoded by the underlying codec. Usually, a container contains an audio and a video stream. A container is also known as encapsulation. Some of the containers are avi, ps, ogg, asf, ts etc [11, 13].

2.4.1 Avi

AVI, the special case of the RIFF (Resource Interchange File Format), is the product of Microsoft and the most generally available format for the audio and video data use on personal computer [6]. It supports wide range of codecs and compress less than MPEG and MOV. It is used for progressive downloads [12].

2.4.2 Ogg

Ogg is the product of Xiph.Org Foundation. It is an open and freely available encapsulation format for streaming media. It can encapsulate a number of audio and video codecs into a single bit stream. It can be used for efficient streaming of High quality media. It can also encapsulate other form of data such as text and metadata [27]. An overview to ogg is also given at [28].

2.4.3 ASF

ASF file format is specially designed to store audio and video data and run over the network [6, 29]. It is the digital media presentation format that support live and on demand digital media. It is specially designed for streaming and/or local playback, independently of the communication protocol [30].

2.4.4 PS and TS

A media has several streams for video, audio and subtitles etc, when all these are mixed into a single steam they are called the Program Stream (PS). Another Stream, called the Transport Stream (TS) [31], was designed to stream video through a network, or satellite where PS gets failed on such channels [11].

2.5 Video Streaming Protocols

Protocols play an important role in streaming videos. Furthermore, transport protocols are also one of the subsystems of the streaming server. There are many protocols available for data transmission but the most widely used protocols for video streaming include Hypertext Transfer Protocol (HTTP)[32], Universal Datagram Protocol (UDP)[5], Real Time Protocol (RTP) [33], Real Time Streaming Protocol (RTSP)[34] etc [4].

The Hypertext Transfer Protocol (HTTP) was originally designed by Tim Berners-Lee. This protocol is used by servers for communication. It is generic, stateless, object oriented protocol used for many tasks [6, 35, 36] such as name servers and distributed object management systems, typing of data representation and communication between user agents and proxies/gateways to other Internet protocols [32]. The User Datagram Protocol (UDP) is transport layer protocol. UDP provides a connectionless but unreliable datagram service over the network [3, 5]. Real time Transport Protocol (RTP) is IP based transport protocol for real time data, primarily designed for multicasting data but can also be used in unicasting [3, 6, 36]. Microsoft Media Server (MMS) Protocol is Microsoft's real time streaming protocol that uses TCP and UDP [37].

2.6 Important Terminologies and Tools

Besides codecs, encapsulations and protocols, delay and frame loss are also of the nature needs due attention. They will be explained in subsequent subsections. Other tools and softwares that were used in designing experimental setup are video

streaming server, VideoLan Client, Camtasia Studio, Microsoft excel, online stopwatch and webcam which can be deciding factors.

2.6.1 Delay

It is the amount of time taken by a bit to travel from source to destination. It is also called the Latency [40, 41]. In real time video streaming, if video packets are not arrived in time they become useless [4].

2.6.2 Frame Loss

It is the number of video frames that are transmitted from source to destination, but do not reach the destination.

2.6.3 Video Streaming Server

In order to stream a video streaming server is required. For efficient streaming, video streaming server plays an important role [38]. A good streaming server is that process multimedia data within timing constraint, it must have interactive controls operations like pausing and resuming video, forward and backing video. Also, the streaming server should retrieve media components synchronously [4].

2.6.4 VideoLan Client

VideoLAN is a nonprofit organization that provides free and open source solution to the multimedia. The VideoLAN software solution is developed under the GNU General Public License (GPL) by the students of Ecole Central Paris and open source developer around the world. VideoLAN Client (VLC v. 2.0.3) is one of the products which can be used as a server for streaming purpose as well as a client for receiving and displaying the stream on many platforms [11].

It can support several encapsulation formats (e.g. avi, ps, ogg, asf, ts), codecs (like dirac, divx2, divx3, h.264, mjpeg, mpeg1, mpeg2, wmv1, wmv2 etc) and as a streaming server it can also contain several protocols such as http, udp, rtp, rtsp etc. Its architecture is given at [11].

VLC also give detailed information about media and codec such as codec of the media, frame rate, bit rate, number of displayed frames, etc. Moreover, VLC provides facility to capture the snapshots of video at different locations. User can stop and play videos at will in VLC. In advance options VLC provide facility to check video frame by frame. Also there are options in VLC to add zoom and effects into a video.

2.6.5 Camtasia Studio

Camtasia Studio is the screen recording and videos editing software product. It is an easy tool for cutting, splicing and combining videos clips. It provides facility to record full video screen or part of it [10]. The screen captured can then easily be saved as video in .avi format. Another tool CamStudio Recorder [39] is also available for this purpose under General Public License.

2.7 Conclusion

This chapter explained the system diagram of the streaming process, in which a raw video is taken as an input and passes through encoding, encapsulation, network, decapsulation and decoding to receive as an output. Also information about the protocols, compression algorithms, containers used in our thesis is given. Besides these, information about streaming server, VLC, Camtasia Studio, and other tools that are used in thesis work are also discussed.

Chapter 3

EXPERIMENTAL TOOLS AND METHOD

3.1 Introduction

This chapter explains two different experimental setups designed for two different scenarios, which are based on the research questions. Later experiments were performed on these scenarios to analyze the research questions. This chapter also gives information about the hardware specifications that are used to design the two experimental setups.

3.2 Computer System Specification

The computer system that was used in the experimental setup has Intel(R) Core (TM) 2 Duo E7200 @2.53 GHz processor, 2GB Ram, DG31PR motherboard, 250GB of SATA hard disk and 128MB of graphics memory. 19 inches Micron CRT monitor was also attached to the system whose screen resolution was 1024x768 (true colors), color quantity was 32 bits and refresh rate was 75 Hz. More detail specifications of the system is given in Appendix A.

Besides these hardware specifications and configurations setting, other softwares were also installed to the system. These softwares include Camtasia Studio, VideoLAN, Microsoft Office and an Online Stopwatch. A webcam was also connected to the system for capturing live videos.

3.3 Using VLC as Streaming Server

The reason for using VLC as a server is that it accommodates several audio and video

codecs, encapsulation methods and protocols for streaming media over a network; furthermore, it was developed for the research purpose. In our thesis we are using the most commonly used encapsulation methods (avi, ps, ogg, asf, ts), its supported codecs like dirac, divx2, divx3, h.264, mjpeg, mpeg1, mpeg2, wmv1, wmv2 and the mostly used protocols such as http, udp, rtp etc for streaming videos.

3.3.1 Streaming Via VLC

In order to stream a live video via webcam or standard stored video run the VLC. In main window click the *"Media"*, down in media click *"Stream"*. A new window will be opened where one can add a standard stored video or select live video with the help of capturing device. In order to stream a stored video, add the video with the help of "Add" button under the *"File"* Tab else under *"Capture Device"* Tab select the video device name for the live video.

After selecting the source, change the caching to 100 ms in *"show more options"*. Now click the "stream" button and then "Next". Now in the opened window select the protocol in *"New Destination"* and "Add". Below the *"New Destination"* select a "Profile" or "create new profile" or "edit" the old profile and then proceed with the "Next" button. Now the video is ready to stream, click *"Stream"* to start streaming the selected video. The complete process is shown stepwise in the Figure 3-1. A more detail explanation is also given at [11] [42-44].

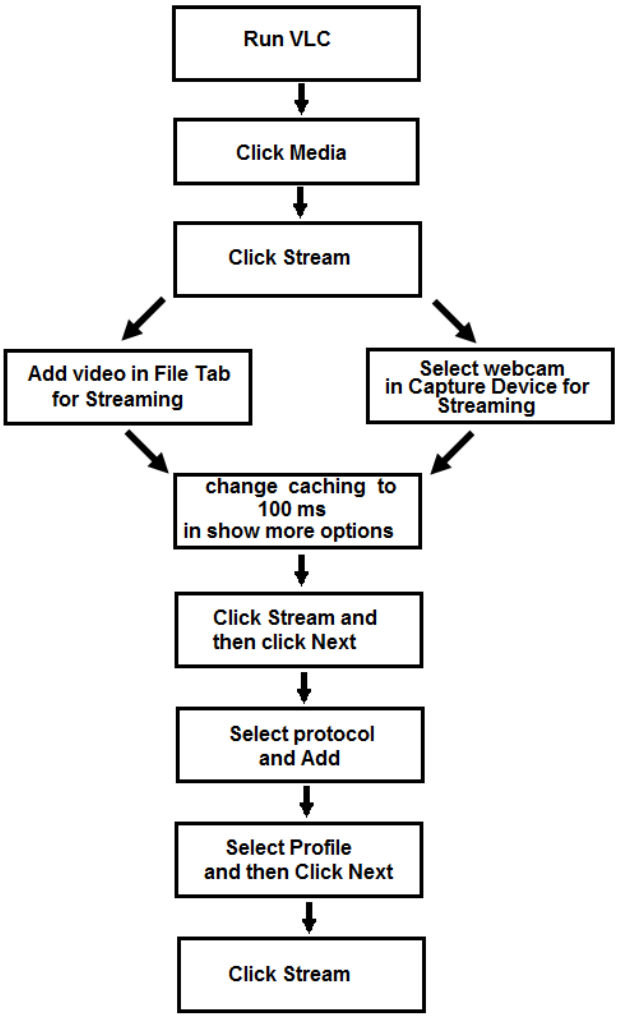

Figure 3-1 Streaming Video Via VLC

3.3.2 Creation of New Profile in VLC

In our thesis, we created a new profile for our experiments. The new profile was created by clicking *"Create a new profile"* button which will open a new window. First *"Enter Profile Name"* and then under *"Encapsulation"* Tab select the desired encapsulation. After selecting the encapsulation, check the *"video"* box and select the video codec in *"video codec"* Tab. Also adjust the *"Bit Rate"*, *"Frame Rate"* and *"Resolution"* in the same Tab. Click the *"Save"* button to save the created named

profile. Figure 3-2 also shows how to create the new profile step by step and more details can be found at [44].

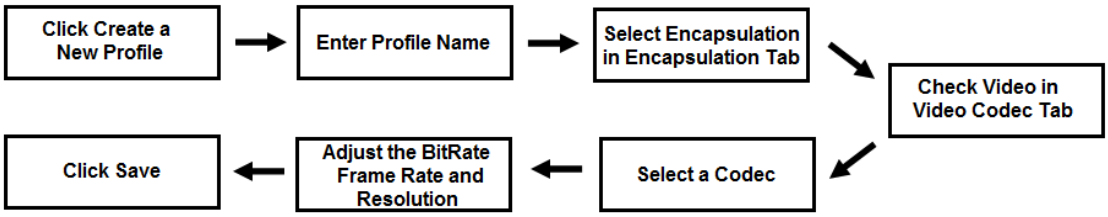

Figure 3-2 Creation of New Profile

3.3.3 Receiving Stream via VLC

VLC is also capable to receive the streamed video from the network. The process is shown in Figure 3-3. On the client side run VLC and click *"Open Network Stream"* under the *"Media"*. A new window will be opened. Here enter the *"URL"* and port of the server which is streaming the video. Also adjust the *"Caching"* to the 100 ms in *"show more options"* in the same window. Click the *"Play"* button to start playing the streamed video. Detail information can also be found at [42, 43].

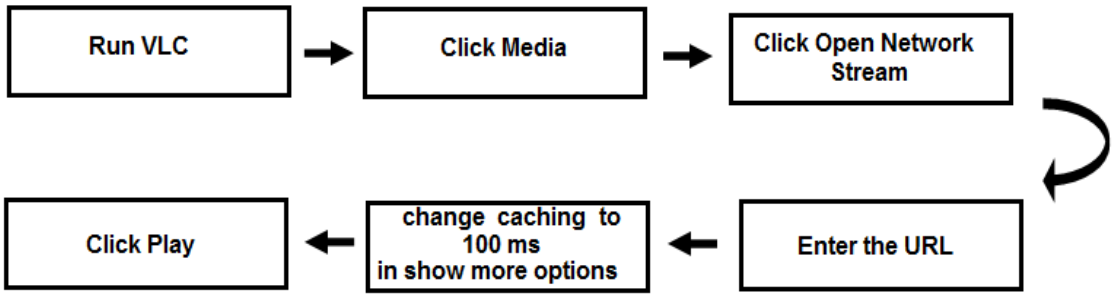

Figure 3-3 Receiving Stream via VLC

3.4 Configuring Camtasia Recorder

Camtasia Recorder is one of the applications of camtasia studio and it was configured with our thesis requirements for conducting experiments. First the camtasia recorder was launched from the camtasia studio. After launching, **"Select Region"** was selected in *"input"* of the *"capture"* menu, so that the selected region of the monitor screen is captured as a video and not the whole screen. In the same *"capture"* menu, under *"output"*, **"File"** was selected, so that the activity on the selected screen is recorded as a video file that can be analyzed later as well.

Besides these changes, in *"tools"* menu *"tools options"* was launched from *"options"*. In *"tools options"* under the *"AVI"* tab *"video compression setup"* was opened through the button *"Video Setup"*. In *"video compression setup"* codec **"TechSmith Screen Capture Codec"** was chosen for capturing the monitor screen as a video. In the same *"AVI"* tab, under *"Video options"* the frame rate was increased and set to **50 frames/sec**, so that the recorded video do not pass over any event of the selected screen. The whole configuration process is also shown diagrammatically in the Figure 3-4.

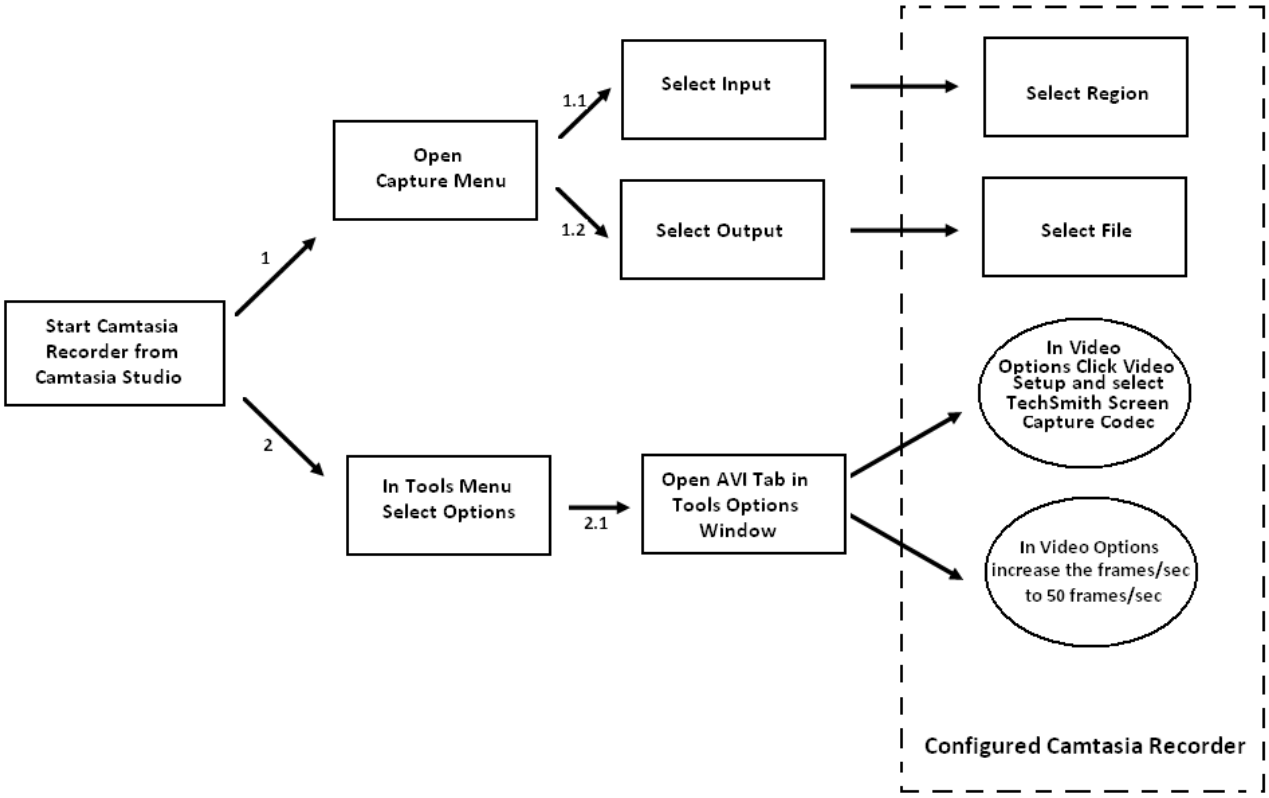

Figure 3-4 Configuration Diagram of Camtasia Recorder

3.5 Video Capturing Device (Webcam)

A simple webcam was also used in designing the experimental setup. This webcam have a fixed frame rate of 30 fps and five types of changeable resolutions (160x120, 176x144, 320x240, 352x288, and 640x480). The webcam also provided the capability of variable zoom, which could be adjusted manually. The webcam would capture the video in raw format of YUY2.

3.6 Experimental Scenarios

Using the system specifications, tools and configuration settings, two different scenarios was developed to test the research questions. Using these scenarios various experiments were perform which are explained in the next chapter. These two scenarios are explained below.

3.6.1 Scenario One

In scenario one, experimental setup was designed for the delay factor. A plug and play webcam was attached to the computer system through a USB port. The webcam was also attached to the monitor in such a way that it could capture the monitor screen and its zoom was adjusted accordingly as shown in Figure 3-5.

In this scenario the webcam is used to capture the online stopwatch on the monitor screen and its video is streamed with the help of VideoLAN. This streamed video was captured on the client application which was another instance of the same application VideoLAN or windows media player. Also this activity of original streamed stopwatch and received stopwatch video was captured with the help of Camtasia Recorder as a video in avi format.

Camtasia Recorder was also configured so that it could capture the activity on screen at 50 fps which was greater capturing frame rate then the frame rate used in experiments. The reason for keeping more capturing frame rate than the frame rate used in streaming experiments is that the resulted avi video must contain all of the streamed frames.

Later this resulted avi video would be played and then paused at different points of time so to obtain two different timers readings-one the original time and other the stream of the original time. As both the timer readings were obtained from the same frame then the difference between the two collected timer readings would represent the delay of the underlying experimental setup. The whole process is shown in Figure 3-5.

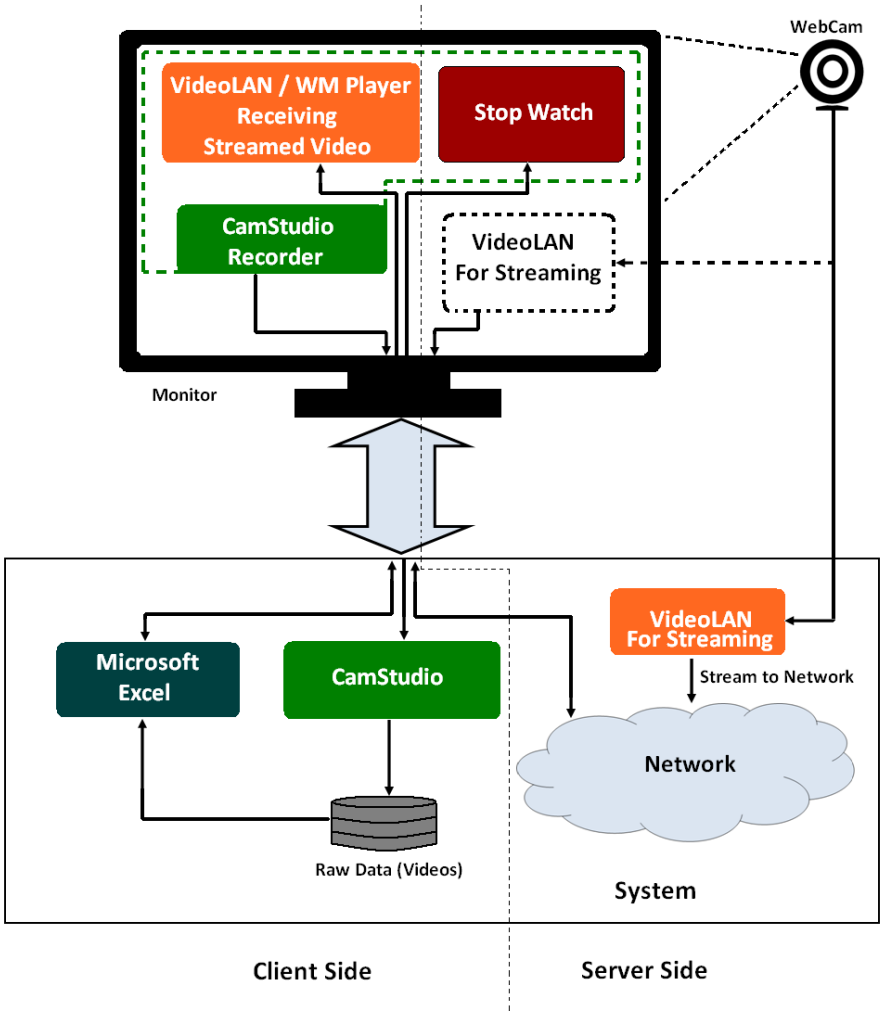

Figure 3-5 Scenario One

The Figure 3-6 shows the screenshot of the single frame containing the original stopwatch time 2.796 seconds and time received on the streaming setup 2.236 seconds. Hence the difference between the two times is 560 milliseconds (2.796 – 2.236 = 0.560 seconds) represent the delay due to the underline setup. In order to get the precise delay five data readings were noted in Microsoft excel and its average was taken.

Figure 3-6 Screenshot of Stopwatch

3.6.2 Scenario Two

In scenario two, experimental setup was designed for percent frame loss factor. Here instead of using the webcam a standard video, named as “paris” and available at [45], was downloaded and used. Video “paris” is the copyright of PictureTel Corporation and available free of charge for research purpose. The video “paris” consists of 1065 CIF (352x288) picture frames with aspect ratio (4:3) and size of 155MB [46] in Y4M format. This video was streamed with the help of VideoLAN from server side installed on the computer system. And then the streamed video was captured on the client application which was another instance of the same application VideoLAN.

This video standard video “paris” would be streamed under different experimental setups and using the statistics of client side VideoLAN, number of frames lost were noted in Microsoft excel sheet. In order to get the precise frame loss five data readings were noted in Microsoft excel and its average was taken. Later percentage frame loss was calculated in same Excel sheet as total frames of standard video were already given with the video. The whole process is shown in Figure 3-7.

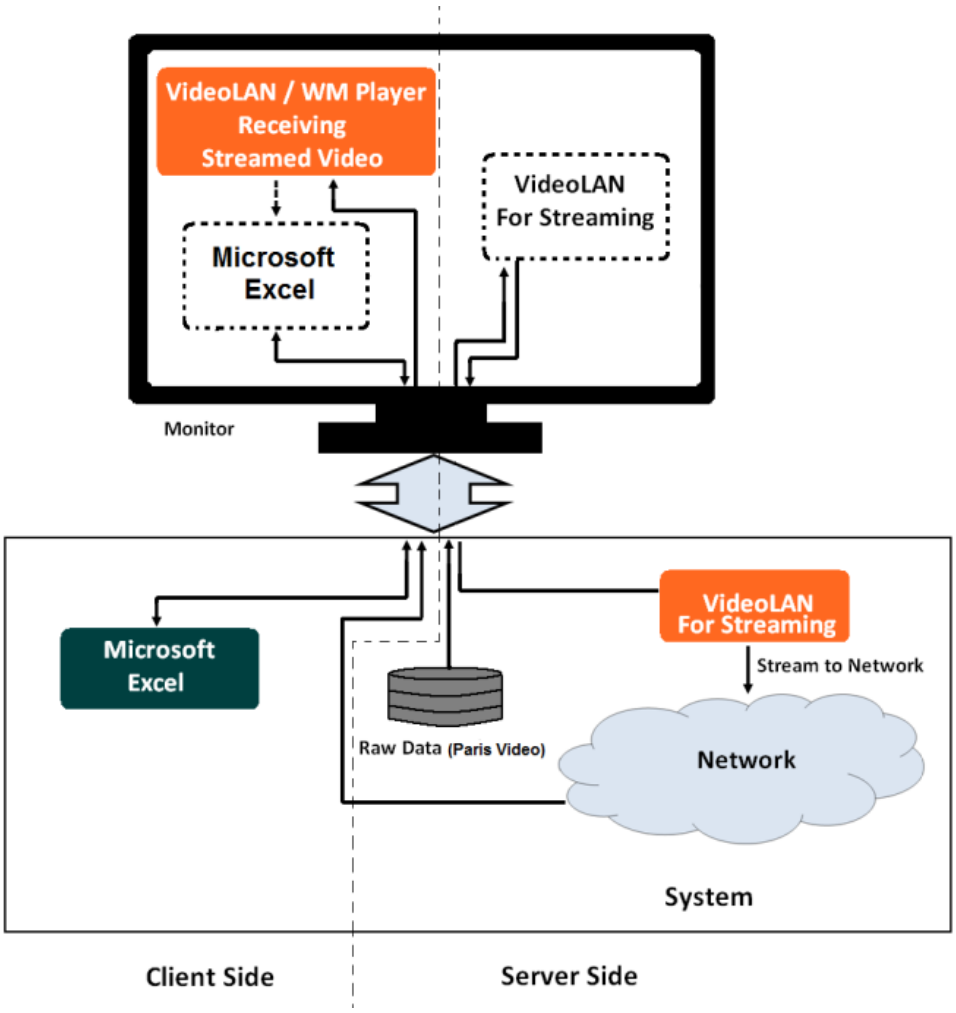

Figure 3-7 Scenario Two

3.7 Conclusion

In this chapter we discussed two experimental setups. These setups are based on the system diagram for streaming discussed in chapter two. The two experimental setups are used for two different scenarios. Information about the system is also given on which these scenarios are tested.

Chapter 4

EXPERIMENTAL SETUPS

4.1 Introduction

This chapter explains different types of experiments performed, based on the experimental setups described in scenario One and scenario Two in the previous chapter. Besides the main experiments performed for the research questions, confirmation experiments are also explained in this chapter. These confirmation experiments are performed so that the data of the actual experiments can be verified from them. They are also referred as sub experiments. Each experiment was repeated five times and data was collected and then transformed into results. The complete data of all these experiments is given in appendix B.

4.2 Experiments of Scenario One

Using experimental setup described in scenario one section 3.6.1, four different experiments and three sub experiments were performed. In these experiments different components of our proposed experimental setup such as codec, encapsulations, protocols, and over all complete stack (combination of codec, encapsulation and protocol) were studied and analyzed against delay. The details about these different parameters and experimental setups are outlined in Table 1 and explained below.

TABLE 1 PARAMETERS USED IN SCENARIO ONE EXPERIMENTS

	Experiment 1 (Effect of codec on delay)	Experiment 2 (Effect of encapsulation on delay)	Experiment 3 (Effect of protocol on delay)	Experiment 4 (Analyzing of overall stack on delay)
Constant Parameters	Protocol	Protocol	Encapsulation	
	Encapsulation	Codec	Codec	
	Resolution	Resolution	Resolution	Resolution
	Bit Rate	Bit Rate	Bit Rate	Bit Rate
	Frame Rate	Frame Rate	Frame Rate	Frame Rate
	Caching for media	Caching for media	Caching for media	Caching for media

4.2.1 Experiment 1 (Effect of Codec on Delay)

In experiment 1 nine codec (dirac, divX2, divX3, h.264, m-jpeg, mpeg1, mpeg2, wmv1, wmv2) were selected based on compatibility and tried one by one while other effecting factors were kept constant. The other effecting constant factors are; (protocol = RTP), (encapsulation = TS), (resolution = 640X480), (bit rate = 1024 kb/s), (Frame rate 30 fps), (Caching for media = 100 milliseconds on both client and server side).

Experiment 1.1 (Confirming the Effect of Codec)

In order to confirm that the result obtained from experiment 1 is really showing the effect of codec on delay the experiment 1.1 above was performed. This experiment was similar to experiment 1. In this experiment the protocol was changed from RTP to HTTP, while other factors were kept same as was in the experiment 1.

4.2.2 Experiment 2 (Effect of Encapsulation on Delay)

In experiment 2 encapsulation schemes (avi, PS, ogg, asf, raw and TS) were selected based on compatibility and tried one by one while other effecting factors that were kept constant are; (protocol = http), (codec = mpeg1), (resolution = 640X480), (bit

rate = 1024 kb/s), (Frame rate 30 fps), (Caching for media = 100 milliseconds on both client and server side).

Experiment 2.1 (Confirming the Effect of Encapsulation on Delay)

For the confirmation of effect of encapsulation on delay experiment 2.1 was performed which was similar, having all factors same as that of experiment 2, but only one factor was changed. The changed factor was codec. In experiment 2 the codec was mpeg1 while in experiment 2.1 the codec was changed to mpeg2.

4.2.3 Experiment 3 (Effect of Protocols on Delay)

In experiment 3 protocols (udp, rtp and http) were selected that are mainly used for streaming and support streaming on the network and factors (encapsulation = TS), (codec = mpeg1), (resolution = 640X480), (bit rate = 1024 kb/s), (Frame rate 30 fps), (Caching for media = 100 milliseconds on both client and server side) were remain constant.

Experiment 3.1 (Confirming the Effect of Protocols on Delay)

For confirming the effect of protocols on delay another experiment 3.1 was performed. Experiment 3.1 had the same setup as was in experiment 3. Similarly the delay effecting factors were also same as in experiment 3 except the codec. Only the codec was changed from mpeg1 to mpeg2.

4.2.4 Experiment 4 (Analyzing of Overall Stack on Delay)

In this experiment protocol, encapsulation and codec were taken collectively based on compatibility and the performance results from the previous experiments. Others

factors (resolution = 640x480), (bit rate = 1024 kb/s), (Frame rate 30 fps), (Caching for media = 100 milliseconds on both client and server side) were kept constant.

4.3 Experiments of Scenario Two

Using scenario two described in section 3.6.2, three different experiments and there sub experiments were performed. In these experiments different components of our proposed experimental setup such as codec, encapsulations and protocols were studied and analyzed against percent frame loss. The details about these different parameters and experimental setups are outlined in Table 2 and explained below.

TABLE 2 PARAMETERS USED IN SCENARIO TWO EXPERIMENTS

	Experiment 5 (Effect of codec on percent frame loss)	**Experiment 6 (Effect of encapsulation on percent frame loss)**	**Experiment 7 (Effect of protocol on percent frame loss)**
Constant Parameters	Protocol	Protocol	Encapsulation
	Encapsulation	Codec	Codec
	Resolution	Resolution	Resolution
	Bit Rate	Bit Rate	Bit Rate
	Frame Rate	Frame Rate	Frame Rate
	Caching for media	Caching for media	Caching for media

4.3.1 Experiment 5 (Effect of Codec on Percent Frames Loss)

In experiment 5 nine codec (dirac, divX2, divX3, h.264, m-jpeg, mpeg1, mpeg2, wmv1, wmv2) were selected based on compatibility and tried one by one for streaming of video while other effecting factors were kept constant. The other effecting constant factors are; (protocol = RTP), (encapsulation = TS), (resolution = 352X288), (bit rate = 1024 kb/s), (Frame rate 30 fps), (Caching for media = 100 milliseconds on both client and server side).

Experiment 5.1 (Confirming the Effect of Codec on Percent Frames Loss)

To confirm the result of experiment 5 –Effect of codec on percent frames loss, this confirmation experiment was performed. In this experiment all the parameters and experimental setup was same as that of experiment 5 except protocol. The protocol was changed from RTP to HTTP and then experiment was performed.

4.3.2 Experiment 6 (Effect of Encapsulation on Percent Frames Loss)

In experiment 6 encapsulation schemes (avi, PS, ogg, asf, raw and TS) were selected based on compatibility and used one by one for streaming the video while other effecting factors that were kept constant are; (protocol = rtp), (codec = mpeg1), (resolution = 352X288), (bit rate = 1024 kb/s), (Frame rate 30 fps), (Caching for media = 100 milliseconds on both client and server side).

Experiment 6.1 (Confirming the Effect of Encapsulations on Percent Frames Loss)

To confirm the result of experiment 6 a confirmation experiment 6.1 was performed. This experiment was similar to experiment 6 but one factor – codec, was changed from mpeg1 to h.264. All other factors were kept same and constant as that of experiment 6.

4.3.3 Experiment 7 (Effect of Protocols on Percent Frames Loss)

In experiment 7 protocols (udp, rtp and http) were selected that are mainly used for streaming and support streaming on the network and factors (encapsulation = TS), (codec = mpeg1), (resolution = 352X288), (bit rate = 1024 kb/s), (Frame rate 30 fps), (Caching for media = 100 milliseconds on both client and server side) were remain constant.

Experiment 7.1 (Confirming the Effect of Protocols on Percent Frames Loss)

For confirming the result of experiment 7 -effect of protocols on percent frames loss the experiment was repeated as experiment 7.1 with some minor changes. Experiment 7.1 was similar to experiment 7 but one constant factor codec was changed from mpeg1 to mpeg2 and then experiment was repeated for all the three protocols. Each protocol was tried in this confirmation experiment and other factors were kept constant, similar to experiment 7.

4.4 Conclusion

In this chapter we explained different experiments which were performed for the two scenarios. Also there confirmation experiments are explained. These experiments also give information about those factors that were kept constant throughout the experiment and their assigned constant values. Each experiment was repeated five times and five times data was collected. The data was saved in Microsoft excel sheet for further processing that is getting averages and then generating results and graphs. The data also given in tabulated form in appendix B.

Chapter 5

RESULTS AND EVALUATION

5.1 Introduction

This chapter explains in details the results of the experiments performed in chapter 4. The data obtained from the experiments performed in chapter 4 was collected in excel sheet where it was processed, also given in appendix B. Averages and percentages were calculated for experiments and then the processed form of data was transformed to the bar charts and lined charts. These bar charts and lined charts are explained in the rest of chapter for every experiment performed in previous chapter. It should also be noted that the bar charts are obtained from the actual experiments while the two line charts are based on the confirmation experiments performed to verify the results of the actual results.

All these charts are shown and explained below in the same sequence as experiments were explained in previous chapter 4.

5.2 Scenario One Results

Using our proposed experimental setup in scenario one outlined in chapter 3, a live video was streamed for duration of half minute. After completing the experiments and data collection, the collected data was transformed into the bar charts and line charts for further analysis. These chats and their analysis are given below.

5.2.1 Result of Experiment 1 (Effect of Codec on Delay)

The result shown in Figure 5-1 is based on the data of experiment 1. It shows that

keeping the underlying setup parameters the same, different codecs experience different video transmission delay. Some codecs have more impact of delay like Dirac and h.264 while some have a less impact like divX3, mpeg1 and mpeg2.

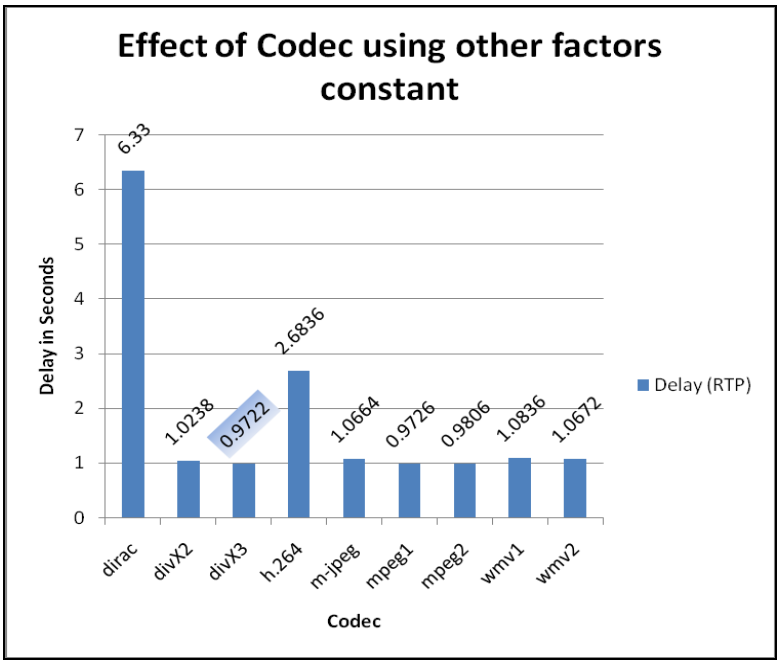

Figure 5-1 Bar Chart of Codec and Delay

Result of Experiment 1.1 (Confirming the Effect of Codec on Delay)

Furthermore, in Figure 5-2, a two lined chart is shown, which is the confirmation chart for the effect of codec on delay (experiment 1). In the chart one line shows the original and actual data of the experiment 1 while another line is the confirmation line obtained from the data of experiment 1.1 while considering Http instead of RTP as the underlying transmission protocol. Although there is a variation in data because of the change in protocols but the shapes of both the lines are same. This similarity in lines shows that codec have its effect on delay. But both lines are not exactly same because different codec have different performance on different protocol stack.

If we conclude the result based on the confirmation line it give almost same result with Dirac and h.264 with more delay and mpeg1, mpeg2 with less delay.

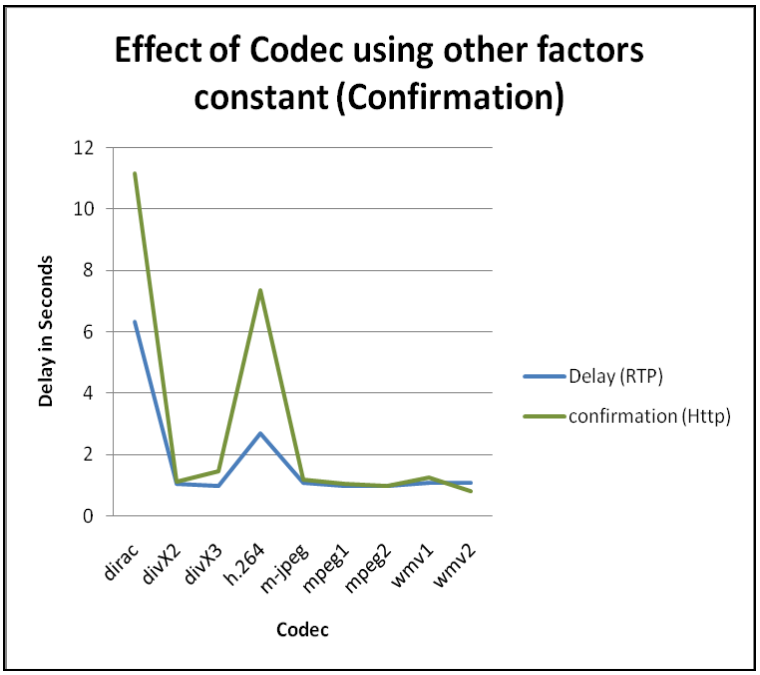

Figure 5-2 Two Lined Chart of Codec and Delay

5.2.2 Result of Experiment 2 (Effect of Encapsulation on Delay)

The results obtained using experiment 2 is shown in Figure 5-3, which presents the effect of encapsulation schemes on delay. It shows that the encapsulation schemes have also an impact on delay and some encapsulations have more impact than the others. In the selected encapsulation schemes avi and ps have greater impact as compared to ogg and asf, which have little impact on delay.

Result of Experiment 2.1 (Confirming the Effect of Encapsulation on Delay)

The Figure 5-4 shows two lined confirmation chart based on data of experiment 2 and 2.1. We can see that both the lines are following similar trend. The similarity in the shapes of the lines shows that the effect on delay due to the encapsulations. From the

confirmation chart it can also be concluded that ps has more impact on delay while ogg and asf has little impact on the delay.

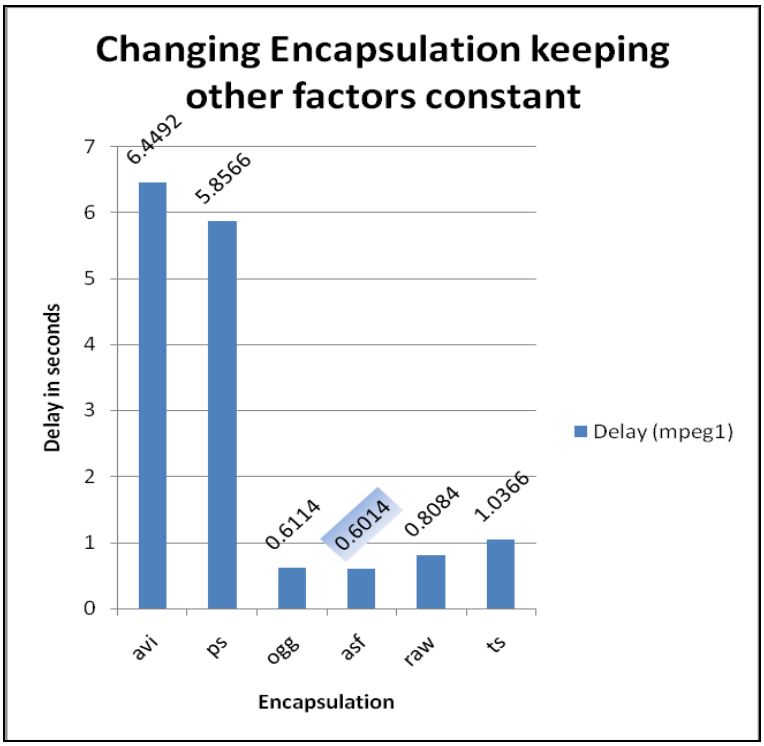

Figure 5-3 Bar Chart of Encapsulation and Delay

It should be noted that the confirmation line starts from PS encapsulation and not from avi. This is because mpeg2 codec had compatibility issues with avi.

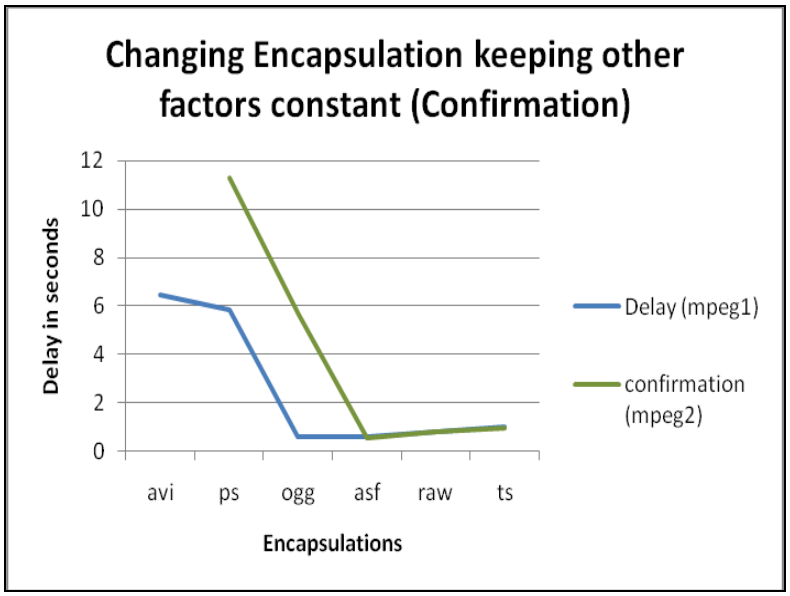

Figure 5-4 Two Lined Chart of Encapsulation and Delay

5.2.3 Result of Experiment 3 (Effect of Protocols on Delay)

The result of Experiment 3, outlined in section 4.2.3 is shown in Figure 5-5. By analyzing the bars in Figure 5-5, it can be easily concluded that RTP has minimum effect on delay. Furthermore, the maximum delay difference is about 100ms, while considering the performance of udp vs. rtp protocol.

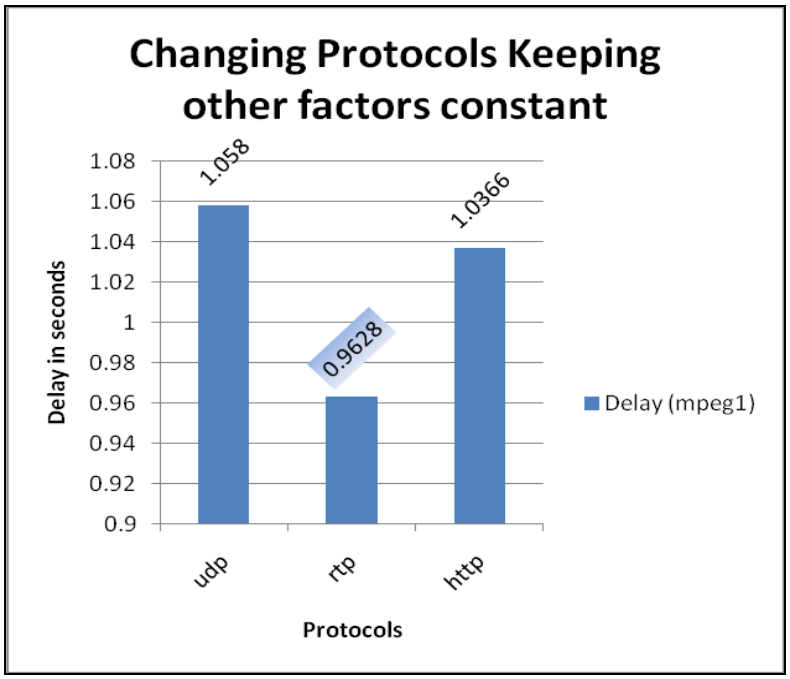

Figure 5-5 Bar Chart of Protocols and Delay

Result of Experiment 3.1 (Confirming the Effect of Protocols on Delay)

Figure 5-6 given below is the confirmation chart. The one, Delay(mpeg1), line is based on the data collected in experiment 3 having codec mpeg1 and the other line is based on the data collected in experiment 3.1 for confirmation while replacing mpeg2 instead of mpeg1 codec.

It can be clearly observe that both lines are similar with little difference. This also confirms the results of Figure 5-5, which shows that there is neglegible impact on delay, if we switch between mpeg1 and mpeg2. Additionally, Figure 5-5 also shows

that communication protocols, such as udp, rtp and http have also very little impact on delay, therefore the results provided in Figure 5-6 is an additional confirmation of our previous results.

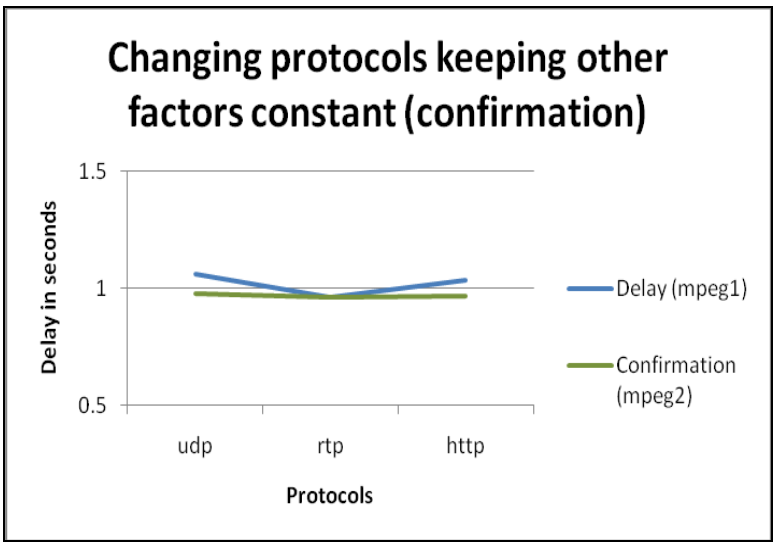

Figure 5-6 Two Lined Chart of Protocols and Delay

5.2.4 Result of Experiment 4 (Analyzing of Overall Stack on Delay)

The Figure 5-7 given below is based on experiment 4, detailed in section 4.2.4. From this bar chart we can conclude that the codec stack (http, mjpeg, mjpeg) has the minimum delay while the codec stack (mms, asf, wmv1) perform the worst. Hence, from the comparision of these stacks performance it can be concluded that (http, mjpeg, mjpeg) performs the best.

It was also noted that the qulaity of the mjpeg was also good as compared to the others.

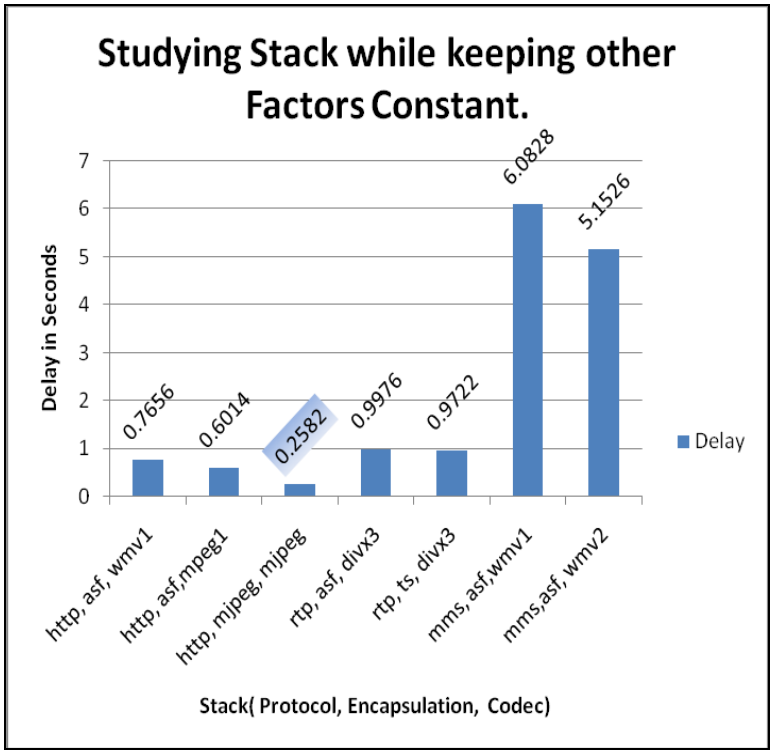

Figure 5-7 Bar Chart of Stack and Delay

5.3 Scenario Two Results

Using our proposed experimental setup in scenario two outlined in Chapter 3, a standard CIF video of duration 35 seconds was streamed. After completing the experiments and data collection, percent frame loss was calculated and the collected results were transformed to the bar charts and line charts for further analysis. These chats and their analysis are given below.

5.3.1 Result of Experiment 5 (Effect of Codec on Percent Frames Loss)

The result shown in Figure 5-8 is based on the data of experiment 5. It shows that keeping the underlying setup parameters the same, different codecs experience different percent frame loss. Some codecs have more impact of percent frame loss like Dirac and h.264 while some have a less impact like wmv1, mpeg2 and m-jpeg.

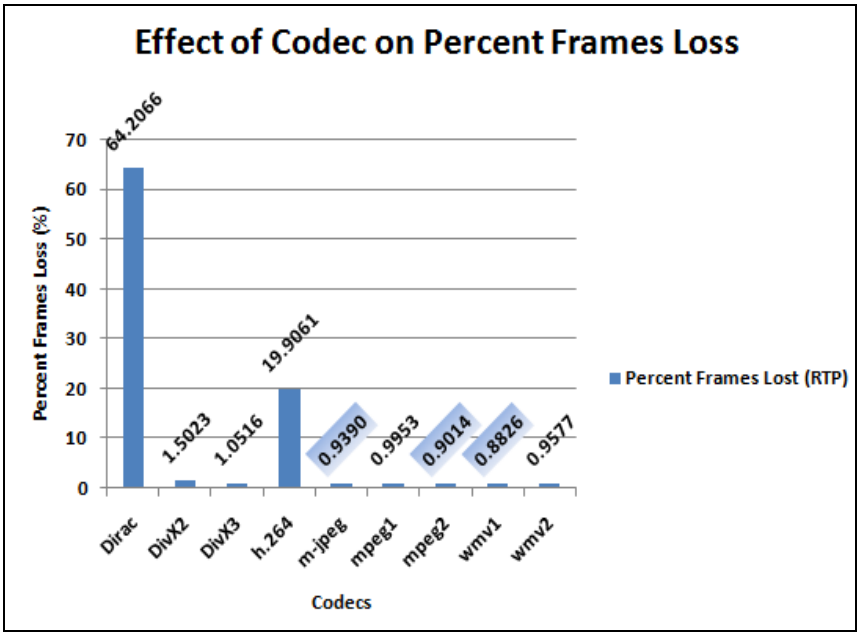

Figure 5-8 Bar Chart of Codec and Percent Frame Loss

Result of Experiment 5.1 (Confirming the Effect of Codec on Percent Frames Loss)

Furthermore, in Figure 5-9, a two lined chart is shown, which is the confirmation chart for the effect of codec on percent frame loss (experiment 5). In the chart one line shows the original and actual data of the experiment 5 while the second line is the confirmation line obtained from the data of experiment 5.1 while considering Http instead of RTP as the underlying transmission protocol. Although there is a variation in data because of the change in protocols but the shapes of both the lines are same. This similarity in lines shows that codec have its effect on percent frame loss. But both lines are not exactly same because different codec have different performance on different protocol stack.

If we conclude the result based on the confirmation line it give almost same result with Dirac and h.264 with more percent frame loss and wmv1, mpeg2 and m-jpeg with less percent frame loss.

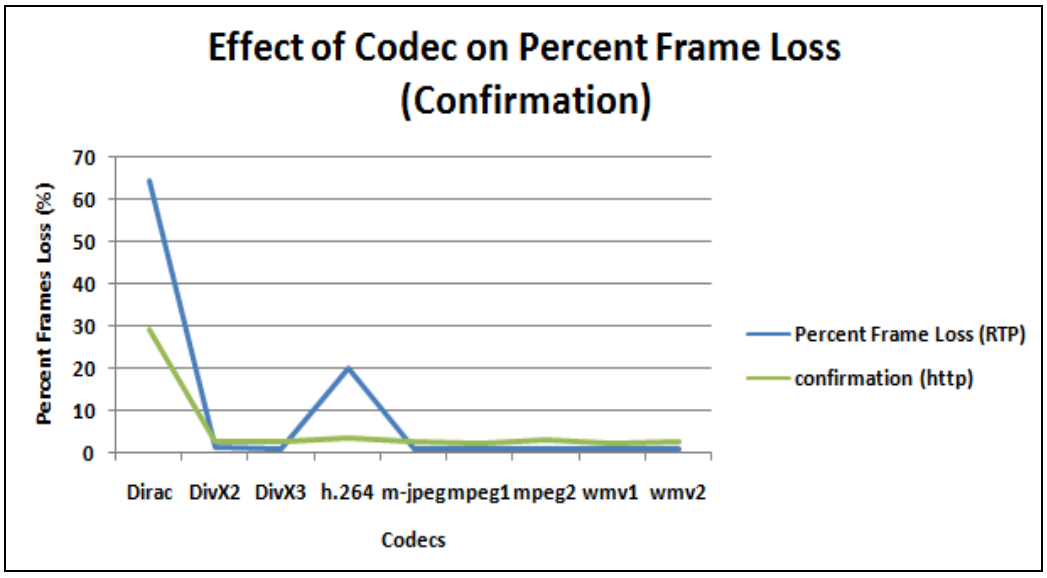

Figure 5-9 Two Lined Chart of Codec and Percent Frame Loss

5.3.2 Result of Experiment 6 (Effect of Encapsulation on Percent Frames Loss)

The result obtained using experiment 6 is shown in Figure 5-10, which presents the effect of encapsulation schemes on percent frame loss. It shows that the encapsulation schemes have also an impact on frame loss and some encapsulations have more impact than the others. In the selected encapsulation schemes avi has greater impact as compared to others, which have little and same impact on percent frame loss.

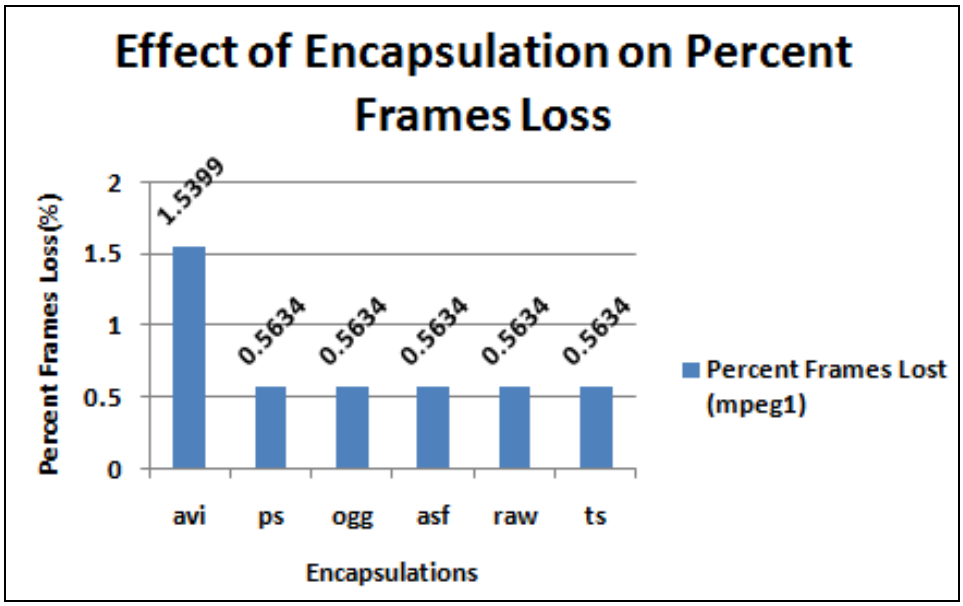

Figure 5-10 Bar Chart of Encapsulation and Percent Frame Loss

Result of Experiment 6.1 (Confirming the Effect of Encapsulation on Percent Frames Loss)

The Figure 5-11 shows two lined confirmation chart based on data of experiment 6 and 6.1. We can see that both the lines are following similar trend. Although there is variability in the shapes of the lines because of different codec parameters yet it follows the trend of the original line which confirms the effect on percent frame loss due to the encapsulations.

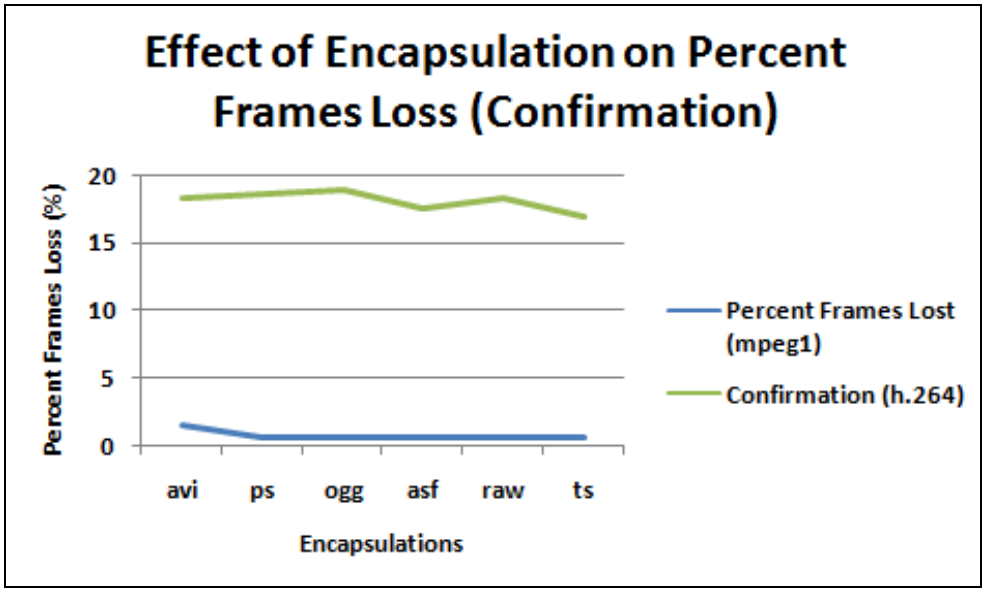

Figure 5-11 Two Lined Chart of Encapsulation and Percent Frame Loss

5.3.3 Result of Experiment 7 (Effect of Protocol on Percent Frames Loss)

The result of Experiment No 7, outlined in section 4.3.3 is shown in Figure 5-12. By analyzing the bars in Figure 5-12, it can be easily concluded that RTP has minimum effect on percent frame loss. Furthermore, the maximum percent frame loss difference is about 4 percent, while considering the performance of udp vs. rtp protocol.

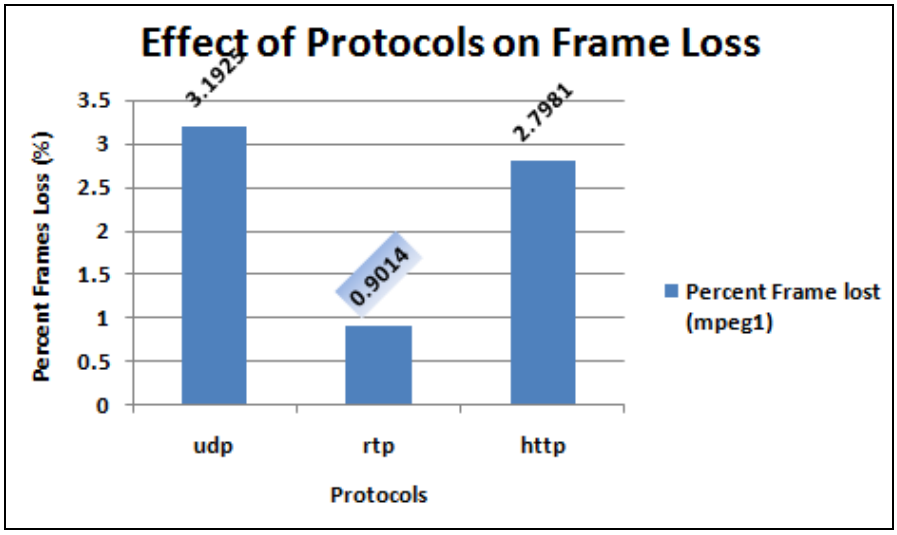

Fig. 20 Figure 5-12 Bar Chart of Protocols and Percent Frame Loss

Result of Experiment 7.1 (Confirming the Effect of Protocols on Percent Frames Loss)

Figure 5-13 given below is the confirmation chart, in which one line is based on the data collected in experiment 7 having codec mpeg1 and the other line is based on the data collected in experiment 7.1 for confirmation while replacing mpeg2 instead of mpeg1 codec.

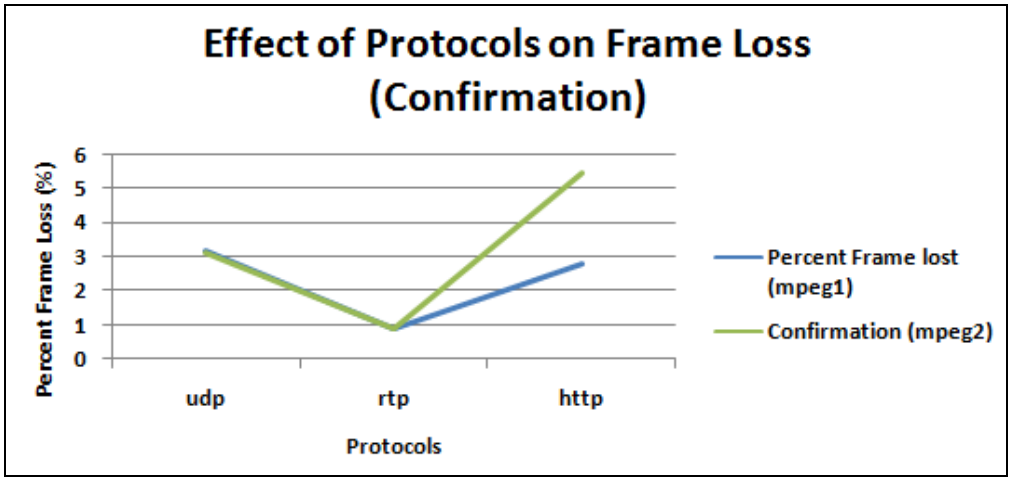

Figure 5-13 Two Lined Chart of Protocols and Percent Frame Loss

It can be clearly observe that both lines are similar with little difference. This also confirms the results of Figure 5-12, which shows that there is neglegible impact on

percent frame loss, if we switch between mpeg1 and mpeg2. Additionally, Figure 5-12 also shows that communication protocols, such as udp, rtp and http have also very little impact on percent frame loss, therefore the results provided in Figure 5-13 is an additional confirmation of our previous results.

5.4 Conclusion

This chapter explained the results obtained from the experiments performed on scenario one and two. It also explains the results of the confirmation experiments. Also the bar and lined charts are explained. The bar charts are obtained from the data of the original experiments performed while the lined charts are obtained from the data of original and conformational experiments performed. Actual collected data of both types of charts is given in appendix B. Moreover, these charts were also analyzed and explained in detail in this chapter.

Chapter 6

CONCLUSION AND FUTURE WORK

6.1 Introduction

This chapter summarizes the whole thesis report. At the beginning we defined the research questions. In pursue of research questions experimental setup was designed and different experiments were performed on them. Based on these experiments results were drawn to answer our research questions. This chapter also gives information about the future work.

6.2 Discussion

Live video streaming is affected by several factors among which two important are delay and frame loss. In our thesis we studied these factors with respect to other factors (Protocols, encapsulations and codec). Experiments were performed in a controlled environment to test the effect of protocols, encapsulations and codec on delay and frame loss while streaming a video. Our goal was to identify the delay and percentage of frame loss trends while systematically changing the combination of different considered codecs, encapsulation methods and protocols. From conducting experiments data was collected and transformed to bar charts for comparison and analysis. The comparison and analysis of the obtained charts are shown and explained chapter 5.

6.3 Conclusion

From the results of scenario one answer to the research question, “What is the effect

of protocol, encapsulation and codec on delay individually and collectively while streaming", can be easily concluded that codec divx3 have less impact on delay. Among the considered encapsulation methods, avi perform the worst and among the considered transmission protocols, rtp results in transmission of video data with least delay impact. Furthermore, in joint stack experiment it was observed that the stack (http, mjpeg, mjpeg) performed best as shown in Figure 5-7.

Similarly, from the results of scenario two answer to the research question, "What is the effect of protocol, Encapsulation and codec on frame drops while streaming", can be concluded that dirac have more effect on percent frame loss, hence, worst performance. Among the considered encapsulation methods, asf performs best. And among the considered transmission protocols, rtp results in transmission of video data with least frame loss.

6.4 Recommendations

After concluding the research thesis and answering the research questions, the work was summarized in table 3. The table 3 compares codecs, encapsulations, protocols and their best combinations (stack) suitable for different application areas. The codecs, encapsulation methods and protocols are recommended as high, medium and low, based on their performance in our research study. It should be noted that in table 3 H represents high, M represents medium and L represents Low where H, M and L represents the level of suitability for the considered purpose, like for achieving high user perceivable quality, h.264 is highly recommended. Similarly, for chatting where real time communication is more important than user perceivable quality, m-jpeg and m-jpeg is highly recommended. Moreover, protocol, rtp, and encapsulation, asf, is highly recommended in most of the cases while codec mpeg2 is moderately

recommended for most of the cases.

TABLE 3 COLLECTIVE COMPARISONS OF CODECS, ENCAPSULATIONS AND PROTOCOLS

	CODEC/ ENCAPSULATION/ PROTOCOL/ STACK	POPULARITY	EDUCATIONAL PURPOSE	CHAT PURPOSE	E-COMMERCE	REAL TIME COMMUNICATION	HIGH QUALITY
CODEC	dirac	L	M	L	H	L	H
	divx2	L	L	L	L	M	L
	divx3	M	L	M	L	M	L
	h.264	H	H	L	M	L	H
	m-jpeg	L	M	H	M	H	L
	mpeg1	L	L	M	L	M	L
	mpeg2	M	M	M	M	M	L
	wmv1	L	L	L	M	M	M
	wmv2	M	M	L	M	M	M
ENCAPSULATION SCHEMES	avi	H	L	L	M	L	H
	ps	L	L	L	M	L	L
	ogg	L	L	H	M	H	L
	asf	M	H	H	H	H	M
	ts	L	M	M	H	M	M
PROTOCOLS	http	H	M	M	H	M	M
	rtp	M	H	H	H	H	H
	udp	L	L	M	L	M	H
STACK	http, asf, wmv1	M	M	M	H	M	M
	http, asf, mpeg1	M	M	M	H	M	M
	http, mjpeg, mjpeg	L	M	H	M	H	L
	rtp, asf, divx3	M	L	M	L	M	L
	rtp, ts, divx3	M	L	M	L	M	L
	mms, asf, wmv1	H	M	L	M	L	M
	mms, asf, wmv2	H	M	L	M	L	M

6.5 Future Work

In future the scope of the experiments can be extended to audio codec as well. And also the combined effect of audio and video can be studied and analyzed as well, for

both the scenarios one and two. Similarly, the results of scenario one and two can be studied and analyzed mutually. Furthermore, more controlled transmission scenarios will be considered in which the performance of these different parameters will be considered using range of limited bandwidth. Additionally, subjective (e.g. mean opinion score) and objective quality evaluation (e.g. PSNR) procedures will be utilized to analyze their impact on video quality.

APPENDIX A: Detailed System Report

Intel(R) Graphics Media Accelerator Driver Report

Report Date:	11/22/2012
Report Time [hr:mm:ss]:	21:45:50
Driver Version:	6.14.10.4885
Operating System:	Windows XP* Professional, Service Pack 2 (5.1.2600)
Default Language:	English
DirectX* Version:	9.0
Physical Memory:	2035 MB
Minimum Graphics Memory:	8 MB
Maximum Graphics Memory:	128 MB
Graphics Memory in Use:	9 MB
Processor:	x86 family 6 Model 23 Stepping 6
Processor Speed:	2533 MHZ
Vendor ID:	8086
Device ID:	29C2
Device Revision:	10

* Accelerator Information *

Accelerator in Use:	Intel(R) G33/G31 Express Chipset Family
Video BIOS:	1508.0
Current Graphics Mode:	1024 by 768 True Color (75 Hz)

* Devices Connected to the Graphics Accelerator *

Active Monitors: 1

* Monitor *

Monitor Name:	Plug and Play Monitor	
Display Type:	Analog	
Gamma Value:	2.92	
DDC2 Protocol:	Supported	
Maximum Image Size:	Horizontal:	12.0 inches
	Vertical:	9.0 inches

Monitor Supported Modes:
640 by 400 (70 Hz), 640 by 480 (60 Hz), 640 by 480 (60 Hz), 640 by 480 (70 Hz), 640 by 480 (72 Hz), 640 by 480 (75 Hz), 640 by 480 (75 Hz), 640 by 480 (85 Hz), 800 by 600 (72 Hz), 800 by 600 (75 Hz), 800 by 600 (75 Hz), 800 by 600 (85 Hz), 1024 by 768 (70 Hz), 1024 by 768 (75 Hz), 1024 by 768 (85 Hz), 1280 by 1024 (60 Hz)

Display Power Management Support:

Standby Mode:	Supported
Suspend Mode:	Supported
Active Off Mode:	Supported

APPENDIX B: Data Obtained From Experiments

TABLE 4 DATA OF EXPERIMENT 1 AND 1.1

S No.	Codec	Experiment 1 (Effect of Codec on Delay)						Experiment 1.1 (Confirming the Effect of Codec on Delay)					
		Original	Received	Delay	Min	Max	average	Original	Received	Delay	min	max	average
1	Dirac	6.535	0.213	6.322	6.278	6.45	6.33	12.125	0.946	11.179	11.137	11.18	11.1544
		7.396	1.117	6.279				13.158	2.021	11.137			
		10.19	3.74	6.45				14.192	3.054	11.138			
		10.922	4.601	6.321				16.298	5.16	11.138			
		10.965	4.687	6.278				18.404	7.224	11.18			
2	DivX2	1.85	0.817	1.033	0.945	1.075	1.0238	4.689	3.57	1.119	1.118	1.161	1.1272
		2.064	1.119	0.945				5.333	4.215	1.118			
		3.225	2.15	1.075				22.104	20.985	1.119			
		3.914	2.88	1.034				23.437	22.318	1.119			
		6.666	5.634	1.032				22.963	21.802	1.161			
3	DivX3	1.505	0.558	0.947	0.947	0.989	0.9722	4.515	3.053	1.462	1.419	1.463	1.4538
		2.451	1.462	0.989				5.203	3.74	1.463			
		2.967	2.02	0.947				5.676	4.214	1.462			
		4.042	3.053	0.989				6.278	4.815	1.463			
		6.106	5.117	0.989				17.028	15.609	1.419			
4	h.264	2.882	0.215	2.667	2.665	2.711	2.6836	13.331	5.977	7.354	7.31	7.355	7.3452
		3.442	0.731	2.711				13.503	6.149	7.354			
		3.913	1.248	2.665				14.492	7.182	7.31			
		4.43	1.764	2.666				14.965	7.61	7.355			
		5.763	3.054	2.709				15.782	8.429	7.353			
5	m-jpeg	1.978	0.903	1.075	1.032	1.075	1.0664	1.506	0.301	1.205	1.16	1.205	1.1782
		2.107	1.032	1.075				1.893	0.688	1.205			
		2.753	1.678	1.075				2.968	1.807	1.161			
		3.569	2.494	1.075				4.257	3.097	1.16			
		4.73	3.698	1.032				10.363	9.203	1.16			
6	mpeg1	1.635	0.688	0.947	0.947	0.989	0.9726	4.214	3.139	1.075	0.989	1.075	1.0366
		1.981	1.032	0.949				5.247	4.258	0.989			
		2.108	1.119	0.989				8.384	7.353	1.031			
		3.054	2.065	0.989				9.631	8.557	1.074			
		4.989	4	0.989				11.481	10.467	1.014			
7	mpeg2	1.205	0.216	0.989	0.946	1.031	0.9806	2.151	1.265	0.886	0.886	1.033	0.9672
		1.894	0.947	0.947				3.184	2.151	1.033			
		3.312	2.366	0.946				5.118	4.173	0.945			
		4.301	3.27	1.031				5.979	5.032	0.947			
		6.065	5.075	0.99				6.881	5.856	1.025			
8	wmv1	1.463	0.388	1.075	1.075	1.118	1.0836	1.677	0.432	1.245	1.205	1.247	1.238
		2.581	1.506	1.075				2.15	0.904	1.246			
		2.839	1.764	1.075				3.139	1.934	1.205			
		3.656	2.538	1.118				4.558	3.311	1.247			
		8.085	7.01	1.075				6.279	5.032	1.247			
9	wmv2	1.205	0.129	1.076	1.034	1.076	1.0672	0.947	0.13	0.817	0.771	0.819	0.8086
		1.592	0.517	1.075				1.291	0.473	0.818			
		4.088	3.054	1.034				2.452	1.634	0.818			
		8.171	7.096	1.075				4.472	3.701	0.771			
		8.558	7.482	1.076				7.57	6.751	0.819			

TABLE 5 DATA OF EXPERIMENT 2 AND 2.1

S No	Encapsulation	Experiment 2 (Effect Of Encapsulation On Delay)						Experiment 2.1 (Confirming The Effect Of Encapsulation On Delay)					
		Original	Received	Delay	Min	max	average	Original	Received	Delay	min	max	average
1	TS	4.214	3.139	1.075	0.989	1.075	1.0366	2.151	1.265	0.886	0.886	1.033	0.9672
		5.247	4.258	0.989				3.184	2.151	1.033			
		8.384	7.353	1.031				5.118	4.173	0.945			
		9.631	8.557	1.074				5.979	5.032	0.947			
		11.481	10.467	1.014				6.881	5.856	1.025			
2	ASF	0.902	0.3	0.602	0.601	0.602	0.6014	1.763	1.204	0.559	0.518	0.6	0.559
		1.935	1.334	0.601				2.493	1.935	0.558			
		3.268	2.667	0.601				3.438	2.838	0.6			
		5.376	4.774	0.602				4.215	3.655	0.56			
		7.568	6.967	0.601				5.848	5.33	0.518			
3	AVI	6.997	0.213	6.784	5.933	6.784	6.4492	6.997	0.213	6.784	5.933	6.784	6.4492
		8.286	1.677	6.609				8.286	1.677	6.609			
		9.362	2.881	6.481				9.362	2.881	6.481			
		10.609	4.17	6.439				10.609	4.17	6.439			
		15.51	9.577	5.933				15.51	9.577	5.933			
4	OGG	1.464	0.818	0.646	0.602	0.646	0.6114	6.623	0.991	5.632	5.632	5.633	5.6322
		2.453	1.85	0.603				7.827	2.195	5.632			
		3.441	2.839	0.602				9.375	3.743	5.632			
		4.99	4.387	0.603				10.58	4.947	5.633			
		8.172	7.569	0.603				11.654	6.022	5.632			
5	PS	6.064	0.216	5.848	5.848	5.89	5.8566	12.255	0.946	11.309	11.309	11.31	11.3094
		7.224	1.334	5.89				13.116	1.806	11.31			
		8.686	2.838	5.848				15.05	3.741	11.309			
		10.363	4.515	5.848				15.824	4.515	11.309			
		11.955	6.106	5.849				17.33	6.02	11.31			
6	RAW	2.796	1.635	1.161	0.3	1.161	0.8084	1.763	1.159	0.604	0.604	0.945	0.7912
		3.355	2.366	0.989				3.354	2.58	0.774			
		4.516	4.043	0.473				5.546	4.601	0.945			
		6.452	5.333	1.119				8.599	7.826	0.773			
		8.385	8.085	0.3				10.965	10.105	0.86			

TABLE 6 DATA OF EXPERIMENT 3 AND 3.1

S No	Protocols	Experiment 3 (Effect Of Protocols On Delay)						Experiment 3.1 (Confirming The Effect Of Protocols On Delay)					
		Original	Received	Delay	Min	max	average	Original	Received	Delay	min	max	average
1	udp	1.332	0.301	1.031	1.031	1.119	1.058	1.291	0.3	0.991	0.936	1.031	0.978
		2.15	1.031	1.119				2.106	1.119	0.987			
		4.171	3.139	1.032				3.784	2.839	0.945			
		5.247	4.171	1.076				5.375	4.439	0.936			
		6.407	5.375	1.032				7.782	6.751	1.031			
2	rtp	1.291	0.346	0.945	0.945	0.989	0.9628	1.291	0.346	0.945	0.945	0.99	0.9628
		2.71	1.721	0.989				2.622	1.636	0.986			
		3.914	2.968	0.946				3.7	2.753	0.947			
		5.118	4.129	0.989				4.774	3.828	0.946			
		6.493	5.548	0.945				6.065	5.075	0.99			
3	http	4.214	3.139	1.075	0.989	1.075	1.0366	2.151	1.265	0.886	0.886	1.033	0.9672
		5.247	4.258	0.989				3.184	2.151	1.033			
		8.384	7.353	1.031				5.118	4.173	0.945			
		9.631	8.557	1.074				5.979	5.032	0.947			
		11.481	10.467	1.014				6.881	5.856	1.025			

TABLE 7 DATA OF EXPERIMENT 4

	Experiment 4. (Effect Of Stack On Delay)						
S. No	**Stack**	**Original**	**Received**	**Delay**	**min**	**max**	**average**
1	**Http, asf, wmv1**	1.29	0.517	0.773	0.689	0.861	0.7656
		1.979	1.118	0.861			
		4.86	4.128	0.732			
		5.203	4.43	0.773			
		6.15	5.461	0.689			
2	**Rtp, asf, divX3**	9.461	8.429	1.032	0.947	1.032	0.9976
		9.805	8.858	0.947			
		10.277	9.289	0.988			
		10.88	9.848	1.032			
		12.213	11.224	0.989			
3	**mms, asf, wmv1**	6.623	0.774	5.849	5.847	6.934	6.0828
		8.256	2.409	5.847			
		9.289	3.397	5.892			
		10.45	4.558	5.892			
		12.17	5.236	6.934			
4	**mms, asf, wmv2**	5.117	0.259	4.858	4.858	6.03	5.1526
		11.137	5.107	6.03			
		11.911	7.01	4.901			
		13.502	8.515	4.987			
		16.512	11.525	4.987			
5	**http, mjpeg, mjpeg**	0.432	0.173	0.259	0.258	0.259	0.2582
		1.506	1.248	0.258			
		2.538	2.28	0.258			
		3.699	3.441	0.258			
		6.537	6.279	0.258			
6	**Rtp, TS, DivX3**	1.505	0.558	0.947	0.947	0.989	0.9722
		2.451	1.462	0.989			
		2.967	2.02	0.947			
		4.042	3.053	0.989			
		6.106	5.117	0.989			
7	**http, asf, mpeg1**	0.902	0.3	0.602	0.601	0.602	0.6014
		1.935	1.334	0.601			
		3.268	2.667	0.601			
		5.376	4.774	0.602			
		7.568	6.967	0.601			

TABLE 8 DATA OF EXPERIMENT 5 AND 5.1

S No.	Codec	Experiment 5 (Effect of Codec on Percent Frame Loss)					Experiment 5.1 (Confirming Effect of Codec on Percent Frame Loss)				
		Total Frames	Displayed	Actual-Displayed	percent loss	Average	Total Frames	Displayed	Actual-Displayed	Percent loss	Average
1	Dirac	1065	337	728	68.3568	64.2066	1065	750	315	29.5775	29.2770
		1065	420	645	60.5634		1065	758	307	28.8263	
		1065	409	656	61.5962		1065	750	315	29.5775	
		1065	368	697	65.4460		1065	752	313	29.3897	
		1065	372	693	65.0704		1065	756	309	29.0141	
2	DivX2	1065	1039	26	2.4413	1.5023	1065	1038	27	2.5352	2.5352
		1065	1056	9	0.8451		1065	1033	32	3.0047	
		1065	1037	28	2.6291		1065	1034	31	2.9108	
		1065	1056	9	0.8451		1065	1033	32	3.0047	
		1065	1057	8	0.7512		1065	1052	13	1.2207	
3	DivX3	1065	1040	25	2.3474	1.0516	1065	1036	29	2.7230	2.4789
		1065	1057	8	0.7512		1065	1035	30	2.8169	
		1065	1057	8	0.7512		1065	1035	30	2.8169	
		1065	1057	8	0.7512		1065	1035	30	2.8169	
		1065	1058	7	0.6573		1065	1052	13	1.2207	
4	h.264	1065	827	238	22.3474	19.9061	1065	1025	40	3.7559	3.5493
		1065	837	228	21.4085		1065	1036	29	2.7230	
		1065	873	192	18.0282		1065	1025	40	3.7559	
		1065	865	200	18.7793		1065	1025	40	3.7559	
		1065	863	202	18.9671		1065	1025	40	3.7559	
5	m-jpeg	1065	1041	24	2.2535	0.9390	1065	1030	35	3.2864	2.5540
		1065	1058	7	0.6573		1065	1036	29	2.7230	
		1065	1058	7	0.6573		1065	1035	30	2.8169	
		1065	1058	7	0.6573		1065	1035	30	2.8169	
		1065	1060	5	0.4695		1065	1053	12	1.1268	
6	mpeg1	1065	1036	29	2.7230	0.9953	1065	1042	23	2.1596	2.2535
		1065	1059	6	0.5634		1065	1036	29	2.7230	
		1065	1059	6	0.5634		1065	1036	29	2.7230	
		1065	1059	6	0.5634		1065	1036	29	2.7230	
		1065	1059	6	0.5634		1065	1055	10	0.9390	
7	mpeg2	1065	1041	24	2.2535	0.9014	1065	1024	41	3.8498	2.8357
		1065	1059	6	0.5634		1065	1041	24	2.2535	
		1065	1059	6	0.5634		1065	1030	35	3.2864	
		1065	1059	6	0.5634		1065	1030	35	3.2864	
		1065	1059	6	0.5634		1065	1049	16	1.5023	
8	wmv1	1065	1042	23	2.1596	0.8826	1065	1034	31	2.9108	2.1596
		1065	1059	6	0.5634		1065	1034	31	2.9108	
		1065	1059	6	0.5634		1065	1044	21	1.9718	
		1065	1059	6	0.5634		1065	1044	21	1.9718	
		1065	1059	6	0.5634		1065	1054	11	1.0329	
9	wmv2	1065	1041	24	2.2535	0.9577	1065	1040	25	2.3474	2.4601
		1065	1059	6	0.5634		1065	1039	26	2.4413	
		1065	1058	7	0.6573		1065	1040	25	2.3474	
		1065	1058	7	0.6573		1065	1035	30	2.8169	
		1065	1058	7	0.6573		1065	1040	25	2.3474	

TABLE 9 DATA OF EXPERIMENT 6 AND 6.1

S No	Encapsulations	Experiment 6 (Effect of Encapsulation on Percent Frame Loss)					Experiment 6.1 (Confirming Effect of Encapsulation on Percent Frame Loss)				
		Total Frames	Displayed	Actual-Displayed	Percent loss	Average	Total Frames	Displayed	Actual-Displayed	percent loss	Average
1	avi	1065	1007	58	5.4460	1.5399	1065	882	183	17.1831	18.2535
		1065	1059	6	0.5634		1065	867	198	18.5915	
		1065	1059	6	0.5634		1065	870	195	18.3099	
		1065	1059	6	0.5634		1065	868	197	18.4977	
		1065	1059	6	0.5634		1065	866	199	18.6854	
2	ps	1065	1059	6	0.5634	0.5634	1065	868	197	18.4977	18.6103
		1065	1059	6	0.5634		1065	867	198	18.5915	
		1065	1059	6	0.5634		1065	875	190	17.8404	
		1065	1059	6	0.5634		1065	864	201	18.8732	
		1065	1059	6	0.5634		1065	860	205	19.2488	
3	ogg	1065	1059	6	0.5634	0.5634	1065	856	209	19.6244	18.9296
		1065	1059	6	0.5634		1065	867	198	18.5915	
		1065	1059	6	0.5634		1065	859	206	19.3427	
		1065	1059	6	0.5634		1065	868	197	18.4977	
		1065	1059	6	0.5634		1065	867	198	18.5915	
4	asf	1065	1059	6	0.5634	0.5634	1065	955	110	10.3286	17.5587
		1065	1059	6	0.5634		1065	862	203	19.0610	
		1065	1059	6	0.5634		1065	856	209	19.6244	
		1065	1059	6	0.5634		1065	857	208	19.5305	
		1065	1059	6	0.5634		1065	860	205	19.2488	
5	raw	1065	1059	6	0.5634	0.5634	1065	863	202	18.9671	18.3662
		1065	1059	6	0.5634		1065	867	198	18.5915	
		1065	1059	6	0.5634		1065	884	181	16.9953	
		1065	1059	6	0.5634		1065	864	201	18.8732	
		1065	1059	6	0.5634		1065	869	196	18.4038	
6	ts	1065	1059	6	0.5634	0.5634	1065	863	202	18.9671	16.9577
		1065	1059	6	0.5634		1065	960	105	9.8592	
		1065	1059	6	0.5634		1065	858	207	19.4366	
		1065	1059	6	0.5634		1065	876	189	17.7465	
		1065	1059	6	0.5634		1065	865	200	18.7793	

TABLE 10 DATA OF EXPERIMENT 7 AND 7.1

S No	Protocols	Experiment 7 (Effect of Protocols on Percent Frame Loss)					Experiment 7.1 (Confirming Effect of Protocols on Percent Frame Loss)				
		Total Frames	Displayed	Actual-Displayed	percent loss	Average	Total Frames	Displayed	Actual-Displayed	percent loss	Average
1	udp	1065	1035	30	2.8169	3.1925	1065	1040	25	2.3474	3.1362
		1065	1022	43	4.0376		1065	1035	30	2.8169	
		1065	1037	28	2.6291		1065	1026	39	3.6620	
		1065	1035	30	2.8169		1065	1035	30	2.8169	
		1065	1026	39	3.6620		1065	1022	43	4.0376	
2	rtp	1065	1041	24	2.2535	0.9014	1065	1041	24	2.2535	0.9014
		1065	1059	6	0.5634		1065	1059	6	0.5634	
		1065	1059	6	0.5634		1065	1059	6	0.5634	
		1065	1059	6	0.5634		1065	1059	6	0.5634	
		1065	1059	6	0.5634		1065	1059	6	0.5634	
3	http	1065	1038	27	2.5352	2.7981	1065	838	227	21.3146	5.4460
		1065	1036	29	2.7230		1065	1049	16	1.5023	
		1065	1030	35	3.2864		1065	1054	11	1.0329	
		1065	1036	29	2.7230		1065	1046	19	1.7840	
		1065	1036	29	2.7230		1065	1048	17	1.5962	

BIBLIOGRAPHY

[1]. Xiaoling Qiu, Haiping Liu, Deshi Li, Song Zhang, Dipak Ghosal, Biswanath Mukerjee, "*Optimizing Http-based Adaptive Video Streaming For Wireless Access Networks*", proceedings of IC-BNMT 2010, IEEE.

[2]. Huong BUI Thi Lan, Hoai Son NGUYEN, "*A Low-delay Push-pull based Application Layer Multicast for P2P Live Video Streaming"*, Third International Conference on Knowledge and Systems Engineering, 2011 IEEE.

[3]. Chunlei Liu, "*Multimedia over IP: RSVP, RTP, RTCP, RTSP",* July, 2000.

[4]. D. Wu, Y. Hou W. Zhu, Y.-Q. Zhang, J. Peha "*Streaming video over the internet: approaches and directions*," IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, pp. 282–300, Mar. 2001.

[5]. J. Postal, Universal Datagram Protocol (UDP), RFC 768. (Online). Avaliable: http://www.ietf.org/rfc/rfc768.txt

[6]. California Software Labs (CSWL), "*Basic Streaming Technology and RTSP Protocol*".

[7]. Civanlar, M.R. Luthra, A. ; Wenger, S. ; Wenwu Zhu ; "*Introduction to the Special Issue on Streaming Video*", IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, No. 3, Mar. 2001.

[8]. Venkataraman, M.; Chatterjee, M.; Chattopadhyay, S., "*Evaluating Quality of Experience for Streaming Video in Real Time*", Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE, Publication Year: 2009, Page(s): 1 – 6.

[9]. Online Stopwatch. (2012). [Online]. Available: http://stopwatch.onlineclock.net/.

[10]. TechSmith, Camtasia Studio, v 2.0.1. (Oct, 2012). [Online]. Available: http://www.techsmith.com/camtasia.html.

[11]. Lexis de Lattre, Johan Bilien, Anil Daoud, Clément Stenac, Antoine Cellerier, Jean-Paul Saman, "*Documentation:Streaming HowTo"*. (2012). [Online]. Available: http://wiki.videolan.org/Documentation:Streaming_HowTo.

[12]. Minnesota Historical Society / State Archives, "*NDIIPP Digital Audio and Video White Paper*", Version 1.0, May 2009.

[13]. Zhi-Gang Li, Zhao-Yang ZHANG, "*Real-Time Streaming and Robust Streaming H.264/AVC video*", Third International Conference on Image and Graphics, 2004 IEEE.

[14]. Ahtsham Ali, Syed Farooq Ali, Nadeem Khan and Shahid Masud, "*Performance Improvement in Motion Estimation of Dirac Wavelet based Video Codec*", ISCIT 2009 IEEE.

[15]. Sinzobakwira Issa, Othman Omar Khalifa, "*Performance analysis of Dirac video codec with H.264/AVC*", ICCCE May, 2010 IEEE.

[16]. K. Onthriar, K.K. Loo, Z. Xue, "*Performance Comparison of Emerging Dirac Video Codec with H.264/AVC*", IEEE, Aug, 2006

[17]. M. Tun, K. K. Loo, J. Cosmas, "*A Novel Rate Control Algorithm for the Dirac Video Codec Based upon the Quality Factor Optimization*", IEEE 2007

[18]. Deniz Özenli , Melih Pazarcı, "*Performance Analysis of Dirac Video Codec in Different Motion Vector Accuracies and Wavelet Lifting Decompositions*", 53rd International Symposium ELMAR-2011, 14-16 September 2011, Zadar, Croatia.

[19]. Hisham Sliman, Mohamed El-Sharkawy, Paul Salama, Maher Rizkalla and Salwa El- Ramly, "*All-Zero Block Detection in VC-1*", 2009 IEEE.

[20]. Shih-Yu Huang and Jia-Shung Wang, "*A low-cost desktop videoconferencing codec: An adaptive Motion-Jpeg Design*", IEEE Vol. 40, No.4, Nov 1994

[21]. Lei Chen, Narasimha Shashidhar, Qingzhong Liu, "*Scalable Secure MJPEG Video Streaming*", IEEE 2012

[22]. E A Jammeh, M Ghanbari, "*Smoothing Transcoded MPEG-1 Video for Efficient Transmission Over Best-Effort IP Networks*", 2003 The Institution of Electrical Engineers.

[23]. O J Morris, "*Mpeg-2: Where did it come from and what is it?*", IEEE 24 Jan 1995.

[24]. Wang Xing Guo, Zheng Wei Guo, Ishfaq Ahmad, "*Mpeg-2 to Mpeg-4 Transcoding*", IEEE 2001.

[25]. DivX LLC (Oct, 2012). [Online] Available: http://www.divx.com/en/node/1001.

[26]. Jay Loomis, Mike Wasson, "*VC-1 Technical Overview*", Microsoft Corporation, Oct 2007. Web link, http://www.microsoft.com/windows/windowsmedia/howto/articles/vc1techoverview.aspx

[27]. S. Pfeiffer, Xiph.Org Foundation, "*Rfc3533: The Ogg Encapsulation Format Version 0*", May 2003, [Online]. Available: http://www.xiph.org/ogg/doc/rfc3533.txt.

[28]. Fadi Almasalha, Ashfaq Khokhar and Shahab Baqai, "*Selective Encryption based Data Security for Ogg Streams*", ICASSP 2010 IEEE.

[29]. Qing Wei, Yongqiang Zhang, Mohua Zhang, Mingyi Cui, "*Study on Synchronization for Streaming Media Based on ASF Format for E-Learning*", 2010 International Conference on Web Information Systems and Mining, IEEE.

[30]. Microsoft Corporation, "*Advance Systems Format (ASF)*", Revision 01.20.03, Dec 2004.

[31]. Cristiano Akamine, Yuzo Iano, Gustavo de Melo Valeira and Gunnar Bedicks, "*Re-Multiplexing ISDB-T BTS Into DVB TS for SFN*", IEEE Transaction on Broadcasting, Vol. 55, No. 4, Dec 2009.

[32]. Tim Berners-Lee, R. Fielding, H. Frystyk, "*Hypertext Transfer Protocol -HTTP/1.0*", RFC 1945, May, 1996.

[33]. H. Schulzrinne, R. Frederick, V. Jacobson, "*RTP: A Transport Protocol for Real-Time Applications"*, IETF RFC 3550, July 2003.

[34]. Schulzrinne, H., et aI, Real Time Streaming Protocol (RTSP), RCF 2326.

[35]. Hypertext Transfer Protocol (HTTP) (Oct, 2012) [Online]. Available: http://www.livinginternet.com/w/ww_http.htm.

[36]. Iraj Sodagar, *"MPEG-DASH: the Standard for Multimedia Streaming over Internet*", Preprint of the article published in IEEE Multimedia, Oct-Nov 2011.

[37]. Microsoft Corporation, "*MS-MMSP: Microsoft Media Server (MMS) Protocol*", Oct 17, 2012

[38]. Vivek Kumar Singh, Suhas Hegde Ankadi and Debabrata Das, *"Study of Combined Effects of Zero Copy and Pre-processing on Input/Output Performance of Multimedia on Demand Streaming Server"*, 2011, IEEE.

[39]. Camstudio open source (Oct, 2012). Free Streaming Video Software. [Online]. Available: http://camstudio.org/.

[40]. Manoj Bhatia, Jonathan Davidson, Satish Kalidindi, Sudipto Mukherjee, James Peters, "*VoIP: An In-Depth Analysis*", (Nov, 2012). [Online]. Available: http://www.ciscopress.com/articles/article.asp?p=606583.

[41]. Tom Sheldon's Linktionary.com. (Sept, 2012). Delay, Latency, and Jitter. [Online]. Available: http://www.linktionary.com/d/delay.html.

[42]. wikiHow to do anything. (Oct, 2012). How to Use Vlc to Stream Audio and Video to Multiple Computers on Your Network Using Multicast. [Online]. Available: http://www.wikihow.com/Use-Vlc-to-Stream-Audio-and-Video-to-Multiple-Computers-on-Your-Network-Using-Multicast.

[43]. wikiHow to do anything. (Oct, 2012). How to Use VLC Media Player to Stream Multimedia to Another Computer. [Online]. Available: http://www.wikihow.com/Use-VLC-Media-Player-to-Stream-Multimedia-to-Another-Computer.

[44]. SPINETIX. (Oct, 2012). Tutorial: Streaming using VLC. [Online]. Available: http://support.spinetix.com/wiki/Tutorial:Streaming_using_VLC

[45]. Paris (Nov, 2012) xiph.org. [Online]. Available: http://media.xiph.org/video/derf/

[46]. (Nov, 2012) Copyright PictureTel Corporation. [Online]. Available: http://media.xiph.org/video/derf/ftp3.itu.int/video-site/sequences/Paris/README_paris.