

PERFORMANCE ENHANCEMENT OF REAL-TIME PROTOCOL

**A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of**

MASTER OF SCIENCE

in Computer Engineering

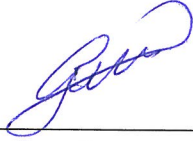
**by
Çağan Selçuk YÜCEL**

**September 2010
İZMİR**

We approve the thesis of **Çağın Selçuk YÜCEL**

Asst. Prof. Dr. Tolga AYAV
Supervisor

Asst. Prof. Dr. Tuğkan TUĞLULAR
Committee Member



Asst. Prof. Dr. Özgür GÜMÜŞ
Committee Member

21 September 2010

Prof. Dr. Sıtkı AYTAÇ
Head of the Department of
Computer Engineering

Assoc. Prof. Dr. Talat YALÇIN
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGEMENTS

Foremost, I would like to thank my advisor, Asst. Prof., Ph.D. Tolga Ayav, for his guidance, patience, and encouragement. Furthermore, I would like to thank my brother, Çağatay Yücel for his support. I would like to thank my parents for their support throughout my education as well as in my graduate study. Finally, I would like to thank dedicate this study to my recently deceased father, Dr. Hilmi Yücel. He was and he is always going to be my guide through any obstacles.

ABSTRACT

PERFORMANCE ENHANCEMENT OF REAL TIME TRANSFER PROTOCOL

A mechanism is described for dynamic adjustment of the performance requirements of multimedia applications. The sending application uses RTP receiver reports to compute packet loss and control mechanism periodically measures the utilization. Based on these and some other metrics the control mechanism can make necessary adjustments. Several experiments have been run in order to tune and evaluate the mechanism. The results indicate that the mechanism can be applied efficiently and the performance of the RTP can be increased.

ÖZET

GERÇEK ZAMANLI PROTOKOLÜN PERFORMANSININ YÜKSELTİLMESİ

Çoklu ortamlı uygulamaların performans ihtiyaçları için dinamik ayarlama sağlayan bir mekanizma tanımlanmıştır. Kontrol mekanizmasında verim ölçümü, yollayan uygulamanın, RTP alıcı raporlarından kayıp paket hesabı yapması ile sağlanır. Bu ölçümlere dayanarak, kontrol mekanizması gerekli ayarlamaları yapabilmektedir. Kontrol mekanizmasını değerlendirmek ve ayarlamak için çeşitli deneyler yapılmıştır. Bu deney sonuçlarında görülmüştür ki; kontrol mekanizması uygulandığında RTP transferinde verim artımı sağlanabilmektedir.

TABLE OF CONTENTS

LIST OF FIGURES	<i>ix</i>
LIST OF TABLES	x
CHAPTER 1. INTRODUCTION.....	1
1.1. Motivation	2
1.2. The Thesis.....	2
1.3. Organization of the Thesis.....	3
CHAPTER 2. REQUIREMENTS OF MULTIMEDIA APPLICATIONS.....	4
2.1. Characteristics of Multimedia Systems	4
2.1.1. Multimedia Data Streams	4
2.1.2. Multimedia Application Categories	5
2.1.3. Resource Demands of Multimedia Applications	6
2.1.4. Quality of Service	7
2.2. Process Management	9
2.2.1. Requirements	9
2.2.2. Scheduling Strategies.....	10
2.3. Multimedia Data Transmission	10
2.3.1. Multi-Point Communication	11
2.3.2. Quality of Service Control.....	12
2.3.3. Flow Control.....	12
CHAPTER 3. RELATED WORK.....	14
CHAPTER 4. NETWORK PROTOCOLS OVERVIEW	15
4.1. Why Can't We Just Use TCP for Audio and Video?	15
4.2. Why Can't We Just Use UDP for Audio and Video?.....	16
CHAPTER 5. THE REAL-TIME TRANSPORT PROTOCOL	18

5.1. RTP Use Scenarios.....	20
5.1.1. Simple Multicast Audio Conference	20
5.1.2. Audio and Video Conference	21
5.1.3. Mixers and Translators	22
5.2. RTP.....	23
5.2.1. Definitions	23
5.2.2. RTP Data Packet.....	25
5.2.3. RTP Header.....	25
The RTP header has the following format:.....	25
5.3. RTCP	28
5.3.1. Packet Formats.....	30
5.3.2. RTCP Transmission Interval	32
5.3.3. RTCP Packet Send & Receive Rules.....	36
5.3.4. Sender & Receiver Reports.....	38
5.3.5. RR: Receiver Report RTCP Packet	42
 CHAPTER 6. PERFORMANCE PARAMETERS	 43
6.1. Performance Parameters for Wireless and Wired LAN	43
6.1.1. Throughput.....	43
6.1.2. Integrity & Reliability	44
6.2. Performance Parameters for Streaming Media.....	44
6.2.1. Delay	44
6.2.2. Jitter	46
6.2.3. Packet Loss Rate	47
 CHAPTER 7. A CONTROL MECHANISM OVER RTP.....	 48
7.1. How to Enhance Performance?	48
7.2. QoS Metrics over RTP	49
7.3. Control Mechanism over RTP	52
7.3.1. Controller	53
7.3.2. Sub-controller	56
7.3.3. Traffic Generator	56
7.4. Protocol.....	57

CHAPTER 8. EXPERIMENT AND THE RESULTS	59
8.1. Experimental Steps	60
8.2. Exploring the Results	61
8.2.1. Result Set 1:	62
8.2.2. Result Set 2:	64
8.2.3. Result Set 3:	66
 CHAPTER 9. CONCLUSION	 68
 REFERENCES	 69

LIST OF FIGURES

Figure 5.1. RTP in relation to network and transport protocols	20
Figure 5.2. The RTP header.....	25
Figure 5.3. RTCP compound packet.....	32
Figure 5.4. RTCP sender report	37
Figure 5.5. RTCP receiver report.....	42
Figure 6.1. Throughput is the largest problem for wireless LAN users	44
Figure 7.1. RTP diagram.....	51
Figure 7.2. Control mechanism over RTP	52
Figure 7.3. PID control	54
Figure 7.4. Controller user interface.....	55
Figure 7.5. Controller user interface, transfer parameters settings panel	56
Figure 7.6. Controller user interface, traffic generator settings panel	57
Figure 8.1. Efficiency vs. Bandwidth	62
Figure 8.2. Efficiency vs. Time	63
Figure 8.4. Efficiency vs. Time, bandwidth is set to 5000 b/sec.	64
Figure 8.5. Efficiency vs. Time, bandwidth is set to 12500 b/sec.	65
Figure 8.6. Efficiency vs. Time, bandwidth is set to 28000 b/sec.	65
Figure 8.7. Efficiency vs. Time, bandwidth is set dynamically by the mechanism.	66
Figure 8.7. Bandwidth vs. Time, bandwidth is set dynamically by the mechanism.	67

LIST OF TABLES

Table 2.1. Requirements of different multimedia application classes.....	8
Table 7.1. Network QoS parameters.....	49

CHAPTER 1

INTRODUCTION

In recent years the Real-time Transport Protocol, RTP [1], has become the protocol of choice for audio/video transport in the Internet. One of the reasons for this success is the flexibility of RTP, which was designed as a protocol framework, rather than a monolithic protocol, allowing it to be tailored to different applications and media formats.

This flexibility has led a number of authors to consider the use of RTP for other scenarios, not necessarily related to audio/video transport. For example, it has recently been suggested that RTP may provide the base for a protocol for the transport of interactive media such as shared whiteboards [2].

Real-time media streams that use RTP are, to some degree, resilient against packet losses. RTP [3] provides all the necessary mechanisms to restore ordering and timing present at the sender to properly reproduce a media stream at a recipient. RTP also provides continuous feedback about the overall reception quality from all receivers -- thereby allowing the sender(s) in the mid-term (in the order of several seconds to minutes) to adapt their coding scheme and transmission behavior to the observed network quality of service (QoS). However, except for a few payload-specific mechanisms [4], RTP makes no provision for timely feedback that would allow a sender to repair the media stream immediately.

This document specifies a modified RTP feedback control mechanism for audio and video conferences, by means of two modifications/additions: Firstly, to achieve timely feedback, the concept of feedback channel is introduced. Secondly, a small number of general-purpose feedback messages are introduced.

1.1. Motivation

The explosive growth of the Internet and mobile computing introduces two main problems in multimedia applications. The first problem is heterogeneity of client devices and their network connections. The client devices may vary from desktop PCs, notebook computers, PDAs to mobile phones, which their capabilities also vary along many axes, including screen size, color depth and processing power [5]. Furthermore, they may connect to the Internet via different networks, such as wired LAN, wireless LAN or wireless WAN.

The second problem is mobility of clients. The clients may be moving while they are accessing multimedia streams. It may cause a problem because the network connections may change from time to time, ranging from a very good network to a congested network.

Both these problems cause poor QoS of the RTP. By these cases and variable parameters such as, bandwidth, packet loss etc., it is essential to have an enhancement in performance of the protocol.

1.2. The Thesis

The two problems described above make it difficult for a multimedia server to provide a streaming service which is appropriate for every client in every situation. A solution approach to the problems above which is presented in this thesis, to have a mechanism working to monitor the real-time traffic and report the QoS periodically. As a result of reporting, some manipulations and adjustments can be performed with the mechanism working parallel to the real-time protocol.

The purpose of this thesis is to develop an approach for performance increase of a network service infrastructure for transferring real-time multimedia streams. The approach allows a client on a network to have optimized multimedia stream from the provider. The service infrastructure should be able to maintain statically appropriate parameters for the network.

The prototype proposed in this thesis is designed for a LAN only. It needs some minor modifications to be applied for larger areas, such as the Internet. Scalability is an important issue to be considered when applying the infrastructure to

the Internet because the number of users may be in order of millions.

1.3. Organization of the Thesis

In general, this thesis consists of three main parts, background information and motivation, main approach to achieve thesis goals and results.

Chapter 1 (this chapter), Introduction, discusses the background of this thesis and gives some application scenarios in which the result of this thesis may be applied.

Chapter 2, Streaming Media, discusses some important aspects in streaming multimedia systems which have relations with this thesis, including the terminology and applications of streaming multimedia systems, networking and network protocols for streaming multimedia systems.

Chapter 3, Related Work, discusses and introduces some similar works that have been achieved considerable.

Chapter 4, Network Protocols Overview, reviews the background of the general network structures that are used today, gives examples and comparisons in use of multimedia streaming.

Chapter 5, RTP - The Real-Time Transport Protocol, explains the architecture of the mainly used existing protocol for multimedia streaming in details.

Chapter 6, Performance Parameters, introduces the parameters that are examined in order to have performance metrics and increase in terms of QoS parameters.

Chapter 7, The Work Done, explains the analysis and implementations in order to achieve the thesis goals. An introduction to the protocol and samples for implementation are presented.

Chapter 8, The Results, discusses the implementation of the RTP control mechanism infrastructure in Java platform. It explains how each component of the infrastructure can be implemented in Java technology and classes. As a result of the thesis study, accomplishments will be presented.

Chapter 9, Conclusion, gives outcome of this thesis and some outlooks in the future.

CHAPTER 2

REQUIREMENTS OF MULTIMEDIA APPLICATIONS

2.1. Characteristics of Multimedia Systems

Streaming media is highly delaying sensitive. A delay of more than a few hundred milliseconds makes many streaming media applications, e.g. VOIP, unusable. On the other hand, streaming media application is loss-tolerant. Packet loss only causes a short glitch in the audio/video playback. These losses can even be recovered through several techniques.

While streaming media are generally carried over UDP for the timely delivery that it provides and its support for multicasting, just UDP does not provide all of the information required by streaming media. For example, UDP does not indicate to the receiver when the media samples were generated, as is required by the receiver to properly determine when to play out the sample. To provide this additional information, another protocol, the Real-Time Protocol (RTP), is used on top of UDP, and is described later in this thesis.

2.1.1. Multimedia Data Streams

According to [6], streaming media is rapid transmission of audio and video in packets over the Internet. But, [7] defines streaming media as delivery of continuous audio, video over the Internet, by feeding the media to the user as the media is viewed. For the purpose of this measurement project, streaming media is defined as transmission of audio or video in packets over either the Internet or an intranet, and feeding the media to the user as the media is viewed. The reason behind this definition is that the measurement system is designed for both intranet and Internet.

The information that can be perceived by the human senses is transported through different media, such as text or sound. Humans communicate with computers by means of various media, or use computers as tools for communication with each other. These

observations led to the following definition of multimedia systems [7]:

“A multimedia system is characterized by the computer-controlled generation, manipulation, presentation, storage, and communication of independent discrete and continuous media.”

The ability to handle continuous media, such as audio or video, distinguishes multimedia systems from most conventional computer systems. Continuous media, while being actually discrete, appear smooth to the human observer when presented regularly and periodically at sufficiently high frequencies. Therefore, continuous media are time dependent, and their processing is subject to time constraints.

2.1.2. Multimedia Application Categories

Home entertainment system is one application of streaming media. Web TV and internet radio is growing in its popularity. Streaming media has been considered as a way for entertainment companies to deliver music and videos to consumers [9]. Extremely low cost phone conversations can be implemented using streaming media technologies such as voice over IP (VOIP). A survey show that over half of Internet users in the US access some form of streaming media [9]. The other application of streaming media is for enterprise use. Teleconferencing through streaming media gives low cost solution for companies. [10] mentions that multimedia conferencing is expected to become a requirement of international companies who want to remain competitive in the global market. Training and sales presentations needs to be delivered to wide audience [9], and remote training with streaming media can reduce company training costs. The ability to transfer streaming media over wireless LANs provides additional benefits. Home entertainment applications can benefit from the portability feature of wireless LANs, allowing people to locate equipment in arbitrary positions in their home without cables.

Multimedia applications can be divided into different categories, each making particular demands for support on the operating system or runtime environment.

Information Systems: The main purpose of such systems is to provide information for users. The requested information is usually stored in databases or media archives. Examples: electronic publishing, online galleries or weather information systems.

Remote Representation: By means of a remote representation system a user can

take part in or monitor events from a remote location.

Examples: distance conferencing or lecturing, virtual reality, or remote robotic agents.

Entertainment: This major application area of multimedia technology is strongly oriented towards audio and video data.

Examples: digital television, video on demand, distributed games or interactive television.

User interaction can help to classify amongst services;

Interactive Services: Interactive services let the user to select the transmitted information. These services can be:

- **Conversational Services.** Services with real-time demands and no relevant buffering, like video conferencing or video surveillance
- **Messaging Services.** Services with temporary storing, like multimedia mail.
- **Retrieval Services.** Information services interactively presenting previously stored information from a database or media collection, for example teleshopping or hospital information systems.

Distribution Services: Distribution services transmit information from a central source to a potentially unknown set of receivers. There are two subcategories that differ in the control possibilities granted the users;

- **Services without User Control:** Services characterized by having one central sender that broadcasts information to all participating users, for example digital television broadcasting.
- **Services with User Control:** These are services allowing the user to choose from the distributed information.

2.1.3. Resource Demands of Multimedia Applications

Applications make use of various system resources, which are provided by the different system components. According to [10], a resource is a system entity that a task requires for the manipulation or presentation of data. Typical resources offered by a multimedia-capable system include for example the CPU, the memory management, the file systems, network access, and special hardware like cameras or sound subsystems.

A resource is categorized as either

- Active or passive. Active resources provide a service, normally by making use of passive resources.
- Exclusive or shared. Exclusive resources can be used by one task at a time only, but shared resources may be used by various tasks concurrently.
- Single or multiple, depending on the number of instances of that particular resource existing in a given system.

The capacity of a resource is measured by a task's ability to perform its work in a given time span using that resource. To support continuous media, even systems comprising high-capacity networks and fast processors require real-time mechanisms to provide the necessary resources within the time constraints specified by the desired quality and by the type of the processed media.

For distributed multimedia applications, the required communication structure is very important. Traditional networking applications like file transfer generate one-to-one traffic, whereas typical multimedia applications such as video conferences or distributed virtual reality games communicate via one-to-many or even many-to-many message paths.

2.1.4. Quality of Service

The term quality of service (QoS) originated in the field of communications and was used to describe the technical characteristics of data transmission. The notion can be generalized in order to describe any service that a system offers, as done for example in [11]: *"Quality of service represents the set of those quantitative and qualitative characteristics of a distributed multimedia system necessary to achieve the required functionality of an application."*

All system entities - especially the operating system components and the network subsystem - must cooperate to provide services that meet the demands of the applications.

The following definitions, cited from [11], are used to classify the quality of a service offered by an active resource:

Throughput: Throughput is defined as the product of the size or number of the data units and of the rate at which the units are processed.

Delay: The local delay is the time a resource needs to complete a task; the global

delay is the total delay a data unit needs for traveling from its source to its destination.

Jitter: Jitter is the maximum allowed variance in inter-arrival the time interval of the data.

Reliability: Reliability issues comprise error frequency and possible error types as well as error- detection and error-correction mechanisms.

Category	Application	Bandwidth	Real-time requirements	Communication structure	Media
Information systems	hospital information systems	medium-very high	high	1-to-1, 1-to-n	high-resolution images, text, video, sound
	electronic publishing	low	none	1-to-1, 1-to-n	text, images
	museums	low-medium	none	1-to-1	(high-resolution) images, text
Remote representation	distance learning, conferencing	high	strong	1-to-n, n-to-m	text, video, sound
	virtual reality	medium-very high	high-very high	1-to-1, n-to-m	text, video, sound
	robotic agents	low-high	high	1-to-1	text, video, sound
	remote task agents	low	none	1-to-1, 1-to-n	text, video, sound, images
Entertainment	digital TV	very high	very high	1-to-n	high-definition video, text, stereo audio
	interactive TV	low	low	m-to-1	text
	video-on-demand	medium-very high	low-very high	1-to-1, 1-to-n	text, video, sound
	widely distributed interactive games	low	low-medium	1-to-n, n-to-m	text

Table 2. Requirements of different multimedia application classes.

To assure timely presentation or manipulation of multimedia data, an application requires either guaranteed services, for which strictly observed quality-of-service guarantees are given by the service-providing system entity, or mostly reliable predictive services, which behave as specified for the most part. In both cases, an application must consult the resource manager that is responsible for a particular system service to make a resource reservation.

An application reserves resources by performing the following steps:

1. Estimating the user's subjective QoS wishes.
2. Mapping these estimations to a technical QoS profile.
3. Negotiating a contract about the QoS values with the resource management.

The actual QoS values in a system can vary over time. A system must therefore permanently monitor its state and take measures to meet the guarantees, or must notify the clients about insufficiencies. The system components should also be prepared to renegotiate the contract if a user decides to change the requirements.

2.2. Process Management

The preceding considerations showed that, from an operating system point of view, real-time scheduling and data transmission are the most crucial topics.

Since this thesis deals primarily with the transmission of multimedia data, processor scheduling will be covered only within this section.

2.2.1. Requirements

The resource management of a real-time system schedules the access of tasks to system resources.

A task is defined as a schedulable system entity such as a thread or a process, and is characterized by its time constraints and resource requirements. The timeliness of the media presentations or manipulations performed by a task is as important as the correctness of the operation.

The aim of a processor resource's management component is to calculate a schedule that allows the timely processing of as many time-critical tasks as possible. A real-time multimedia scheduler must take into account externally defined time constraints as well as time limits formed by internal data dependencies.

The real-time requirements of multimedia applications are usually depending on the application type-less demanding than those of traditional command and control systems, which often have a direct physical impact:

- Multimedia systems are usually more fault tolerant than command and control systems.
- While missing a deadline is not desirable, it generally does not have severe physical consequences.
- Continuous media data is most often the result of the periodic sampling of a

signal. Hence, tasks require the processor resource periodically; such tasks are much easier to schedule than sporadic tasks.

- The bandwidth demand can often be adjusted dynamically to the available capacities.

2.2.2. Scheduling Strategies

The following paragraphs cover only periodic tasks, which request the processors at constant intervals. Two algorithms dominate the scheduler implementations in multimedia-capable operating systems:

Earliest Deadline First (EDF): EDF is an optimal dynamic algorithm. Any beginning of a new period of a task triggers a reordering of the complete schedule, according to the dead lines of the scheduled tasks.

Rate-Monotonic Scheduling: Rate-monotonic scheduling is an optimal static, priority-driven algorithm. During the set-up phase, each task is assigned a static priority according to its requested processing rate: the shorter a task's period, the higher is its priority. Afterwards, each task is processed with this priority, and no rearrangement of the priorities is necessary as long as no new task appears.

Both algorithms can be extended by dividing each task into a mandatory part, which secures an acceptable result, and an optional part, which refines the work. Tasks are primarily scheduled with respect to the mandatory parts, while the optional parts are only processed during under load periods. This arrangement allows scaling of continuous media.

2.3. Multimedia Data Transmission

Real-time multimedia applications, such as video on demand, use the communication subsystem in different ways than traditional applications such as file transfer or electronic mail do. Obviously, the prevailing service model of the Internet - the best-effort, point-to-point delivery service of IP - does not reflect the needs of continuous media transmission. The special needs of real-time multimedia network traffic that also led to the proposed Integrated Services extensions for the Internet

architecture will be discussed later in Section 2.3.

2.3.1. Multi Point Communication

Multi-point communication structures, which are characterized by a group of m senders transmitting data to a group of n recipients, are often necessary for distributed multimedia applications.

Typical scenarios include computer supported cooperative work (CSCW), multimedia conferencing, distributed virtual reality, distributed simulation, digital television, or distance teaching.

To avoid inefficient repeated sending of n identical messages, the communication system must offer multicast transmission. Multicasting requires support about the following issues:

Multicast Group and Address Management: To dynamically participate in multicast sessions, any participant should be able to initiate, join or leave multicast groups at any time.

Setting up a multicast group requires the assignment and propagation of a multicast group address, used as the destination address for all data sent to this group.

Transport Reliability and Flow Control: New mechanisms must be developed that offer reliability without creating huge amounts of control traffic. To avoid costly retransmissions of packets dropped due to network congestion, flow control techniques must be used to adapt the sent traffic quantity to the network congestion state along all multicast data paths.

Routing: Multicasting asks for routing protocols that reduce duplicate traffic on any shared link to a minimum, preferably to one copy. In addition, out of both network load and security concerns, data should only be sent along links that have at least one recipient.

Naturally, a routing algorithm should choose the optimal path for the transmitted data, but with multicasting, it must consider the paths to a (potentially widely spread) set of receivers.

2.3.2. Quality of Service Control

The general QoS criteria listed in Section 2.1.4 are very suitable to describe the QoS aspects of multimedia-data transmission. The network type-independent QoS parameter list includes:

- Connection establishment delay.
- Connection establishment failure probability.
- Sustained or peak throughput rate.
- Transit delay.
- Delay variation.
- Loss rate.
- Error rate.
- Security and Protection parameters.
- Priority.

Some of these parameters are negotiable between network systems and applications, whereas others are fixed or statistical characteristics of a network. Each networking technique might extend the above list by specific parameters.

2.3.3. Flow Control

The main task of flow control is to protect the receiver from an unmanageable flood of packets and from consequent data loss due to reception buffer overflow. For that purpose, senders and receivers can reach traffic agreements that are independent from the underlying communication system.

There are two main approaches to flow control:

Window-Based Flow Control: Window-based systems work with respect to the maximum amount of data that the sender is allowed to sent without having received an acknowledgment for data already transmitted. The sizes of both sending and reception windows must be adapted to the participants' capacities and to the characteristics of a particular network.

Rate-Based Flow Control: In a rate-based system, a sender forwards data at a reasonable rate that is chosen in order to allow network and receivers to cope with the

transmitted volume.

Rate-controlled systems do not necessarily enforce an acknowledgment of the received data. However, some feedback about the current transmission quality is useful for adaptive systems that allow media scaling (dynamic bandwidth control).

CHAPTER 3

RELATED WORK

A scalable feedback control mechanism for video sources has also been proposed in [4]. This mechanism is also end-to-end and defines network states according to feedback information from the receivers. This work differs from our approach in the sense that a probabilistic polling mechanism with increasing search scope and a randomly delayed reply scheme is used to supply the source with feedback information. With RTP the receiver reports are multicast periodically so that an explicit probing mechanism is not required. Another feedback controller is presented in [5]. However, this controller requires that the switches send their buffer occupancies and service rates back to the source. It was not designed to scale for multicast distributions. The control of the application by a network manager that monitors the network elements via standardized management information bases is presented in [6]. The network manager notifies applications of network congestion based on feedback received from switches and routers. This approach has serious scaling problems.

CHAPTER 4

NETWORK PROTOCOLS OVERVIEW

TCP is not appropriate for transmitting media packets; because re-transmissions of packets to provide connection oriented service causes end to end delays and they cannot be tolerated in real-time applications. UDP is much more applicable than TCP for media data. Because TCP performs congestion control and retransmission of packets loss are not applicable for Multimedia applications. By adding additional data top of UDP real-time multimedia data is sent over network.

4.1. Why Can't We Just Use TCP for Audio and Video?

For delivering audio and video for playback, TCP may be appropriate. Also, with sufficiently long buffering and adequate average throughput, near-real-time delivery using TCP can be successful, as practiced by the Netscape WWW browser. TCP may often run over highly lossy networks (e.g., the German X.25 network) with acceptable throughput, even though the uncompensated losses would make audio or video communication impossible.

However, for real-time delivery of audio and video, TCP and other reliable transport protocols such as XTP are inappropriate. The three main reasons are:

- Reliable transmission is inappropriate for delay-sensitive data such as real-time audio and video. By the time the sender has discovered the missing packet and retransmitted it, at least one round-trip time, likely more, has elapsed. The receiver either has to wait for the retransmission, increasing delay and incurring an audible gap in play out, or discard the retransmitted packet, defeating the TCP mechanism.
- Standard TCP implementations force the receiver application to wait, so that packet losses would always yield increased delay. Note that a single packet lost repeatedly could drastically increase delay, which would persist at least until the end of talk spurt.
- TCP cannot support multicast.

- The TCP congestion control mechanisms decreases the congestion window when packet losses are detected ("slow start"). Audio and video, on the other hand, have "natural" rates that cannot be suddenly decreased without starving the receiver. For example, standard PCM audio requires 64 kb/s, plus any header overhead, and cannot be delivered in less than that. Video could be more easily throttled simply by slowing the acquisition of frames at the sender when the transmitter's send buffer is full, with the corresponding delay. The correct congestion response for these media is to change the audio/video encoding, video frame rate, or video image size at the transmitter, based, for example, on feedback received through RTCP receiver report packets.

An additional small disadvantage is that the TCP and RTP headers are larger than a UDP header (40 bytes for TCP and RTP 3.6, 32 bytes for RTP 4.0, compared to 8 bytes). Also, these reliable transport protocols do not contain the necessary time stamp and encoding information needed by the receiving application, so that they cannot replace RTP. (They would not need the sequence number as these protocols assure that no losses or reordering takes place.)

While LANs often have sufficient bandwidth and low enough losses not to trigger these problems, TCP does not offer any advantages in that scenario either, except for the recovery from rare packet losses. Even in a LAN with no losses, the TCP slow start mechanism would limit the initial rate of the source for the first few round-trip times.

4.2. Why Can't We Just Use UDP for Audio and Video?

RTP provides end-to-end delivery services for data with real-time characteristics, such as interactive audio and video. Those services include payload type identification sequence numbering, time stamping and delivery monitoring. Applications typically run RTP on top of UDP to make use of its multiplexing and check-sum services; both protocols contribute parts of the transport protocol functionality.

But UDP itself alone does not define a technique for synchronizing. In order to use only UDP a feedback channel must be defined for QoS. However, RTP may be used with other suitable underlying network or transport protocols.

RTP supports data transfer to multiple destinations using multicast distribution if provided by the underlying network.

Note that RTP itself does not provide any mechanism to ensure timely delivery or provide other quality-of-service guarantees, but relies on lower-layer services to do so. It does not guarantee delivery or prevent out-of-order delivery, nor does it assume that the underlying network is reliable and delivers packets in sequence. The sequence numbers included in RTP allow the receiver to reconstruct the sender's packet sequence, but sequence numbers might also be used to determine the proper location of a packet, for example in video decoding, without necessarily decoding packets in sequence.

CHAPTER 5

THE REAL-TIME TRANSPORT PROTOCOL

RTP has been designed within the Internet Engineering Task Force (IETF) [13]. Note that the “transport protocol” could be misleading, as it is mostly used together with UDP, also designated as a transport protocol. The name emphasizes, however, that RTP is an end-to-end protocol. To avoid misunderstandings, it may help to clear up some of the things that RTP does not attempt to do. RTP has no notion of a connection; it may operate over either connection-oriented or connectionless lower-layer protocols. It has no dependencies on particular address formats and only requires that framing and segmentation is taken care of by lower layers. RTP offers no reliability mechanisms. It is typically implemented as part of the application and not of the operating system kernel. RTP consists of two parts, a data part and a control part.

Continuous media data like audio and video is carried in RTP data packets. The functionality of the control packets is described below. If RTP packets are carried in UDP datagram, data and control packets use two consecutive ports, with the data port always being the lower, even numbered one. If other protocols serve underneath RTP (e.g., RTP directly over ATM AAL5), it is possible to carry both in a single lower-layer protocol data unit, with control followed by data.

Application Level Framing: The key architectural principle is Application Level Framing. The idea is that the application should break down the data into groups or aggregates, which are meaningful to the application, and which do not depend on a specific network technology. These data aggregates are called Application Data Units (ADUs). The frame boundaries of ADUs should be preserved by lower levels, that is, by the network system.

The rule, by which the size of an ADU is chosen, states that an application must be able to process each ADU separately and potentially out of order with respect to other ADUs.

So, the loss of some ADUs, even if a retransmission is triggered, does not stop the receiving application from processing other ADUs. Therefore and to express data loss in terms meaningful to the application, each ADU must contain a name that

allows the receiver to understand the place of an ADU in the sequence of ADUs produced by the sender. Hence, RTP data units carry sequence numbers and timestamps, so that a receiver can determine the time and sequence relation between ADUs.

The ADU is also the main unit of error recovery. Because each ADU is a meaningful data entity to the receiving application, the application itself can decide about how to cope with a lost data unit: real-time applications, such as digital video broadcasting, might prefer to simply ignore a few lost frames instead of delaying the presentation until the lost frames are retransmitted, whereas file transfer applications cannot accept the loss of a single data unit.

In addition, if the application has the choice of how to deal with a lost unit, the sender can decide whether to buffer the data units for potential retransmission, to recompute the lost units if requested again, or to send new data that which diminishes the harm done by the loss of the original ADU.

Integrated Layer Processing: Because application level framing breaks down the data in pieces that an application can handle separately from other data units, all processing of a single, complete ADU can be done in one integrated processing step for reasons of efficiency. This engineering principle is called Integrated Layer Processing.

While the authors of [11] agree that layered isolation, as employed in conventional protocol stacks, is suitable for the network layer and below, they argue that many of the functions of the transport and above layers could be structured in a way that would permit the use of the more efficient Integrated Layer Processing.

Especially for RISC systems, whose performance is substantially limited by the costs of memory cycles, an integrated processing loop is more efficient than several, isolated steps that each read the data from memory, possibly process it in some way, and write it back to memory.

RTP is used in combination with other network or transport protocols typically on top of UDP - as visualized in Figure 5.1.

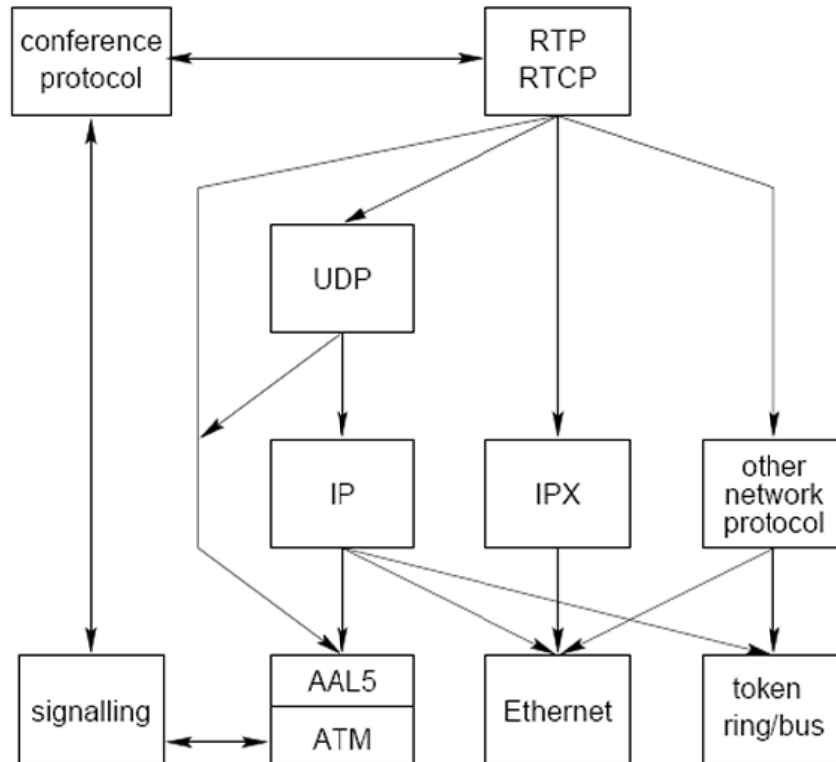


Figure 5.1. RTP in relation to network and transport protocols

5.1. RTP Use Scenarios

The following sections describe some aspects of the use of RTP. The examples were chosen to illustrate the basic operation of applications using RTP, not to limit what RTP may be used for. In these examples, RTP is carried on top of IP and UDP and follows the conventions established by the profile for audio and video specified in the companion RFC 3551.

5.1.1. Simple Multicast Audio Conference

A working group of the IETF meets to discuss the latest protocol document, using the IP multicast services of the Internet for voice communications. Through some allocation mechanism the working group chair obtains a multicast group address and pair of ports. One port is used for audio data, and the other is used for control (RTCP) packets. This address and port information is distributed to the intended participants. If

privacy is desired, the data and control packets may be encrypted, in which case an encryption key must also be generated and distributed. The exact details of these allocation and distribution mechanisms are beyond the scope of RTP.

The audio conferencing application used by each conference participant sends audio data in small chunks of, say, 20 ms duration. Each chunk of audio data is preceded by an RTP header; RTP header and data are in turn contained in a UDP packet. The RTP header indicates what type of audio encoding (such as PCM, ADPCM or LPC) is contained in each packet so that senders can change the encoding during a conference, for example, to accommodate a new participant that is connected through a low-bandwidth link or react to indications of network congestion. The Internet, like other packet networks, occasionally loses and reorders packets and delays them by variable amounts of time. To cope with these impairments, the RTP header contains timing information and a sequence number that allow the receivers to reconstruct the timing produced by the source, so that in this example, chunks of audio are contiguously played out the speaker every 20 ms. This timing reconstruction is performed separately for each source of RTP packets in the conference. The sequence number can also be used by the receiver to estimate how many packets are being lost. Since members of the working group join and leave during the conference, it is useful to know who is participating at any moment and how well they are receiving the audio data. For that purpose, each instances of the audio application in the conference periodically multicasts a reception report plus the name of its user on the RTCP (control) port. The reception report indicates how well the current speaker is being received and may be used to control adaptive encodings. In addition to the user name, other identifying information may also be included subject to control bandwidth limits. A site sends the RTCP BYE packet when it leaves the conference.

5.1.2. Audio and Video Conference

If both audio and video media are used in a conference, they are transmitted as separate RTP sessions. That is, separate RTP and RTCP packets are transmitted for each medium using two different UDP port pairs and/or multicast addresses. There is no direct coupling at the RTP level between the audio and video sessions, except that a user participating in both sessions should use the same distinguished (canonical) name

in the RTCP packets for both so that the sessions can be associated. One motivation for this separation is to allow some participants in the conference to receive only one medium if they choose. Further explanation is given in Section 5.2. Despite the separation, synchronized playback of a source's audio and video can be achieved using timing information carried in the RTCP packets for both sessions.

5.1.3. Mixers and Translators

So far, I have assumed that all sites want to receive media data in the same format. However, this may not always be appropriate. Consider the case where participants in one area are connected through a low-speed link to the majority of the conference participants who enjoy high-speed network access. Instead of forcing everyone to use a lower-bandwidth, reduced-quality audio encoding, an RTP-level relay called a *mixer* may be placed near the low-bandwidth area. This mixer resynchronizes incoming audio packets to reconstruct the constant 20 ms spacing generated by the sender, mixes these reconstructed audio streams into a single stream, translates the audio encoding to a lower-bandwidth one and forwards the lower-bandwidth packet stream across the low-speed link. These packets might be unicast to a single recipient or multicast on a different address to multiple recipients. The RTP header includes a means for mixers to identify the sources that contributed to a mixed packet so that correct talker indication can be provided at the receivers. Some of the intended participants in the audio conference may be connected with high bandwidth links but might not be directly reachable via IP multicast. For example, they might be behind an application-level firewall that will not let any IP packets pass. For these sites, mixing may not be necessary, in which case another type of RTP-level relay called a *translator* may be used. Two translators are installed, one on either side of the firewall, with the outside one funneling all multicast packets received through a secure connection to the translator inside the firewall.

The translator inside the firewall sends them again as multicast packets to a multicast group restricted to the site's internal network. Mixers and translators may be designed for a variety of purposes. An example is a video mixer that scales the images of individual people in separate video streams and composites them into one video

stream to simulate a group scene. Other examples of translation include the connection of a group of hosts speaking only IP/UDP to a group of hosts that understand only ST-II, or the packet-by-packet encoding translation of video streams from individual sources without resynchronization or mixing

5.1.4. Layered Encoding

Multimedia applications should be able to adjust the transmission rate to match the capacity of the receiver or to adapt to network congestion. Many implementations place the responsibility of rate adaptivity at the source. This does not work well with multicast transmission because of the conflicting bandwidth requirements of heterogeneous receivers. The result is often a least-common denominator scenario, where the smallest pipe in the network mesh dictates the quality and fidelity of the overall live multimedia broadcast.

Instead, responsibility for rate-adaptation can be placed at the receivers by combining a layered encoding with a layered transmission system. In the context of RTP over IP multicast, the source can stripe the progressive layers of a hierarchically represented signal across multiple RTP sessions each carried on its own multicast group. Receivers can then adapt to network heterogeneity and control their reception bandwidth by joining only the appropriate subset of the multicast groups.

5.2. RTP

5.2.1. Definitions

RTP payload: The data transported by RTP in a packet, for example audio samples or compressed video data. The payload format and interpretation are beyond the scope of this document.

RTP packet: A data packet consisting of the fixed RTP header, a possibly empty list of contributing sources and the payload data. Some underlying protocols may require an encapsulation of the RTP packet to be defined. Typically one packet of the underlying protocol contains a single RTP packet, but several RTP packets may be

contained if permitted by the encapsulation method.

RTCP packet: A control packet consisting of a fixed header part similar to that of RTP data packets, followed by structured elements that vary depending upon the RTCP packet type.

Typically, multiple RTCP packets are sent together as a compound RTCP packet in a single packet of the underlying protocol; this is enabled by the length field in the fixed header of each RTCP packet.

Port: The "abstraction that transport protocols use to distinguish among multiple destinations within a given host computer. TCP/IP protocols identify ports using small positive integers." [9] The transport selectors (TSEL) used by the OSI transport layer are equivalent to ports. RTP depends upon the lower-layer protocol to provide some mechanism such as ports to multiplex the RTP and RTCP packets of a session.

Transport address: The combination of a network address and port that identifies a transport-level endpoint, for example an IP address and a UDP port. Packets are transmitted from a source transport address to a destination transport address.

RTP media type: An RTP media type is the collection of payload types which can be carried within a single RTP session. The RTP Profile assigns RTP media types to RTP payload types.

Multimedia session: A set of concurrent RTP sessions among a common group of participants. For example, a videoconference (which is a multimedia session) may contain an audio RTP session and a video RTP session.

RTP session: An association among a set of participants communicating with RTP. A participant may be involved in multiple RTP sessions at the same time. In a multimedia session, each medium is typically carried in a separate RTP session with its own RTCP packets unless the encoding itself multiplexes multiple media into a single data stream. A participant distinguishes multiple RTP sessions by reception of different sessions using different pairs of destination transport addresses, where a pair of transport addresses comprises one network address plus a pair of ports for RTP and RTCP. All participants in an RTP session may share a common destination transport address pair, as in the case of IP multicast, or the pairs may be different for each participant, as in the case of individual unicast network addresses and port pairs. In the unicast case, a participant may receive from all other participants in the session using the same pair of ports, or may use a distinct pair of ports for each.

5.2.2. RTP Data Packet

RTP data packets consist of a 12-byte header followed by the payload, e.g., a video frame or a sequence of audio samples. The payload may be wrapped again into an encoding-specific layer.

5.2.3. RTP Header

The RTP header has the following format:

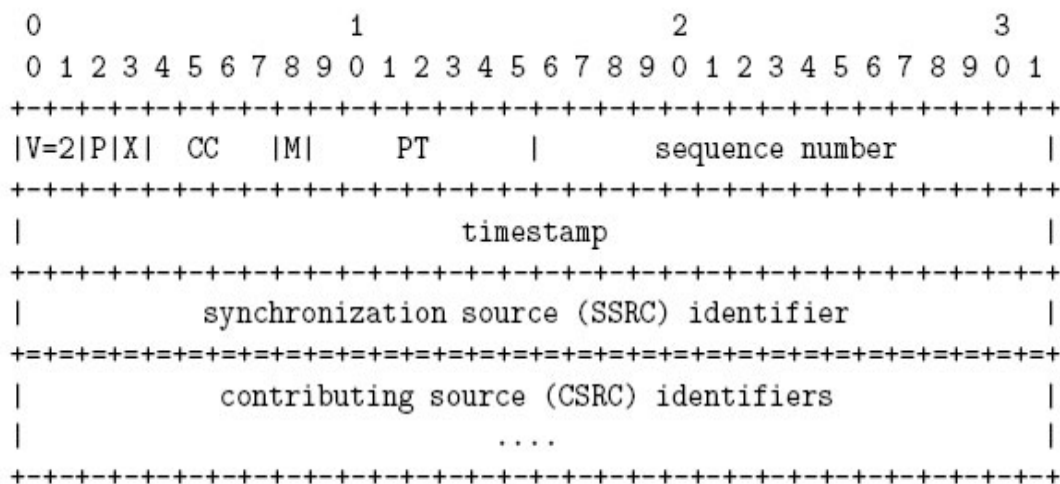


Figure 5.2. The RTP header

The first twelve octets are present in every RTP packet, while the list of CSRC identifiers is present only when inserted by a mixer. The fields have the following meaning:

Version (V): This field identifies the version of RTP. The version defined by this specification is two (2).

(The value 1 is used by the first draft version of RTP and the value 0 is used by the protocol initially implemented in the "vat" audio tool.)

Padding (P): If the padding bit is set, the packet contains one or more additional padding octets at the end which are not part of the payload. The last octet of the padding

contains a count of how many padding octets should be ignored, including itself.

Padding may be needed by some encryption algorithms with fixed block sizes or for carrying several RTP packets in a lower-layer protocol data unit.

Extension (X): If the extension bit is set, the fixed header must be followed by exactly one header extension

CSRC count (CC): The CSRC count contains the number of CSRC identifiers that follow the fixed header.

Marker (M): The interpretation of the marker is defined by a profile. It is intended to allow significant events such as frame boundaries to be marked in the packet stream. A profile may define additional marker bits or specify that there is no marker bit by changing the number of bits in the payload type field.

Payload type (PT): This field identifies the format of the RTP payload and determines its interpretation by the application. A profile may specify a default static mapping of payload type codes to payload formats. Additional payload type codes may be defined dynamically through non-RTP means. A set of default mappings for audio and video is specified in the companion RFC 3551 [14]. A RTP source may change the payload type during a session, but this field should not be used for multiplexing separate media streams. A receiver must ignore packets with payload types that it does not understand.

Sequence number: The sequence number increments by one for each RTP data packet sent, and may be used by the receiver to detect packet loss and to restore packet sequence. The initial value of the sequence number should be random (unpredictable) to make known-plaintext attacks on encryption more difficult, even if the source itself does not encrypt, because the packets may flow through a translator that does. Techniques for choosing unpredictable numbers are discussed in [15].

Timestamp: The timestamp reflects the sampling instant of the first octet in the RTP data packet. The sampling instant must be derived from a clock whose increments monotonically and linearly in time to allow synchronization and jitter calculations. The resolution of the clock must be sufficient for the desired synchronization accuracy and for measuring packet arrival jitter (one tick per video frame is typically not sufficient). The clock frequency is dependent on the format of data carried as payload and is specified statically in the profile or payload format specification that defines the format, or may be specified dynamically for payload formats defined through non-RTP means.

If RTP packets are generated periodically, the nominal sampling instant as determined from the sampling clock is to be used, not a reading of the system clock. As an example, for fixed-rate audio the timestamp clock would likely increment by one for each sampling period. If an audio application reads blocks covering 160 sampling periods from the input device, the timestamp would be increased by 160 for each such block, regardless of whether the block is transmitted in a packet or dropped as silent.

The initial value of the timestamp should be random, as for the sequence number. Several consecutive RTP packets will have equal timestamps if they are (logically) generated at once, e.g., belong to the same video frame. Consecutive RTP packets may contain timestamps that are not monotonic if the data is not transmitted in the order it was sampled, as in the case of MPEG interpolated video frames. (The sequence numbers of the packets as transmitted will still be monotonic.)

RTP timestamps from different media streams may advance at different rates and usually have independent, random offsets. Therefore, although these timestamps are sufficient to reconstruct the timing of a single stream, directly comparing RTP timestamps from different media is not effective for synchronization.

Instead, for each medium the RTP timestamp is related to the sampling instant by pairing it with a timestamp from a reference clock (wall-clock) that represents the time when the data corresponding to the RTP timestamp was sampled. The reference clock is shared by all media to be synchronized. The timestamp pairs are not transmitted in every data packet, but at a lower rate in RTCP SR packets.

The sampling instant is chosen as the point of reference for the RTP timestamp because it is known to the transmitting endpoint and has a common definition for all media, independent of encoding delays or other processing. The purpose is to allow synchronized presentation of all media sampled at the same time.

Applications transmitting stored data rather than data sampled in real-time typically use a virtual presentation timeline derived from wall-clock time to determine when the next frame or other unit of each medium in the stored data should be presented. In this case, the RTP timestamp would reflect the presentation time for each unit. That is, the RTP timestamp for each unit would be related to the wall-clock time at which the unit becomes current on the virtual presentation timeline. Actual presentation occurs some time later as determined by the receiver.

An example describing live audio narration of prerecorded video illustrates the

significance of choosing the sampling instant as the reference point. In this scenario, the video would be presented locally for the narrator to view and would be simultaneously transmitted using RTP. The "sampling instant" of a video frame transmitted in RTP would be established by referencing its timestamp to the wall-clock time when that video frame was presented to the narrator.

The sampling instant for the audio RTP packets containing the narrator's speech would be established by referencing the same wall-clock time when the audio was sampled. The audio and video may even be transmitted by different hosts if the reference clocks on the two hosts are synchronized by some means such as NTP. A receiver can then synchronize presentation of the audio and video packets by relating their RTP timestamps using the timestamp pairs in RTCP SR packets.

SSRC: The SSRC field identifies the synchronization source. This identifier should be chosen randomly, with the intent that no two synchronization sources within the same RTP session will have the same SSRC identifier. Although the probability of multiple sources choosing the same identifier is low, all RTP implementations must be prepared to detect and resolve collisions. Section 8 describes the probability of collision along with a mechanism for resolving collisions and detecting RTP-level forwarding loops based on the uniqueness of the SSRC identifier. If a source changes its source transport address, it must also choose a new SSRC identifier to avoid being interpreted as a looped source.

CSRC list: The CSRC list identifies the contributing sources for the payload contained in this packet. The number of identifiers is given by the CC field. If there are more than 15 contributing sources, only 15 can be identified. CSRC identifiers are inserted by mixers using the SSRC identifiers of contributing sources. For example, for audio packets the SSRC identifiers of all sources that were mixed together to create a packet, are listed allowing correct talker indications at the receiver.

5.3. RTCP

The RTP control protocol (RTCP) is based on the periodic transmission of control packets to all participants in the session, using the same distribution mechanism as the data packets.

The underlying protocol must provide multiplexing of the data and control

packets, for example using separate port numbers with UDP. RTCP performs four functions:

1. The primary function is to provide feedback on the quality of the data distribution. This is an integral part of the RTP's role as a transport protocol and is related to the flow and congestion control functions of other transport protocols. The feedback may be directly useful for control of adaptive encodings [16, 17], but experiments with IP multicasting have shown that it is also critical to get feedback from the receivers to diagnose faults in the distribution. Sending reception feedback reports to all participants allows one who is observing problems to evaluate whether those problems are local or global. With a distribution mechanism like IP multicast, it is also possible for an entity such as a network service provider who is not otherwise involved in the session to receive the feedback information and act as a third-party monitor to diagnose network problems. This feedback function is performed by the RTCP sender and receiver reports.

2. RTCP carries a persistent transport-level identifier for an RTP source called the canonical name or CNAME.

Since the SSRC identifier may change if a conflict is discovered or a program is restarted, receivers require the CNAME to keep track of each participant. Receivers may also require the CNAME to associate multiple data streams from a given participant in a set of related RTP sessions, for example to synchronize audio and video. Inter-media synchronization also requires the NTP and RTP timestamps included in RTCP packets by data senders.

3. The first two functions require that all participants send RTCP packets, therefore the rate must be controlled in order for RTP to scale up to a large number of participants. By having each participant send its control packets to all the others, each can independently observe the number of participants. This number is used to calculate the rate at which the packets are sent.

4. A fourth, OPTIONAL function is to convey minimal session control information, for example participant identification to be displayed in the user interface. This is most likely to be useful in "loosely controlled" sessions where participants enter and leave without membership control or parameter negotiation. RTCP serves as a convenient channel to reach all the participants, but it is not necessarily expected to

support all the control communication requirements of an application. A higher-level session control protocol, which is beyond the scope of this document, may be needed.

5.3.1. Packet Format

This specification defines several RTCP packet types to carry a variety of control information:

SR: Sender report, for transmission and reception statistics from participants that are active senders

RR: Receiver report, for reception statistics from participants that are not active senders and in combination with SR for active senders reporting on more than 31 sources

SDES: Source description items, including CNAME

BYE: Indicates end of participation

APP: Application-specific functions

Each RTCP packet begins with a fixed part similar to that of RTP data packets, followed by structured elements that may be of variable length according to the packet type but must end on a 32-bit boundary. The alignment requirement and a length field in the fixed part of each packet are included to make RTCP packets "stackable". Multiple RTCP packets can be concatenated without any intervening separators to form a compound RTCP packet that is sent in a single packet of the lower layer protocol, for example UDP.

There is no explicit count of individual RTCP packets in the compound packet since the lower layer protocols are expected to provide an overall length to determine the end of the compound packet.

Each individual RTCP packet in the compound packet may be processed independently with no requirements upon the order or combination of packets. However, in order to perform the functions of the protocol, the following constraints are imposed:

- Reception statistics (in SR or RR) should be sent as often as bandwidth constraints will allow to maximize the resolution of the statistics, therefore each periodically transmitted compound RTCP packet must include a report packet.

- New receivers need to receive the CNAME for a source as soon as possible to

identify the source and to begin associating media for purposes such as lip-sync, so each compound RTCP packet must also include the SDES CNAME except when the compound RTCP packet is split for partial encryption.

- The number of packet types that may appear first in the compound packet needs to be limited to increase the number of constant bits in the first word and the probability of successfully validating RTCP packets against misaddressed RTP data packets or other unrelated packets.

Thus, all RTCP packets must be sent in a compound packet of at least two individual packets, with the following format:

Encryption prefix: If and only if the compound packet is to be encrypted, it must be prefixed by a random 32-bit quantity redrawn for every compound packet transmitted. If padding is required for the encryption, it must be added to the last packet of the compound packet.

SR or RR: The first RTCP packet in the compound packet must always be a report packet to facilitate header validation. This is true even if no data has been sent or received, in which case an empty RR must be sent, and even if the only other RTCP packet in the compound packet is a BYE.

Additional RRs: If the number of sources for which reception statistics are being reported exceeds 31, the number that will fit into one SR or RR packet, then additional RR packets should follow the initial report packet.

SDES: An SDES packet containing a CNAME item must be included in each compound RTCP packet.

Other source description items may optionally be included if required by a particular application, subject to bandwidth constraints.

BYE or APP: Other RTCP packet types, including those yet to be defined, may follow in any order, except that BYE should be the last packet sent with a given SSRC/CSRC. Packet types may appear more than once.

An individual RTP participant should send only one compound RTCP packet per report interval in order for the RTCP bandwidth per participant to be estimated correctly except when the compound RTCP packet is split for partial encryption. If there are too many sources to fit all the necessary RR packets into one compound RTCP packet without exceeding the maximum transmission unit (MTU) of the network path, then only the subset that will fit into one MTU should be included in each interval. The

subsets should be selected round-robin across multiple intervals so that all sources are reported.

It is recommended that translators and mixers combine individual RTCP packets from the multiple sources they are forwarding into one compound packet whenever feasible in order to amortize the packet overhead.

An example RTCP compound packet as might be produced by a mixer is shown in Figure 5.3. If the overall length of a compound packet would exceed the MTU of the network path, it should be segmented into multiple shorter compound packets to be transmitted in separate packets of the underlying protocol. This does not impair the RTCP bandwidth estimation because each compound packet represents at least one distinct participant. Note that each of the compound packets must begin with an SR or RR packet.

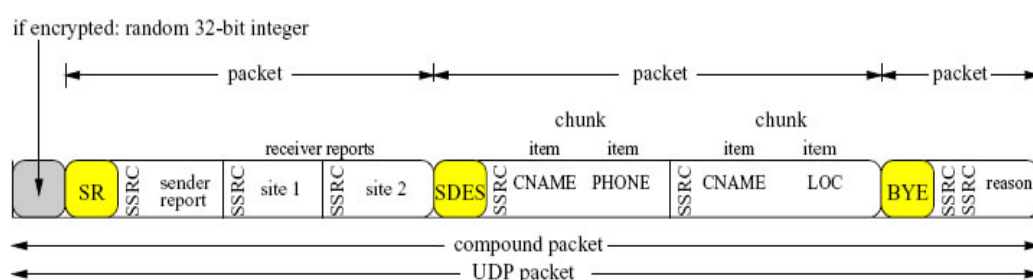


Figure 5.3. RTCP compound packet

5.3.2. RTCP Transmission Interval

RTP is designed to allow an application to scale automatically over session sizes ranging from a few participants to thousands. For example, in an audio conference the data traffic is inherently self-limiting because only one or two people will speak at a time, so with multicast distribution the data rate on any given link remains relatively constant independent of the number of participants. However, the control traffic is not self-limiting. If the reception reports from each participant were sent at a constant rate, the control traffic would grow linearly with the number of participants. Therefore, the rate must be scaled down by dynamically calculating the interval between RTCP packet transmissions.

For each session, it is assumed that the data traffic is subject to an aggregate limit called the "session bandwidth" to be divided among the participants. This bandwidth might be reserved and the limit enforced by the network. If there is no reservation, there may be other constraints, depending on the environment, that establish the "reasonable" maximum for the session to use, and that would be the session bandwidth. The session bandwidth may be chosen based on some cost or a priori knowledge of the available network bandwidth for the session. It is somewhat independent of the media encoding, but the encoding choice may be limited by the session bandwidth.

Often, the session bandwidth is the sum of the nominal bandwidths of the senders expected to be concurrently active. For teleconference audio, this number would typically be one sender's bandwidth. For layered encodings, each layer is a separate RTP session with its own session bandwidth parameter.

The session bandwidth parameter is expected to be supplied by a session management application when it invokes a media application, but media applications may set a default based on the single-sender data bandwidth for the encoding selected for the session. The application may also enforce bandwidth limits based on multicast scope rules or other criteria. All participants must use the same value for the session bandwidth so that the same RTCP interval will be calculated.

Bandwidth calculations for control and data traffic include lower- layer transport and network protocols (e.g., UDP and IP), since that is what the resource reservation system would need to know. The application can also be expected to know which of these protocols are in use. Link level headers are not included in the calculation since the packet will be encapsulated with different link level headers as it travels.

The control traffic should be limited to a small and known fraction of the session bandwidth: small so that the primary function of the transport protocol to carry data is not impaired; known so that the control traffic can be included in the bandwidth specification given to a resource reservation protocol, and so that each participant can independently calculate its share. The control traffic bandwidth is in addition to the session bandwidth for the data traffic. It is recommended that the fraction of the session bandwidth added for RTCP be fixed at 5%. It is also recommended that 1/4 of the RTCP bandwidth be dedicated to participants that are sending data so that in sessions with a large number of receivers but a small number of senders, newly joining

participants will more quickly receive the CNAME for the sending sites. When the proportion of senders is greater than $1/4$ of the participants, the senders get their proportion of the full RTCP bandwidth. While the values of these and other constants in the interval calculation are not critical, all participants in the session must use the same values so the same interval will be calculated. Therefore, these constants should be fixed for a particular profile.

A profile may specify that the control traffic bandwidth may be a separate parameter of the session rather than a strict percentage of the session bandwidth. Using a separate parameter allows rate-adaptive applications to set an RTCP bandwidth consistent with a "typical" data bandwidth that is lower than the maximum bandwidth specified by the session bandwidth parameter.

The profile may further specify that the control traffic bandwidth may be divided into two separate session parameters for those participants which are active data senders and those which are not; let us call the parameters S and R . Following the recommendation that $1/4$ of the RTCP bandwidth be dedicated to data senders, the recommended default values for these two parameters would be 1.25% and 3.75%, respectively. When the proportion of senders is greater than $S/(S+R)$ of the participants, the senders get their proportion of the sum of these parameters. Using two parameters allows RTCP reception reports to be turned off entirely for a particular session by setting the RTCP bandwidth for non-data-senders to zero while keeping the RTCP bandwidth for data senders non-zero so that sender reports can still be sent for inter-media synchronization. Turning off RTCP reception reports is not recommended because they are needed for the functions listed at the beginning of Section 6, particularly reception quality feedback and congestion control. However, doing so may be appropriate for systems operating on unidirectional links or for sessions that don't require feedback on the quality of reception or liveliness of receivers and that have other means to avoid congestion.

The calculated interval between transmissions of compound RTCP packets should also have a lower bound to avoid having bursts of packets exceed the allowed bandwidth when the number of participants is small and the traffic isn't smoothed according to the law of large numbers. It also keeps the report interval from becoming too small during transient outages like a network partition such that adaptation is delayed when the partition heals. At application startup, a delay should be imposed

before the first compound RTCP packet is sent to allow time for RTCP packets to be received from other participants so the report interval will converge to the correct value more quickly. This delay may be set to half the minimum interval to allow quicker notification that the new participant is present. The recommended value for a fixed minimum interval is 5 seconds.

An implementation may scale the minimum RTCP interval to a smaller value inversely proportional to the session bandwidth parameter with the following limitations:

- For multicast sessions, only active data senders may use the reduced minimum value to calculate the interval for transmission of compound RTCP packets.
- For unicast sessions, the reduced value may be used by participants that are not active data senders as well, and the delay before sending the initial compound RTCP packet may be zero.
- For all sessions, the fixed minimum should be used when calculating the participant timeout interval so that implementations which do not use the reduced value for transmitting RTCP packets are not timed out by other participants prematurely.
- The recommended value for the reduced minimum in seconds is 360 divided by the session bandwidth in kilobits/second. This minimum is smaller than 5 seconds for bandwidths greater than 72 kb/s.

The algorithm described in Appendix X was designed to meet the goals outlined in this section. It calculates the interval between sending compound RTCP packets to divide the allowed control traffic bandwidth among the participants. This allows an application to provide fast response for small sessions where, for example, identification of all participants is important, yet automatically adapt to large sessions. The algorithm incorporates the following characteristics:

- The calculated interval between RTCP packets scales linearly with the number of members in the group. It is this linear factor which allows for a constant amount of control traffic when summed across all members.
- The interval between RTCP packets is varied randomly over the range [0.5, 1.5] times the calculated interval to avoid unintended synchronization of all participants. The first RTCP packet sent after joining a session is also delayed by a random variation of half the minimum RTCP interval.
- A dynamic estimate of the average compound RTCP packet size is calculated,

including all those packets received and sent, to automatically adapt to changes in the amount of control information carried.

- Since the calculated interval is dependent on the number of observed group members, there may be undesirable startup effects when a new user joins an existing session, or many users simultaneously join a new session. These new users will initially have incorrect estimates of the group membership, and thus their RTCP transmission interval will be too short. This problem can be significant if many users join the session simultaneously. To deal with this, an algorithm called "timer reconsideration" is employed. This algorithm implements a simple back-off mechanism which causes users to hold back RTCP packet transmission if the group sizes are increasing.

- When users leave a session, either with a BYE or by timeout, the group membership decreases, and thus the calculated interval should decrease. A "reverse reconsideration" algorithm is used to allow members to more quickly reduce their intervals in response to group membership decreases.

- BYE packets are given different treatment than other RTCP packets. When a user leaves a group, and wishes to send a BYE packet, it may do so before its next scheduled RTCP packet. However, transmission of BYEs follows a back-off algorithm which avoids floods of BYE packets should a large number of members simultaneously leave the session.

This algorithm may be used for sessions in which all participants are allowed to send. In that case, the session bandwidth parameter is the product of the individual sender's bandwidth times the number of participants and the RTCP bandwidth is 5% of that.

The rules for how to send and what to do when receiving an RTCP packet are outlined here.

5.3.3. RTCP Packet Send & Receive Rules

An implementation which is constrained to two-party unicast operation should still use randomization of the RTCP transmission interval to avoid unintended synchronization of multiple instances operating in the same environment.

To execute these rules, a session participant must maintain several pieces of state:

tp: the last time an RTCP packet was transmitted;
tc: the current time;
tn: the next scheduled transmission time of an RTCP packet;
Pmembers: the estimated number of session members at the time tn was last recomputed;

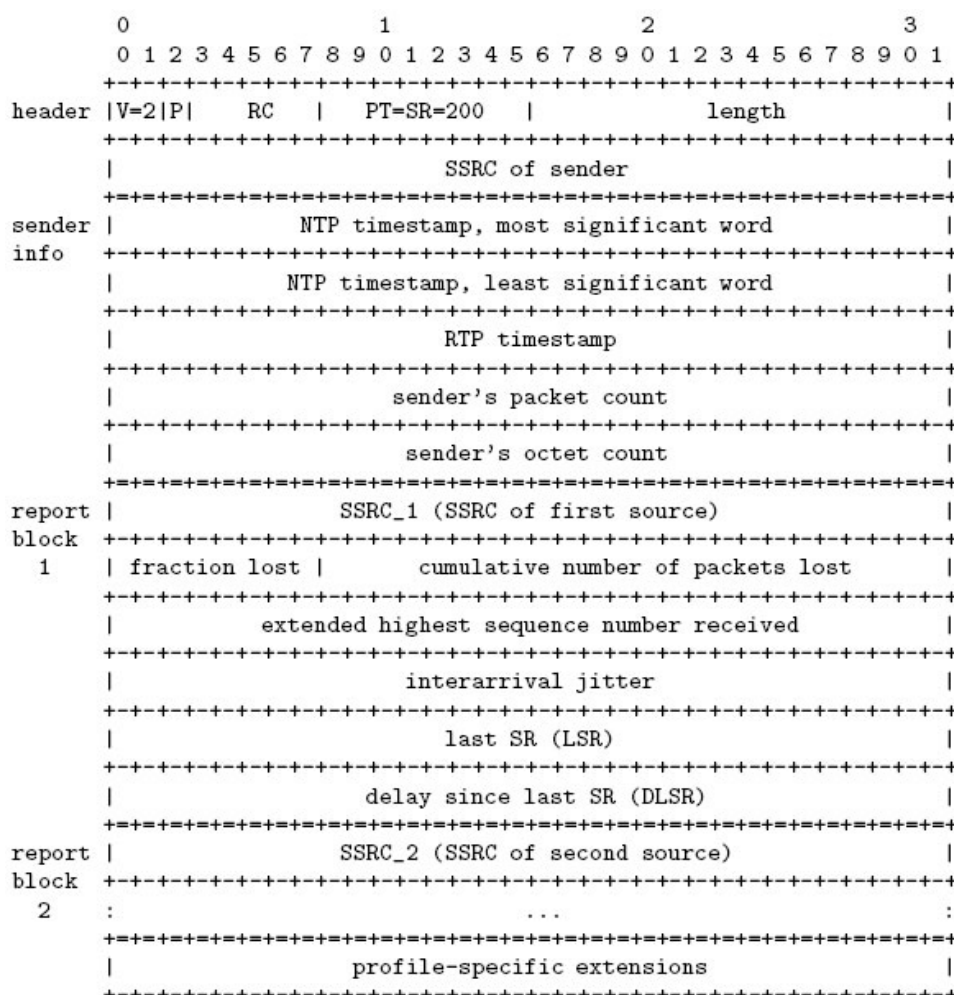


Figure 5.4. RTCP sender report

Members: the most current estimate for the number of session members;
Senders: the most current estimate for the number of senders in the session;
rtcp_bw: The target RTCP bandwidth, i.e., the total bandwidth that will be used for RTCP packets by all members of this session, in octets per second. This will be a specified fraction of the "session bandwidth" parameter supplied to the application at startup.

`we_sent`: Flag that is true if the application has sent data since the 2nd previous RTCP report was transmitted.

`avg_rtcp_size`: The average compound RTCP packet size, in octets, over all RTCP packets sent and received by this participant. The size includes lower-layer transport and network protocol headers (e.g., UDP and IP).

`Initial`: Flag that is true if the application has not yet sent an RTCP packet.

5.3.4. Sender & Receiver Reports

The sender report packet consists of three sections, possibly followed by a fourth profile-specific extension section if defined. The first section, the header, is 8 octets long.

Version (V): Identifies the version of RTP, which is the same in RTCP packets as in RTP data packets. The version defined by this specification is two (2).

Padding (P): If the padding bit is set, this individual RTCP packet contains some additional padding octets at the end which are not part of the control information but are included in the length field. The last octet of the padding is a count of how many padding octets should be ignored, including itself (it will be a multiple of four). Padding may be needed by some encryption algorithms with fixed block sizes. In a compound RTCP packet, padding is only required on one individual packet because the compound packet is encrypted as a whole. Thus, padding must only be added to the last individual packet, and if padding is added to that packet, the padding bit must be set only on that packet.

Reception report count (RC): The number of reception report blocks contained in this packet. A value of zero is valid.

Packet type (PT): Contains the constant 200 to identify this as an RTCP SR packet.

Length: The length of this RTCP packet in 32-bit words minus one, including the header and any padding. (The offset of one makes zero a valid length and avoids a possible infinite loop in scanning a compound RTCP packet, while counting 32-bit words avoids a validity check for a multiple of 4.)

SSRC: The synchronization source identifier for the originator of this SR packet. The second section, the sender information, is 20 octets long and is present in every sender report packet. It summarizes the data transmissions from this sender. The fields

have the following meaning:

NTP timestamp: Indicates the wall-clock time when this report was sent so that it may be used in combination with timestamps returned in reception reports from other receivers to measure round-trip propagation to those receivers. Receivers should expect that the measurement accuracy of the timestamp may be limited to far less than the resolution of the NTP timestamp.

The measurement uncertainty of the timestamp is not indicated as it may not be known. On a system that has no notion of wall-clock time but does have some system-specific clock such as "system uptime", a sender may use that clock as a reference to calculate relative NTP timestamps. It is important to choose a commonly used clock so that if separate implementations are used to produce the individual streams of a multimedia session, all implementations will use the same clock. Until the year 2036, relative and absolute timestamps will differ in the high bit so (invalid) comparisons will show a large difference; by then one hopes relative timestamps will no longer be needed. A sender that has no notion of wall-clock or elapsed time may set the NTP timestamp to zero.

RTP timestamp: Corresponds to the same time as the NTP timestamp (above), but in the same units and with the same random offset as the RTP timestamps in data packets. This correspondence may be used for intra- and inter-media synchronization for sources whose NTP timestamps are synchronized, and may be used by media-independent receivers to estimate the nominal RTP clock frequency. Note that in most cases this timestamp will not be equal to the RTP timestamp in any adjacent data packet. Rather, it must be calculated from the corresponding NTP timestamp using the relationship between the RTP timestamp counter and real-time as maintained by periodically checking the wall-clock time at a sampling instant.

Sender's packet count: The total number of RTP data packets transmitted by the sender since starting transmission up until the time this SR packet was generated. The count should be reset if the sender changes its SSRC identifier.

Sender's octet count: The total number of payload octets (i.e., not including header or padding) transmitted in RTP data packets by the sender since starting transmission up until the time this SR packet was generated. The count should be reset if the sender changes its SSRC identifier. This field can be used to estimate the average payload data rate.

The third section contains zero or more reception report blocks depending on the number of other sources heard by this sender since the last report. Each reception report block conveys statistics on the reception of RTP packets from a single synchronization source.

Receivers should not carry over statistics when a source changes its SSRC identifier due to a collision.

These statistics are:

SSRC_n (source identifier): The SSRC identifier of the source to which the information in this reception report block pertains.

Fraction lost: The fraction of RTP data packets from source SSRC_n lost since the previous SR or RR packet was sent, expressed as a fixed point number with the binary point at the left edge of the field. (That is equivalent to taking the integer part after multiplying the loss fraction by 256.) This fraction is defined to be the number of packets lost divided by the number of packets expected, as defined in the next paragraph. If the loss is negative due to duplicates, the fraction lost is set to zero note that a receiver cannot tell whether any packets were lost after the last one received, and that there will be no reception report block issued for a source if all packets from that source sent during the last reporting interval have been lost.

Cumulative number of packets lost: The total number of RTP data packets from source SSRC_n that have been lost since the beginning of reception. This number is defined to be the number of packets expected less the number of packets actually received, where the number of packets received includes any which are late or duplicates.

Thus, packets that arrive late are not counted as lost, and the loss may be negative if there are duplicates. The number of packets expected is defined to be the extended last sequence number received, as defined next, less the initial sequence number received.

Extended highest sequence number received: The low 16 bits contain the highest sequence number received in an RTP data packet from source SSRC_n, and the most significant 16 bits extend that sequence number with the corresponding count of sequence number cycles. Note that different receivers within the same session will generate different extensions to the sequence number if their start times differ significantly.

Inter-arrival jitter: An estimate of the statistical variance of the RTP data packet inter-arrival time measured in timestamp units and expressed as an unsigned integer. The inter-arrival jitter J is defined to be the mean deviation (smoothed absolute value) of the difference D in packet spacing at the receiver compared to the sender for a pair of packets. As shown in the equation below, this is equivalent to the difference in the "relative transit time" for the two packets; the relative transit time is the difference between a packet's RTP timestamp and the receiver's clock at the time of arrival, measured in the same units.

If S_i is the RTP timestamp from packet i , and R_i is the time of arrival in RTP timestamp units for packet i , then for two packets i and j , D may be expressed as

$$D(i,j) = (R_j - R_i) - (S_j - S_i) = (R_j - S_j) - (R_i - S_i) \quad (5.1)$$

The inter-arrival jitter should be calculated continuously as each data packet i is received from source $SSRC_n$, using this difference D for that packet and the previous packet $i-1$ in order of arrival (not necessarily in sequence), according to the formula

$$J(i) = J(i-1) + (|D(i-1,i)| - J(i-1))/16 \quad (5.2)$$

Whenever a reception report is issued, the current value of J is sampled.

The jitter calculation must conform to the formula specified here in order to allow profile-independent monitors to make valid interpretations of reports coming from different implementations. This algorithm is the optimal first-order estimator and the gain parameter $1/16$ gives a good noise reduction ratio while maintaining a reasonable rate of convergence.

Last SR timestamp (LSR): The middle 32 bits out of 64 in the NTP timestamp received as part of the most recent RTCP sender report (SR) packet from source $SSRC_n$. If no SR has been received yet, the field is set to zero.

Delay since last SR (DLSR): The delay, expressed in units of $1/65536$ seconds, between receiving the last SR packet from source $SSRC_n$ and sending this reception report block. If no SR packet has been received yet from $SSRC_n$, the DLSR field is set to zero.

5.3.5. RR: Receiver Report RTCP Packet

The format of the receiver report (RR) packet is the same as that of the SR packet except that the packet type field contains the constant 201 and the five words of sender information are omitted (these are the NTP and RTP timestamps and sender's packet and octet counts). The remaining fields have the same meaning as for the SR packet. An empty RR packet (RC = 0) must be put at the head of a compound RTCP packet when there is no data transmission or reception to report.

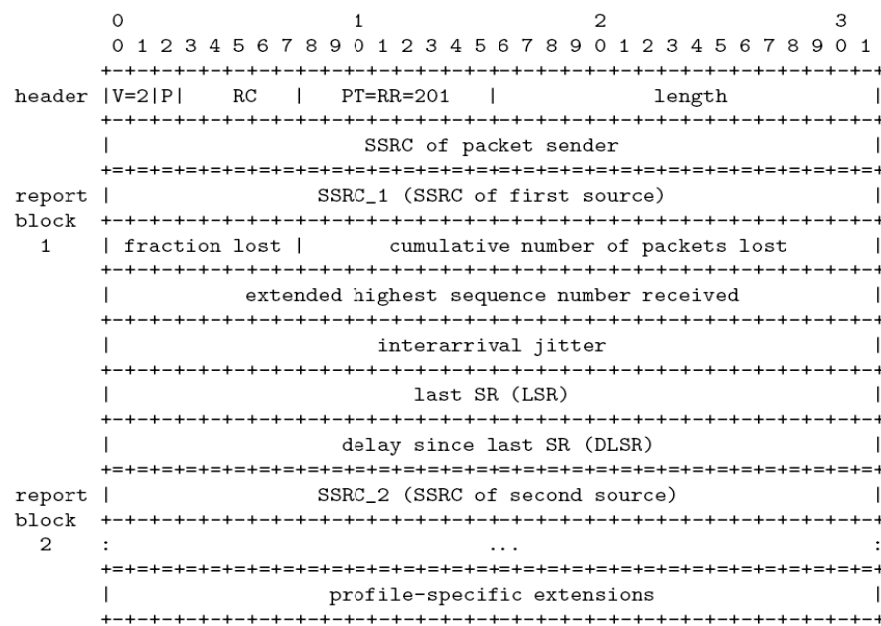


Figure 5.5. RTCP receiver report

CHAPTER 6

PERFORMANCE PARAMETERS

A scalable feedback control mechanism for video sources has also been proposed in [6]. This mechanism is also end-to-end and defines network states according to feedback information from the receivers. This work differs from my approach in the sense that a probabilistic polling mechanism with increasing search scope and a randomly delayed reply scheme is used to supply the source with feedback information. With RTP the receiver reports are multicast periodically so that an explicit probing mechanism is not required.

6.1. Performance Parameters for Wireless and Wired LAN

6.1.1. Throughput

Fluckiger defines throughput as “The bit rate between two communicating end systems is the number of binary digits that the network is capable of accepting and delivering per unit time” [18]. Throughput in 802.11 wireless LAN is not as high as wired LANs (e.g. Gigabit Ethernet) though it is still better than dial up modem. The current maximum throughput for wireless LAN is 54 Mbps, implemented in 802.11a and 802.11g. [19]. Though this throughput is already high enough for streaming applications alone, there may be some performance impact if traffic is loaded with heavy background application such as FTP. Figure 6.1 shows a survey held by Cisco [20] shows that throughput is the largest problem experienced by wireless LAN users.

During the demonstration, the steaming audio and VOIP used consumes a maximum of 32Kbps. Clearly, this is far below the maximum throughput of a wireless LAN.

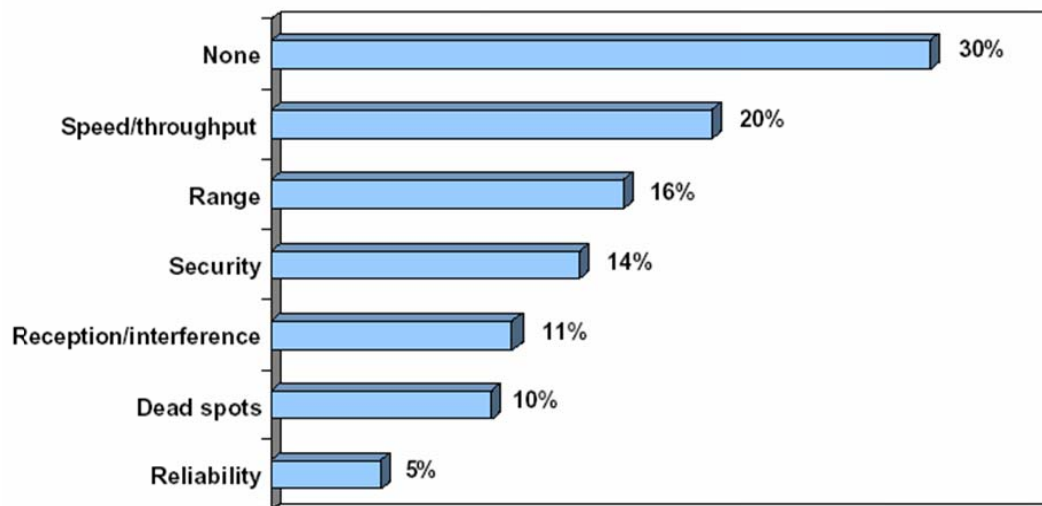


Figure 6.1. Throughput is the largest problem for wireless LAN users

6.1.2. Integrity & Reliability

Data transmitted in a wireless network tend to experience more interference than data transmitted in wired network. This affect reliability and integrity of data transferred. Though this issue does not significantly impact connection oriented protocols that can retransmit information, this issue impact connectionless protocol such as UDP, because there is no error control implemented in connectionless protocol. Even though wireless LANs exhibit high error rates, when streaming media is transferred over a wireless LAN, it is rarely done using a protocol such as TCP that can recover from errors. This is because the time that it takes a connection oriented protocol to recover from errors will introduce delay that substantially degrades the streaming media, while the end-user can rarely perceive a few lost or damaged streaming media packets.

6.2. Performance Parameters for Streaming Media

6.2.1. Delay

The end-to-end packet delay is the time spent by a packet travelling from source to destination [21]. It is the sum of transmission processing, queuing delays in routers,

propagation delays, and end-system processing delays along path from source to destination [22].

Packet delay is a key consideration associated with transmission of voice. The delay can be resulted from processing packets or heavy line utilization [23]. When a slight delay occurs in transferring data, it is usually not noticeable. Furthermore, if packets containing data are lost during transmission, the sender normally retransmits these packets. However, small delay or loss in transferring voice packets results in disruption of speech intelligibility. [22] mentions that delays between 150 and 400 milliseconds can be acceptable. However, delays exceeding 400 milliseconds can seriously hinder the interactivity in voice conversation.

Delayed voice packets are usually considered as being lost. If packets do not arrive in time they are either covered by a period of silence or synthetic speech [23]. Since a period of silence involves simply generating nothing to compensate for delayed packets, it will certainly generate loss of speech intelligibility for the user. A better method to address loss is to use synthetic speech. In this process, the receiver attempts to reconstruct the lost packets through previous correctly received packets. More advance method for reconstructing delayed packets is by using Forward Error Correction (FEC) techniques: Through this technique, the sender sends redundant information along with the original information. Thus, the original data can be reconstructed from redundant information that the receiver received [22]. However the cost of this is additional bandwidth required to send the redundant information. The delay value can be obtained by knowing the time difference between sender and receiver. Thus, when packet i is sent at time S_i , and arrives at destination at time R_i , the delay associated with packet i is:

$$D_i = R_i - S_i \quad (6.1)$$

In measurement system, the receiver time can be obtained from packets captured by Ethereal. However, obtaining the sender time is slightly complex. The approach used by [24], is to approximate the sender's sending time by interpreting the RTP timestamps and sampling rate. The basic idea of the calculation is that rather than comparing the delay from the packet's sending time, I measure the packet delay with the first packet generated by the server. Thus, the monitor only needs to obtain the time when the first

packet is sent from the server and the sampling rate of the stream. The sampling rate is defined as the number of samples or snapshots taken of a particular signal in a given amount of time. According to RFC 1889 [25], the RTP timestamp will increase for each consecutive packet according to the sampling rate. For example, if a server generates RTP packets every 20 ms, and each packet's timestamp increases by 160 for each consecutive packet, the sampling rate of that stream is 160 samples per 20 ms or equal to 8 KHz.

Assumed that the server has sent 2 packets, P2 and P1, in a row, the sampling rate can be calculated by the server using this following formula:

$$\text{Sampling rate} = (TS2 - TS1) / (T2 - T1)$$

Where:

$TS2$ = Timestamp value of P2

$TS1$ = Timestamp value of P1

$T2$ = sending time of P2

$T1$ = sending time of P1

Having got the sampling rate and time when the first packet was sent, the delay for each packet can be obtained.

$$\text{Planned time elapsed since P1} = (TS_i - TS1) / \text{Sampling rate}$$

$$\text{Actual time elapsed since P1} = T_i - T1$$

$$\text{Delay} = \text{Actual time elapsed} - \text{Planned time elapsed}$$

Where

$P1$ = First packet sent

$T1$ = Receiving time of P1

$TS1$ = First packet timestamps

P_i = Packet received

T_i = Receiving time of P_i

TS_i = received timestamps

6.2.2. Jitter

Jitter is delay variation. The RTP standard [25] defines jitter as a smoothed function of delay differences between consecutive packets over time. Another definition of Jitter (derived from Kurose & Ross[22]) is that jitter is the fluctuation of delay from

packet to packet. Streaming media applications usually remove jitter by temporarily storing received packets in buffer instead of playing the packets directly to users. In accordance with the RTP standard, for my measurement system, the jitter of one packet is the difference between the delays of this packet compared with delay of previous packet. Thus, for P_i , the jitter is as follows:

$$J_i = D_i - D_{i-1}$$

where:

P_i = Packet arrives at i th

J_i = Jitter of packet i th

D_i = Delay of packet i th

D_{i-1} = Delay of packet (i th -1)

The measurement system ignores the effect of lost packets on the jitter calculation. The only way it could consider them would be to treat them as having infinite delay, but that would make the jitter calculations meaningless.

6.2.3. Packet Loss Rate

There are several different ways of defining the packet loss rate. [24] defines packet loss rates as fractions of packets that do not arrive in the receiver at all, while [21] defines packet loss as packets that do not arrive in receiver in a given time interval.

The sequence numbers included in RTP allow the receiver to construct the sender's packet sequence. In this measurement system, packet loss will be packets that do not arrive in order. For example, if at t_1 P_1 arrives, then at t_2 , P_3 arrives instead of P_2 , then P_2 is simply counted as a lost packet. The reason underlying this is that the measurement system does not have access to the applications. Though streaming applications can reconstruct packets that do not arrive in order, each application will have different waiting time of lost packets. Thus the measurement system must be prepared to wait a different amount of time for lost packets. Kurose & Ross [22] argue that packet loss rates between 1% and 20% can be tolerated. However, this argument depends on the types of applications of streaming media. There are a few techniques explained in section 3.2.1 that normally can be used to minimize the impact of packet loss on streaming media application.

CHAPTER 7

A CONTROL MECHANISM OVER RTP

The main goal of this thesis is to achieve a controllable environment in order to provide an approach to increase performance of a streaming media transfer over RTP. In order to have this main goal to be accomplished it is essential to analyze the actual transfer of a media using RTP. The study performed to enhance the performance requires monitoring the transfer. Hence, a control mechanism that monitors and helps to analyze should be provided. In this section of the thesis report, this control mechanism will be introduced and explained in details.

7.1. How to Enhance Performance?

In general, it is desired to have the integrity preserved while the data is transferred. In other words, whole amount of data packages should be at the destination after the transfer is completed. In RTP transfers, timeliness of RTP packet arrivals are important for that particular packet to be evaluated in time, because of this fact, any late packet can be considered as a lost packet for multimedia transfer. There are some interpolation techniques to replace the lost segment for the transfer but these are not considered in the context of this study. However an approach is presented in this study, for minimizing the negative effects of the late packets.

In this study, the performance concept is mainly based on the transfer integrity. While streaming a media, real-time data, it is aimed to maintain high percentage of data transfer. The quality of service can be represented by this notion. The more the data is transferred untainted, the more performance can be experienced with that transfer and protocol. And of course there is also the concern of the data transfer time. It is desired to have the possible minimum time of travel for the transferred packets. Having the travel time possible minimum has a major affect on performance.

In order to ensure the QoS for RTP, researchers have come out with some QoS parameters as to be observed. In general, the focus of this thesis, in this notion, is the Network QoS, based on the QoS Framework as suggested by [26].

This means that the analysis that has been taken out is to evaluate the network environment in which RTP communication is being monitored. It simply means that, by ensuring the performance concept that is mentioned above, QoS requirements will be met.

7.2. QoS Metrics over RTP

There are several QoS parameters that have been identified to be implemented in this study. Table 7.1 provides a list of Network QoS parameters available as derived from [27].

In the analysis, three QoS parameters, namely delay, jitter, and packet loss rate have been selected. The main justification for this selection is such that the study focuses on the performance of RTP communication over different networking conditions and environment, and hence the time and reliability would be main concerns for evaluation. The results of studies described by [28] also shown that high delay and high delay variability (jitter) has been experienced by a large number of Internet paths that resulting in poor RTP performance. It should be noted that these QoS parameters have also been considered in some previous studies [29] and [30].

<i>Category</i>	<i>Parameters</i>
<i>Timeliness</i>	Delay Response time Jitter
<i>Bandwidth</i>	Systems-level data rate Application-level data rate Transaction time
<i>Reliability</i>	Mean time to failure (MTTF) Mean time to repair (MTTR) Mean time between failures (MTBF) Percentage of time available Packet loss rate Bit error rate

Table 7.1. Network QoS parameters

In order to be able to make adjustments on a particular system, one needs to study the system to maintain the key aspects and functionalities for it. Hence, before mentioning to enhance performance for real-time networking, keen observations and analysis should take place.

It will be essential to introduce the key performance metric that are taken under consideration for this thesis study. Mainly, network performance parameters are introduced above, but for the study only one parameter, bandwidth is selected.

BW: Bandwidth; in computer networking and computer science, bandwidth (digital bandwidth or network bandwidth) is a measure of available or consumed data communication resources expressed in bits/second.

Bandwidth typically means the net bit rate, channel capacity or the maximum throughput of a logical or physical communication path in a digital communication system.

For example, bandwidth tests measure the maximum throughput of a computer network. The reason for this usage is that according to Hartley's law, the maximum data rate of a physical communication link is proportional to its bandwidth in hertz, which is sometimes called frequency bandwidth, radio bandwidth or analog bandwidth, the last especially in computer networking literature.

Bandwidth may also refer to consumed bandwidth, corresponding to achieve throughput, the average rate of successful data transfer through a communication path (in my case the RTP transfer). This sense leads related part of my thesis study referred as bandwidth management or as bandwidth allocation. The main difference between general bandwidth management and our approach is that, control mechanism lets the user adjust the bandwidth of the allocated amount of the transfer channel, for the best performance interest of the RTP traffic.

The main approach to adjust the bandwidth in my study can be explained as follows;

There is a specific value for the data buffer (k) in the RTP library. This buffer value, k is constant for a given default value of bandwidth. This default value is assigned by the library in the initiation. In this case it is 8000 b/sec. as default value. While the application is writing data to the RTP transfer channel, it waits for k/BW sec. between the packets that are assigned to be written. Hence the data density for my application stays at a certain value for BW. In order to maintain a higher density, hence

a more transfer for RTP data, the user can adjust the BW value bigger than before. This will cause less waiting time at the buffer while the data is being written. Keeping in mind that, the transport medium has a roof capacity for bandwidth .Our assigned value cannot be over this capacity value.

The transfer protocol, package structures, reports are examined. As a result of this study, insufficient parts are detected and made essential modifications for the study to progress accordingly. Simply RTP works as follows;

- RTP sender starts a RTP session
- RTP sender starts the RTP stream
- RTP receiver joins to the session
- RTP channel between sender and receiver is established
- RTCP channel between sender and receiver is established
- SR and RR packet flow starts.

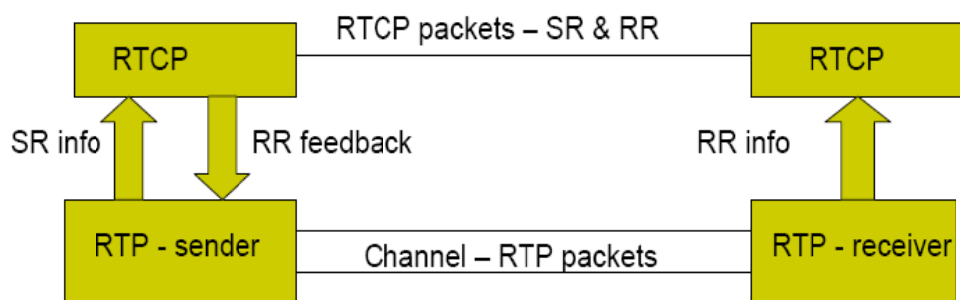


Figure 7.1. RTP diagram

As a result of this establishment, real-time data flow begins. As can be observed from the Figure 7.1, the data flow is one-directional and the control flow- RTCP channel- is bi-directional. It is chosen to explain and implement the topic in this manner, because of the fact that, it is easier to understand, control and visualize. But as matter of fact; it is possible and desirable to have mentioned network nodes are both RTP senders and receivers.

Package structures, network flow analysis indicated that, to enhance performance, these questions should be studied;

- What information is needed from the RTP receiver?

- What are the most important performance metrics?
- How can RTP sender manage the RTP session for high performance?
- Even if it is real-time data, should there be a packet reconstruction?
- Is jlibRTP sufficient?

To accomplish the goals for this thesis, including the answers to above questions, the “Control Mechanism over RTP” will be introduced.

7.3. Control Mechanism over RTP

The control mechanism monitors the current performance state of the transfer. It reports to the user so that the user can make the suitable adjustments.

Figure 7.2 shows the settlement of the control mechanism with the RTP transfer mechanism.

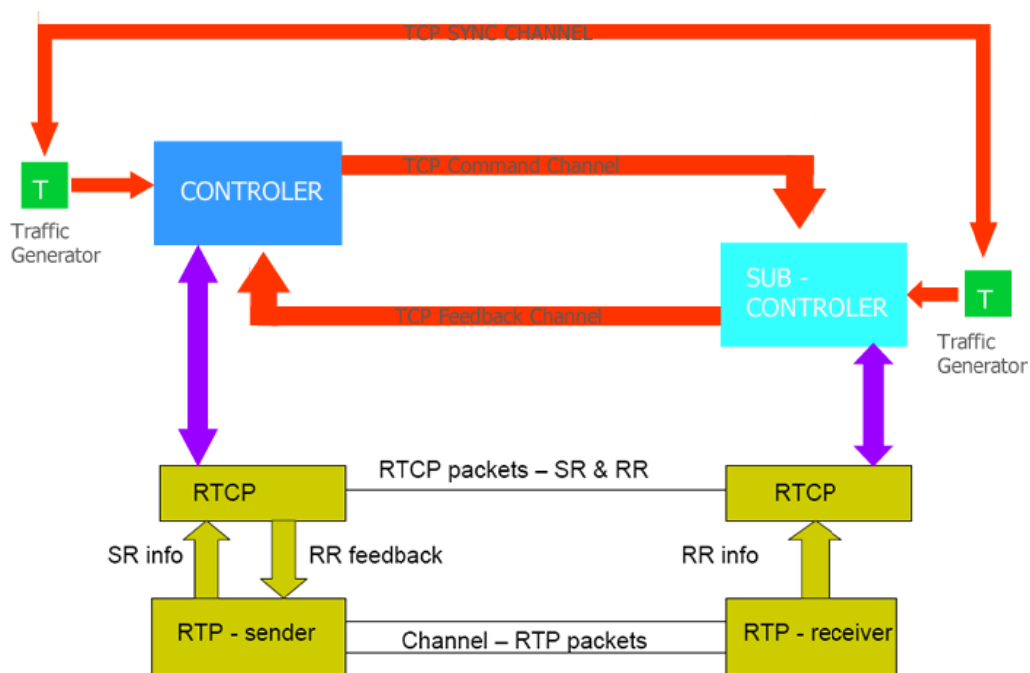


Figure 7.2. Control mechanism over RTP

7.3.1. Controller

This is the main, key element of the thesis study. The controller sits on top of RTP sender enabling the RTP sender, send the real-time data accordingly to the adjustments made by the controller.

First of all, user (RTP sender) can initiate a session by defining a specific port for corresponding transfer. In case of multiple RTP receivers controller will define various ports for each listener (RTP receiver). Approves join requests and in necessary conditions for the best of the session and hence the transfer.

The controller unit can function manually and automatically.

As the most important feature of this component, control logic is introduced. In order to optimize the parameters which are considered as performance metrics for this study, and enhance the performance accordingly, a function is performed. This function is managed by the controller and uses PID controller logic.

A proportional–integral–derivative controller (PID controller) is a generic control loop feedback mechanism (controller) widely used in industrial control systems – a PID is the most commonly used feedback controller. A PID controller calculates an "error" value as the difference between a measured process variable and a desired set point. The controller attempts to minimize the error by adjusting the process control inputs. In the absence of knowledge of the underlying process, PID controllers are the best controllers. However, for best performance, the PID parameters used in the calculation must be tuned according to the nature of the system – while the design is generic, the parameters depend on the specific system.

The PID controller calculation involves three separate parameters, and is accordingly sometimes called three-term control: the proportional, the integral and derivative values, denoted P, I, and D. The proportional value determines the reaction to the current error, the integral value determines the reaction based on the sum of recent errors, and the derivative value determines the reaction based on the rate at which the error has been changing. The weighted sum of these three actions is used to adjust the process via a control element. Heuristically, these values can be interpreted in terms of time: P depends on the present error, I on the accumulation of past errors, and D is a prediction of future errors, based on current rate of change.

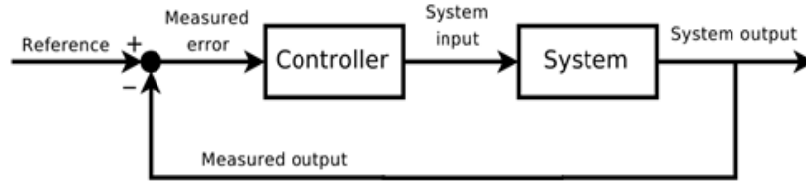


Figure 7.3. PID control

By tuning the three constants in the PID controller algorithm, the controller can provide control action designed for specific process requirements. The response of the controller can be described in terms of the responsiveness of the controller to an error, the degree to which the controller overshoots the set point and the degree of system oscillation.

In this study, the controller measures RTP traffic performance outputs periodically. A constant amount of TCP feedback (the feedback channel that is implemented by me, that measures the parameters and system outputs) packets are waited before the algorithm starts. Hence, the data about data integrity (percentage of the transferred packets to send packets) and the arrival time for each send portion is monitored. From these measurements, differences from the desired value can be obtained and by tuning the parameters of concern; in this specific case; BW according to the PID algorithm;

$$BW = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{d}{dt} e(t) \quad (7.1)$$

optimal system inputs are obtained .The algorithm starts with one parameter at a time and continues for the other parameters, this process is performed continuously in order to have the optimized values for the concerned parameters. The controller tunes the parameters by oscillating them according to the instant system traffic and desired system output values. The algorithm can be examined as following;

```

err = (desired - nowVerim);
firstder = middleerr-olderr;
secondder = err - middleerr;
differr = secondder - firstder;
    if (nterm == 0)    {    totalerr = err;    }
delta = err * KP + totalerr * KI + differr * KD;
PIDout += delta;
  
```

An administrator for the transfer process can manage the flow manually by observing the status for the network elements.

User interface of the implementation for this study provides a panel for controlling the transfer. Settings for the concerned parameters can be done, traffic generator's setting can be done and also participant management can be performed from this panel. The controller administrator has also the list and the information about the RTP receivers, information that indicates the efficiency about that particular user's reception so the user can drop a weak recipient. This property gives the administrator an effective control over the RTP session and enabling the user performs adjustment to enhance the performance.

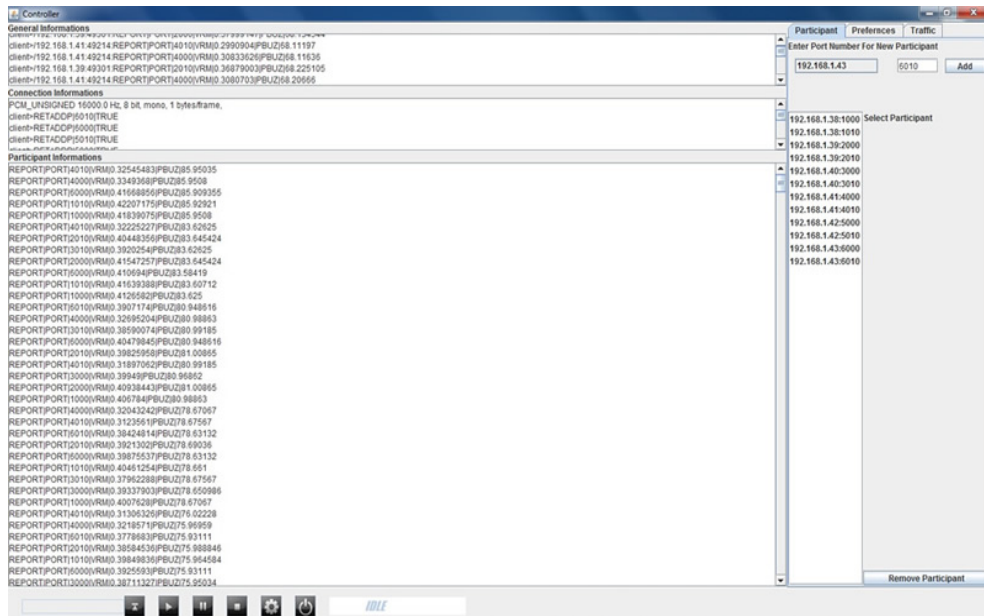


Figure 7.4. Controller user interface

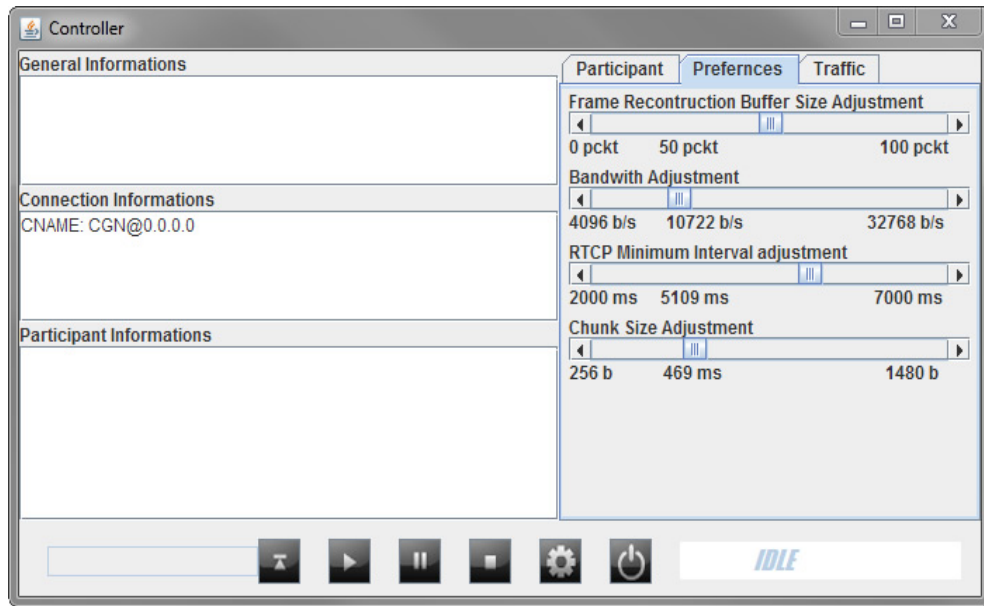


Figure 7.5. Controller user interface, transfer parameters settings panel

7.3.2. Sub-controller

All the receivers for the corresponding related sender use a sub-controller. These elements provide information, report for the controller and apply the comments given by the controller. These reports are used to evaluate the traffic performance and help the controller to adjust the efficiency. Between the main controller and the sub-controller there is a TCP connection for the important, light weight reports and comments.

7.3.3. Traffic Generator

This network traffic simulator is designed to have the tests as similar as possible to the real traffic. This simulator enables the mechanism to perform under real-like environment. Traffic generator produces TCP packages. TCP packages are chosen because of the TCP connection control messages, helping to crowd the traffic. The traffic generator produces TCP packets according to the given assigned period value. For example when 200 is assigned, the generator will produce and send TCP packages every 200 msec.s to the assigned port.

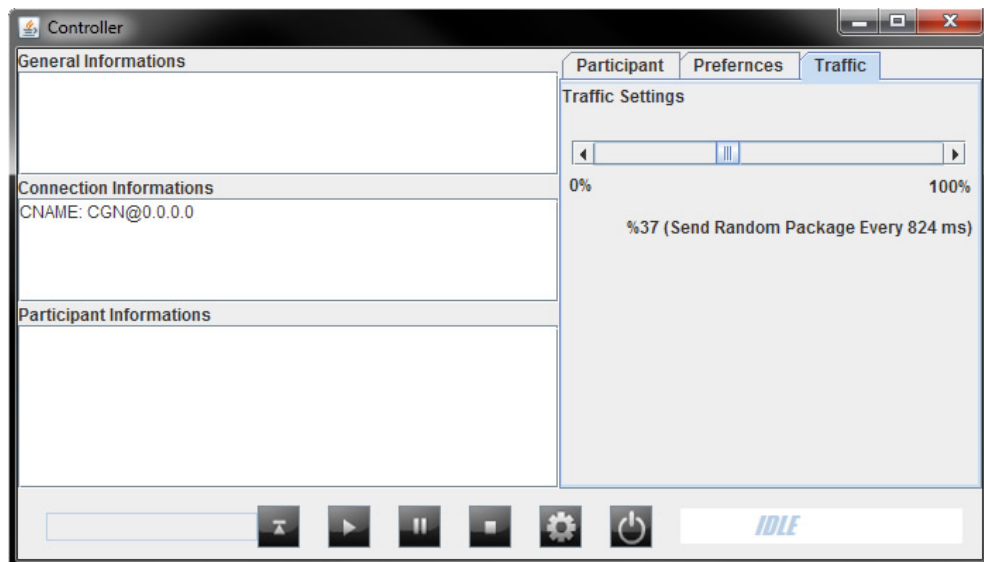


Figure 7.6. Controller user interface, traffic generator settings panel

7.4. Protocol

- ADDP – At RTP receiver side, initiates another listener.
- RETADDP – Confirm for the ADDP
- KILL - At RTP receiver side, terminates the specified listener.
- BW <int> - The integer value for the bandwidth parameter.
- MI <int> - The integer value for the minimum interval parameter
- FR <int> - The integer value for the frame reconstruction parameter.
- CS <int> - The integer value for the chunk size parameter.
- FPSN-First packet sequence number. This value indicates the first real- time data packet received by the RTP receiver. All other measurements take place according to this value. This is because of the fact that every RTP receiver might join to the session hence receive different amounts of the data with different beginning order to have a healthy measurement controller should know the amount of the data and starting point for the measurements.
- REPORT – efficiency and QoS information are carried out.
- The controller measures the efficiency as follows;

- Efficiency = the number of packets arrived / Last packet time stamp – FPSN value.

- For the QoS definition, we introduce a new parameter called PBUZ (paket başı ulaşım zamanı) the time spent for a packets arrival. The shorter time is, the efficient the transfer.

CHAPTER 8

EXPERIMENT AND THE RESULTS

After studying the real-time protocol in theory and in implementation with related library – jlibRTP - and making the observation about the flow of the RTP traffic, the study has come to a theoretical approach to enhance the performance. Next step was to test it for practical accomplishments.

In order to test the control mechanism that has been introduced in the passing chapter, first it is decided that to have RTP traffic flow between hosts in controlled traffic. First of all, a controlled environment for the study is prepared to function. For this purpose, a RTP implementation is chosen, jlibRTP. Being plain structured, open-sourced and improvable made this particular library selected amongst many others. Java programming language is chosen for this study, to implement necessary insufficiencies in the library and the essential codes for the control mechanism and testing.

One way RTP data traffic was generated to keep the simplicity for the simulation. One RTP sender is prepared to send a real-time multimedia data. In this case it is an audio stream to make the test results more sensible. The audio stream is in a specific format for ease of transfer. The format adjustments and encoding issues are not concerned with this study.

On the other side, the other host, RTP receiver is prepared to join the session that is initiated by the RTP sender. The idea is to enable the RTP receiver to join to the session after a random period of time after the session has been started. When this issue is solved, necessary implementations made for being capable to measure the traffic and performance for this – late registration – case. This implementation will be covered in more detail later in this chapter.

Controlled traffic environment is needed to avoid the misleading effects of the real traffic. So a traffic generator is implemented and integrated to the test interface. This generator produces TCP packets and sends them to the RTP receiver via a decided port. The generator produces and sends these packets in user-defined periods. This property gives control of the data traffic amount in the flow at the considered time.

8.1. Experimental Steps

In this section the steps are mentioned during the experiment.

1. RTP sender initiates a session for RTP receivers.
2. A RTP receiver sends a request to join the session. These requests can be sent after the stream has started.
3. Traffic generator starts. The traffic generator starts after the session is accompanied, this is because to make sure of the connection between the sender and receiver.
4. While streaming the audio data, sender plays it while it is sending and also the receiver plays it during reading.
5. During the transfer, periodic measurements for performance in terms of arrived integrated data efficiency and the transport time per packet (PBUZ).

These are calculated according to these formulas;

- Arrived data integrity =

The number of arrived packets / Last packet sequence number – FPSN

- Transport time per packet (PBUZ) =

Last packet timestamp – first packet time stamp / Last packet sequence number – FPSN

Here a new term is introduced, FPSN (first packet sequence number); this is a parameter that is added to the protocol for handling late registration. When a receiver joins to the session after it has started, unless it is not media on demand application, this late registered receiver will start to receive the stream from the point where the sender is currently at. Hence, in order to have correct measurements for performance, it is essential to know the exact packet count that is sent to each receiver.

These measurements took place over the actual traffic without altering the traffic. The data that are needed to make the measurements are taken not from RTCP in order not to affect the performance. There is a messenger thread working in the same manner as RTCP to collect the data for the measurements. This messenger channel is between the controller and sub-controller which are also connected to the sender and receiver. Through this channel sub-controller sends the parameters to the controller indicating the current status of the receiver.

6. In RTP senders GUI, user can follow the current stats about the performance, and the receiver status. Having been able to observe the current status, user can perform the suitable actions to gain an increase. For example, user can vary the traffic amount, change the bandwidth for the flow, chunk size determine the frame reconstruction time and minimum interval for the RTCP packets. These changes are sent to the sub-controller via TCP channel and sub-controller makes the necessary adjustments commended by the controller. All these adjustments can be performed by the auto-control mechanism.

7. During these measurements and adjustments input data for charting graphics are being collected by the chart generator that is integrated into the implementation.

8.2. Exploring the Results

In this section, test results of the thesis work will be displayed and evaluated in form of graphical display.

First, the traffic simulated environment measurements will be presented. Various bandwidth values are tested to see the effects on efficiency. Then, on traffic simulated environment, control mechanism is launched. Meeting the desired outcomes is the goal of this thesis in experimental case.

These measurements are results of a test setting that consist of two laptops, working on windows 7 operating system and a wireless router without internet connection. In local network signal power of the router is reduced in order to support traffic generator to simulate the traffic more efficiently, resulting the packet delays and losses.

It should be kept in mind that, all results introduced are average outcomes of several similar tests with same predefined conditions, like traffic setting, signal power setting and other parameters' settings. As a disadvantage of working with wireless network, tests are mostly unique but they characterize similar.

In order to ease the understanding, the results are presented in result sets, referring each test setting's average.

8.2.1. Result Set 1:

In this set, the aim is to observe the behavior and affect of the bandwidth parameter on the transfer. To achieve this, bandwidth is incremented from the beginning of the transfer, till the end. According outcomes are as follows;

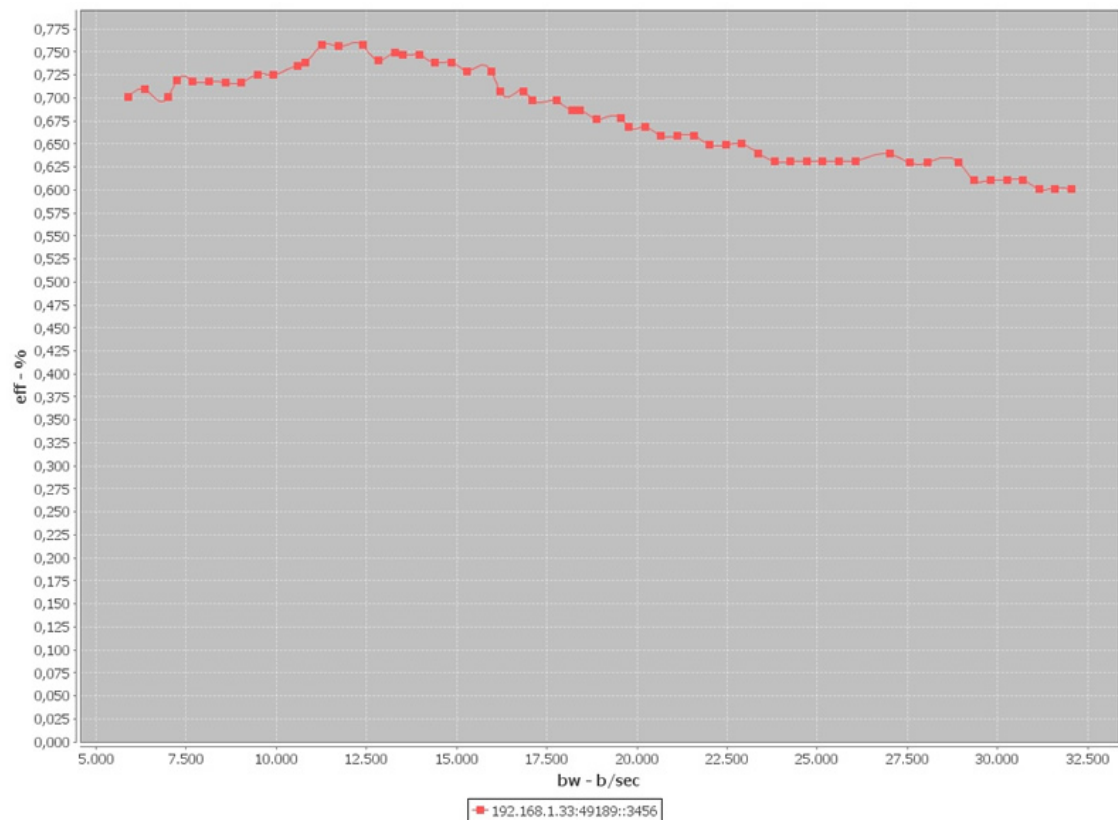


Figure 8.1. Efficiency vs. Bandwidth

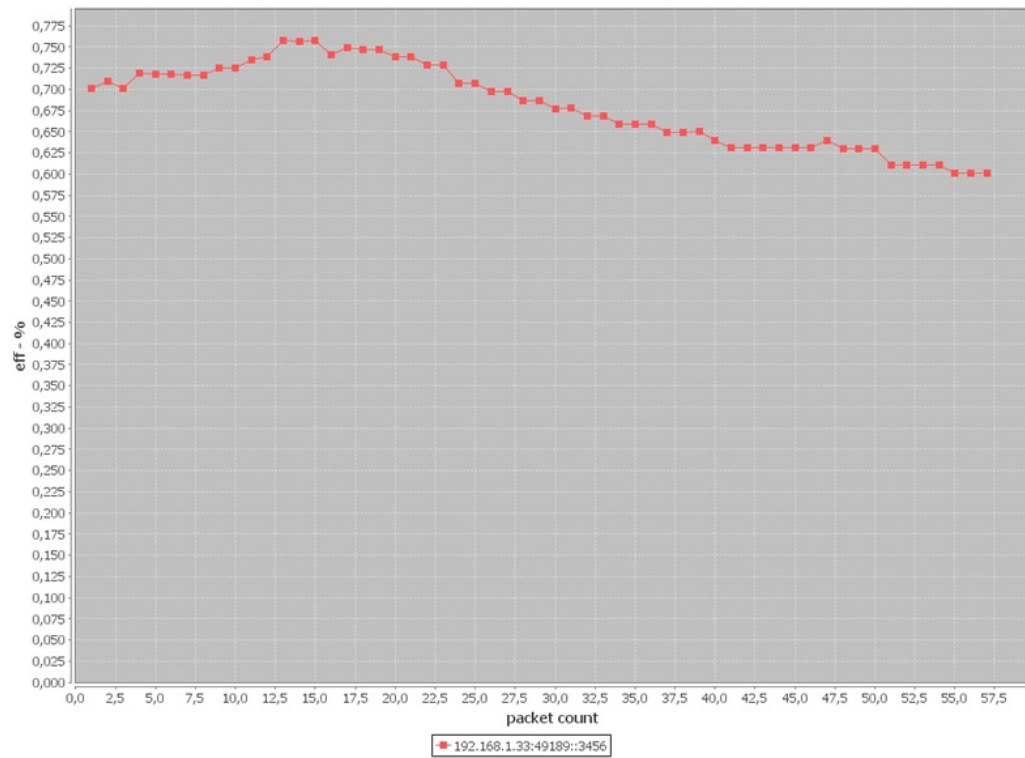


Figure 8.2. Efficiency vs. Time

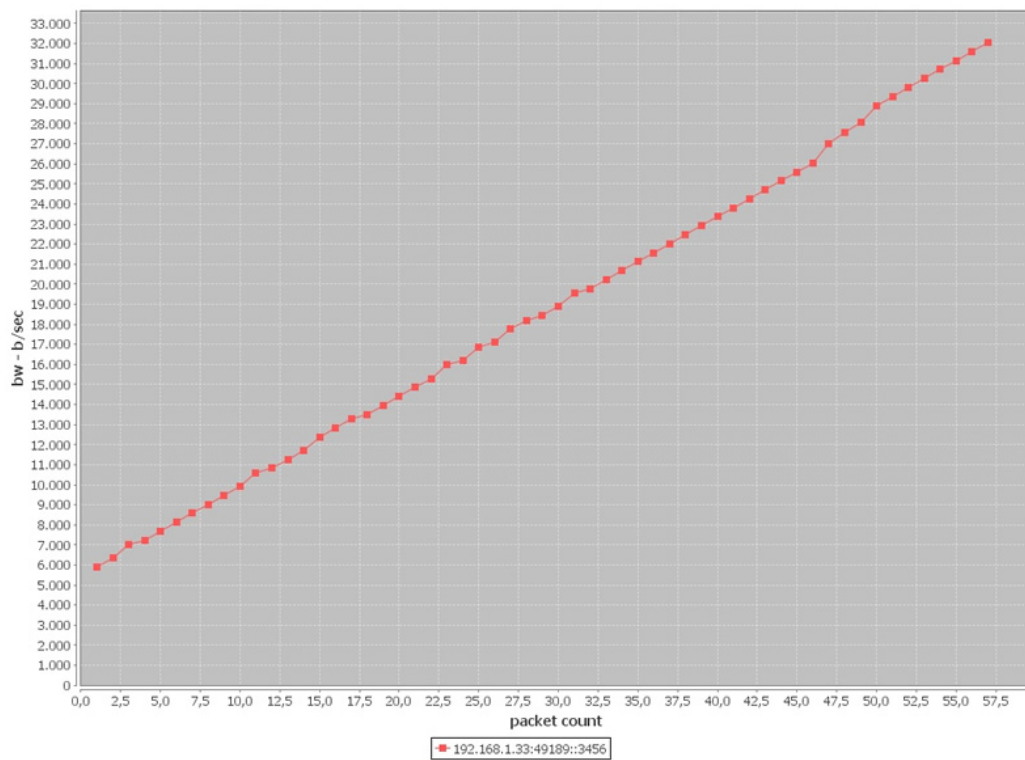


Figure 8.3. Bandwidth vs. Time

As it can clearly be seen; during the transfer of the real time data, RTP bandwidth is increased from 6000 b/sec up to 32.000 b/sec. Efficiency graphics shows that for the particular transfer, the best efficiency is gained by having the bandwidth values between 10.000 b/sec and 15.000 b/sec.

Next step will be choosing some set points to measure the same transfer under same conditions.

8.2.2. Result Set 2:

In this step of the experiment, bandwidth values, 5000 b/sec, 12500 b/sec and 28.000 b/sec are chosen.

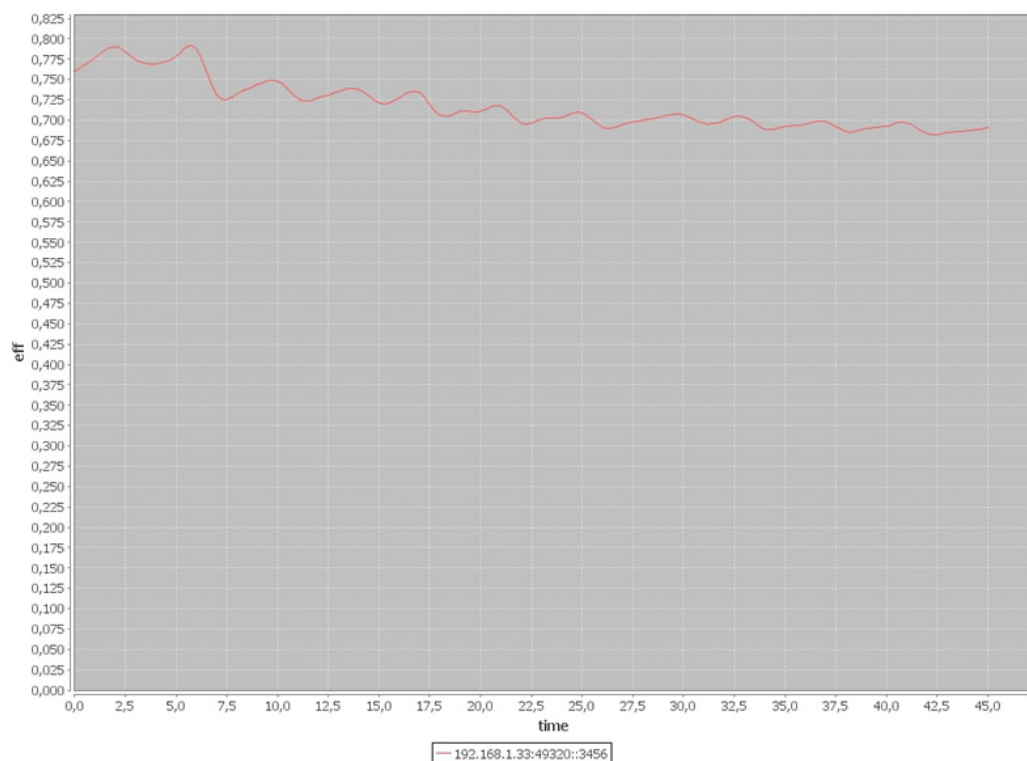


Figure 8.4. Efficiency vs. Time, bandwidth is set to 5000 b/sec.

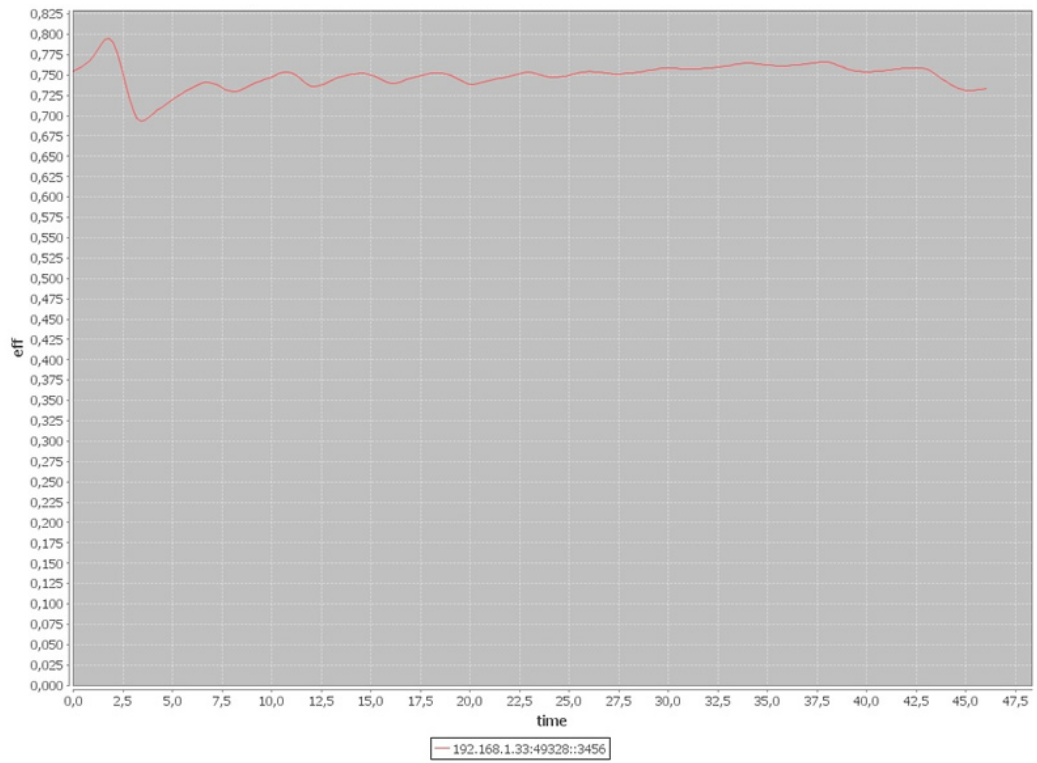


Figure 8.5. Efficiency vs. Time, bandwidth is set to 12500 b/sec.

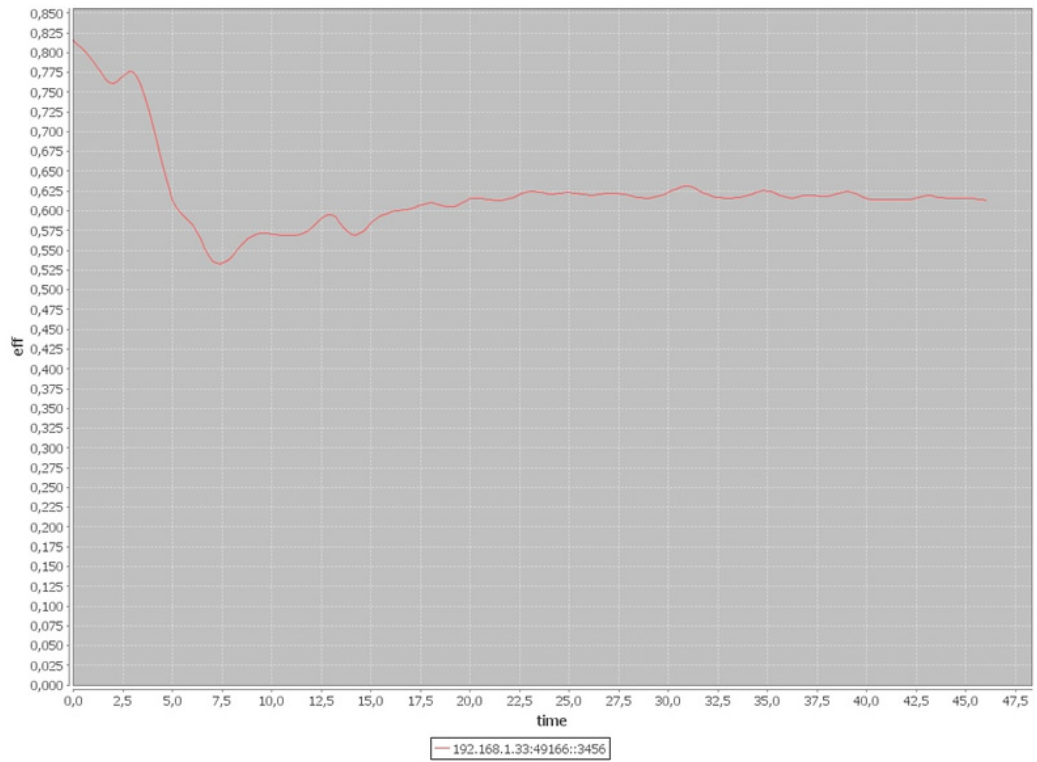


Figure 8.6. Efficiency vs. Time, bandwidth is set to 28000 b/sec.

As it can clearly be seen; during the transfer of the real time data, when RTP bandwidth is set to 5000 b/sec. and 28000 b/sec. transfer efficiency is decreased. Efficiency graphics shows that for the particular transfer, the best efficiency is gained by having the bandwidth set to 12500 b/sec. Next step will be launching the control mechanism and make the same test with same variables.

8.2.3. Result Set 3:

In this step of the experiment, bandwidth value will be set to the library default in the beginning and it is expected that control mechanism will raise the bandwidth value until the desired efficiency value is met and continue to maintain it with adjusting the bandwidth value.

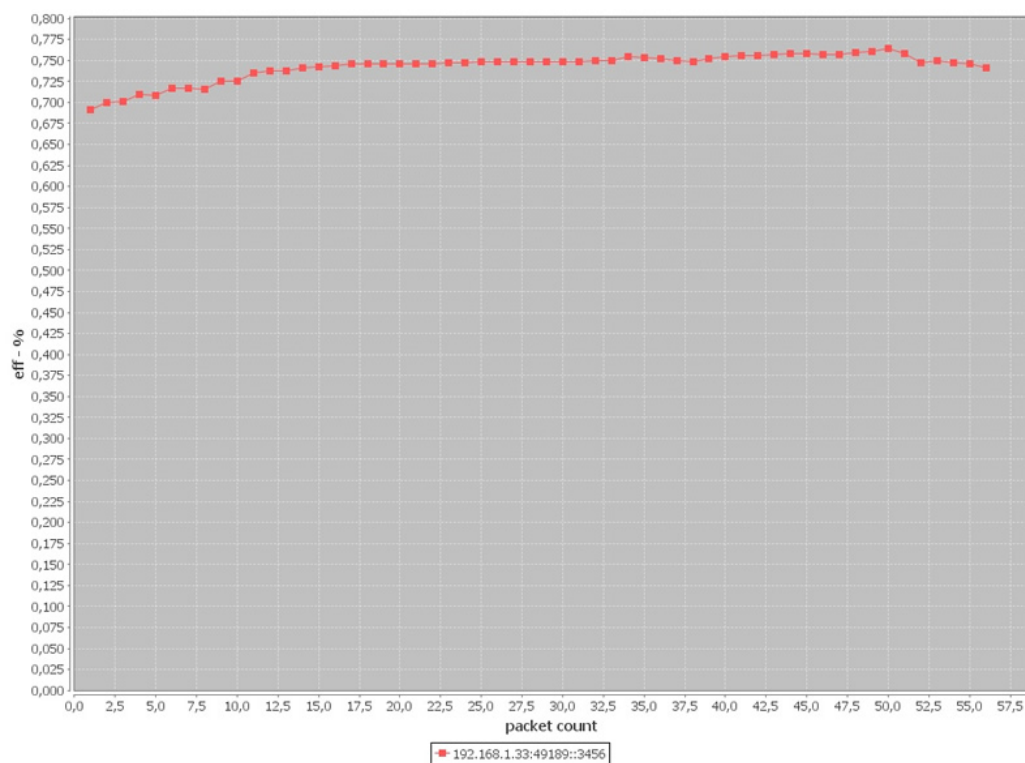


Figure 8.7. Efficiency vs. Time, bandwidth is set dynamically by the mechanism.

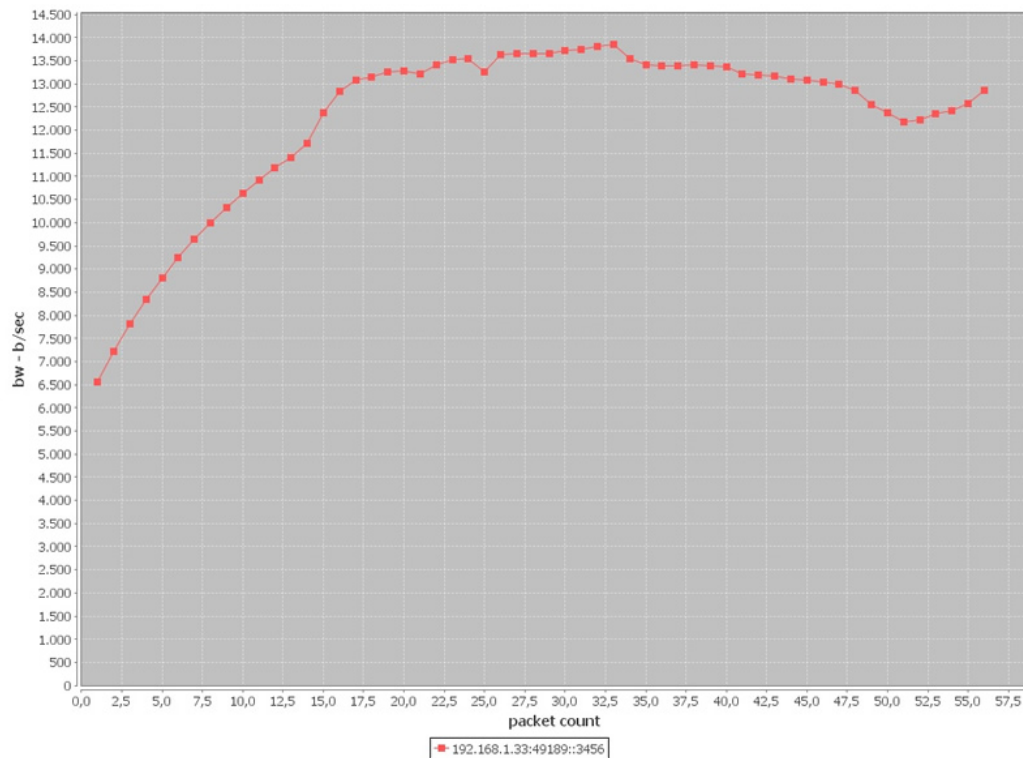


Figure 8.7. Bandwidth vs. Time, bandwidth is set dynamically by the mechanism.

As it can clearly be seen; during the transfer of the real time data, when RTP bandwidth is set to 4000 b/sec. by library default, the efficiency is below 70% so the control mechanism start to increase the bandwidth value until the desired efficiency value, 75% is met. Then control mechanism tries to maintain the efficiency value around 75% by increasing and decreasing the bandwidth value. The mostly calculate value for this transfer, to have the desired efficiency is around 13000b7sec and 13500 b/sec.

As originally observed the optimum interval for this test set up was 10.000 b/sec and 15.000 b/sec. Having efficiency value around 13000 b/sec and in this particular interval proves our mechanism.

CHAPTER 9

CONCLUSION

In this study, gaining advanced knowledge about; real-time traffic, real-time data specifications, real-time transport protocol was desired. Real-time data applications and systems are becoming more popular day by day, in a sense that it is leading in some areas, like media on demand applications, video conferencing, VoIP, etc.

In order to accomplish thesis goals, several RTP libraries are examined. One of them was chosen and modified. Modifications made the existing library more efficient and capable of meeting thesis goals. These modifications made after analyzing and observing the data flow and system workout. After analysis and modifications the main idea of this study – performance enhancement of RTP – became the point of attention. First, performance parameters which are going to be the reference points for this study were determined. Theoretical study leads the idea of optimizing these performance metrics for enhancing the transport performance. The idea of having a control mechanism over RTP was convenient. Essential measurements and adjustment were performed by this mechanism. Last the set up for testing the control mechanism is prepared and as an outcome of the experiment, the collected data indicated the significant increase for the performance is gained, proving my theoretical study valid.

Throughout the study, RFC 3550 was a huge guide for RTP and I hope this thesis study will be an efficient guide for the ones that will make more advanced research on this field.

REFERENCES

- [1] C. Perkins, I. Kouvelas, O. Hodson, V. Hardman, M. Handley, J-C. Bolot, A. Vega-Garcia, and S. Fosse-Parisis, "*RTP Payload for Redundant Audio Data*", RFC 2198, September 1997.
- [2] I. Kouvelas, O. Hodson, V. Hardman, and J. Crowcroft. "*Redundancy control in real-time Internet audio conferencing*". In Proceedings of AVSPN'97, Aberdeen, Scotland, September 1997.
- [3] R.E. Blahut. "*Theory and Practice of Error Control Codes*". Addison Wesley, 1983.
- [4] S. Floyd and K. Fall. "*Promoting the use of end-to-end congestion control in the internet*". Submitted to IEEE/ACM Transactions on Networking, February 1998.
- [5] M. Handley. "*An examination of Mbone performance*". USC/ISI Research Report: ISI/RR-97-450, April 1997
- [6] International Engineering Consortium, 2002. "*Streaming Media Production*", 20/05/2010. http://www.iec.org/online/tutorials/desk_stream.html
- [7] University of Illinois. ION Resources – Streaming Media, 1999. "An Overview of Streaming Media", 20/05/2010. <http://illinois.online.uillinois.edu/IONresources/tutorials/streamingMedia/Introduction.html>
- [8] S. Gibbs, C. Breiteneder, D. Tsichritzis. "*Data Modelling of Time-Based Media*", 1993.
- [9] Network World, Inc. 1998. "*Network World Fusion. Script: Streaming Media Audio Primer*", 20/05/2010. <http://www.nwfusion.com/primers/streaming/streamingscript.html>
- [10] B.O. Szuprowicz, "*Multimedia Networking*". McGraw-Hill, Inc. USA. 1995.
- [11] A. Vogel, B. Kerherve, G. Bochmann, J. Gecsei, Distributed Multimedia and QOS, *A Survey in IEEE MultiMedia*, 2(2), 1995, 10-19.
- [12] J-C. Bolot and A. Vega-Garcia. "*The case for FEC based error control for packet audio in the Internet*". To appear in ACM Multimedia Systems.
- [13] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "*RTP: A transport protocol for real-time applications*", RFC 1889, January 1996.
- [14] R. Shirdokar, J. Kabara, and P. Krishnamurthy, "*A QoS-based Indoor Wireless Data Network Design for VoIP Applications*". IEEE Publications, 2001.

- [15] J. Rosenberg and H. Schulzrinne. An A/V profile extension for generic forward error correction in RTP.
- [16] C. Bormann, L. Cline, G. Deisher, T. Gardos, C. Maciocco, D. Newell, J. Ott, S. Wenger, and C. Zhu, RTP payload format for the 1998 version of ITU-T recommendation H.263 video (H.263+).
- [17] L. Vicisano, L. Rizzo, and Crowcroft J, TCP-like congestion control for layered multicast data transfer. *Proc. IEEE INFOCOM'98, 1998*.
- [18] F. Fluckiger, “*Understanding Networked Multimedia Application and Technology*”. Prentice Hall. Great Britain. 1995.
- [19] Geiger, J. INT Media Group. 15 April 2002. “*Making the Choice: 802.11a or 802.11g*”, 20/05/2010.
http://www.80211-planet.com/tutorials/article/0,4000,10724_1009431,00.
- [20] CISCO Thought Leadership Series. Wireless LAN Benefits, Study Conducted by NOP World – Technology on Behalf of CISCO Systems. USA. Fall 2001.
- [21] J.L.A. Fonseca, M.A. Stanton, “*A Methodology for Performance Analysis of Real-Time Continuous Media Applications*”.12th International Workshop on Distributed Systems:Operations and Management DSOM'2001 Nancy France, 2001.
- [22] J.F. Kurose, K.W. Ross, “*Computer Networking : A Top Down Approach Featuring The Internet*”. Addison Wesley Longman, Inc. USA, 2001.
- [23] G. Held, “*Voice and Data Internetworking*”. McGraw-Hill, 2000.
- [24] U. Schwantag, USA. 1997.”*Measuring Quality of Service Parameters*”.
20/05/2010.<http://ns.uoregon.edu/ursula/thesis/node23.html>
- [25]H. Schulzrinne, S. Casner, S. Frederick, and V. Jacobson, RFC 1889: RTP: A Transport Protocol for real-time applications. IETF. January 1996. pp.75.
- [26] S. Jha, and M. Hassan, “*Engineering Internet QoS*”. Artech House. ISBN: 1580533418.2002.
- [27] B. Goode, Voice Over Internet Protocol (VoIP). *Proc. The IEEE. Volume 90. No. 9. September 2002*.
- [28] A.P. Markopoulou, F.A. Tobagi, and M.J. Karam, Assessing the Quality of Voice communications over Internet Backbones, *IEEE/ACM Transactions on Networking, Vol.11, No. 5, October 2003*.
- [29] A. Kos, B. Klepec, and S. Tomažič, “*Techniques for Performance Improvement of VoIP Applications*”. IEEE MELECON 2002. Cairo, Egypt. May 2002.
- [30] J. Feigin, K. Pahlavan, and M. Yliantilla, “*Hardware-Fitted Modeling and*

Simulation of VoIP over a Wireless LAN". IEEE, VTC'2000, Boston, IEEE Publications, 2000.