IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 3, Ver. II (May-Jun. 2014), PP 78-84
www.iosrjournals.org

Real Time Communication over Modified UDP Protocol

Ishant Raj, Kamaljeet Singh, Rahul Raj, T. J. Parvat
[1](Department of Computer Engineering, SIT Lonavala, India)
[2](Department of Computer Engineering, SIT Lonavala, India)
[3](Department of Computer Engineering, SIT Lonavala, India)
[4](Department of Computer Engineering, SIT Lonavala, India)

Abstract*: The objective of this paper is to provide a uniform means of transmitting data (audio, video) to real time constraints over multicast or unicast network services and providing much enhanced experience of the Real-time communication over the existing systems. In this paper we will explore usage of the dominating technologies in the real time systems and shows how to design and develop communications within such systems. In particular, it proposes reliable protocol over UDP transport layer that gains less than one millisecond average round trip time (RTT) and processes dozens of thousands of messages every second at each and every port in network. This paper presents the current internet protocols: Real-time Transport Protocol (RTP), RTP Control Protocol (RTCP) and Real-time Streaming Protocol (RTSP). Together these protocols may provide controlled delivery of multimedia traffic over the Internet. The feedback mechanisms for the quality of service (QoS) monitoring, such as delay, jitter and packet loss calculations, are described in detail.*

Keywords : *header compression; jitter; multimedia traffic; real-time;RTP;RTCP;*

I. INTRODUCTION

Fast and reliable communications is a basic requirement in almost all modern applications, but the real time systems take it to extreme and require real time responses from the network. In some cases, the requirements are so tough that may require special hardware to achieve desired performance .In the past twenty-five years several different, wired networks have been designed and built. But not all of them are dependable, i.e. suitable for accommodating real-time traffic, or traffic with Quality of Service (QoS) requirements. Some were designed with QoS in mind, some had QoS added as an after-thought, and some do not support QoS requirements at all. Multimedia services, such as video conferencing, Internet telephony and streaming audio/video, have recently been introduced for the millions of users of the Internet. The popularity of these services and the feedback received has clearly revealed that some modifications and extensions to the current internet protocols are needed to be able to support real-time applications better. Minimization of the end-to-end delay, accurate synchronization of the voice and video streams and a feedback mechanism for the quality of service monitoring are some of the main requirements of these various multimedia applications.

In the process of transmitting real-time communications from one point to another, multiple steps are involved and various protocols are used. First, some type of signaling and call control is needed to establish, modify, and terminate a call. Within the public switched telephone network (PSTN), a circuit-switched network, Signaling System 7 (SS7) is used for call setup and termination. For packet-based networks, both the SIP and H.323 protocols provide call control. After the calling session is established, the audio or video input needs to be sampled and converted to a digital format. Next, the sampled data is encapsulated into Real-time Transport Protocol (RTP) packets. RTP is specifically designed for the needs of real-time communication over a packet-based network.

Then, the RTP packet is encapsulated into a network transport protocol, which is most often the User Datagram Protocol (UDP). Alternatively, the Transmission Control Protocol (TCP) can be used for encapsulation; however, because TCP is a guaranteed transport-level protocol, the additional time needed to occasionally retransmit TCP packets can add enough latency (the time between sending and receiving packets) to the transmission so that the received audio is unintelligible. Throughout the transmission of the RTP packets, the Real-time Control Protocol (RTCP) is used to monitor the quality of an RTP session.

Next, the network transport protocol, UDP or TCP, is encapsulated into an IP packet, which is then encapsulated into the link layer protocol — Ethernet, for example. The link layer packet is then transmitted to the destination computer(s). Figure 1 shows the encapsulation process, from the encapsulation of the RTP packet to encapsulation of the link layer packet.

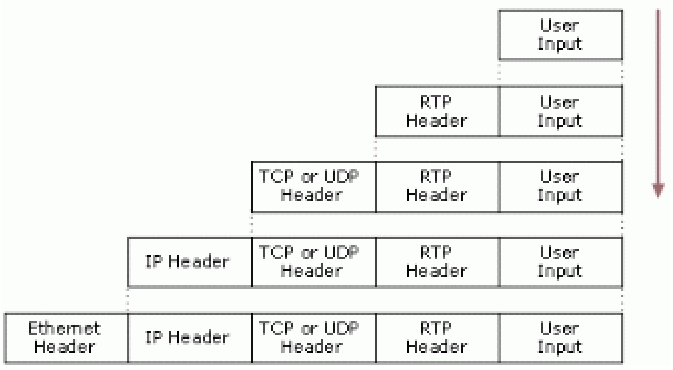


Fig 1: Real-Time Communication Protocol Encapsulation

II. RTP

RTP [4] is a real-time end-to-end transport protocol. However, considering RTP as a transport protocol may be misleading because it is mostly used upon UDP, which is also considered as a transport protocol. On the other hand, RTP is very closely coupled to the Auctioning Scheme application it carries. So, RTP is best viewed as a framework that applications can use to implement a new single protocol. RTP doesn't guarantee timely delivery of packets, nor does it keep the packets in sequence. RTP gives the responsibility for recovering lost segments and re-sequencing of the packets for the application layer. There are a couple of benefits in doing so. The application may accept less than perfect delivery and with video or speech there usually is no time for retransmissions. Also the sender may provide, instead of retransmission, new or updated data that tries to fix the consequences of the original loss. What RTP then provides, is:

- Payload type identification
- Source identification
- Sequence numbering
- Time stamping

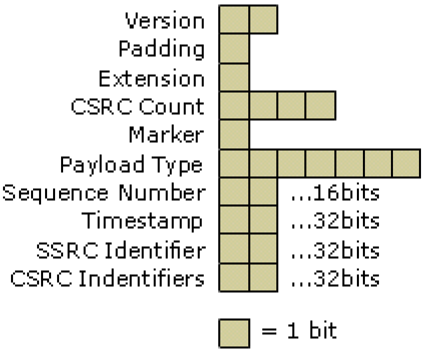


Fig 2: RTP Packet Format

III. RTCP

The RTP data transport is augmented by a control protocol (RTCP), which provide RTP session participants feedback on the quality of data distribution. The underlying protocol must provide multiplexing of the data and control packets, with UDP this is usually implemented using separate port numbers. The format of RTCP is similar to RTP, type indication is at same position.

- QoS monitoring and congestion control
- Session size estimation and scaling
- Identification

The RTCP [1] packets contain direct information for quality-of-service monitoring. The sender reports (SR) and receiver reports (RR) exchange information on packet losses, delay and delay jitter. This information may be used to implement a TCP like flow control mechanism upon UDP at the application level using adaptive encodings. A network management tool may monitor the network load based on the RTCP packets without receiving the actual data or detect the faulty parts of the network. The RTCP packets carry also a transport-level identifier (called a canonical name) for a RTP source, which is used to keep track of each participant. Source description packets may also contain other textual information (user's name, email address) about the source.

Albeit the source of the RTP packets is already identified by the SSRC identifier, an application may use multiple RTP streams, which can be easily associated with this textual information. The RTCP packets are sent periodically by each session member in multicast fashion to the other participants. The more there are participants the more RTCP messages should be exchanged. That's why the fraction of the control traffic must be limited. There is in fact a trade-off between up-to-date information and the amount of the control traffic. The control traffic load is scaled with the data traffic load so that it makes up about 5% of the total data traffic.

Each RTCP [5] packet starts with a header similar to that of RTP data packet. The payload type field identifies the type of packet, there are 5 RTCP payload types (200-204) defined:

- Sender Report (SR):-It contains information regarding the quality of the RTP session.
- Receiver Report (RR):-It contains information regarding the quality of the RTP session.
- Source Description (SDES):-It contains information regarding the identity of each participant in the RTP session.
- Goodbye (BYE):-It indicates that one or more sources are no longer active in the RTP session.
- Application-defined packet (APP):-It is for experimental use by new applications.

Participants in an RTP session send RR packet types, and, if they are active senders, send SR packet types. The RR packet has two sections, the header and report blocks, as shown in Fig 3. There is one report block for each source.

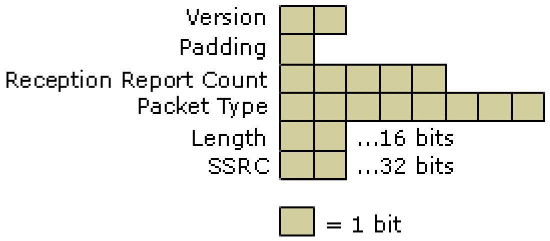


Fig 3: RTCP RR and SR Header Format

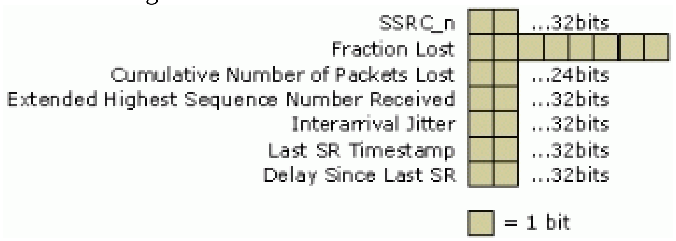


Fig 4: RTCP Report Block

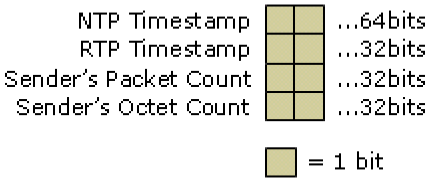


Fig 5: RTCP SR Information

IV. RTSP

RTSP is a client-server multimedia presentation control protocol .RTSP is an Internet protocol specification which define the different aspects which make out a streaming session between a streaming RTSP/RTP server and a player client. There are other media streaming protocols in operation for example the big bunch of different streaming methods which came up with the new HTML5 standard. We just deal here with RTSP and RTP.

RTSP (Real Time Streaming Protocol) is defined in the Internet standard draft RFC 2326 [10]. Despite of its name RTSP actually does not transport any media like video and audio. RTSP is a companion protocol for RTP which is doing the actual work of the media transport. RTSP covers all aspects for the control of an RTP streaming session. A player connects to the RTSP connection handler of a streaming server and exchanges RTSP requests with the server. These requests and their responses are defined in RFC 2326. For example the RTSP requests are used to

- select the streams which the client would like to play,
- query the format of the stream (codecs) and the transport method (for example UDP or TCP based),
- start and stop a streaming session

An RTSP/RTP server starts a stream after some requests from the client when the client sends an RTSP PLAY request. To handle RTSP requests from the client a streaming server must implement an RTSP parser to interpret the client messages. In RTSP each session has a state. In this it is required to keep the client's state up-to-date. Client changes state when it receives a reply from the server according to the following state diagram (fig 6).

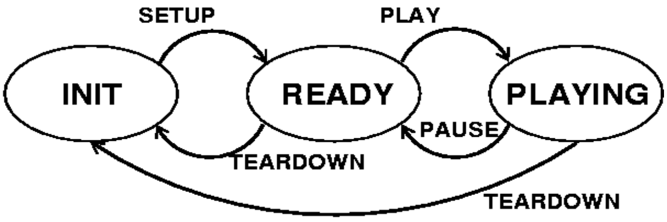


Fig 6: State Diagram

V. MULTIMEDIA TRAFFIC

Streaming Live A/V: (Like broadcast TV/radio but via Internet, Can't pause, rewind. The time between request and display 1 to 10 seconds .Continuous playout).

Streaming Stored A/V: (Like DVD but via Internet Prerecorded multimedia content, user may pause, rewind, Forward, The time between request and display 1 to 10 seconds after display start the playout must be continuous)

NON REAL TIME COMMUNICATION

Consider One Example in fig (7) of downloading a video from internet. The Video has already been made or exist in Server System .A Client HTTP is used to access the downloaded video to the Client Site. The Production, transmission, and use all happen at different times.

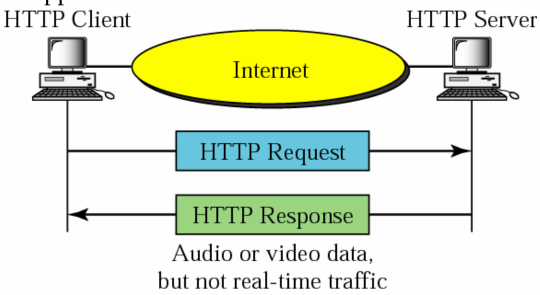
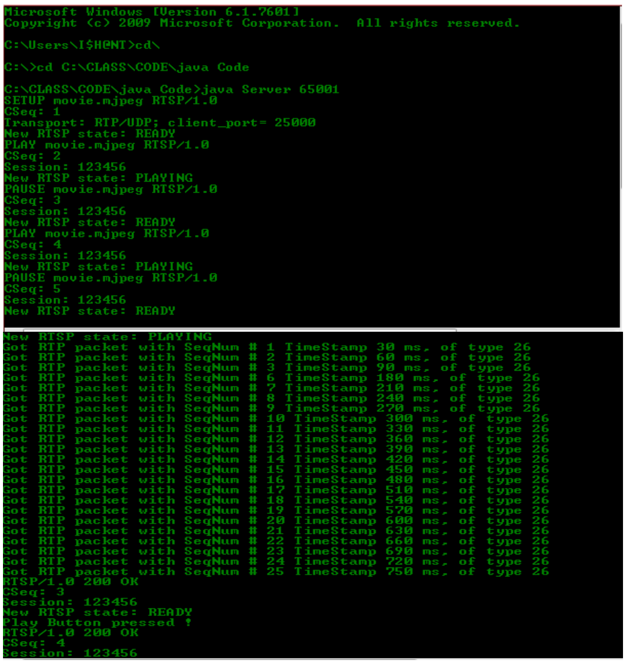


Fig 7: Non Real-time Traffic

The Snapshot tells how Packet is transfer and receive from Server to client using cmd in fig (8).



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\I$H@NT>cd\

C:\>cd C:\CLASS\CODE\java Code

C:\CLASS\CODE\java Code>java Server 65001
SETUP movie.mjpeg RTSP/1.0
CSeq: 1
Transport: RTP/UDP; client_port= 25000
New RTSP state: READY
PLAY movie.mjpeg RTSP/1.0
CSeq: 2
Session: 123456
New RTSP state: PLAYING
PAUSE movie.mjpeg RTSP/1.0
CSeq: 3
Session: 123456
New RTSP state: READY
PLAY movie.mjpeg RTSP/1.0
CSeq: 4
Session: 123456
New RTSP state: PLAYING
PAUSE movie.mjpeg RTSP/1.0
CSeq: 5
Session: 123456
New RTSP state: READY
```

```
New RTSP state: PLAYING
Got RTP packet with SeqNum # 1 TimeStamp 30 ms, of type 26
Got RTP packet with SeqNum # 2 TimeStamp 60 ms, of type 26
Got RTP packet with SeqNum # 3 TimeStamp 90 ms, of type 26
Got RTP packet with SeqNum # 6 TimeStamp 180 ms, of type 26
Got RTP packet with SeqNum # 7 TimeStamp 210 ms, of type 26
Got RTP packet with SeqNum # 8 TimeStamp 240 ms, of type 26
Got RTP packet with SeqNum # 9 TimeStamp 270 ms, of type 26
Got RTP packet with SeqNum # 10 TimeStamp 300 ms, of type 26
Got RTP packet with SeqNum # 11 TimeStamp 330 ms, of type 26
Got RTP packet with SeqNum # 12 TimeStamp 360 ms, of type 26
Got RTP packet with SeqNum # 13 TimeStamp 390 ms, of type 26
Got RTP packet with SeqNum # 14 TimeStamp 420 ms, of type 26
Got RTP packet with SeqNum # 15 TimeStamp 450 ms, of type 26
Got RTP packet with SeqNum # 16 TimeStamp 480 ms, of type 26
Got RTP packet with SeqNum # 17 TimeStamp 510 ms, of type 26
Got RTP packet with SeqNum # 18 TimeStamp 540 ms, of type 26
Got RTP packet with SeqNum # 19 TimeStamp 570 ms, of type 26
Got RTP packet with SeqNum # 20 TimeStamp 600 ms, of type 26
Got RTP packet with SeqNum # 21 TimeStamp 630 ms, of type 26
Got RTP packet with SeqNum # 22 TimeStamp 660 ms, of type 26
Got RTP packet with SeqNum # 23 TimeStamp 690 ms, of type 26
Got RTP packet with SeqNum # 24 TimeStamp 720 ms, of type 26
Got RTP packet with SeqNum # 25 TimeStamp 750 ms, of type 26
RTSP/1.0 200 OK
CSeq: 3
Session: 123456
New RTSP state: READY
Play Button pressed !
RTSP/1.0 200 OK
CSeq: 4
Session: 123456
```

Fig 8: Client Server Communication Using RTP

REAL TIME COMMUNICATION

Consider one Example of Video Conferencing in fig (9) which a camera is connected to server that transmits video information as it produced. Everything that happens at the server site can be displayed on the computer at the client site. This is both multimedia (video) and real-time traffic (production and use at the same time).

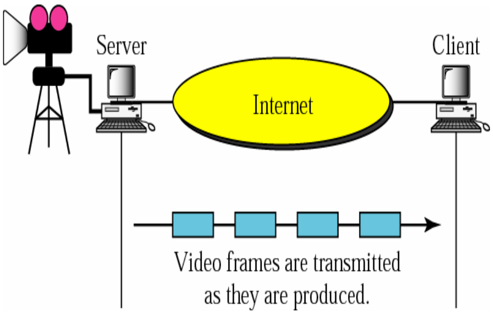


Fig 9: Real-time Traffic

This is Real time communication, using webcam we use to connect client-server. For this java media Framework (JMF) is used. The basic source code of webcam is as

CaptureDeviceInfo webcamInfooo = new CaptureDeviceInfo ("Camera", newMediaLocator ("vfw: //0"), null);
DeviceListVector = CaptureDeviceManager.getDeviceList (null);
MediaLocator webcamMediaLocator=webcamInfooo.getLocator ();

VI. ROUND TRIP DELAY

Receiver reports may be used to estimate the round-trip delay between sender and receiver. The receiver report includes the LSR (timestamp from the last sender report received) and DLSR (delay since last sender report received) fields, from which the sender can directly calculate the round-trip delay according the formula 1,where A is the time instant when the receiver report was received by the sender.

$$D = A\text{-}LSR\text{-}DLSR \quad (1)$$

The fig (10) shows the round-trip calculation against the time axis. The middle 32-bits of NTP timestamp are copied by the receiver to LSR field and the delay since last particular sender's report is stored until a corresponding receiver report is sent. It should be noted that as the minimum interval between consecutive reception reports is defined to be 5s, the delay estimate cannot be used as a real-time measure.

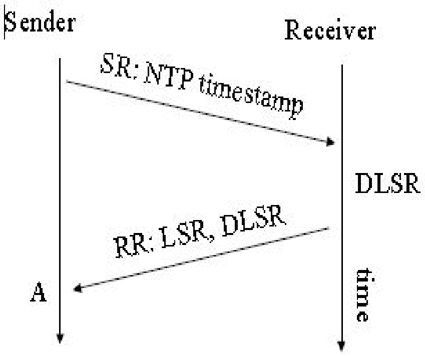


Fig 10: Round-trip Calculation

VII. INTER-ARRIVAL JITTER

RTP and RTCP provide information, such as time stamps and inter arrival jitter values that real-time communications applications can use to compensate for jitter during a session. An application's jitter buffers use the time stamps and inter arrival jitter values to make adjustments so that a smooth, even flow of packets is received. Applications use the information received from the RTP and RTCP packets to calculate the difference in transit time for two packets. The calculation they use is:

$$D\ (n,\ n\text{-}1) = [R\ (n)\text{-}S\ (n)]\text{-}[R\ (n\text{-}1)\text{-}S\ (n\text{-}1)]$$

Where D(n,n-1) is the difference in transit time for packets n and n-1, S represents the time when packets (n,n-1) were sent, and R represents the time when packets (n,n-1) were received. The difference in transmit time, D (n, *n-1*), is then used in the following formula, as described in RFC 1889 [4], “RTP: A Transport Protocol for Real-Time Communications,” to determine inter arrival packet jitter, J (*n)*, as a smoothed running value of an RTP session:

$$J(n) = J(n-1) + (|D(n, n-1)| - J(n-1))/16$$

It is proposed that the change in this jitter estimate could indicate congestion before it leads to packet loss.

VIII. PACKET LOSS

The receiver reports also contain information about the lost packets. The fraction of lost packets is defined to be the number of packets lost divided by the number of packets expected, which are calculated based on actually received packets and the highest sequence number received in RTP packets. A cumulative number of packets lost is also maintained. These packet loss measures may be used as congestion indication for the sender to reduce the application's sending rate.

IX. Quality Of Service

The RTP and RTCP protocols, jitter control mechanisms, and acoustic echo canceller provide applications with information and tools to monitor and improve the quality of real-time communications; however, none of these protocols or technologies has control over the underlying networking environment. QoS, a combination of IETF-defined protocols, such as Differentiated Services (Diff-Serv) and 802.1p, is used to provide different levels of control over the underlying networking environment and to provide varying degrees of quality of service.

On receiving a RTCP receiver report, the sender analyses the packet loss and delay measures and classifies each receiver either as unloaded, loaded or congested. After that the sender's bandwidth is adjusted. The adjustment is done either based on the receiver with the highest average loss rate or based on certain proportions of unloaded, loaded and congested receivers. The bad thing in the former approach is that a receiver with a low speed link may provide low quality also to all the other receivers. On the other hand, the latter approach will let some amount of congested receivers to suffer continuously. The bandwidth is adjusted using a multiplicative decrease but only an additive increase to be able to react rapidly to congestion, still being beware of too rapid increase after the congestion.

X. Header Compression

The 12-byte RTP header together with 20-byte TCP and 20-byte IP-header produces a quite high overhead to the payloads. This overhead becomes a major problem in low-bandwidth links such as dial-up modems at 14.4 or 28.8 kbps. The brand-new RFC 2508 [9] presents a compression method which reduces the RTP/TCP/IP header to only two bytes for the most packets. The main idea is that half of the bytes of the TCP and IP headers remain constant over the life of the connection. After sending the uncompressed header once, these fields may be dropped off from the compressed headers that follow. From the RTP headers it can be seen that although several fields change in every packet, i.e. sequence number and timestamp, the difference from packet to packet is often constant and of good use in compressor and decompressor. It is stated that there is no use of compressing RTCP packets, which constitute only 5% of the bandwidth. Also additional memory for saving the context of SDES items should be needed.

XI. Conclusion

The RTP protocol seems to suite the delivery of the real time traffic pretty well. The RTP protocol provides timing information and the identification of the source and the payload type for the multimedia applications. The accompanying control protocol, RTCP, provides information about the perceived quality of service. However, there are some limitations in the scalability of the RTP scheme. Header overhead may become a problem on low-speed links or on large trunk lines. Thus, a header compression scheme and a user multiplexing scheme are presented. Also the amount of the control traffic may need to be limited. Large multicast groups may utilize timer reconsideration and enhanced group membership sampling to avoid congestion and memory problems.

Acknowledgements

We thank first and foremost thank our teachers Prof. T. J. Parvat, Prof. J. P. Chavan, Prof. V. N. Dhawas for their support throughout the development of the project. We also thank our guide from DRDO Mrs. Nirja Thakur for her cooperation and continuous efforts in the project. We also thank all our friends, colleagues and professors who have helped us in the development of this system.

REFERENCES

[1] Audio/Video Transport (AVT) Working Group, IETF. http://www.ietf.org/html.charters/avtcharter.html

[2] _ITU-T Recommendation H.323: Packet-Based Multimedia Communications Systems. February 1998.

[3] ITU-T Recommendation H.225.0: Media Stream packetization and Synchronization on Non- Guaranteed Quality of Service LANs. November 1996.

[4] RTP: A Transport Protocol for Real-Time Applications. IETF RFC 1889. January 1996.

[5] RTP Profile for Audio and Video Conferences with Minimal Control. IETF RFC 1890. January 1996.

[6] Rosenberg, J. Schultzrinne, H.Timer Reconsideration for Enhanced RTP Scalability. IETF Draft. July 1997.

[7] Schultzrinne, H. Rosenberg, J. Sampling of the Group Membership in RTP. IETF Draft. November 1998.

[8] Busse, I. Deffner, B. Schultzrinne, H. Dynamic QoS Control of Multimedia Applications based on RTP. Computer Communications. January 1996.

[9] Compressing IP/UDP/RTP Headers for Low-Speed Serial Links. IETF RFC 2508. February 1999.

[10] RTSP: A Transport Protocol for Real-Time Streaming Applications. IETF RFC 2326