Telecommunication Standardization Sector **AVC-866**

Original: English

SOURCE : Ami Amir & Yizhak Idan RADVision Ltd.
e-mail:amir@radvision.rad.co.il or idan@radvision.rad.co.il
voice: +972 3 645 5220
fax: +972 3 647 6669

TITLE : Adaptation of RTP to H.323

DATE: December 27, 1995

The following is a proposal for the RTP Annex of H.225.0

This AVC is based on the Nov 20, 1995 Internet Draft of Audio-Video Transport WG draft-ietf-avt-rtp-08.txt by Schulzrinne/Casner/Frederick/Jacobson.

Annex A of this document defines the H.323/H.225.0 RTP profile.

Major changes to the RTP document include:

1. Removal of the RTP annexes A to D which deal with implementation recommendations,
2. Removal of Chapter 9 (Security)
3. Definition of RTCP as optional. Using two control protocols is a sure mechanism to get into deadlocks. For example the RTCP BYE command can be used to close a channel and the same can be done by H.245. There should be an overriding of such RTCP commands in channels controlled by H.245.
4. Removal of refrences to mixers, since we feel that mixers should not be implemented,
5. Removal of what we feel are redundant parts, in an attempt to shorten the document (and make Mr. Bigi happy).

~~Internet Engineering Task Force~~ ~~Audio Video Transport WG~~
~~Internet Draft~~ ~~Schulzrinne/Casner/Frederick/Jacobson~~
~~draft-ietf-avt-rtp-08.txt~~ ~~GMD/ISI/Xerox/LBNL~~
~~November 20, 1995~~
~~Expires: 3/1/96~~

RTP: A Transport Protocol for Real-Time Applications

ABSTRACT

RTP provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services. RTP does not address resource reservation and does not guarantee quality-of-service for real-time services. The data transport is augmented by a control protocol (RTCP) to allow monitoring of the data delivery in a manner scaleable to large multicast networks, and to provide minimal control and identification functionality. RTP and RTCP are designed to be independent of the underlying transport and network layers. The protocol supports the use of RTP-level translators and mixers.

This Annex to H.225.0 ~~specification~~ is derived from a specification which is a product of the Audio/Video Transport working
~~—~~group within the Internet Engineering Task Force.

1 Introduction

This Annex specifies the real-time transport protocol (RTP), which 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, timestamping and delivery monitoring. Applications typically run RTP on top of UDP to make use of its multiplexing and checksum services; both protocols contribute parts of the transport protocol functionality. However, RTP may be used with other suitable underlying network or transport protocols (see Section 10). 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.

This document defines RTP, consisting of two closely-linked parts:

o the real-time transport protocol (RTP), to carry data that has real-time properties.

o the RTP control protocol (RTCP), to monitor the quality of service and to convey information about the participants in an on-going session. The latter aspect of RTCP may be sufficient for "loosely controlled" sessions, i.e., where there is no explicit membership control and set-up, but it is not necessarily intended to support all of an application's control communication requirements. This functionality may be fully or partially subsumed by a separate session control protocol, which is beyond the scope of this document. The use of RTCP is optional.

~~which is beyond the scope of this document~~.

In addition to this document, a complete specification of RTP for a particular application will require one or more companion documents (see Section 12):

o a profile specification document, which defines a set of payload type codes and their mapping to payload formats (e.g., media encoding). A profile may also define extensions or modifications to RTP that are specific to a particular class of applications. Typically an application will operate under only one profile. A profile for audio and video data may be found in the companion Annex A~~RFC TBD.~~

o payload format specification documents, which define how a particular payload, such as an audio or video encoding, is to be carried in RTP. Payload format is defined in Annex yyy.

2.~~1 Mixers and~~ Translators
Mixers

~~So far, we 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.~~
(There is no need for Mixers in H.323 - centralized MCUs can be viewed as representing H.323 terminals - and appear as RTP endpoints)

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~~ a 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.~~ An H.323 gateway is an RTP Translator.

Details of the operation of ~~mixers and~~ translators are given in Section 7.

3 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 (see below), and the payload data. Some underlying protocols may require an encapsulation othe 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 (see Section 10).

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. The formats are defined in Section 6. 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." [3] 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 session: The association among a set of participants communicating with RTP. For each participant, the session is defined by a particular pair of destination transport addresses (one network address plus a port pair for RTP and RTCP). The destination transport address pair may be common for all participants, as in the case of IP multicast, or may be different for each, as in the case of individual unicast network addresses plus a common port pair. In a multimedia session, each medium is carried in a separate RTP session with its own RTCP packets. The multiple RTP sessions are distinguished by different port number pairs and/or different multicast addresses.

Synchronization source (SSRC): The source of a stream of RTP packets, identified by a 32-bit numeric SSRC identifier carried in the RTP header so as not to be dependent upon the network address.

All packets from a synchronization source form part of the same timing and sequence number space, so a receiver groups packets by synchronization source for playback. Examples of synchronization sources include the sender of a stream of packets derived from a signal source such as a microphone or a camera, or an RTP mixer (see below). A synchronization source may change its data format, e.g., audio encoding, over time. The SSRC identifier is a randomly chosen value meant to be globally unique within a particular RTP session (see Section 8). A participant need not use the same SSRC identifier for all the RTP sessions in a multimedia session; the binding of the SSRC identifiers is provided through RTCP (see Section 6.4.1). If a participant generates multiple streams in one RTP session, for example from separate video cameras, each must be identified as a different SSRC.

~~Contributing source (CSRC): A source of a stream of RTP packets that has contributed to the combined stream produced by an RTP mixer (see below). The mixer inserts a list of the SSRC identifiers of the sources that contributed to the generation of a particular packet into the RTP header of that packet. This list is called the CSRC list. An example application is audio conferencing where a mixer indicates all the talkers whose speech was combined to produce the outgoing packet, allowing the receiver to indicate the current talker, even though all the audio packets contain the same SSRC identifier (that of the mixer).~~ CSRC will be set to 0.

End system: An application that generates the content to be sent in RTP packets and/or consumes the content of received RTP packets. An end system can act as one or more synchronization sources in a particular RTP session, but typically only one.

~~Mixer: An intermediate system that receives RTP packets from one or more sources, possibly changes the data format, combines the packets in some manner and then forwards a new RTP packet. Since the timing among multiple input sources will not generally be synchronized, the mixer will make timing adjustments among the streams and generate its own timing for the combined stream. Thus, all data packets originating from a mixer will be identified as having the mixer as their synchronization source.~~

Translator: An intermediate system that forwards RTP packets with their synchronization source identifier intact. Examples of translators include devices that convert encodings without mixing, replicators from multicast to unicast, and application-

level filters in firewalls.

Monitor: An application that receives RTCP packets sent by participants in an RTP session, in particular the reception reports, and estimates the current quality of service for distribution monitoring, fault diagnosis and long-term statistics. The monitor function is likely to be built into the application(s) participating in the session, but may also be a separate application that does not otherwise participate and does not send or receive the RTP data packets. These are called third party monitors. A Gatekeeper should be an RTP monitor.

Non-RTP means: Protocols and mechanisms that may be needed in addition to RTP to provide a usable service. In particular, for multimedia conferences, a conference control application may distribute multicast addresses and keys for encryption, negotiate the encryption algorithm to be used, and define dynamic mappings between RTP payload type values and the payload formats they represent for formats that do not have a predefined payload type value. For simple applications, electronic mail or a conference database may also be used. The specification of such protocols and mechanisms is outside the scope of this document.

4 Byte Order, Alignment, and Time Format

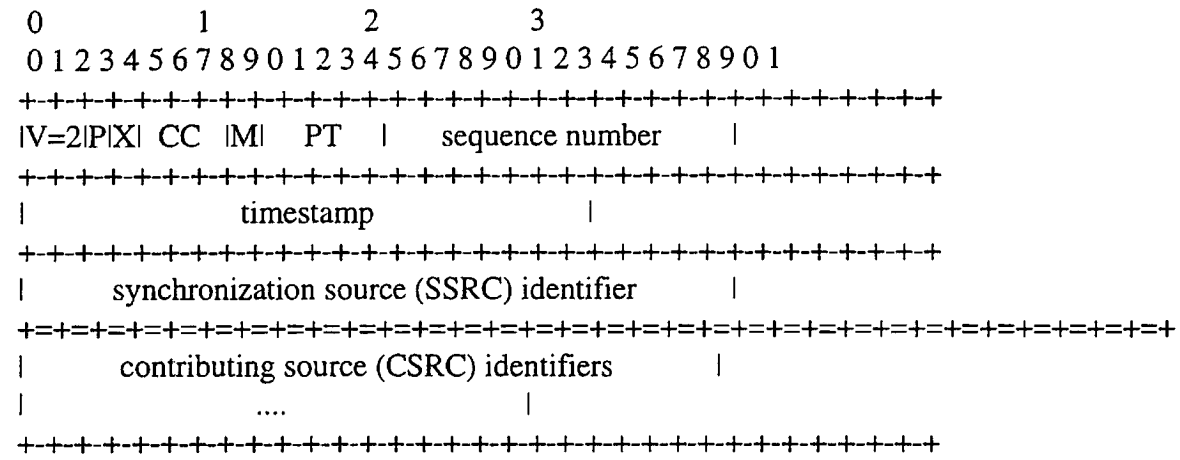
All integer fields are carried in network byte order, that is, most significant byte (octet) first. This byte order is commonly known as big-endian. The transmission order is described in detail in [4]. Unless otherwise noted, numeric constants are in decimal (base 10). All header data is aligned to its natural length, i.e., 16-bit fields are aligned on even offsets, 32-bit fields are aligned at offsets divisible by four, etc. Octets designated as padding have the value zero.

Wallclock time (absolute time) is represented using the timestamp format of the Network Time Protocol (NTP), which is in seconds relative to 0h UTC on 1 January 1900 [5]. The full resolution NTP timestamp is a 64-bit unsigned fixed-point number with the integer part in the first 32 bits and the fractional part in the last 32 bits. In some fields where a more compact representation is appropriate, only the middle 32 bits are used; that is, the low 16 bits of the integer part and the high 16 bits of the fractional part. The high 16 bits of the integer part must be determined independently. If NTP is not available to a Terminal, it shall set its NTP to 0.

5 RTP Data Transfer Protocol

5.1 RTP Fixed Header Fields

The RTP header has the following format:



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): 2 bits
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): 1 bit
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. 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): 1 bit
If the extension bit is set, the fixed header is followed by exactly one header extension, with a format defined in Section 5.3.1.

CSRC count (CC):- Shall be set to four zeroes (0000).~~4 bits~~
~~The CSRC count contains the number of CSRC identifiers that follow the fixed header.~~

marker (M): 1 bit
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 (see Section 5.3).

payload type (PT): 7 bits
This field identifies the format of the RTP payload and determines its interpretation by the application. A profile specifies a default static mapping of payload type codes to payload formats. Additional payload type codes may be defined dynamically through non-RTP means (see Section 3). An initial set of default mappings for audio and video is specified in the companion profile Internet-Draft draft-ietf-avt-profile , and may be extended in future editions of the Assigned Numbers RFC [6]. An RTP sender emits a single RTP payload type at any given time; this field is not intended for multiplexing separate media streams (see Section 5.2).

sequence number: 16 bits
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 is 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 [7].

timestamp: 32 bits
The timestamp reflects the sampling instant of the first octet in the RTP data packet. The sampling instant must be derived

from a clock that increments monotonically and linearly in time to allow synchronization and jitter calculations (see Section 6.3.1).~~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.~~

Use of the timestamp for H.225.0 is defined in the accompanying profile document in Annex A.

The initial value of the timestamp is random, as for the sequence number. Several consecutive RTP packets may 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 monotone~~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 monotone~~monotonic~~.)

SSRC: 32 bits

The SSRC field identifies the synchronization source. This identifier is chosen randomly, with the intent that no two synchronization sources within the same RTP session will have the same SSRC identifier. An example algorithm for generating a random identifier is presented in Appendix A.6. 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: 0 to 15 items, 32 bits each

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 may 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 indication at the receiver.

5.2 Multiplexing RTP Sessions

Review

For efficient protocol processing, the number of multiplexing points should be minimized, as described in the integrated layer processing design principle [1]. In RTP, multiplexing is provided by the destination transport address (network address and port number) which define an RTP session. For example, in a teleconference composed of audio and video media encoded separately, each medium should be carried in a separate RTP session with its own destination transport address. It is not intended that the audio and video be carried in a single RTP session and demultiplexed based on the payload type or SSRC fields. Interleaving packets with different payload types but using the same SSRC would introduce several problems:

1. If one payload type were switched during a session, there would be no general means to identify which of the old values the new one replaced.

2. An SSRC is defined to identify a single timing and sequence number space. Interleaving multiple payload types would require different timing spaces ifthe media clock rates differ and would require different sequence number spaces to tell which payload type suffered packet loss.

3. The RTCP sender and receiver reports (see Section 6.3) can only describe one timing and sequence number space per SSRC and do not carry a payload type field.

~~4. An RTP mixer would not be able to combine interleaved streams of incompatible media into one stream.~~

45. Carrying multiple media in one RTP session precludes: the use of different network paths or network resource allocations if appropriate; reception of a subset of the media if desired, for example just audio if video would exceed the available bandwidth; and receiver implementations that use separate processes for the different media, whereas using separate RTP sessions permits either single- or multiple-process implementations.

Using a different SSRC for each medium but sending them in the same RTP session would avoid the first three problems but not the last

—~~one~~two.

If multiplexing is used, each medium shall have a different SSRC

5.3 Profile-Specific Modifications to the RTP Header

The header may be tailored through modifications or additions defined in a profile specification while still allowing profile-independent monitoring and recording tools to function.

- The marker bit and payload type field carry profile-specific information, but they are allocated in the fixed header since many applications are expected to need them and might otherwise have to add another 32-bit word just to hold them. The octet containing these fields may be redefined by a profile to suit different requirements, for example with a more or fewer marker bits. If there are any marker bits, one should be located in the most significant bit of the octet since profile-independent monitors may be able to observe a correlation between packet loss patterns and the marker bit.

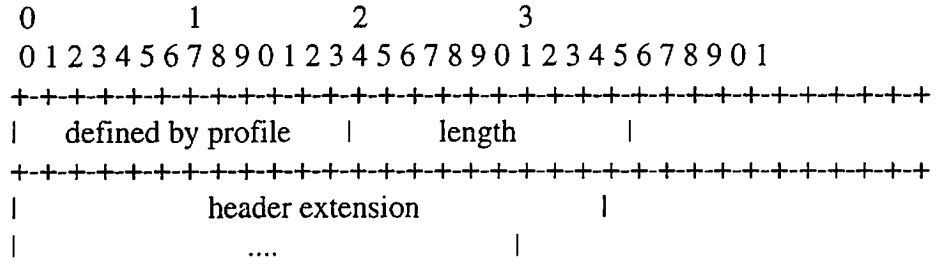
- Additional information that is required for a particular payload format, such as a video encoding, should be carried in the payload section of the packet. This might be in a header that is always present at the start of the payload section, or might be indicated by a reserved value in the data pattern.

- If a particular class of applications needs additional functionality independent of payload format, the profile under which those applications operate should define additional fixed fields to follow immediately after the SSRC field of the existing fixed header. Those applications will be able to quickly and directly access the additional fields while profile-independent monitors or recorders can still process the RTP packets by interpreting only the first twelve octets.

5.3.1 RTP Header Extension

An extension mechanism is provided to allow individual implementations to experiment with new payload-format-independent functions that require additional information to be carried in the RTP data packet header. This mechanism is designed so that the header extension may be ignored by other interoperating implementations that have not been extended.

Note that this header extension is intended only for limited use. Most potential uses of this mechanism would be better done another way, using the methods described in the previous section. For example, a profile-specific extension to the fixed header is less expensive to process because it is not conditional nor in a variable location. Additional information required for a particular payload format should not use this header extension, but should be carried in the payload section of the packet.



If the X bit in the RTP header is one, a variable-length header extension is appended to the RTP header, following the CSRC list if present. The header extension contains a 16-bit length field that counts the number of 32-bit words in the extension, excluding the four-octet extension header (therefore zero is a valid length). Only a single extension may be appended to the RTP data header. To allow multiple interoperating implementations to each experiment independently with different header extensions, or to allow a particular implementation to experiment with more than one type of header extension, the first 16 bits of the header extension are left open for distinguishing identifiers or parameters. The format of these 16 bits is to be defined by the profile specification under which the implementations are operating. This RTP specification does not define any header extensions itself.

6 RTP Control Protocol -- RTCP

The use of RTCP is optional.

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 [8,9], 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, described below in Section 6.3.

2. RTCP carries a persistent transport-level identifier for an RTP source called the canonical name or CNAME, Section 6.4.1. 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 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.

3. By having each participant send its control packets to all the others, each can independently observe the number of participants.

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 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.

Note that H.323 defines Multicast as a conference with no control mechanism and no feedback from the receivers.

6.1 RTCP 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

SDES: Source description items, including CNAME

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 always end on a 32-bit boundary. The alignment requirement and a length field in the fixed part are included to make RTCP packets "stackable". Multiple RTCP packets may 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:

- o 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 should include a report packet.

- o 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 should also include the SDES CNAME.

o The number of packet types that may appear first in the compound packet should 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 recommended:

SR or RR: The first RTCP packet in the compound packet must always be a report packet to facilitate header validation as described in Appendix A.2. This is true even if no data has been sent nor received, in which case an empty RR is 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.

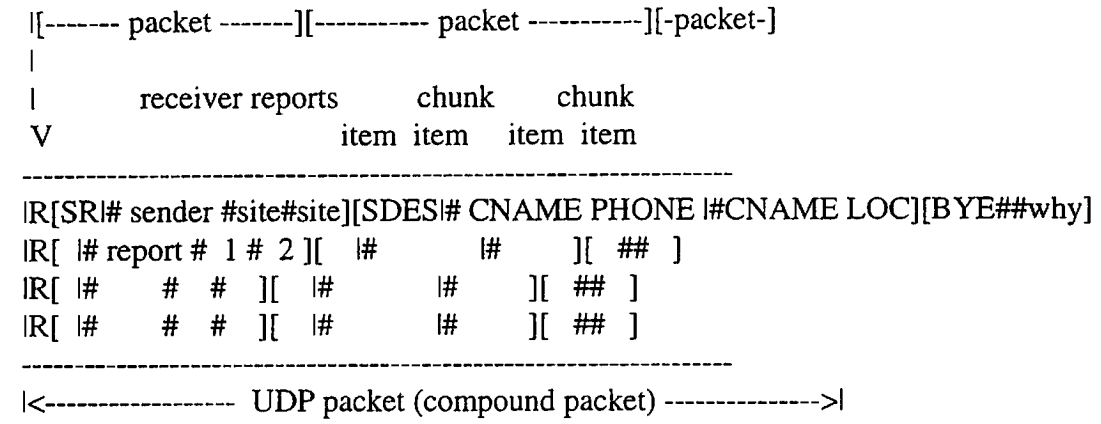
SDES: An SDES packet containing a CNAME item must be included in each compound RTCP packet. Other source description items may optionally be included if required by a particular application, subject to bandwidth constraints (see Section 6.2.2).

It is advisable for translators and mixers to combine individual RTCP packets from the multiple sources they are forwarding into one compound packet whenever feasible in order to amortize the packet overhead (see Section 7). An example RTCP compound packet as might be produced by a mixer is shown in Fig. 1. If the overall length of a compound packet would exceed the maximum transmission unit (MTU) of the network path, it may be segmented into multiple shorter compound packets to be transmitted in separate packets of the underlying protocol. Note that each of the compound packets must begin with an SR or RR packet.

An implementation may ignore incoming RTCP packets with types unknown to it. Additional RTCP packet types may be registered with the Internet Assigned Numbers Authority (IANA).

6.2 RTCP Transmission Interval

if encrypted: random 32-bit integer
|



#: SSRC/CSRC

Figure 1: Example of an RTCP compound packet

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.

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, or it might just be a reasonable share. The session bandwidth may be chosen based or 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. The session bandwidth parameter is expected to be supplied by a session management application when it invokes a media application, but media applications may also 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.

6.2.1 Maintaining the number of session members

New sites are added

to the count when they are heard, and an entry for each is created in a table indexed by the SSRC or CSRC identifier (see Section 8.2) to keep track of them. New entries may not be considered valid until multiple packets carrying the new SSRC have been received (see

Appendix A.1). Entries may be deleted from the table when an RTCP BYE packet with the corresponding SSRC identifier is received.

6.2.2 Allocation of source description bandwidth

This specification defines several source description (SDES) items in addition to the mandatory CNAME item, such as NAME (personal name)

6.3 Sender and Receiver Reports

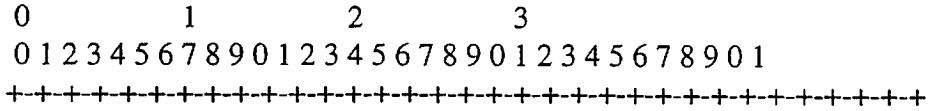
RTP receivers provide reception quality feedback using RTCP report packets which may take one of two forms depending upon whether or not the receiver is also a sender. The only difference between the sender report (SR) and receiver report (RR) forms, besides the packet type code, is that the sender report includes a 20-byte sender information section for use by active senders. The SR is issued if a site has sent any data packets during the interval since issuing the last report or the previous one, otherwise the RR is issued.

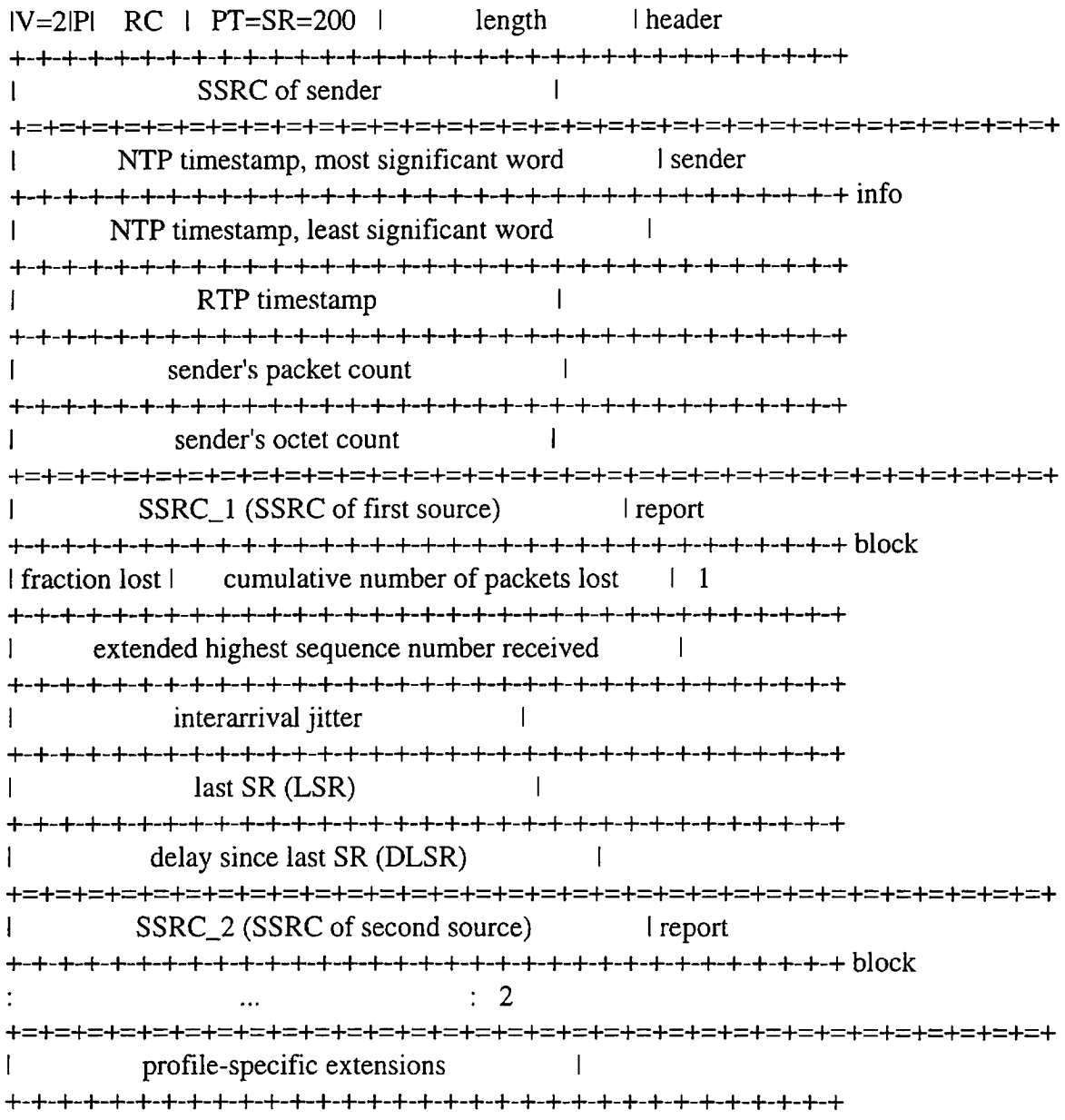
Both the SR and RR forms include zero or more reception report blocks, one for each of the synchronization sources from which this receiver has received RTP data packets sinthe last report. Reports are not issued for contributing sources listed in the CSRC list. Each reception report block provides statistics about the data received from the particular source indicated in that block. Since a maximum of 31 reception report blocks will fit in an SR or RR packet, additional RR packets may be stacked after the initial SR or RR packet as needed to contain the reception reports for all sources heard during the interval since the last report.

The next sections define the formats of the two reports, how they may

be extended in a profile-specific manner if an application requires additional feedback information, and how the reports may be used. Details of reception reporting by translators and mixers is given in Section 7.

6.3.1 SR: Sender report RTCP packet





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. The fields have the following meaning:

version (V): 2 bits

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): 1 bit

If the padding bit is set, this RTCP packet contains some additional padding octets at the end which are not part of the control information. The last octet of the padding is a count of how many padding octets should be ignored. Padding may be needed by some encryption algorithms with fixed block sizes. In a compound RTCP packet, padding should only be required on the last individual packet because the compound packet is encrypted as a whole.

reception report count (RC): 5 bits
The number of reception report blocks contained in this packet. A value of zero is valid.

packet type (PT): 8 bits
Contains the constant 200 to identify this as an RTCP SR packet.

length: 16 bits
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: 32 bits
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: 64 bits
Indicates the wallclock 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. A sender that can keep track of elapsed time but has no notion of wallclock time may use the elapsed time since joining

the session instead. This is assumed to be less than 68 years, so the high bit will be zero. It is permissible to use the

sampling clock to estimate elapsed wallclock time. A sender that has no notion of wallclock or elapsed time shall ~~may~~ set the NTP timestamp to zero.

RTP timestamp: 32 bits
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 is calculated from the corresponding NTP timestamp using the relationship between the RTP timestamp counter and real time as maintained by periodically checking the wallclock time at a sampling instant.

sender's packet count: 32 bits
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 is reset if the sender changes its SSRC identifier.

sender's octet count: 32 bits
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 is 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 do not carry over statistics when a source changes its SSRC identifier due to a collision. These statistics are:

SSRC_n (source identifier): 32 bits
The SSRC identifier of the source to which the information in this reception report block pertains.

fraction lost: 8 bits
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. An implementation is shown in Appendix A.3. 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: 24 bits
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. This may be calculated as shown in Appendix A.3.

extended highest sequence number received: 32 bits
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, which may be maintained according to the algorithm in Appendix A.1. Note that different receivers within the same session will generate different extensions to the sequence number if their start times differ significantly.

interarrival jitter: 32 bits Review

Note that since H.245 also provides jitter information, RTCP jitter calculation should not be used.

~~An estimate of the statistical variance of the RTP data packet interarrival time, measured in timestamp units and expressed as an unsigned integer. The interarrival 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 Si is the RTP timestamp from packet i, and Ri 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)=(Rj-Ri)-(Sj-Si)=(Rj-Sj)-(Ri-Si)~~

~~The interarrival jitter is 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=J+(|D(i-1,i)|-J)/16~~

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

~~The jitter calculation is prescribed here 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 [11]. A sample implementation is shown in Appendix A.8.~~

last SR timestamp (LSR): 32 bits

The middle 32 bits out of 64 in the NTP timestamp (as explained in Section 4) 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): 32 bits

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.

Let SSRC_r denote the receiver issuing this receiver report. Source SSRC_n can compute the round propagation delay to SSRC_r by recording the time A when this reception report block is received. It calculates the total round-trip time A-LSR using the last SR timestamp (LSR) field, and then subtracting this field to leave the round-trip propagation delay as (A- LSR - DLSR). This is illustrated in Fig. 2.

This may be used as an approximate measure of distance to cluster receivers, although some links have very asymmetric delays.

6.3.2 RR: Receiver report RTCP packet

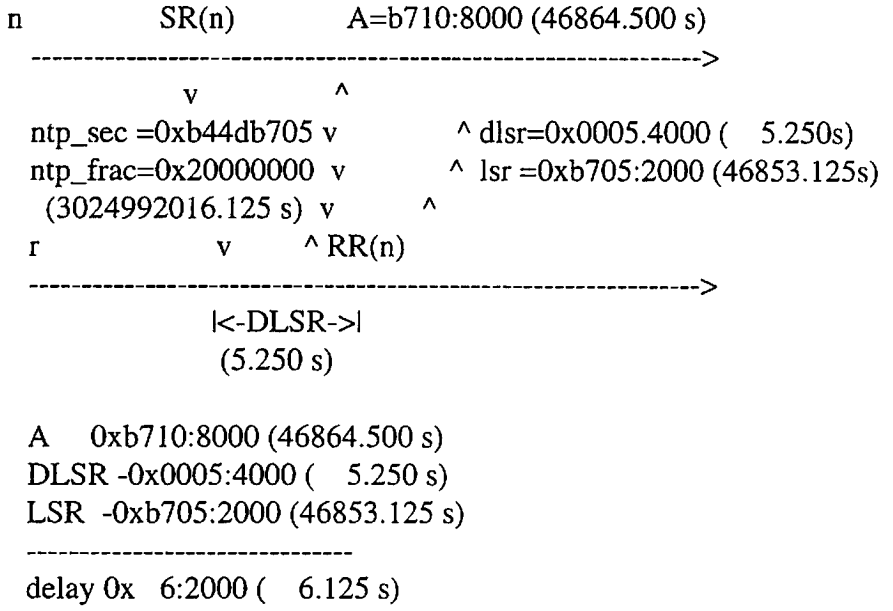
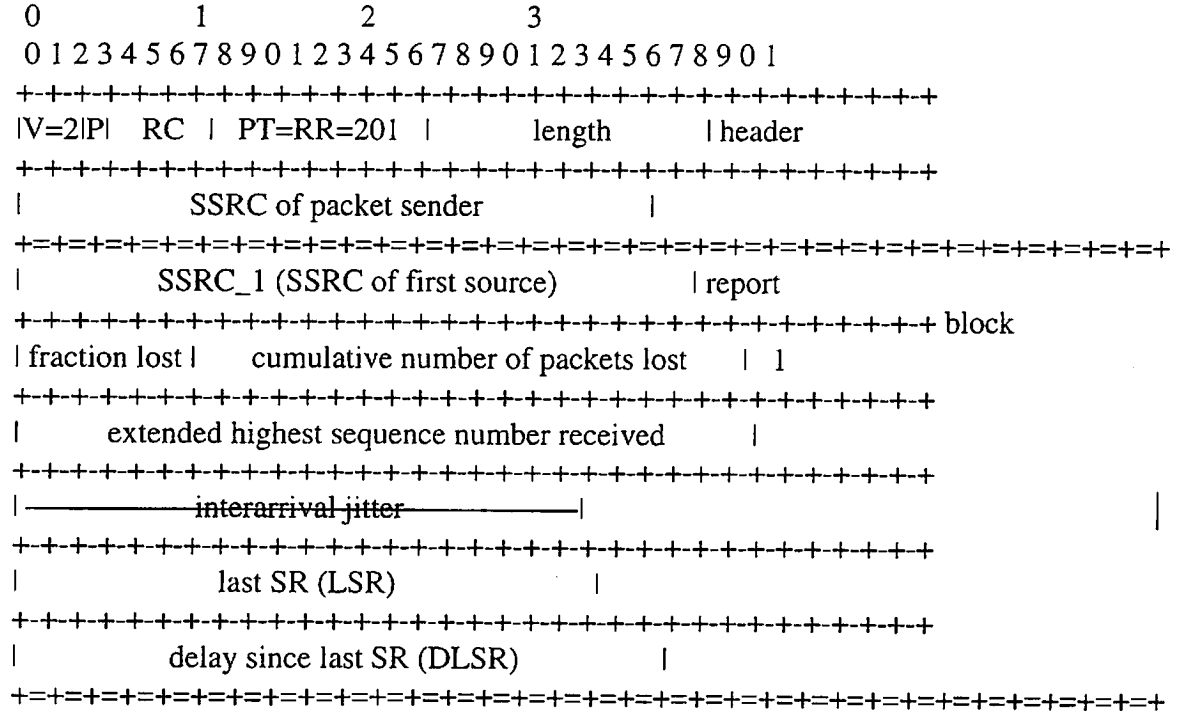
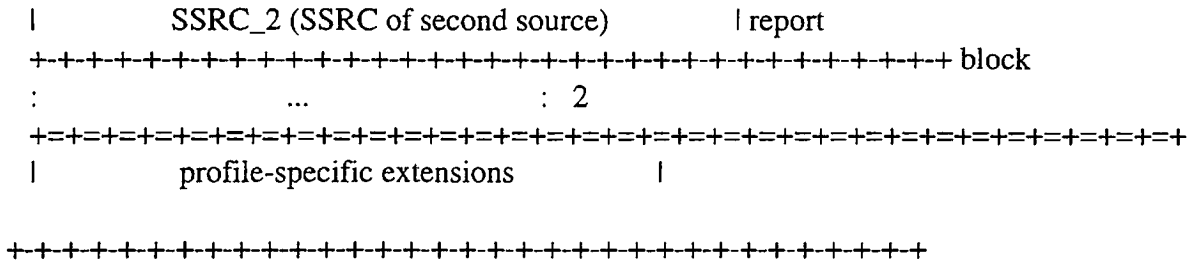


Figure 2: Example for round-trip time computation





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) is put at the headof a compound RTCP packet when there is no data transmission or reception to report.

6.3.3 Extending the sender and receiver reports Review

A profile should define profile- or application-specific extensions to the sender report and receiver if there is additional information that should be reported regularly about the sender or receivers. This method should be used in preference to defining another RTCP packet type because it requires less overhead:

o fewer octets in the packet (no RTCP header or SSRC field);

o simpler and faster parsing because applications running under that profile would be programmed to always expect the extension fields in the directly accessible location after the reception reports.

If additional sender information is required, it should be included first in the extension for sender reports, but would not be present in receiver reports. If information about receivers is to be included, that data may be structured as an array of blocks parallel to the existing array of reception report blocks; that is, the number of blocks would be indicated by the RC field.

6.3.4 Analyzing sender and receiver reports

It is expected that reception quality feedback will be useful not only for the sender but also for other receivers and third-party monitors. The sender may modify its transmissions based on the feedback; receivers can determine whether problems are local,

regional or global; network managers may use profile-independent monitors that receive only the RTCP packets and not the corresponding RTP data packets to evaluate the performance of their networks for multicast distribution.

Cumulative counts are used in both the sender information and receiver report blocks so that differences may be calculated between any two reports to make measurements over both short and long time periods, and to provide resilience against the loss of a report. The

difference between the last two reports received can be used to estimate the recent quality of the distribution. The NTP timestamp is included so that rates may be calculated from these differences over the interval between two reports. Since that timestamp is independent of the clock rate for the data encoding, it is possible to implement encoding- and profile-independent quality monitors.

An example calculation is the packet loss rate over the interval between two reception reports. The difference in the cumulative number of packets lost gives the number lost during that interval. The difference in the extended last sequence numbers received gives the number of packets expected during the interval. The ratio of these two is the packet loss fraction over the interval. This ratio should equal the fraction lost field if the two reports are consecutive, but otherwise not. The loss rate per second can be obtained by dividing the loss fraction by the difference in NTP timestamps, expressed in seconds. The number of packets received is the number of packets expected minus the number lost. The number of packets expected may also be used to judge the statistical validity of any loss estimates. For example, 1 out of 5 packets lost has a lower significance than 200 out of 1000.

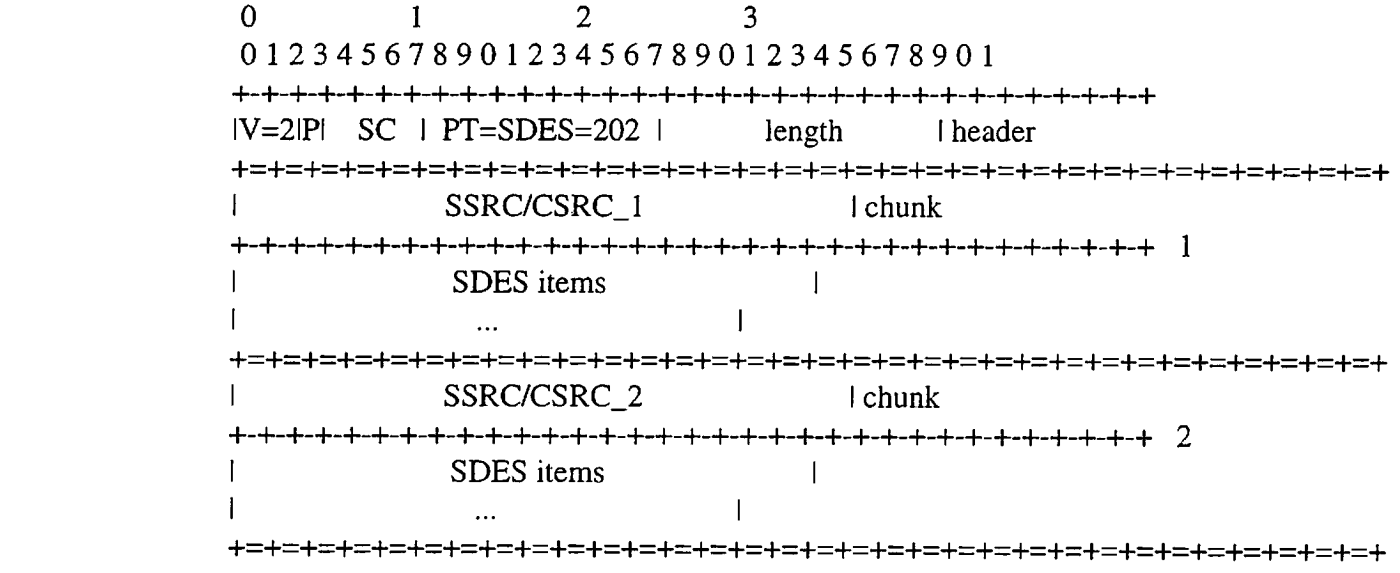
From the sender information, a third-party monitor can calculate the average payload data rate and the average packet rate over an interval without receiving the data. Taking the ratio of the two gives the average payload size. If it can be assumed that packet loss is independent of packet size, then the number of packets received by a particular receiver times the average payload size (or the corresponding packet size) gives the apparent throughput available to that receiver.

In addition to the cumulative counts which allow long-term packet loss measurements using differences between reports, the fraction lost field provides a short-term measurement from a single report.

This becomes more important as the size of a session scales up enough that reception state information might not be kept for all receivers or the interval between reports becomes long enough that only one report might have been received from a particular receiver.

The interarrival jitter field provides a second short-term measure of network congestion. Packet loss tracks persistent congestion while the jitter measure tracks transient congestion. The jitter measure may indicate congestion before it leads to packet loss. Since the interarrival jitter field is only a snapshot of the jitter at the time of a report, it may be necessary to analyze a number of reports from one receiver over time or from multiple receivers, e.g., within a single network.

6.4 SDES: Source description RTCP packet



The SDES packet is a three-level structure composed of a header and zero or more chunks, each of which is composed of items describing the source identified in that chunk. The items are described individually in subsequent sections.

version (V), padding (P), length:
As described for the SR packet (see Section 6.3.1).

packet type (PT): 8 bits
Contains the constant 202 to identify this as an RTCP SDES packet.

source count (SC): 5 bits

The number of SSRC/CSRC chunks contained in this SDES packet. A value of zero is valid but useless.

Each chunk consists of an SSRC/CSRC identifier followed by a list of zero or more items, which carry information about the SSRC/CSRC. Each chunk starts on a 32-bit boundary. Each item consists of an 8-bit type field, an 8-bit octet count describing the length of the text (thus, not including this two-octet header), and the text itself. Note that the text can be no longer than 255 octets, but this is consistent with the need to limit RTCP bandwidth consumption.

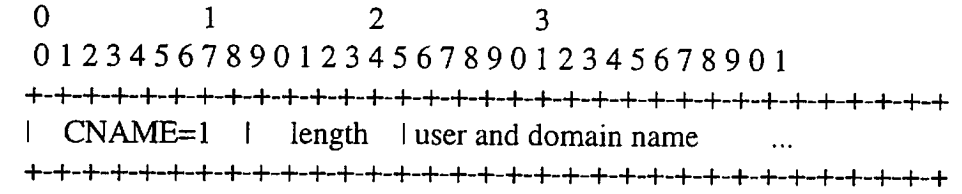
The text is encoded according to the UTF-2 encoding specified in Annex F of ISO standard 10646 [12,13]. This encoding is also known as UTF-8 or UTF-FSS. It is described in "File System Safe UCS Transformation Format (FSS_UTF)", X/Open Preliminary Specification, Document Number P316 and Unicode Technical Report #4. US-ASCII is a subset of this encoding and requires no additional encoding. The presence of multi-octet encodings is indicated by setting the most significant bit of a character to a value of one.

Items are contiguous, i.e., items are not individually padded to a 32-bit boundary. Text is not null terminated because some multi-octet encodings include null octets. The list of items in each chunk is terminated by one or more null octets, the first of which is interpreted as an item type of zero to denote the end of the list, and the remainder as needed to pad until the next 32-bit boundary. A chunk with zero items (four null octets) is valid but useless.

End systems send one SDES packet containing their own source identifier (the same as the SSRC inthe fixed RTP header). A mixer sends one SDES packet containing a chunk for each contributing source from which it is receiving SDES information, or multiple complete SDES packets in the format above if there are more than 31 such sources (see Section 7).

The SDES items currently defined are described in the next sections. Only the CNAME item is mandatory. Some items shown here may be useful only for particular profiles, but the item types are all assigned from one common space to promote shared use and to simplify profile-independent applications. Additional items may be defined in a profile by registering the type numbers with IANA.

6.4.1 CNAME: Canonical end-point identifier SDES item Review



The CNAME identifier has the following properties:

o Because the randomly allocated SSRC identifier may change if a conflict is discovered or if a program is restarted, the CNAME item is required to provide the binding from the SSRC identifier to an identifier for the source that remains constant.

o Like the SSRC identifier, the CNAME identifier should also be unique among all participants within one RTP session.

o To provide a binding across multiple media tools used by one participant in a set of related RTP sessions, the CNAME should be fixed for that participant.

o To facilitate third-party monitoring, the CNAME should be suitable for either a program or a person to locate the source.

(We need to define CNAME and tie it into the Connection ID or better yet - to Conference ID).

~~Therefore, the CNAME should be derived algorithmically and not entered manually, when possible. To meet these requirements, the following format should be used unless a profile specifies an alternate syntax or semantics. The CNAME item should have the format "user@host", or "host" if a user name is not available as on single-user systems. For both formats, "host" is either the fully qualified domain name of the host from which the real-time data originates, formatted according to the rules specified in RFC 1034 [14], RFC 1035 [15] and Section 2.1 of RFC 1123 [16]; or the standard ASCII representation of the host's numeric address on the interface used for the RTP communication. For example, the standard ASCII representation of an IP Version 4 address is "dotted decimal", also known as dotted quad. Other address types are expected to have ASCII representations that are mutually unique. The fully qualified domain name is more convenient for a human observer and may avoid the need to send a NAME item in addition, but it may be difficult or impossible to obtain reliably in some operating environments. Applications that may be run in such environments should use the ASCII representation of the address instead.~~

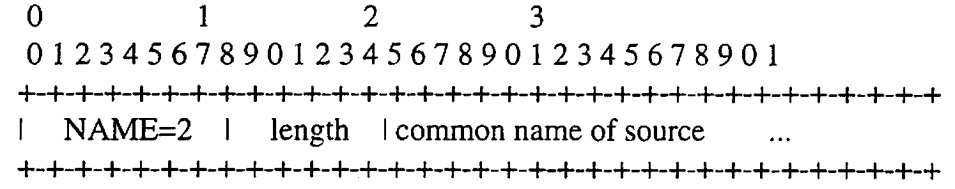
~~Examples are "doe@sleepy.megacorp.com" or "doe@192.0.2.89" for a~~

~~multi-user system. On a system with no user name, examples would be "sleepy.megacorp.com" or "192.0.2.89".~~

~~The user name should be in a form that a program such as "finger" or "talk" could use, i.e., it typically is the login name rather than the personal name. The host name is not necessarily identical to the one in the participant's electronic mail address.~~

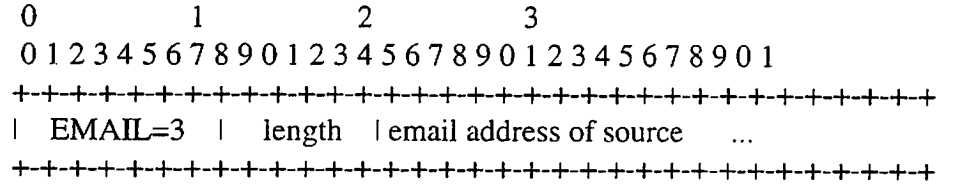
~~This syntax will not provide unique identifiers for each source if an application permits a user to generate multiple sources from one host. Such an application would have to rely on the SSRC to further identify the source, or the profile for that application would have to specify additional syntax for the CNAME identifier.~~

6.4.2 NAME: User name SDES item



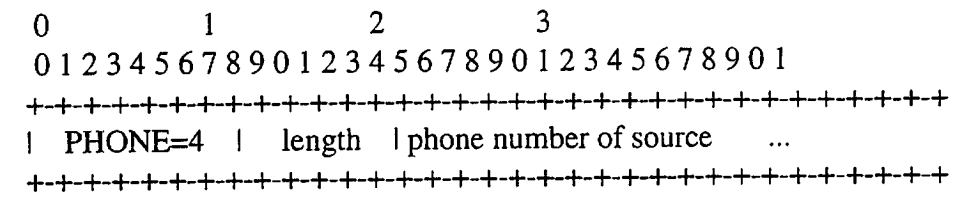
This is the real name used to describe the source, e.g., "John Doe, Bit Recycler, Megacorp". It may be in any form desired by the user. For applications such as conferencing, this form of name may be the most desirable for display in participant lists, and therefore might be sent most frequently of those items other than CNAME. Profiles may establish such priorities. The NAME value is expected to remain constant at least for the duration of a session. It should not be relied upon to be unique among all participants in the session.

6.4.3 EMAIL: Electronic mail address SDES item



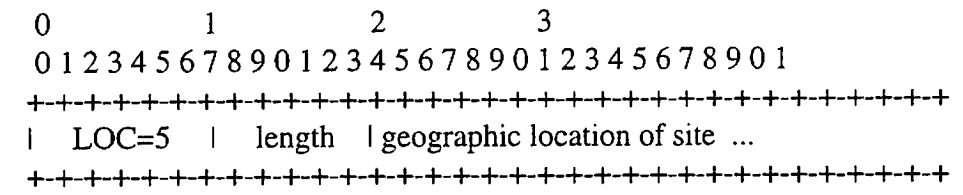
The email address is formatted according to RFC 822 [19], for example, "John.Doe@megacorp.com". The EMAIL value is expected to remain constant for the duration of a session.

6.4.4 PHONE: Phone number SDES item



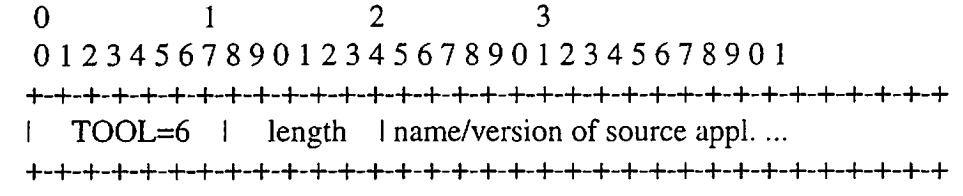
The phone number should be formatted with the plus sign replacing the international access code. For example, "+1 908 555 1212" for a number in the United States.

6.4.5 LOC: Geographic user location SDES item



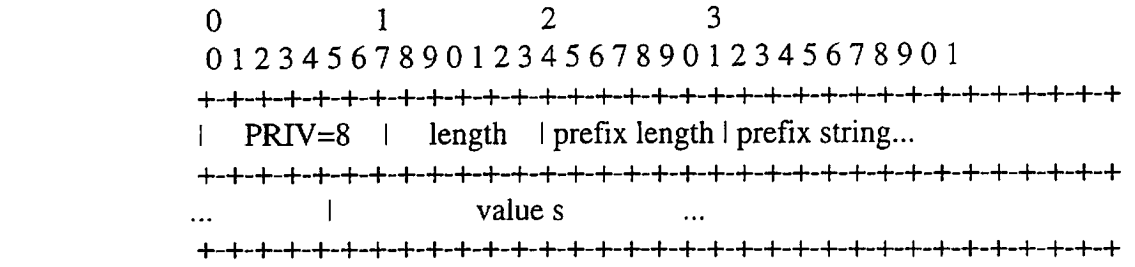
Depending on the application, different degrees of detail are appropriate for this item. For conference applications, a string like "Murray Hill, New Jersey" may be sufficient, while, for an active badge system, strings like "Room 2A244, AT&T BL MH" might be appropriate. The degree of detail is left to the implementation and/or user, but format and content may be prescribed by a profile. The LOC value is expected to remain constant for the duration of a session, except for mobile hosts.

6.4.6 TOOL: Application or tool name SDES item



A string giving the name and possibly version of the application generating the stream, e.g., "videotool 1.2". This information may be useful for debugging purposes and is similar to the Mailer or Mail-System-Version SMTP headers. The TOOL value is expected to remain constant for the duration of the session.

6.4.7 PRIV: Private extensions SDES item

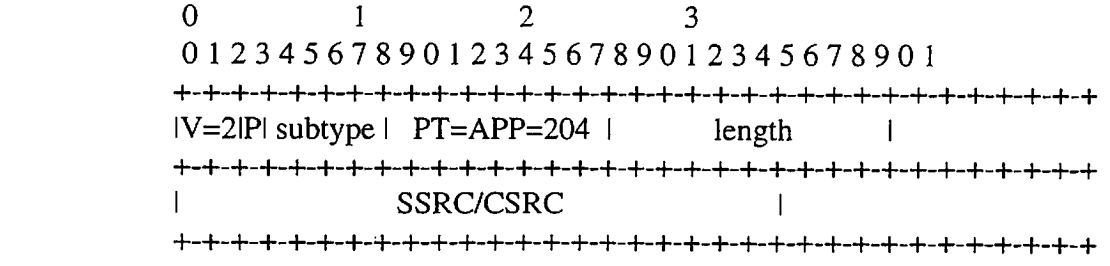


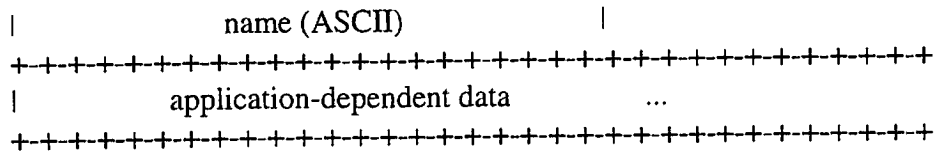
This item is used to define experimental or application-specific SDES extensions. The item contains a prefix consisting of a length-string pair, followed by the value string filling the remainder of the item and carrying the desired information. The prefix length field is 8 bits long. The prefix string is a name chosen by the person defining the PRIV item to be unique with respect to other PRIV items this application might receive. The application creator might choose to use the application name plus an additional subtype identification if needed. Alternatively, it is recommended that others choose a name based on the entity they represent, then coordinate the use of the name within that entity.

Note that the prefix consumes some space within the item's total length of 255 octets, so the prefix should be kept as short as possible. This facility and the constrained RTCP bandwidth should not be overloaded; it is not intended to satisfy all the control communication requirements of all applications.

SDES PRIV prefixes will not be registered by IANA. If some form of the PRIV item proves to be of general utility, it should instead be assigned a regular SDES item type registered with IANA so that no prefix is required. This simplifies use and increases transmission efficiency.

6.6 APP: Application-defined RTCP packet





The APP packet is intended for experimental use as new applications and new features are developed, without requiring packet type value registration. APP packets with unrecognized names should be ignored. After testing and if wider use is justified, it is recommended that each APP packet be redefined without the subtype and name fields and registered with the Internet Assigned Numbers Authority using an RTCP packet type.

version (V), padding (P), length:
As described for the SR packet (see Section 6.3.1).

subtype: 5 bits
May be used as a subtype to allow a set of APP packets to be defined under one unique name, or for any application-dependent data.

packet type (PT): 8 bits
Contains the constant 204 to identify this as an RTCP APP packet.

name: 4 octets
A name chosen by the person defining the set of APP packets to be unique with respect to other APP packets this application might receive. The application creator might choose to use the

application name, and then coordinate the allocation of subtype values to others who want to define new packet types for the application. Alternatively, it is recommended that others choose a name based on the entity they represent, then coordinate the use of the name within that entity. The name is interpreted as a sequence of four ASCII characters, with uppercase and lowercase characters treated as distinct.

application-dependent data: variable length
Application-dependent data may or may not appear in an APP packet. It is interpreted by the application and not RTP itself. It must be a multiple of 32 bits long.

In addition to end systems, RTP supports the notion of "translators" ~~and "mixers~~", which could be considered as "intermediate systems" at the RTP level. Although this support adds some complexity to the protocol, the need for these functions has been clearly established by experiments with multicast audio and video applications in the Internet. Example uses of translators ~~and mixers~~ given in Section 2.~~3~~ stem from the presence of firewalls and low bandwidth connections, both of which are likely to remain.

7.1 General Description

Translator: Forwards RTP packets with their SSRC identifier intact; this makes it possible for receivers to identify individual sources even though packets from all the sources pass through the same translator and carry the translator's network source address. Some kinds of translators will pass through the data untouched, but others may change the encoding of the data and thus the RTP data payload type and timestamp. If multiple data packets are re-encoded into one, or vice versa, a translator must assign new sequence numbers to the outgoing packets. Losses in the incoming packet stream may induce corresponding gaps in the outgoing sequence numbers. Receivers cannot detect the presence of a translator unless they know by some other means what payload type or transport address was used by the original source.

7.2 RTCP Processing in Translators

In addition to forwarding data packets, perhaps modified, translators and mixers must also process RTCP packets. In many cases, they will take apart the compound RTCP packets received from end systems to aggregate SDES information and to modify the SR or RR packets. Retransmission of this information may be triggered by the packet arrival or by the RTCP interval timer of the translator or mixer itself.

A translator that does not modify the data packets, for example one that just replicates between a multicast address and a unicast address, may simply forward RTCP packets unmodified as well. A translator that transforms the payload in some way must make corresponding transformations in the SR and RR information so that it still reflects the characteristics of the data and the reception quality. These translators must not simply forward RTCP packets. In general, a translator should not aggregate SR and RR packets from different sources into one packet since that would reduce the accuracy of the propagation delay measurements based on the LSR and

DLSR fields.

SR sender information: A translator does not generate its own sender information, but forwards the SR packets received from one cloud to the others. The SSRC is left intact but the sender information must be modified if required by the translation. If a translator changes the data encoding, it must change the "sender's byte count" field. If it also combines several data packets into one output packet, it must change the "sender's packet count" field. If it changes the timestamp frequency, it must change the "RTP timestamp" field in the SR packet.

SR/RR reception report blocks: A translator forwards reception reports received from one cloud to the others. Note that these flow in the direction opposite to the data. The SSRC is left intact. If a translator combines several data packets into one output packet, and therefore changes the sequence numbers, it must make the inverse manipulation for the packet loss fields and the "extended last sequence number" field. This may be complex. In the extreme case, there may be no meaningful way to translate the reception reports, so the translator may pass on no reception report at all or a synthetic report based on its own reception. The general rule is to do what makes sense for a particular translation.

A translator does not require an SSRC identifier of itsown, but may choose to allocate one for the purpose of sending reports about what it has received. These would be sent to all the connected clouds, each corresponding to the translation of the data stream as sent to that cloud, since reception reports are normally multicast to all participants.

SDES: Translators typically forward without change the SDES information they receive from one cloud to the others, but may, for example, decide to filter non-CNAME SDES information if bandwidth is limited. The CNAMEs must be forwarded to allow SSRC identifier collision detection to work. A translator that generates its own RR packets must send SDES CNAME information about itself to the same clouds that it sends those RR packets.

BYE: Translators forward BYE packets unchanged. Translators with their own SSRC should generate BYE packets with that SSRC identifier if they are about to cease forwarding packets.

APP: Translators forward APP packets unchanged.

Mixer: Receives streams of RTP data packets from one or more sources, possibly changes the data format, combines the streams in some manner and then forwards the combined stream. Since the timing among multiple input sources will not generally be synchronized, the mixer will make timing adjustments among the streams and generate its own timing for the combined stream, so it is the synchronization source. Thus, all data packets forwarded by a mixer will be marked with the mixer's own SSRC identifier. In order to preserve the identity of the original sources contributing to the mixed packet, the mixer should insert their SSRC identifiers into the CSRC identifier list following the fixed RTP header of the packet. A mixer that is also itself a contributing source for some packet should explicitly include its own SSRC identifier in the CSRC list for that packet.
The ONLY mixers provided for shall be MCUs

7.2 RTCP Processing in Mixers

Since a mixer generates a new data stream of its own, it does not pass through SR or RR packets at all and instead generates new information for both sides.

SR sender information: A mixer does not pass through sender information from the sources it mixes because the characteristics of the source streams are lost in the mix. As a synchronization source, the mixer generates its own SR packets with sender information about the mixed data stream and sends them in the same direction as the mixed stream.

SR/RR reception report blocks: A mixer generates its own reception reports for sources in each cloud and sends them out only to the same cloud. It does not send these reception reports to the other clouds and does not forward reception reports from one cloud to the others because the sources would not be SSRCs there (only CSRCs).

SDES: Mixers typically forward without change the SDES information they receive from one cloud to the others, but may, for example, decide to filter non-CNAME SDES information if bandwidth is limited. The CNAMEs must be forwarded to allow SSRC identifier collision detection to work. (An identifier in a CSRC list generated by a mixer might collide with an SSRC identifier generated by an end system.) A mixer must send SDES CNAME

~~information about itself to the same clouds that it sends SR or RR packets.~~

~~Since mixers do not forward SR or RR packets, they will typically be extracting SDES packets from a compound RTCP packet. To minimize overhead, chunks from the SDES packets may be aggregated into a single SDES packet which is then stacked on an SR or RR packet originating from the mixer. The RTCP packet rate may be different on each side of the mixer.~~

~~A mixer that does not insert CSRC identifiers may also refrain from forwarding SDES CNAMEs. In this case, the SSRC identifier spaces in the two clouds are independent. As mentioned earlier, this mode of operation creates a danger that loops can't be detected.~~
~~APP: The treatment of APP packets by mixers is application-specific.~~

~~7.4 Cascaded Mixers~~

~~Do not allow Mixer cascades~~

~~An RTP session may involve a collection of mixers and translators as shown in Figure 3. If two mixers are cascaded, such as M2 and M3 in the figure, packets received by a mixer may already have been mixed and may include a CSRC list with multiple identifiers. The second mixer should build the CSRC list for the outgoing packet using the CSRC identifiers from already-mixed input packets and the SSRC identifiers from unmixed input packets. This is shown in the output arc from mixer M3 labeled M3:89(64,45) in the figure. As in the case of mixers that are not cascaded, if the resulting CSRC list has more than 15 identifiers, the remainder cannot be included.~~

8 SSRC Identifier Allocation and Use Review and define

The SSRC identifier carried in the RTP header and in various fields of RTCP packets is a random 32-bit number that is required to be globally unique within an RTP session. It is crucial that the number be chosen with care in order that participants on the same network or starting at the same time are not likely to choose the same number.

It is not sufficient to use the local network address (such as an IPv4 address) for the identifier because the address may not be unique. Since RTP translators and mixers enable interoperation among multiple networks with different address spaces, the allocation patterns for addresses within two spaces might result in a much higher rate of collision than would occur with random allocation.

Multiple sources running on one host would also conflict.

It is also not sufficient to obtain an SSRC identifier simply by calling random() without carefully initializing the state. An example of how to generate a random identifier is presented in Appendix A.6.

8.2 Loop Detection

Although the probability of SSRC identifier collision is low, all RTP implementations must be prepared to detect collisions and take the appropriate actions to resolve them. If a source discovers at any time that another source is using the same SSRC identifier as its own, it must send an RTCP BYE packet for the old identifier and choose another random one. If a receiver discovers that two other sources are colliding, it may keep the packets from one and discard the packets from the other when this can be detected by different source transport addresses or CNAMEs. The two sources are expected to resolve the collision so that the situation doesn't last.

o A mixer can close a loop by sending to the same transport destination upon which it receives packets, either directly or through another mixer or translator. In this case a source might show up both as an SSRC on a data packet and a CSRC in a mixed data packet.

A source may discover that its own packets are being looped, or that packets from another source are being looped (a third-party loop).

Both loops and collisions in the random selection of a source identifier result in packets arriving with the same SSRC identifier but a different source transport address, which may be that of the end system originating the packet or an intermediate system. Consequently, if a source changes its source transport address, it must also choose a new SSRC identifier to avoid being interpreted as a looped source. Loops or collisions occurring on the far side of a translator or mixer cannot be detected using the source transport address if all copies of the packets go through the translator or mixer, however collisions may still be detected when chunks from two RTCP SDES packets contain the same SSRC identifier but different CNAMEs.

To detect and resolve these conflicts, an RTP implementation must include an algoritsimilar to the one described below. It ignores packets from a new source or loop that collide with an established

source. It resolves collisions with the participant's own SSRC identifier by sending an RTCP BYE for the old identifier and choosing a new one. However, when the collision was induced by a loop of the participant's own packets, the algorithm will choose a new identifier only once and thereafter ignore packets from the looping source transport address. This is required to avoid a flood of BYE packets.

This algorithm depends upon the source transport address being the same for both RTP and RTCP packets from a source. The algorithm would require modifications to support applications that don't meet this constraint.

This algorithm requires keeping a table indexed by source identifiers and containing the source transport address from which the identifier was (first) received, along with other state for that source. Each SSRC or CSRC identifier received in a data or control packet is looked up in this table in order to process that data or control information. For control packets, each element with its own SSRC, for example an SDES chunk, requires a separate lookup. (The SSRC in a reception report block is an exception.) If the SSRC or CSRC is not found, a new entry is created. These table entries are removed when an RTCP BYE packet is received with the corresponding SSRC, or after no packets have arrived for a relatively long time (see Section 6.2.1).

In order to track loops of the participant's own data packets, it is also necessary to keep a separate list of source transport addresses (not identifiers) that have been found to be conflicting. Note that this should be a short list, usually empty. Each element in this list stores the source address plus the time when the most recent conflicting packet was received. An element may be removed from the list when no conflicting packet has arrived from that source for a time on the order of 10 RTCP report intervals (see Section 6.2).

For the algorithm as shown, it is assumed that the participant's own source identifier and state are included in the source identifier table. The algorithm could be restructured to first make a separate comparison against the participant's own source identifier.

```
IF the SSRC or CSRC identifier is not found in the source
   identifier table:
THEN create a new entry storing the source transport address
     and the SSRC or CSRC along with other state.
```

```
        CONTINUE with normal processing.

    (identifier is found in the table)

    IF the source transport address from the packet matches
      the one saved in the table entry for this identifier:
    THEN CONTINUE with normal processing.

    (an identifier collision or a loop is indicated)

    IF the source identifier is not the participant's own:
    THEN IF the source identifier is from an RTCP SDES chunk
           containing a CNAME item that differs from the CNAME
           in the table entry:
         THEN (optionally) count a third-party collision.
ELSE (optionally) count a third-party loop.
         ABORT processing of data packet or control element.

    (a collision or loop of the participant's own data)

    IF the source transport address is found in the list of
     conflicting addresses:
    THEN IF the source identifier is not from an RTCP SDES chunk
           containing a CNAME item OR if that CNAME is the
           participant's own:
         THEN (optionally) count occurrence of own traffic looped.
              mark current time in conflicting address list entry.
              ABORT processing of data packet or control element.
    log occurrence of a collision.
    create a new entry in the conflicting address list and
    mark current time.
    send an RTCP BYE packet with the old SSRC identifier.
    choose a new identifier.
    create a new entry in the source identifier table with the
     old SSRC plus the source transport address from the packet
     being processed.
    CONTINUE with normal processing.
```

In this algorithm, packets from a newly conflicting source address will be ignored and packets from the original source will be kept. (If the original source was through a mixer and later the same source is received directly, the receiver may be well advised to switch unless other sources in the mix would be lost.) If no packets arrive

from the original source for an extended period, the table entry will be timed out and the new source will be able to take over. This might occur if the original source detects the collision and moves to a new source identifier, but in the usual case an RTCP BYE packet will be received from the original source to delete the state without having to wait for a timeout.

When a new SSRC identifier is chosen due to a collision, the candidate identifier should first be looked up in the source identifier table to see if it was already in use by some other source. If so, another candidate should be generated and the process repeated.

A loop of data packets to a multicast destination can cause severe network flooding. All mixers and translators are required to implement a loop detection algorithm like the one here so that they can break loops. This should limit the excess traffic to no more than one duplicate copy of the original traffic, which may allow the

session to continue so that the cause of the loop can be found and fixed. However, in extreme cases where a mixer or translator does not properly break the loop and high traffic levels result, it may be necessary for end systems to cease transmitting data or control packets entirely. This decision may depend upon the application. An error condition should be indicated as appropriate. Transmission might be attempted again periodically after a long, random time (on the order of minutes).

~~9 Security~~

Covered by H.323

10 RTP over Network and Transport Protocols

This section describes issues specific to carrying RTP packets within particular network and transport protocols. The following rules apply unless superseded by protocol-specific definitions outside this specification.

RTP relies on the underlying protocol(s) to provide demultiplexing of RTP data and RTCP control streams. For UDP and similar protocols, RTP uses an even port number and the corresponding RTCP stream uses the next higher (odd) port number. If an application is supplied with an odd number for use as the RTP port, it should replace this number with the next lower (even) number.

RTP data packets contain no length field or other delineation, therefore RTP relies on the underlying protocol(s) to provide a length indication. The maximum length of RTP packets is limited only by the underlying protocols.

A profile may specify a framing method to be used even when RTP is carried in protocols that do provide framing in order to allow carrying several RTP packets in one lower-layer protocol data unit, such as a UDP packet. Carrying several RTP packets in one network or transport packet reduces header overhead and may simplify synchronization between different streams.

11 Summary of Protocol Constants

This section contains a summary listing of the constants defined in this specification.

The RTP payload type (PT) constants are defined in profiles rather than this document. However, the octet of the RTP header which contains the marker bit(s) and payload type must avoid the reserved values 200 and 201 (decimal) to distinguish RTP packets from the RTCP SR and RR packet types for the header validation procedure described in Appendix A.1. For the standard definition of one marker bit and a 7-bit payload type field as shown in this specification, this restriction means that payload types 72 and 73 are reserved.

11.1 RTCP packetypes

abbrev.	name	value
SR	sender report	200
RR	receiver report	201
SDES	source description	202
APP	application-defined	204

These type values were chosen in the range 200-204 for improved header validity checking of RTCP packets compared to RTP packets or other unrelated packets. When the RTCP packet type field is compared to the corresponding octet of the RTP header, this range corresponds to the marker bit being 1 (which it usually is not in data packets) and to the high bit of the standard payload type field being 1 (since the static payload types are typically defined in the low half). This range was also chosen to be some distance numerically from 0 and 255

since all-zeros and all-ones are common data patterns.

Since all compound RTCP packets must begin with SR or RR, these codes were chosen as an even/odd pair to allow the RTCP validity check to test the maximum number of bits with mask and value.

11.2 SDES types

abbrev.	name	value
END	end of SDES list	0
CNAME	canonical name	1
NAME	user name	2
EMAIL	user's electronic mail address	3
PHONE	user's phone number	4
LOC	geographic user location	5
TOOL	name of application or tool	6
NOTE	notice about the source	7
PRIV	private extensions	8

12 RTP Profiles and Payload Format Specifications

Within this specification, the following items have been identified for possible definition within a profile

RTP data header: The octet in the RTP data header that contains the marker bit and payload type field may be redefined by a profile to suit different requirements, for example with more or fewer marker bits (Section 5.3, p. 11).

Payload types: Assuming that a payload type field is included, the

profile will usually define a set of payload formats (e.g., media encodings) and a default static mapping of those formats to payload type values. Some of the payload formats may be defined by reference to separate payload format specifications. For each payload type defined, the profile must specify the RTP timestamp clock rate to be used (Section 5.1, p. 10).

RTP data header additions: Additional fields may be appended to the fixed RTP data header if some additional functionality is required across the profile's class of applications independent of payload type (Section 5.3, p. 11).

RTP data header extensions: The contents of the first 16 bits of the RTP data header extension structure must be defined if use of that mechanism is to be allowed under the profile for implementation-specific extensions (Section 5.3.1, p. 12).

RTCP packet types: New application-class-specific RTCP packet types may be defined and registered with IANA.

RTCP report interval: A profile should specify that the values suggested in Section 6.2 for the constants employed in the calculation of the RTCP report interval will be used. Those are the RTCP fraction of session bandwidth, the minimum report interval, and the bandwidth split between senders and receivers. A profile may specify alternate values if they have been demonstrated to work in a scalable manner.

SR/RR extension: An extension section may be defined for the RTCP SR and RR packets if there is additional information that should be reported regularly about the sender or receivers (Section 6.3.3, p. 23).

SDES use: The profile may specify the relative priorities for RTCP SDES items to be transmitted or excluded entirely (Section 6.2.2); an alternate syntax or semantics for the CNAME item (Section 6.4.1); the format of the LOC item (Section 6.4.5); the semantics and use of the NOTE item (Section 6.4.7
transport layer protocol to carry RTP packets may be required.

Transport mapping: A mapping of RTP and RTCP to transport-level addresses, e.g., UDP ports, other than the standard mapping defined in Section 10, p. 39 may be specified.

Encapsulation: An encapsulation of RTP packets may be defined to allow multiple RTP data packets to be carried in one lower-layer packet or to provide framing over underlying protocols that do not already do so (Section 10, p. 39).

Table of Contents

AVC-866

Annex A - RTP Profile

Telecommunication Standardization Sector

Origina: English

SOURCE : Yizhak Idan, RADVision Ltd.

e-mail: idan@radvision.rad.co.il

voice: +972 3 645 5220

fax: +972 3 647 6669

TITLE : RTP Profile for H.323 Videoconference Using H.225.0 Media Stream Packetization and Synchronization for Visual Telephone Systems on Non-Guaranteed Quality of Service LANs

DATE: December 27, 1995

Main modifications:

1. Definition of PCM (mu-Law and A-Law) as minimal conformance audio payloads on the LAN. DVI4 was omitted and A-Law was added.
2. Addition of guidelines concerning the timestamp use for video frames (see our mail to the reflector on 12/24/95).
3. Omission of encryption guidelines.
4. Addition of the possibility interleaving several different media payloads in the same RTP packet (multiplexed data).
5. Removal of implementation annexes and references to IETF documents.

Internet Engineering Task Force — Audio-Video Transport Working Group
Internet Draft — Schulzrinne
draft-ietf-avt-profile-06.txt — GMD Fokus
November 20, 1995
Expires: 4/1/96

RTP Profile for H.225.0 Audio and Video Conferences with Minimal Control

STATUS OF THIS MEMO

This document is an Internet Draft. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet Drafts.

Internet Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet Drafts as reference material or to cite them other than as ``work in progress''.

To learn the current status of any Internet Draft, please check the ``1id-abstracts.txt'' listing contained in the Internet Drafts Shadow Directories on ftp.is.co.za (Africa), nic.nordu.net (Europe), munnari.oz.au (Pacific Rim), ds.internic.net (US East Coast), or ftp.isi.edu (US West Coast).

Distribution of this document is unlimited.

ABSTRACT

This memo describes a profile for the use of the real-time transport protocol (RTP), version 2, and the associated control protocol, RTCP, within audio and video multiparticipant conferences with minimal control. It provides interpretations of generic fields within the RTP specification suitable for H.323 audio and video conferences. In particular, this document defines a set of default mappings from payload type numbers to encodings.

The document also describes how audio and video data may be carried within RTP. It defines a set of standard encodings and their names when used within RTP. However, the encoding definitions are independent of the

particular transport mechanism used. The descriptions provide pointers to reference implementations and the detailed standards. ~~This document is meant as an aid for implementors of audio, video and other real-time multimedia applications~~.

1 Introduction

This profile defines aspects of RTP left unspecified in the RTP Annex X ~~Version 2 protocol definition (RFC TBD). This profile is intended for the use within audio and video conferences with minimal session control. In particular, no support for the negotiation of parameters or membership control is provided. The profile is expected to be useful in sessions where no negotiation or membership control are used (e.g., using the static payload types and the membership indications provided by RTCP), but this profile may also be useful in conjunction with a higher-level control protocol.~~

Use of this profile occurs by use of the appropriate applications; there is no explicit indication by port number, protocol identifier or the like.

~~Other profiles may make different choices for the items specified here.~~

2 RTP and RTCP Packet Forms and Protocol Behavior

The section "RTP Profiles and Payload Format Specification" enumerates a number of items that can be specified or modified in a profile. This section addresses these items. Generally, this profile follows the default and/or recommended aspects of the RTP specification.

RTP data header: The standard format of the fixed RTP data header is used (one marker bit).

Payload types: Static payload types are defined in Section 6.

RTP data header additions: No additional fixed fields are appended to the RTP data header.

RTP data header extensions: No RTP header extensions are defined, but applications operating under this profile may use such extensions. Thus, applications should not assume that the RTP

header X bit is always zero and should be prepared to ignore the header extension. If a header extension is defined in the future, that definition must specify the contents of the first 16 bits in such a way that multiple different extensions can be identified.

RTCP packet types: No additional RTCP packet types are defined by this profile specification.

RTCP report interval: The suggested constants are to be used for the RTCP report interval calculation.

SR/RR extension: No extension section is defined for the RTCP SR or RR packet.

SDES use: Applications may use any of the SDES items described.
~~While CNAME information is sent every reporting interval, other items should be sent only every fifth reporting interval.~~

~~Security: The RTP default security services are also the default under this profile.~~

~~String-to-key mapping: A user-provided string ("pass phrase") is hashed with the MD5 algorithm to a 16-octet digest. An n-bit key is extracted from the digest by taking the first n bits from the digest. If several keys are needed with a totalof 128 bits or less (as for triple DES), they are extracted in order from that digest. The octet ordering is specified in RFC 1423, Section 2.2. (Note that some DES implementations require that the 56-bit key be expanded into 8 octets by inserting an odd parity bit in the most significant bit of the octet to go with each 7 bits of the key.)~~

~~It is suggested that pass phrases are restricted to ASCII letters, digits, the hyphen, and white space to reduce the the chance of transcription errors when conveying keys by phone, fax, telex or email.~~

~~The pass phrase may be preceded by a specification of the encryption algorithm. Any characters up to the first slash (ASCII 0x2f) are taken as the name of the encryption algorithm. The encryption format specifiers should be drawn from RFC 1423 or any additional identifiers registered with IANA. If no slash is present, DES-CBC is assumed as default. The encryption algorithm specifier is case sensitive.~~

~~The pass-phrase typed by the user is transformed to a canonical form before applying the hash algorithm. For that purpose, we define return, tab, or vertical tab as well as all characters contained in the Unicode space characters table. The transformation consists of the following steps: (1) convert the input string to the ISO 10646 character set, using the UTF-8 encoding as specified in Annex P to ISO/IEC 10646-1:1993 (ASCII characters require no mapping, but ISO 8859-1 characters do); (2) remove leading and trailing white space characters; (3) replace one or more contiguous white space characters by a single space (ASCII or UTF-8 0x20); (4) convert all letters to lower case and replace sequences of characters and non-spacing accents with a single character, where possible. A minimum length of 16 key characters (after applying the transformation) should be enforced by the application, while applications must allow up to 256 characters of input.~~

—Underlying protocol: The profile specifies the use of RTP over unicast and multicast UDP. (This does not precluthe use of these definitions when RTP is carried by other lower-layer protocols.)

Transport mapping: The standard mapping of RTP and RTCP to transport-level addresses is used.

Encapsulation: No encapsulation of RTP packets is specified.

3 Registering Payload Types

This profile defines a set of standard encodings and their payload types when used within RTP. Other encodings and their payload types are to be registered with the Internet Assigned Numbers Authority (IANA). When registering a new encoding/payload type, the following information should be provided:

o name and description of encoding, in particular the RTP timestamp clock rate; the names defined here are 3 or 4 characters long to allow a compact representation if needed;

o indication of who has change control over the encoding (for example, ISO, CCITT/ITU, other international standardization bodies, a consortium or a particular company or group of companies);

o any operating parameters or profiles;

o a reference to a further description, if available, for example (in order of preference) an RFC, a published paper, a patent filing, a technical report, documented source code or a computer manual;

o for proprietary encodings, contact information (postal and e-mail type);

o the payload type value for this profile, if necessary (see below).

Note that not all encodings to be used by RTP need to be assigned a

static payload type. Non-RTP means beyond the scope of this memo (such as directory services or invitation protocols) may be used to establish a dynamic mapping between a payload type drawn from the range 96-127 and an encoding. For implementor convenience, this profile contains descriptions of encodings which do not currently have a static payload type assigned to them.

The available payload type space is relatively small. Thus, new static payload types are assigned only if the following conditions are met:

o The encoding is of interest to the Internet community at large.

o It offers benefits compared to existing encodings and/or is required for interoperation with existing, widely deployed conferencing or multimedia systems.

o The description is sufficient to build a decoder.

4 Audio

4.1 Encoding-Independent Recommendations

For applications which send no packets during silence, the first packet of a talkspurt (first packet after a silence period) is distinguished by setting the marker bit in the RTP data header. Applications without silence suppression set the bit to zero.

The RTP clock rate used for generating the RTP timestamp is independent of the number of channels and the encoding; it equals the

number of sampling periods per second. For N-channel encodings, each sampling period (say, 1/8000 of a second) generates N samples. (This terminology is standard, but somewhat confusing, as the total number of samples generated per second is then the sampling rate times the channel count.)

If multiple audio channels are used, channels are numbered left-to-right, starting at one. In RTP audio packets, information from lower-numbered channels precedes that from higher-numbered channels. For more than two channels, the convention followed by the AIFF-C audio interchange format should be followed [1], using the following notation:

l left
r right
c center

S surround
F front
R rear

channels	description	channel 1	2	3	4	5	6
2	stereo	l	r				
3		l	r	c			
4	quadrophonic	Fl	Fr	Rl	Rr		
4		l	c	r	S		
5		Fl	Fr	Fc	Sl	Sr	
6		l	lc	c	r	rc	S

Samples for all channels belonging to a single sampling instant must be within the same packet. The interleaving of samples from different channels depends on the encoding. General guidelines are given in Section 4.2 and 4.3.

The sampling frequency should be drawn from the set: 8000, 11025, 16000, 22050, 24000, 32000, 44100 and 48000 Hz. (The Apple Macintosh computers have native sample rates of 22254.54 and 11127.27, which can be converted to 22050 and 11025 with acceptable quality by dropping 4 or 2 samples in a 20 ms frame.) However, most audio

encodings are defined for a more restricted set of sampling frequencies. Receivers should be prepared to accept multi-channel audio, but may choose to only play a single channel.

The following recommendations are default operating parameters. Applications should be prepared to handle other values. The ranges given are meant to give guidance to application writers, allowing a set of applications conforming to these guidelines to interoperate without additional negotiation. These guidelines are not intended to restrict operating parameters for applications that can negotiate a set of interoperable parameters, e.g., through a conference control protocol.

For packetized audio, the default packetization interval should have a duration of 20 ms, unless otherwise noted when describing the encoding. The packetization interval determines the minimum end-to-end delay; longer packets introduce less header overhead but higher delay and make packet loss more noticeable. For non-interactive applications such as lectures or links with severe bandwidth constraints, a higher packetization delay may be appropriate. A receiver should accept packets representing between 0 and 200 ms of audio data. This restriction allows reasonable buffer sizing for the

receiver.

4.2 Guidelines for Sample-Based Audio Encodings

In sample-based encodings, each audio sample is represented by a fixed number of bits. Within the compressed audio data, codes for individual samples may span octet boundaries. An RTP audio packet may contain any number of audio samples, subject to the constraint that the number of bits per sample times the number of samples per packet yields an integral octet count. Fractional encodings produce less than one octet per sample.

The duration of an audio packet is determined by the number of samples in the packet.

For sample-based encodings producing one or more octets per sample, samples from different channels sampled at the same sampling instant are packed in consecutive octets. For example, for a two-channel encoding, the octet sequence is (left channel, first sample), (right channel, first sample), (left channel, second sample), (right channel, second sample), ... For multi-octet encodings, octets are transmitted in network byte order (i.e., most significant octet

first).

The packing of sample-based encodings producing less than one octet per sample is encoding-specific.

4.3 Guidelines for Frame-Based Audio Encodings

Frame-based encodings encode a fixed-length block of audio into another block of compressed data, typically also of fixed length. For frame-based encodings, the sender may choose to combine several such frames into a single message. The receiver can tell the number of frames contained in a message since the frame duration is defined as part of the encoding.

For frame-based codecs, the channel order is defined for the whole block. That is, for two-channel audio, right and left samples are coded independently, with the encoded frame for the left channel preceding that for the right channel.

All frame-oriented audio codecs should be able to encode and decode several consecutive frames within a single packet. Since the frame size for the frame-oriented codecs is given, there is no need to use a separate designation for thesame encoding, but with different number of frames per packet.

4.4 Audio Encodings

encoding	sample/frame	bits/sample	ms/frame
1016	frame	N/A	30
DVI4	sample	4	
G721	sample	4	
G722	sample	8	
G728	frame	N/A	2.5
GSM	frame	N/A	20
L8	sample	8	
L16	sample	16	
LPC	frame	N/A	20
MPA	frame	N/A	
PCMA	sample	8	
PCMU	sample	8	
VDVI	sample	var.	

Table 1: Properties of Audio Encodings

The characteristics of standard audio encodings are shown in Table 1 and their payload types are listed in Table 2.

4.4.1 1016

Encoding 1016 is a frame based encoding using code-excited linear prediction (CELP) and is specified in Federal Standard FED-STD 1016 [2,3,4,5].

The U. S. DoD's Federal-Standard-1016 based 4800 bps code excited linear prediction voice coder version 3.2 (CELP 3.2) Fortran and C simulation source codes are available for worldwide distribution at no charge (on DOS diskettes, but configured to compile on Sun SPARC stations) from: Bob Fenichel, National Communications System, Washington, D.C. 20305, phone +1-703-692-2124, fax +1-703-746-4960.

4.4.2 DVI4

DVI4 is specified, with pseudo-code, in [6] as the IMA ADPCM wave type. A specification titled "DVI ADPCM Wave Type" can also be found in the Microsoft Developer Network Development Library CD ROM published quarterly by Microsoft. The relevant section is found under Product Documentation, SDKs, Multimedia Standards Update, New Multimedia Data Types and Data Techniques, Revision 3.0, April 15, 1994. However, the encoding defined here as DVI4 differs in two respects from these recommendations:

o The header contains the predicted value rather than the first sample value.

o IMA ADPCM blocks contain odd number of samples, since the first sample of a block is contained just in the header (uncompressed), followed by an even number of compressed samples. DVI4 has an even number of compressed samples only, using the 'predict' word from the header to decode the first sample.

Each packet contains a single DVI block. The profile only defines the 4-bit-per-sample version, while IMA also specifies a 3-bit-per-sample encoding.

The "header" word for each channel has the following structure:

```
int16  predict;  /* predicted value of first sample
                    from the previous block (L16 format) */
u_int8 index;    /* current index into stepsize table */
u_int8 reserved; /* set to zero by sender, ignored by receiver */
```

Packing of samples for multiple channels is for further study.

The document IMA Recommended Practices for Enhancing Digital Audio Compatibility in Multimedia Systems (version 3.0) contains the algorithm description. It is available from

Interactive Multimedia Association
48 Maryland Avenue, Suite 202
Annapolis, MD 21401-8011
USA
phone: +1 410 626-1380

4.4.3 G721

G721 is specified in ITU recommendation G.721. Reference implementations for G.721 are available as part of the CCITT/ITU-T Software Tool Library (STL) from the ITU General Secretariat, Sales Service, Place du Nations, CH-1211 Geneve 20, Switzerland. The library is covered by a license.

4.4.4 G722

G722 is specified in ITU-T recommendation G.722, "7 kHz audio-coding within 64 kbit/s".

4.4.5 G728

G728 is specified in ITU-T recommendation G.728, "Coding of speech at 16 kbit/s using low-delay code excited linear prediction".

4.4.6 GSM

GSM (group speciale mobile) denotes the European GSM 06.10 provisional standard for full-rate speech transcoding, prI-ETS 300 036, which is based on RPE/LTP (residual pulse excitation/long term prediction) coding at a rate of 13 kb/s [7,8,9]. The standard can be obtained from

ETSI (European Telecommunications Standards Institute)
ETSI Secretariat: B.P.152
F-06561 Valbonne Cedex
France
Phone: +33 92 94 42 00
Fax: +33 93 65 47 16

4.4.7 L8

L8 denotes linear audio data, using 8-bits of precision with an offset of 128, that is, the most negative signal is encoded as zero.

4.4.8 L16

L16 denotes uncompressed audio data, using 16-bit signed representation with 65535 equally divided steps between minimum and maximum signal level, ranging from -32768 to 32767. The value is represented in two's complement notation and network byte order.

4.4.9 LPC

LPC designates an experimental linear predictive encoding contributed by Ron Frederick, Xerox PARC, which is based on an implementation written by Ron Zuckerman, Motorola, posted to the Usenet group comp.dsp on June 26, 1992.

4.4.10 MPA

MPA denotes MPEG-I or MPEG-II audio encapsulated as elementary streams. The encoding is defined in ISO standards ISO/IEC 11172-3 and 13818-3. The encapsulation is specified in work in progress [10], Section 3. The authors can be contacted at

Don Hoffman
Sun Microsystems, Inc.

Mail-stop UMPK14-305
2550 Garcia Avenue
Mountain View, California 94043-1100
USA
electronic mail: don.hoffman@eng.sun.com

Sampling rate and channel count are contained in the payload. MPEG-I audio supports sampling rates of 32000, 44100, and 48000 Hz (ISO/IEC 11172-3, section 1.1; "Scope"). MPEG-II additionally supports ISO/IEC

11172-3 Audio...").

4.4.11 PCMA

PCMA is specified in CCITT/ITU-T recommendation G.711. Audio data is encoded as eight bits per sample, after logarithmic scaling. Code to convert between linear and A-law companded data is available in [6]. A detailed description is given by Jayant and Noll [11].

4.4.12 PCMU

PCMU is specified in CCITT/ITU-T recommendation G.711. Audio data is encoded as eight bits per sample, after logarithmic scaling. Code to convert between linear and mu-law companded data is available in [6]. PCMU is the encoding used for the Internet media type audio/basic. A detailed description is given by Jayant and Noll [11].

4.4.13 VDVI

VDVI is a variable-rate version of DVI4, yielding speech bit rates of between 10 and 25 kb/s. It is specified for single-channel operation only. It uses the following encoding:

DVI4 codeword	VDVI bit pattern
0	00
1	010
2	1100
3	11100
4	111100
5	1111100
6	11111100
7	11111110
8	10
9	011
10	1101
11	11101
12	111101
13	1111101
14	11111101
15	11111111

5 Video

The following video encodings are currently defined, with their abbreviated names used for identification:

5.1 CelB

The CELL-B encoding is a proprietary encoding proposed by Sun Microsystems. The byte stream format is described in work in progress [12]. The author can be contacted at

Michael F. Speer
Sun Microsystems Computer Corporation
2550 Garcia Ave MailStop UMPK14-305
Mountain View, CA 94043
United States
electronic mail: michael.speer@eng.sun.com

5.2 JPEG

The encoding is specified in ISO Standards 10918-1 and 10918-2. The RTP payload format is as specified in work in progress [13]. Further information can be obtained from

Steven McCanne
Lawrence Berkeley National Laboratory
M/S 46A-1123
One Cyclotron Road
Berkeley, CA 94720
United States
Phone: +1 510 486 7520
electronic mail: mccanne@ee.lbl.gov

5.3 H261

The encoding is specified in CCITT/ITU-T standard H.261. The packetization and RTP-specific properties are described in work in progress [14]. Further information can be obtained from

Thierry Turletti
Office NE 43-505
Telemedia, Networks and Systems
Laboratory for Computer Science
Massachusetts Institute of Technology

545 Technology Square
Cambridge, MA 02139
United States
electronic mail: turletti@clove.lcs.mit.edu

5.4 MPV

MPV designates the use MPEG-I and MPEG-II video encoding elementary streams as specified in ISO Standards ISO/IEC 11172 and 13818-2, respectively. The RTP payload format is as specified in work in progress [10], Section 3. See the description of the MPA audio encoding for contact information.

5.5 MP2T

MP2T designates the use of MPEG-II transport streams, for either audio or video. The encapsulation is described in work in progress, [10], Section 2. See the description of the MPA audio encoding for contact information.

5.6 nv

The encoding is implemented in the program 'nv', version 4, developed at Xerox PARC by Ron Frederick. Further information is available from the author:

Ron Frederick
Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304
United States
electronic mail: frederic@parc.xerox.com

6 Payload Type (PT) Definitions

Table 2 defines this profile's static payload type values for the PT field of the RTP data header. ~~A new RTP payload format specification may be registered with the IANA by name, and may also be assigned a static payload type value from the range marked in Section 3.~~

In addition, payload type values in the range 96-127 may be defined dynamically through a conference control protocol, which is beyond the scope of this document. For example, a session directory could specify that for a given session, payload type 96 indicates PCMU

encoding, 8,000 Hz sampling rate, 2 channels. The payload type range marked 'reserved' has been set aside so that RTCP and RTP packets can be reliably distinguished (see Section "Summary of Protocol Constants" of the RTP protocol specification).

An RTP source emits a single RTP payload type at any given ~~time; the interleaving of several RTP payload types in a single RTP session is not allowed, but~~. Mmultiple RTP sessions may be used in parallel to send multiple media. The payload types currently defined in this profile carry either audio or video~~, but not both~~. However, it is allowed to define payload types that combine several media, e.g., audio and video, with appropriate separation in the payload format. Session participants agree through mechanisms beyond the scope of this specification on the set of payload types allowed in a given session. This set may, for example, be defined by the capabilities of the applications used, negotiated by a conference control protocol or established by agreement between the human participants.

Audio applications operating under this profile should, at minimum, be able to send and receive payload types 0 (PCMU) and 8 (PCMA)~~ and 5 (DVI4)~~. This allows interoperability without format negotiation and successful negotation with a conference control protocol.

All current video encodings use a timestamp frequency of 90,000 Hz, the same as the MPEG presentation time stamp frequency. This frequency yields exact integer timestamp increments for the typical 24 (HDTV), 25 (PAL), and 29.97 (NTSC) and 30 Hz (HDTV) frame rates and 50, 59.94 and 60 Hz field rates. While 90 kHz is the recommended rate for future video encodings used within this profile, other rates are possible. However, it is not sufficient to use the video frame rate (typically between 15 and 30 Hz) because that does not provide adequate resolution for typical synchronization requirements when calculating the RTP timestamp corresponding to the NTP timestamp in an RTCP SR packet [15]. The timestamp resolution must also be sufficient for the jitter estimate contained in the receiver reports.
H.323 video RTP packets header timestamp indicates the frame count as well as the accumulated sent bytes count. The 2 most significant bytes of the RTP header timestamp are cyclic frame count and the 2 least significant bytes are cyclic accumulated sent bytes count. The initial value of the timestamp will be randomly chosen.

The standard video encodings and their payload types are listed in Table 2.

7 Port Assignment

As specified in the RTP protocol definition, RTP data is to be carried on an even UDP port number and the corresponding RTCP packets are to be carried on the next higher (odd) port number.

Applications operating under this profile may use any such UDP port pair. For example, the port pair may be allocated randomly by a session management program. A single fixed port number pair cannot be required because multiple applications using this profile are likely to run on the same host, and there are some operating systems that do not allow multiple processes to use the same UDP port with different multicast addresses.

PT	encoding name	audio/video (A/V)	clock rate (Hz)	channels (audio)
0	PCMU	A	8000	1
1	1016	A	8000	1
2	G721	A	8000	1
3	GSM	A	8000	1
4	unassigned	A	8000	1
5	DVI4	A	8000	1
6	DVI4	A	16000	1
7	LPC	A	8000	1
8	PCMA	A	8000	1
9	G722	A	8000	1
10	L16	A	44100	2
11	L16	A	44100	1
12	unassigned	A		
13	unassigned	A		
14	MPA	A	90000	(see text)
15	G728	A	8000	1
16--23	unassigned	A		
24	unassigned	V		
25	CelB	V	90000	
26	JPEG	V	90000	
27	unassigned	V		
28	nv	V	90000	
29	unassigned	V		
30	unassigned	V		
31	H261	V	90000	
32	MPV	V	90000	
33	MP2T	AV	90000	
34--71	unassigned	?		

72--76	reserved	N/A	N/A	N/A
77--95	unassigned	?		
96--127	dynamic	?		

Table 2: Payload types (PT) for standard audio and video encodings

However, port numbers 5004 and 5005 have been registered for use with this profile for those applications that choose to use them as the default pair. Applications that operate under multiple profiles may use this port pair as an indication to select this profile if they are not subject to the constraint of the previous paragraph. Applications need not have a default and may require that the port pair be explicitly specified. The particular port numbers were chosen to lie in the range above 5000 to accomodate port number allocation practice within the Unix operating system, where port numbers below 1024 can only be used by privileged processes and port numbers

between 1024 and 5000 are automatically assigned by the operating system.