INTERNET-DRAFT 2 March 2003
Internet Engineering Task Force Expires: 2 September 2003
Audio/Video Transport Working Group

Timur Friedman, Paris 6
Ramon Caceres, ShieldIP
Alan Clark, Telchemy
Editors

RTP Extended Reports (RTP XR)

draft-ietf-avt-rtcp-report-extns-03.txt

Status of this Memo

Copyright Notice

Abstract

This document defines the extended report (XR) packet type for the RTP control protocol (RTCP). XR packets are composed of report blocks, and seven block types are defined here. The purpose of the extended reporting format is to convey information that supplements the six statistics that are contained in the report blocks used by RTCP's sender report (SR) and receiver report (RR) packets. Some applications, such as multicast inference of network characteristics

(MINC) or voice over IP (VoIP) monitoring, require other and more detailed statistics. In addition to the block types defined here, additional block types may be defined in the future by adhering to the framework that this document provides.

Table of Contents

1. Introduction

This document defines the extended report (XR) packet type for the

RTP control protocol (RTCP) [7]. XR packets convey information beyond that already contained in the reception report blocks of RTCP's sender report (SR) or receiver report (RR) packets. The information is of use across RTP profiles, and so is not appropriately carried in SR or RR profile-specific extensions. Information used for network management falls into this category, for instance.

The definition is broken out over the three following sections of this document, starting with a general framework and finishing with the specific information conveyed. The framework defined by Section 2 contains common header information followed by a series of components called report blocks. Section 3 defines the format common to such blocks. Section 4 defines seven block types.

Seven report block formats are defined by this document:

- Loss RLE Report Block (Section 4.1): Run length encoding of RTP packet loss reports.

- Duplicate RLE Report Block (Section 4.2): Run length encoding of reports of RTP packet duplicates.

- Timestamp Report Block (Section 4.3): A list of timestamps of received RTP packets.

- Statistics Summary Report Block (Section 4.4): Statistics on RTP packet sequence numbers, losses, duplicates, jitter, and TTL values.

- Receiver Timestamp Report Block (Section 4.5): Receiver-end timestamps that complement the sender-end timestamps already defined for RTCP.

- DLRR Report Block (Section 4.6): The delay since the last Receiver Timestamp Report Block was received, allowing non-senders to calculate round-trip times.

- VoIP Metrics Report Block (Section 4.7): Metrics for monitoring Voice over IP (VoIP) calls.

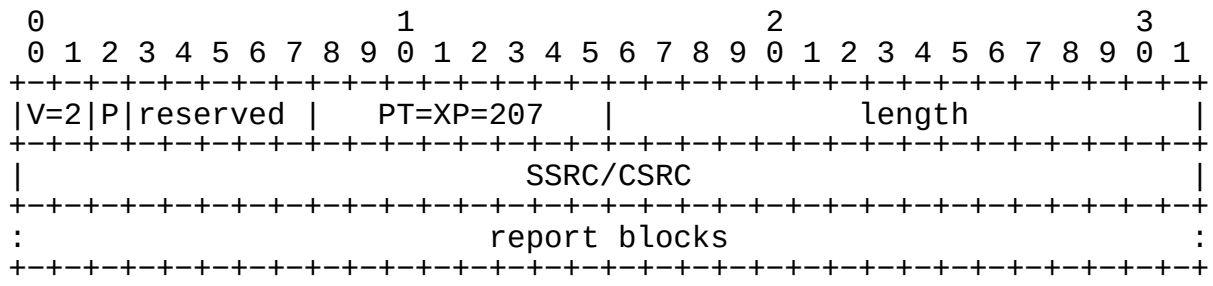
These blocks are defined within a minimal framework: a type field and a length field are common to all XR blocks. The purpose is to maintain flexibility and to keep overhead low. Other block formats, beyond the seven defined here, may be defined within this framework as the need arises.

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [1] and indicate requirement levels for compliance with this specification.

2. XR Packet Format

The XR packet consists of a header of two 32-bit words, followed by a number, possibly zero, of extended report blocks.



version (V): 2 bits
: Identifies the version of RTP. This specification applies to RTP version two.

padding (P): 1 bit
: If the padding bit is set, this XR packet contains some additional padding octets at the end. The semantics of this field are identical to the semantics of the padding field in the SR packet, as defined by the RTP specification.

reserved: 5 bits
: This field is reserved for future definition. In the absence of such definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

packet type (PT): 8 bits
: Contains the constant 207 to identify this as an RTCP XR packet. This value is registered with the Internet Assigned Numbers Authority (IANA), as described in Section 5.1.

length: 16 bits
: As described for the RTP sender report (SR) packet (see Section 6.3.1 of the RTP specification [7]). Briefly, the length of this XR packet in 32-bit words minus one, including the header and any padding.

SSRC: 32 bits
The synchronization source identifier for the originator of this XR packet.

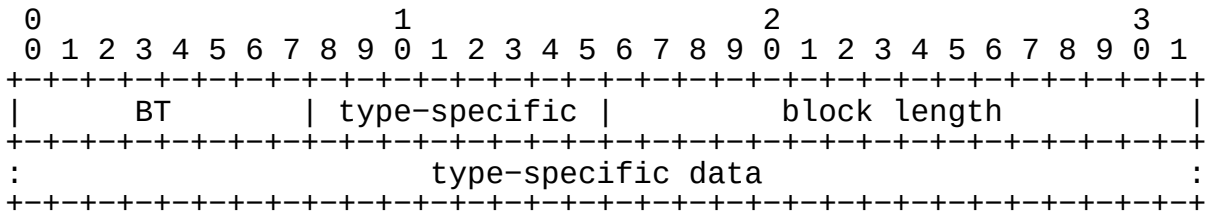
report blocks: variable length.
Zero or more extended report blocks. In keeping with the extended report block framework defined below, each block MUST consist of one or more 32-bit words.

3. Extended Report Block Framework

Extended report blocks are stacked, one after the other, at the end of an XR packet. An individual block's length is a multiple of 4 octets. The XR header's length field describes the total length of the packet, including these extended report blocks.

Each block has block type and length fields that facilitate parsing. A receiving application can demultiplex the blocks based upon their type, and can use the length information to locate each successive block, even in the presence of block types it does not recognize.

An extended report block has the following format:



block type (BT): 8 bits
Identifies the block format. Seven block types are defined in Section 4. Additional block types may be defined in future specifications. This field's name space is managed by the Internet Assigned Numbers Authority (IANA), as described in Section 5.2.

type-specific: 8 bits
The use of these bits is determined by the block type definition.

block length: 16 bits
The length of this report block including the header, in 32-bit words minus one. If the block type definition permits, zero is an acceptable value, signifying a block that consists of only

the BT, type-specific, and block length fields, with a null type-specific data field.

type-specific data: variable length
The use of this field is defined by the particular block type, subject to the constraint that it MUST be a multiple of 32 bits long. If the block type definition permits, It MAY be zero bits long.

4. Extended Report Blocks

This section defines seven extended report blocks: block types for losses, duplicates, packet reception timestamps, detailed reception statistics, receiver timestamps, receiver inter-report delays, and voice over IP (VoIP) metrics. An implementation SHOULD ignore incoming blocks with types either not relevant or unknown to it. Additional block types MUST be registered with the Internet Assigned Numbers Authority (IANA) [5], as described in Section 5.2.

4.1 Loss RLE Report Block

This block type permits detailed reporting upon individual packet receipt and loss events. Such reports can be used, for example, for multicast inference of network characteristics (MINC) [8]. With MINC, one can discover the topology of the multicast tree used for distributing a source's RTP packets, and of the loss rates along links within that tree. Or they could be used to provide raw data to a network management application.

Since a Boolean trace of lost and received RTP packets is potentially lengthy, this block type permits the trace to be compressed through run length encoding. To further reduce block size, loss event reports can be systematically dropped from the trace in a mechanism called thinning that is described below and that is studied in [9].

A participant that generates a Loss RLE Report Block should favor accuracy in reporting on observed events over interpretation of those events whenever possible. Interpretation should be left to those who observe the report blocks. Following this approach implies that accounting for Loss RLE Report Blocks will differ from the accounting for the generation of the SR and RR packets described in the RTP specification [7] in the following two areas: per-sender accounting and per-packet accounting.

In its per-sender accounting, an RTP session participant SHOULD NOT make the receipt of a threshold minimum number of RTP packets a condition for reporting upon the sender of those packets. This accounting technique differs from the technique described in Section

6.2.1 and Appendix A.1 of the RTP specification that allows a threshold to determine whether a sender is considered valid.

In its per-packet accounting, an RTP session participant SHOULD treat all sequence numbers as valid. This accounting technique differs from the technique described in Appendix A.1 of the RTP specification that suggests ruling a sequence number valid or invalid on the basis of its contiguity with the sequence numbers of previously received packets.

Sender validity and sequence number validity are interpretations of the raw data. Such interpretations are justified in the interest, for example, of excluding the stray old packet from an unrelated session from having an effect upon the calculation of the RTCP transmission interval. The presence of stray packets might, on the other hand, be of interest to a network monitoring application.

One accounting interpretation that is still necessary is for a participant to decide whether the 16 bit sequence number has rolled over. Under ordinary circumstances this is not a difficult task. For example, if packet number 65,535 (the highest possible sequence number) is followed shortly by packet number 0, it is reasonable to assume that there has been a rollover. However it is possible that the packet is an earlier one (from 65,535 packets earlier). It is also possible that the sequence numbers have rolled over multiple times, either forward or backward. The interpretation becomes more difficult when there are large gaps between the sequence numbers, even accounting for rollover, and when there are long intervals between received packets.

The per-packet accounting technique mandated here is for a participant to keep track of the sequence number of the packet most recently received from a sender. For the next packet that arrives from that sender, the sequence number MUST be judged to fall no more than 32,768 packets ahead or behind the most recent one, whichever choice places it closer. In the event that both choices are equally distant (only possible when the distance is 32,768), the choice MUST be the one that does not require a rollover. Appendix A.1 presents an algorithm that implements this technique.

Each block reports on a single source, identified by its SSRC. The receiver that is supplying the report is identified in the header of the RTCP packet.

Choice of beginning and ending RTP packet sequence numbers for the trace is left to the application. These values are reported in the block. The last sequence number in the trace MAY differ from the sequence number reported on in any accompanying SR or RR report.

Note that because of sequence number wraparound the ending sequence number MAY be less than the beginning sequence number. A Loss RLE Report Block MUST NOT be used to report upon a range of 65,534 or greater in the sequence number space, as there is no means to identify multiple wraparounds.

The trace described by a Loss RLE report consists of a sequence of Boolean values, one for each sequence number of the trace. A value of one represents a packet receipt, meaning that one or more packets having that sequence number have been received since the most recent wraparound of sequence numbers (or since the beginning of the RTP session if no wraparound has been judged to have occurred). A value of zero represents a packet loss, meaning that there has been no packet receipt for that sequence number as of the time of the report. If a packet with a given sequence number is received after a report of a loss for that sequence number, a later Loss RLE report MAY report a packet receipt for that sequence number.

The encoding itself consists of a series of 16 bit units called chunks that describe sequences of packet receipts or losses in the trace. Each chunk either specifies a run length or a bit vector, or is a null chunk. A run length describes between 1 and 16,383 events that are all the same (either all receipts or all losses). A bit vector describes 15 events that may be mixed receipts and losses. A null chunk describes no events, and is used to round out the block to a 32 bit word boundary.

The mapping from a sequence of lost and received packets into a sequence of chunks is not necessarily unique. For example, the following trace covers 45 packets, of which the 22nd and 24th have been lost and the others received:

1111 1111 1111 1111 1111 1010 1111 1111 1111 1111 1111 1

One way to encode this would be:

bit vector 1111 1111 1111 111
bit vector 1111 1101 0111 111
bit vector 1111 1111 1111 111
null chunk

Another way to encode this would be:

```
run of 21 receipts
bit vector 0101 1111 1111 111
run of 9 receipts
null chunk
```

The choice of encoding is left to the application. As part of this freedom of choice, applications MAY terminate a series of run length and bit vector chunks with a bit vector chunk that runs beyond the sequence number space described by the report block. For example, if the 44th packet in the same sequence were lost:

```
1111 1111 1111 1111 1111 1010 1111 1111 1111 1111 1110 1
```

This could be encoded as:

```
run of 21 receipts
bit vector 0101 1111 1111 111
bit vector 1111 1110 1000 000
null chunk
```

In this example, the last five bits of the second bit vector describe a part of the sequence number space that extends beyond the last sequence number in the trace. These bits have been set to zero.

All bits in a bit vector chunk that describe a part of the sequence number space that extends beyond the last sequence number in the trace MUST be set to zero, and MUST be ignored by the receiver.

A null packet MUST appear at the end of a Loss RLE Report Block if the number of run length plus bit vector chunks is odd. The null chunk MUST NOT appear in any other context.

Caution should be used in sending Loss RLE Report Blocks because, even with the compression provided by run length encoding, they can easily consume bandwidth out of proportion with normal RTCP packets. The block type includes a mechanism, called thinning, that allows an application to limit report sizes.

A thinning value, T, selects a subset of packets within the sequence number space: those with sequence numbers that are multiples of 2^T. Packet reception and loss reports apply only to those packets. T can vary between 0 and 15. If T is zero then every packet in the sequence number space is reported upon. If T is fifteen then one in

every 32,768 packets is reported upon.

Suppose that the trace just described begins at sequence number 13,821. The last sequence number in the trace is 13,865. If the trace were to be thinned with a thinning value of T=2, then the following sequence numbers would be reported upon: 13,824, 13,828, 13,832, 13,836, 13,840, 13,844, 13,848, 13,852, 13,856, 13,860, 13,864. The thinned trace would be as follows:

1 1 1 1 1 0 1 1 1 1 0

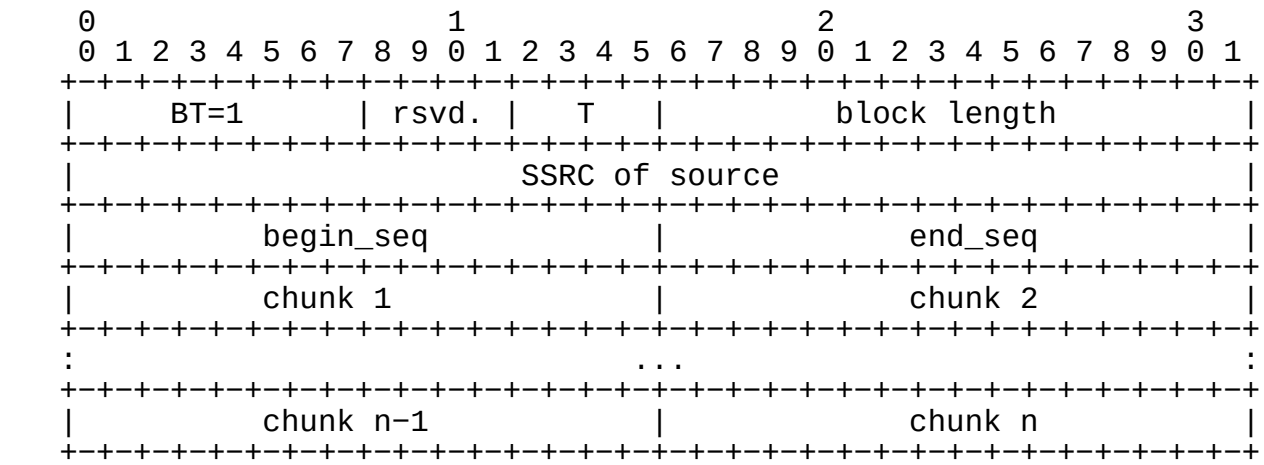
This could be encoded as follows:

bit vector 1111 1011 1100 000
null chunk

The last four bits in the bit vector, representing sequence numbers 13,868, 13,872, 13,876, and 13,880, extend beyond the trace and are thus set to zero and are ignored by the receiver. With thinning, the loss of the 22nd packet goes unreported because its sequence number, 13,842, is not a multiple of four. Packet receipts for all sequence numbers that are not multiples of four also go unreported. However, in this example thinning has permitted the Loss RLE Report Block to be shortened by one 32 bit word.

Choice of the thinning value is left to the application.

The Loss RLE Report Block has the following format:



block type (BT): 8 bits
A Loss RLE Report Block is identified by the constant 1.

rsvd.: 4 bits
This field is reserved for future definition. In the absence of such definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

thinning (T): 4 bits
The amount of thinning performed on the sequence number space. Only those packets with sequence numbers 0 mod 2^T are reported on by this block. A value of 0 indicates that there is no thinning, and all packets are reported on. The maximum thinning is one packet in every 32,768 (amounting to two packets within each 16-bit sequence space).

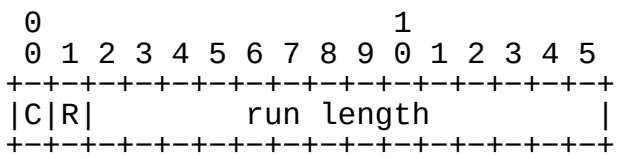
block length: 16 bits
Defined in Section 3.

begin_seq: 16 bits
The first sequence number that this block reports on.

end_seq: 16 bits
The last sequence number that this block reports on plus one.

chunk i: 16 bits
There are three chunk types: run length, bit vector, and terminating null, defined in the following sections. If the chunk is all zeroes then it is a terminating null chunk. Otherwise, the leftmost bit of the chunk determines its type: 0 for run length and 1 for bit vector.

4.1.1 Run Length Chunk

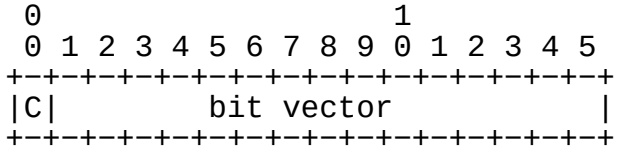


chunk type (C): 1 bit
A zero identifies this as a run length chunk.

run type (R): 1 bit
Zero indicates a run of 0s. One indicates a run of 1s.

run length: 14 bits
A value between 1 and 16,383. The value MUST not be zero for a run length chunk (zeroes in both the run type and run length fields would make the chunk a terminating null chunk). Run lengths of 15 or less MAY be described with a run length chunk despite the fact that they could also be described as part of a bit vector chunk.

4.1.2 Bit Vector Chunk

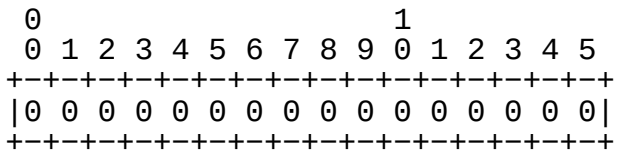


chunk type (C): 1 bit
A one identifies this as a bit vector chunk.

bit vector: 15 bits
The vector is read from left to right, in order of increasing sequence number (with the appropriate allowance for wraparound).

4.1.3 Terminating Null Chunk

This chunk is all zeroes.



4.2 Duplicate RLE Report Block

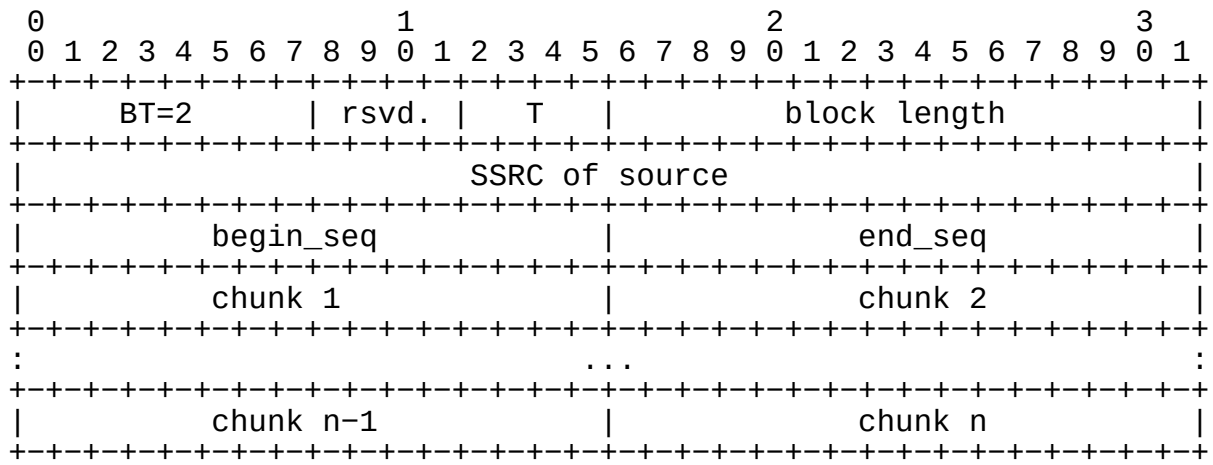
This block type permits per-sequence-number reports on duplicates in a source's RTP packet stream. Such information can be used for network diagnosis, and provide an alternative to packet losses as a basis for multicast tree topology inference.

The Duplicate RLE Report Block format is identical to the Loss RLE Report Block format. Only the interpretation is different, in that the information concerns packet duplicates rather than packet losses. The trace to be encoded in this case also consists of zeros and ones, but a zero here indicates the presence of duplicate packets for a given sequence number, whereas a one indicates that no duplicates were received.

The existence of a duplicate for a given sequence number is determined over the entire reporting period. For example, if packet number 12,593 arrives, followed by other packets with other sequence numbers, the arrival later in the reporting period of another packet numbered 12,593 counts as a duplicate for that sequence number. The duplicate does not need to follow immediately upon the first packet of that number. Care must be taken that a report does not cover a range of 65,534 or greater in the sequence number space.

No distinction is made between the existence of a single duplicate packet and multiple duplicate packets for a given sequence number. Note also that since there is no duplicate for a lost packet, a loss is encoded as a one in a Duplicate RLE Report Block.

The Duplicate RLE Report Block has the following format:



block type (BT): 8 bits
A Duplicate RLE Report Block is identified by the constant 2.

rsvd.: 4 bits
This field is reserved for future definition. In the absence of such definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

thinning (T): 4 bits
As defined in Section 4.1.

block length: 16 bits
Defined in Section 3.

begin_seq: 16 bits
As defined in Section 4.1.

end_seq: 16 bits
As defined in Section 4.1.

chunk i: 16 bits
As defined in Section 4.1.

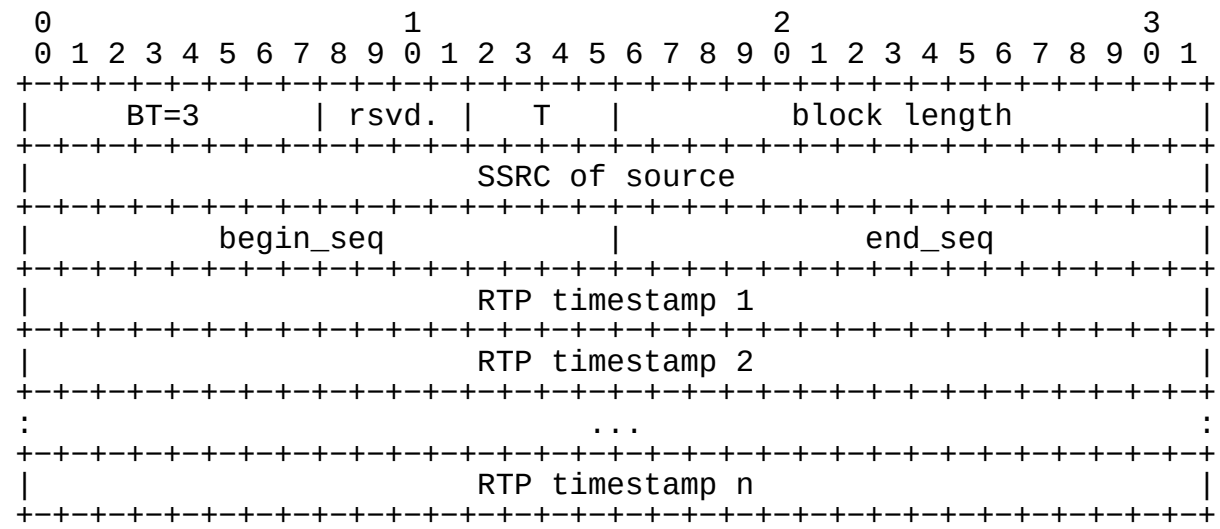
4.3 Timestamp Report Block

This block type permits per-sequence-number reports on packet receipt timestamps for a given source's RTP packet stream. Such information can be used for MINC inference of the topology of the multicast tree used to distribute the source's RTP packets, and of the delays along the links within that tree. It can also be used to measure partial path characteristics and to model distributions for packet jitter.

At least one packet MUST have been received for each sequence number reported upon in this block. If this block type is used to report timestamps for a series of sequence numbers that includes lost packets, several blocks are required. If duplicate packets have been received for a given sequence number, and those packets differ in their receiver timestamps, any timestamp other than the earliest MUST NOT be reported. This is to ensure consistency among reports.

Timestamps consume more bits than loss or duplicate information, and do not lend themselves to run length encoding. The use of thinning is encouraged to limit the size of Timestamp Report Blocks.

The Timestamp Report Block has the following format:



block type (BT): 8 bits
A Timestamp Report Block is identified by the constant 3.

rsvd.: 4 bits
This field is reserved for future definition. In the absence of such definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

thinning (T): 4 bits
As defined in Section 4.1.

block length: 16 bits
Defined in Section 3.

begin_seq: 16 bits
As defined in Section 4.1.

end_seq: 16 bits
As defined in Section 4.1.

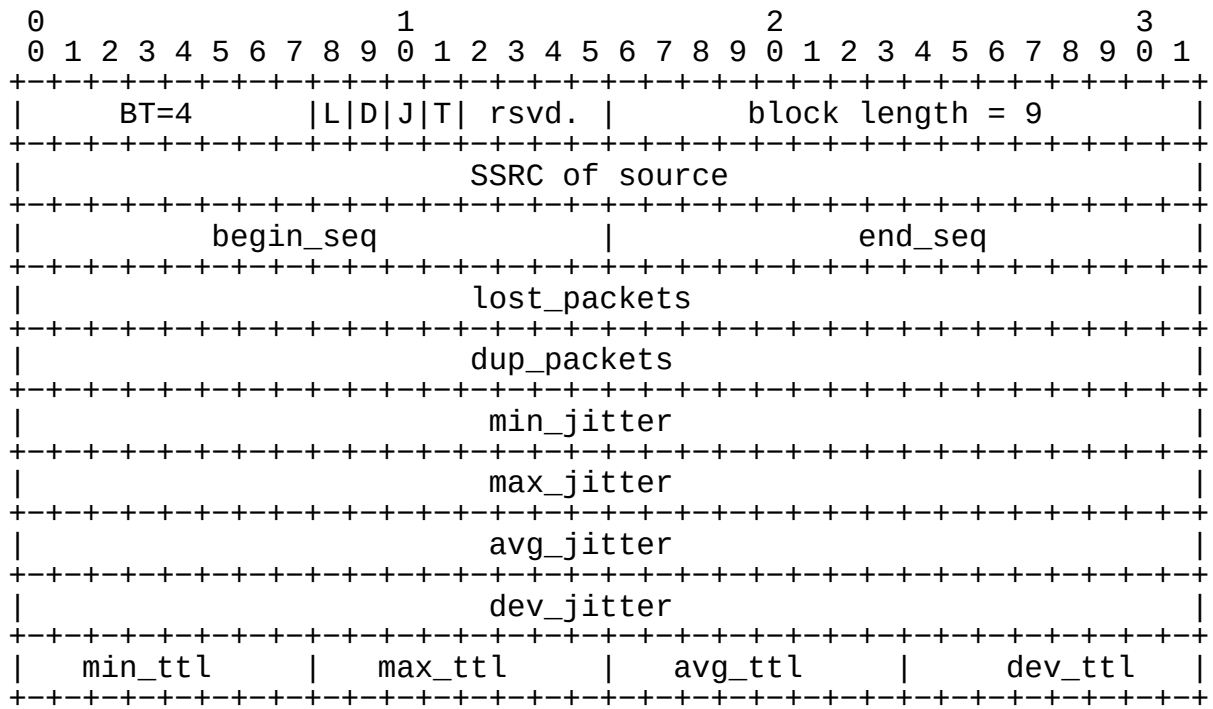
RTP timestamp i: 32 bits
The timestamp reflects the packet arrival time at the receiver. It is preferable for the timestamp to be established at the link layer interface, or in any case as close as possible to the wire arrival time. Units and format are the same as for the timestamp in RTP data packets. As opposed to RTP data packet timestamps, in which nominal values may be used instead of system clock values in order to convey information useful for periodic playout, the receiver timestamps should reflect the actual time as closely as possible. The initial value of the timestamp is random, and subsequent timestamps are offset from this value.

4.4 Statistics Summary Report Block

This block reports statistics beyond the information carried in the standard RTCP packet format, but not as fine grained as that carried in the report blocks previously described. Information is recorded about lost packets, duplicate packets, jitter measurements, and TTL values (TTL values being taken from the TTL field of IPv4 packets, if the data packets are carried over IPv4). Such information can be useful for network management.

The report block contents are dependent upon a bit vector carried in the first part of the header. Not all parameters need to be reported in each block. Flags indicate which are and which are not reported. The fields corresponding to unreported parameters MUST be set to zero. The receiver MUST ignore any Statistics Summary Report Block with a non-zero value in any field flagged as unreported.

The Statistics Summary Report Block has the following format:



block type (BT): 8 bits
A Statistics Summary Report Block is identified by the constant 4.

loss report flag (L): 1 bit
Bit set to 1 if the lost_packets field contains a report, 0 otherwise.

duplicate report flag (D): 1 bit
Bit set to 1 if the dup_packets field contains a report, 0 otherwise.

jitter flag (J): 1 bit
Bit set to 1 if the min_jitter, max_jitter, avg_jitter, and dev_jitter fields all contain reports, 0 if none of them do.

TTL flag (T): 1 bit
Bit set to 1 if the min_ttl, max_ttl, avg_ttl, and dev_ttl fields all contain reports, 0 if none of them do.

rsvd.: 4 bits
This field is reserved for future definition. In the absence of such definition, the bits in this field MUST be set to zero and

MUST be ignored by the receiver.

block length: 16 bits
The constant 9, in accordance with the definition of this field in Section 3.

begin_seq: 16 bits
As defined in Section 4.1.

end_seq: 16 bits
As defined in Section 4.1.

lost_packets: 32 bits
Number of lost packets in the above sequence number interval.

dup_packets: 32 bits
Number of duplicate packets in the above sequence number interval.

min_jitter: 32 bits
The minimum relative transit time between two packets in the above sequence number interval. All jitter values are measured as the difference between a packet's RTP timestamp and the reporter's clock at the time of arrival, measured in the same units.

max_jitter: 32 bits
The maximum relative transit time between two packets in the above sequence number interval.

avg_jitter: 32 bits
The average relative transit time between each two packet series in the above sequence number interval.

dev_jitter: 32 bits
The standard deviation of the relative transit time between each two packet series in the above sequence number interval.

min_ttl: 8 bits
The minimum TTL value of data packets in sequence number range.

max_ttl: 8 bits
The maximum TTL value of data packets in sequence number range.

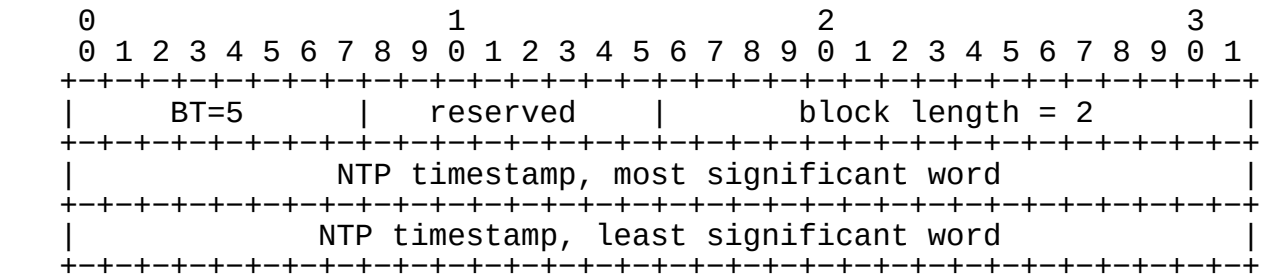
avg_ttl: 8 bits
The average TTL value of data packets in sequence number range.

dev_ttl: 8 bits

The standard deviation of TTL values of data packets in sequence number range.

4.5 Receiver Timestamp Report Block

This block extends RTCP's timestamp reporting so that non-senders may also send timestamps. It recapitulates the NTP timestamp fields from the RTCP Sender Report [7, Sec. 6.3.1]. A non-sender may estimate its RTT to other participants, as proposed in [11], by sending this report block and receiving DLRR Report Blocks (see next section) in reply.



block type (BT): 8 bits
A Receiver Timestamp Report Block is identified by the constant 5.

reserved: 8 bits
This field is reserved for future definition. In the absence of such definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

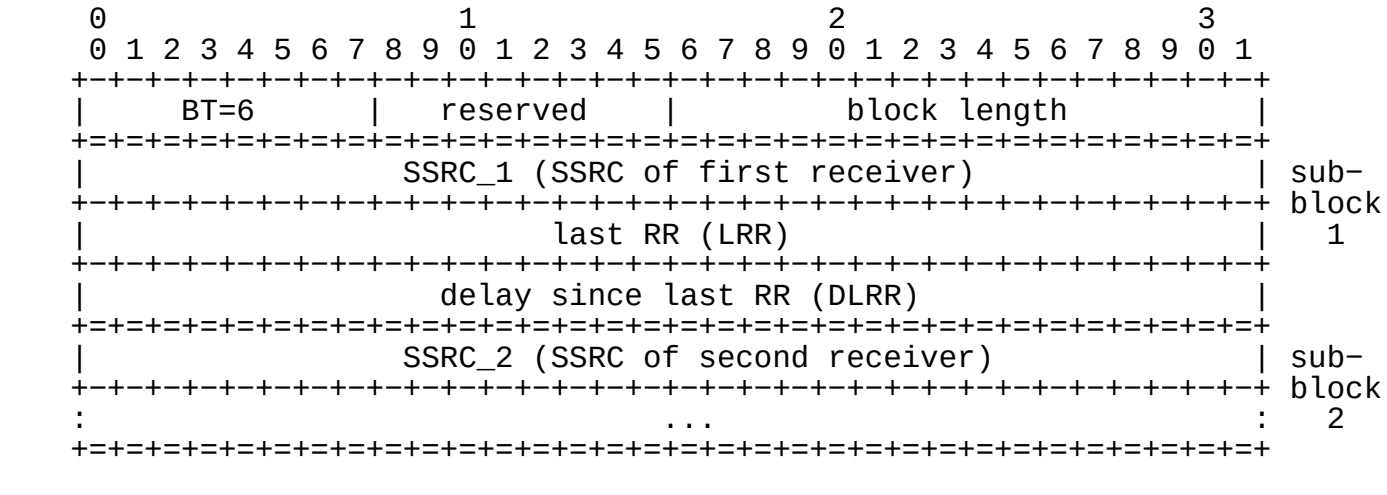
block length: 16 bits
The constant 2, in accordance with the definition of this field in Section 3.

NTP timestamp: 64 bits
Indicates the wallclock time when this block was sent so that it may be used in combination with timestamps returned in DLRR Report Blocks (see next section) 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 report block 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 report sender that has no notion of wallclock or elapsed time may set the NTP timestamp to zero.

4.6 DLRR Report Block

This block extends RTCP's delay since last sender report (DLSR) mechanism [7, Sec. 6.3.1] so that non-senders may also calculate round trip times, as proposed in [11]. It is termed DLRR for delay since last receiver report, and may be sent in response to a Receiver Timestamp Report Block (see previous section) from a receiver to allow that receiver to calculate its round trip time to the respondent. The report consists of one or more 3 word sub-blocks: one sub-block per receiver report.



block type (BT): 8 bits
A DLRR Report Block is identified by the constant 6.

reserved: 8 bits
This field is reserved for future definition. In the absence of such definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

block length: 16 bits
Defined in Section 3.

last RR timestamp (LRR): 32 bits
The middle 32 bits out of 64 in the NTP timestamp (as explained in the previous section) received as part of a Receiver

Timestamp Report Block from participant SSRC_n. If no such block has been received, the field is set to zero.

delay since last RR (DLRR): 32 bits
The delay, expressed in units of 1/65536 seconds, between receiving the last Receiver Timestamp Report Block from participant SSRC_n and sending this DLRR Report Block. If no Receiver Timestamp Report Block has been received yet from SSRC_n, the DLRR field is set to zero (or the DLRR is omitted entirely). Let SSRC_r denote the receiver issuing this DLRR Report Block. Participant SSRC_n can compute the round-trip propagation delay to SSRC_r by recording the time A when this Receiver Timestamp 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 [7, Fig. 2].

4.7 VoIP Metrics Report Block

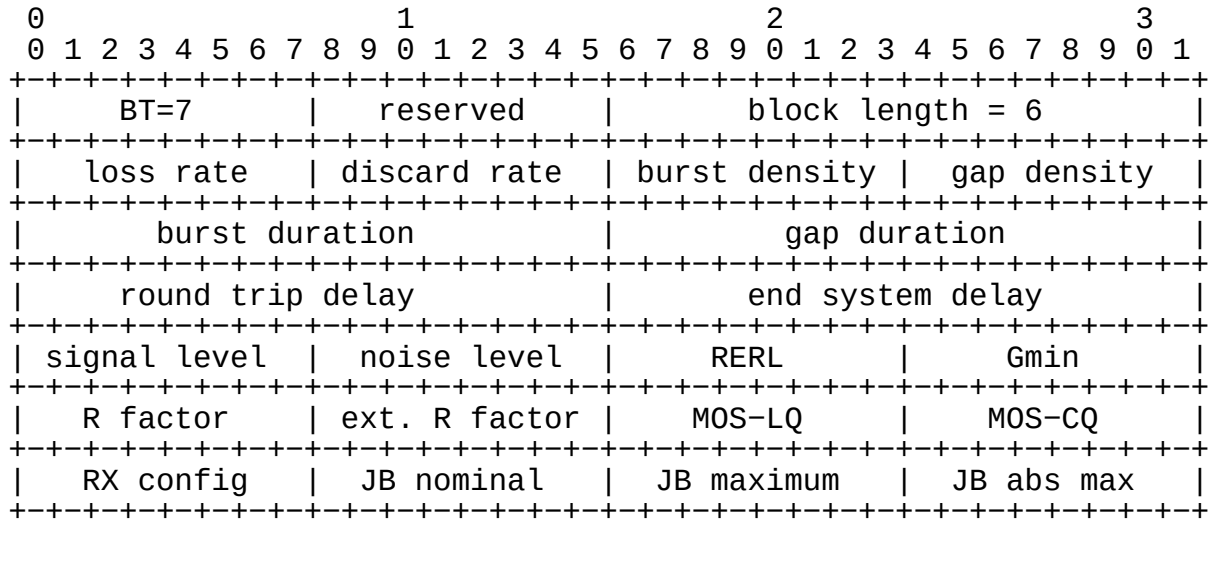
The VoIP Metrics Report Block provides metrics for monitoring voice over IP (VoIP) calls. These metrics include packet loss and discard metrics, delay metrics, analog metrics, and voice quality metrics. The block reports separately on packets lost on the IP channel, and those that have been received but then discarded by the receiving jitter buffer. It also reports on the combined effect of losses and discards, as both have equal effect on call quality.

In order to properly assess the quality of a Voice over IP call it is desirable to consider the degree of burstiness of packet loss [10]. Following a Gilbert-Elliott model [2], a period of time, bounded by lost and/or discarded packets, with a high rate of losses and/or discards is a "burst," and a period of time between two bursts is a "gap." Bursts correspond to periods of time during which the packet loss rate is high enough to produce noticeable degradation in audio quality. Gaps correspond to periods of time during which only isolated lost packets may occur, and in general these can be masked by packet loss concealment. Delay reports include the transit delay between RTCP end points and the VoIP end system processing delays, both of which contribute to the user perceived delay. Additional metrics include signal, echo, noise, and distortion levels. Call quality metrics include R factors (as described by the E Model defined in [2]) and mean opinion scores (MOS scores).

An implementation that sends these blocks SHOULD send at least one every ten seconds for the duration of the call, SHOULD send one whenever a codec type change or other significant change occurs, SHOULD send one when significant call quality degradation is detected

and SHOULD send one upon call termination. Implementations MUST provide values for all the fields defined here. For certain metrics, if the value is undefined or unknown, then the specified default or unknown field value MUST be provided.

The block is encoded as seven 32-bit words:



block type (BT): 8 bits
A VoIP Metrics Report Block is identified by the constant 7.

reserved: 8 bits
This field is reserved for future definition. In the absence of such definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

block length: 16 bits
The constant 6, in accordance with the definition of this field in Section 3.

The remaining fields are described in the following six sections: Packet Loss and Discard Metrics, Delay Metrics, Signal Related Metrics, Call Quality or Transmission Quality Metrics, Configuration Metrics, and Jitter Buffer Parameters.

4.7.1 Packet Loss and Discard Metrics

It is very useful to distinguish between packets lost by the network and those discarded due to jitter. Both have equal effect on the

quality of the voice stream however having separate counts helps identify the source of quality degradation. These fields MUST be populated.

loss rate: 8 bits
The fraction of RTP data packets from the source lost since the beginning of reception, expressed as a fixed point number with the binary point at the left edge of the field. This value is calculated by dividing the total number of packets lost (after the effects of applying any error protection such as FEC) by the total number of packets expected, multiplying the result of the division by 256, limiting the maximum value to 255 (to avoid overflow), and taking the integer part. The numbers of duplicated packets and discarded packets do not enter into this calculation. Since receivers cannot be required to maintain unlimited buffers, a receiver MAY categorize late-arriving packets as lost. The degree of lateness that triggers a loss SHOULD be significantly greater than that which triggers a discard.

discard rate: 8 bits
The fraction of RTP data packets from the source that have been discarded since the beginning of reception, due to late or early arrival, under-run or overflow at the receiving jitter buffer. This value is expressed as a fixed point number with the binary point at the left edge of the field. It is calculated by dividing the total number of packets discarded (excluding duplicate packet discards) by the total number of packets expected, multiplying the result of the division by 256, limiting the maximum value to 255 (to avoid overflow), and taking the integer part.

4.7.2 Burst Metrics

A burst, informally, is a period of high packet losses and/or discards. Formally, a burst is defined as a longest sequence of packets bounded by lost or discarded packets with the constraint that within a burst any sequence of successive packets that were received, and not discarded due to delay variation, is of length less than a value Gmin.

A gap, informally, is a period of low packet losses and/or discards. Formally, a gap is defined as any of the following: (a) the period from the start of an RTP session to the receipt time of the last received packet before the first burst, (b) the period from the end of the last burst to either the time of the report or the end of the RTP session, whichever comes first, or (c) the period of time between two bursts.

For the purpose of determining if a lost or discarded packet near the start or end of an RTP session is within a gap or a burst it is assumed that the RTP session is preceded and followed by at least Gmin received packets, and that the time of the report is followed by at least Gmin received packets.

A gap has the property that any lost or discarded packets within the gap must be preceded and followed by at least Gmin packets that were received and not discarded. This gives a maximum loss/discard density within a gap of: 1 / (Gmin + 1).

A Gmin value of 16 is RECOMMENDED as it results in gap characteristics that correspond to good quality (i.e. low packet loss rate, a minimum distance of 16 received packets between lost packets) and hence differentiates nicely between good and poor quality periods.

For example, a 1 denotes a received, 0 a lost, and X a discarded packet in the following pattern covering 64 packets:

```
1111011111111111111111X111X1011110111111111111111X111111111
|---------gap----------|--burst---|------------gap------------|
```

The burst consists of the twelve packets indicated above, starting at a discarded packet and ending at a lost packet. The first gap starts at the beginning of the session and the second gap ends at the time of the report.

If the packet spacing is 10 ms and the Gmin value is the recommended value of 16, the burst duration is 120 ms, the burst density 0.33, the gap duration 230 ms + 290 ms = 520 ms, and the gap density 0.04.

This would result in reported values as follows (see field descriptions for semantics and details on how these are calculated):

loss density	12, which corresponds to 5%
discard density	12, which corresponds to 5%
burst density	84, which corresponds to 33%
gap density	10, which corresponds to 4%
burst duration	120, value in milliseconds
gap duration	520, value in milliseconds

burst density: 8 bits

The fraction of RTP data packets within burst periods since the beginning of reception that were either lost or discarded. This value is expressed as a fixed point number with the binary point

at the left edge of the field. It is calculated by dividing the total number of packets lost or discarded (excluding duplicate packet discards) within burst periods by the total number of packets expected within the burst periods, multiplying the result of the division by 256, limiting the maximum value to 255 (to avoid overflow), and taking the integer part.

gap density: 8 bits
The fraction of RTP data packets within inter-burst gaps since the beginning of reception that were either lost or discarded. The value is expressed as a fixed point number with the binary point at the left edge of the field. It is calculated by dividing the total number of packets lost or discarded (excluding duplicate packet discards) within gap periods by the total number of packets expected within the gap periods, multiplying the result of the division by 256, limiting the maximum value to 255 (to avoid overflow), and taking the integer part.

burst duration: 16 bits
The mean duration, expressed in milliseconds, of the burst periods that have occurred since the beginning of reception. The duration of each period is calculated based upon the packets that mark the beginning and end of that period. It is equal to the timestamp of the end packet, plus the duration of the end packet, minus the timestamp of the beginning packet. If the actual values are not available, estimated values MUST be used. If there have been no burst periods, the burst duration value MUST be zero.

gap duration: 16 bits
The mean duration, expressed in milliseconds, of the gap periods that have occurred since the beginning of reception. The duration of each period is calculated based upon the packet that marks the end of the prior burst and the packet that marks the beginning of the subsequent burst. It is equal to the timestamp of the subsequent burst packet, minus the timestamp of the prior burst packet, plus the duration of the prior burst packet. If the actual values are not available, estimated values MUST be used. In the case of a gap that occurs at the beginning of reception, the sum of the timestamp of the prior burst packet and the duration of the prior burst packet are replaced by the reception start time. In the case of a gap that occurs at the end of reception, the timestamp of the subsequent burst packet is replaced by the reception end time. If there have been no gap periods, the gap duration value MUST be zero.

4.7.3 Delay Metrics

For the purpose of the following definitions, the RTP interface is the interface between the RTP instance and the voice application (i.e. FEC, de-interleaving, de-multiplexing, jitter buffer). For example, the time delay due to RTP payload multiplexing would be considered to be part of the voice application or end-system delay whereas delay due to multiplexing RTP frames within a UDP frame would be considered part of the RTP reported delay. This distinction is consistent with the use of RTCP for delay measurements.

round trip delay: 16 bits
The most recently calculated round trip time between RTP interfaces, expressed in milliseconds. This value is the time of receipt of the most recent RTCP packet from source SSRC, minus the LSR (last SR) time reported in its SR (sender report), minus the DLSR (delay since last SR) reported in its SR. A non-zero LSR value is REQUIRED in order to calculate round trip delay. A value of 0 is permissible during the first two or three RTCP exchanges as insufficient data may be available to determine delay however MUST be populated as soon as a delay estimate is available.

end system delay: 16 bits
The most recently estimated end system delay, expressed in milliseconds. End system delay is defined as the total encoding, decoding and jitter buffer delay determined at the reporting endpoint. This is the time required for an RTP frame to be buffered, decoded, converted to analog form, looped back at the local analog interface, encoded, and made available for transmission as an RTP frame. The manner in which this value is estimated is implementation dependent. This parameter MUST be provided in all VoIP metrics reports.

Note that the one way symmetric VoIP segment delay may be calculated from the round trip and end system delays as follows. If the round trip delay is denoted RTD and the end system delays associated with the two endpoints are ESD(A) and ESD(B) then:

one way symmetric voice path delay = (RTD + ESD(A) + ESD(B)) / 2

4.7.4 Signal Related Metrics

The following metrics are intended to provide real time information related to the non-packet elements of the voice over IP system to assist with the identification of problems affecting call quality. The values identified below must be determined for the received audio signal. The information required to populate these fields may not be

available in all systems, although it is strongly recommended that this data SHOULD be provided to support problem diagnosis.

signal level: 8 bits
The voice signal relative level is defined as the ratio of the signal level to a reference digital milliwatt, expressed in decibels as a signed integer in two's complement form. This is measured only for packets containing speech energy. The intent of this metric is not to provide a precise measurement of the signal level but to provide a real time indication that the signal level may be excessively high or low.

signal level = 10 log 10 (rms talkspurt power (mW))

A value of 127 indicates that this parameter is unavailable. Typical values should be generally in the -15 to -20 dBm range.

noise level: 8 bits
The noise level is defined as the ratio of the silent period background noise level to a reference digital milliwatt, expressed in decibels as a signed integer in two's complement form.

noise level = 10 log10 (rms silence power (mW))

A value of 127 indicates that this parameter is unavailable.

residual echo return loss (RERL): 8 bits
The residual echo return loss is defined as the sum of the measured echo return loss (ERL) and the echo return loss enhancement (ERLE) expressed in dB as a signed integer in two's complement form. It defines the ratio of a transmitted voice signal that is reflected back to the talker. A low level of echo return loss (say less than 20 dB) in conjunction with some delay can lead to hollowness or audible echo. A high level of echo return loss (say over 40 dB) is preferable.

The ERL and ERLE parameters are often available directly from the echo canceler contained within the VoIP end system. They relate to the echo on the external network attached to the end point.

In the case of a VoIP gateway this would be line echo that typically occurs at 2-4 wire conversion points in the network. Echo return loss from typical 2-4 wire conversions can be in the 8-12 dB range. A line echo canceler can provide an ERLE of 30 dB or more and hence reduce this to 40-50 dB. In the case of an IP phone this could be residual acoustic echo from speakerphone operation, and may more correctly be termed terminal coupling loss (TCL). A typical handset

would result in 40-50 dB of echo due to acoustic feedback.

Typical values for RERL are as follows:

(i) IP gateway connected to circuit switched network with 2 wire loop
Without echo cancellation, typical 2-4 wire converter ERL of 12 dB
RERL = ERL + ERLE = 12 + 0 = 12 dB

(ii) IP gateway connected to circuit switched network with 2 wire loop
With echo canceler that improves echo by 30 dB
RERL = ERL + ERLE = 12 + 30 = 42 dB

(iii) IP phone with conventional handset
Acoustic coupling from handset speaker to microphone 40 dB
Residual echo return loss = TCL = 40 dB

If we denote the "local" end of the VoIP path as A and the remote end as B and if the sender loudness rating (SLR) and receiver loudness rating (RLR) are known for A (default values 8 dB and 2 dB respectively), then the echo loudness level at end A (talker echo loudness rating or TELR) is given by:

TELR(A) = SRL(A) + ERL(B) + ERLE(B) + RLR(A)

TELR(B) = SRL(B) + ERL(A) + ERLE(A) + RLR(B)

Hence in order to incorporate echo into a voice quality estimate at the A end of a VoIP connection it is desirable to send the ERL + ERLE value from B to A.

For an IP phone with handset this metric MUST be set to the designed or measured terminal coupling loss, which would typically be 40-50 dB.

For a PC softphone or speakerphone this metric MUST be set to either the value reported by the acoustic echo canceler or to 127 to indicate an undefined value.

For an IP gateway with ERL and ERLE measurements this metric MUST be reported as ERL + ERLE.

For an IP gateway without ERL and ERLE measurement capability then this metric MUST be reported as 12 dB if line echo cancellation is disabled and 40 dB if line echo cancellation is enabled.

Gmin
See Configuration Parameters (Section 4.7.6, below).

4.7.5 Call Quality or Transmission Quality Metrics

The following metrics are direct measures of the call quality or transmission quality, and incorporate the effects of codec type, packet loss, discard, burstiness, delay etc. These metrics may not be available in all systems however SHOULD be provided in order to support problem diagnosis.

R factor: 8 bits
: The R factor is a voice quality metric describing the segment of the call that is carried over this RTP session. It is expressed as an integer in the range 0 to 100, with a value of 94 corresponding to "toll quality" and values of 50 or less regarded as unusable. This metric is defined as including the effects of delay, consistent with ITU-T G.107 [4] and ETSI TS 101 329-5 [2].

  A value of 127 indicates that this parameter is unavailable.

ext. R factor: 8 bits
: The external R factor is a voice quality metric describing the segment of the call that is carried over a network segment external to the RTP segment, for example a cellular network. Its values are interpreted in the same manner as for the RTP R factor. This metric is defined as including the effects of delay, consistent with ITU-T G.107 [4] and ETSI TS 101 329-5 [2], and relates to the outward voice path from the Voice over IP termination for which this metrics block applies.

  A value of 127 indicates that this parameter is unavailable.

Note that an overall R factor may be estimated from the RTP segment R factor and the external R factor, as follows:

R total = RTP R factor + ext. R factor − 94

MOS-LQ: 8 bits
: The estimated mean opinion score for listening quality (MOS-LQ) is a voice quality metric on a scale from 1 to 5, in which 5 represents excellent and 1 represents unacceptable. This metric is defined as not including the effects of delay and can be compared to MOS scores obtained from listening quality (ACR) tests. It is expressed as an integer in the range 10 to 50, corresponding to MOS x 10. For example, a value of 35 would correspond to an estimated MOS score of 3.5.

  A value of 127 indicates that this parameter is unavailable.

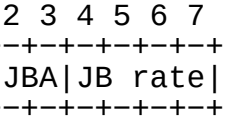
MOS-CQ: 8 bits
The estimated mean opinion score for conversational quality (MOS-CQ) is defined as including the effects of delay and other effects that would affect conversational quality. The metric may be calculated by converting an R factor determined according to ITU-T G.107 [4] or ETSI TS 101 329-5 [2] into an estimated MOS using the equation specified in G.107. It is expressed as an integer in the range 10 to 50, corresponding to MOS x 10, as for MOS-LQ.

A value of 127 indicates that this parameter is unavailable.

4.7.6 Configuration Parameters

Gmin: 8 bits
The gap threshold. This field contains the value used for this report block to determine if a gap exists. The recommended value of 16 corresponds to a burst period having a minimum density of 6.25% of lost or discarded packets, which may cause noticeable degradation in call quality; during gap periods, if packet loss or discard occurs, each lost or discarded packet would be preceded by and followed by a sequence of at least 16 received non-discarded packets. Note that lost or discarded packets that occur within Gmin packets of a report being generated may be reclassified as being part of a burst or gap in later reports. ETSI TS 101 329-5 [2] defines a computationally efficient algorithm for measuring burst and gap density using a packet loss/discard event driven approach. This algorithm is reproduced in Appendix A.2 of the present document. Gmin MUST not be zero and MUST be provided.

receiver configuration byte (RX config): 8 bits
This byte consists of the following fields:



packet loss concealment (PLC): 2 bits
Standard (11) / enhanced (10) / disabled (01) / unspecified (00). When PLC = 11 then a simple replay or interpolation algorithm is being used to fill-in the missing packet. This is typically able to conceal isolated lost packets with loss rates under 3%. When PLC = 10 then an enhanced interpolation algorithm is being used. This would

typically be able to conceal lost packets for loss rates of 10% or more. When PLC = 01 then silence is inserted in place of lost packets. When PLC = 00 then no information is available concerning the use of PLC however for some codecs this may be inferred.

jitter buffer adaptive (JBA): 2 bits
Adaptive (11) / non-adaptive (10) / reserved (01)/ unknown (00). When the jitter buffer is adaptive then its size is being dynamically adjusted to deal with varying levels of jitter. When non-adaptive, the jitter buffer size is maintained at a fixed level. When either adaptive or non-adaptive modes are specified then the jitter buffer size parameters below MUST be specified.

jitter buffer rate (JB rate): 4 bits
J = adjustment rate (0-15). This represents the implementation specific adjustment rate of a jitter buffer in adaptive mode. This parameter is defined in terms of the approximate time taken to fully adjust to a step change in peak to peak jitter from 30 ms to 100 ms such that:

adjustment time = 2 * J * frame size (ms)

This parameter is intended only to provide a guide to the degree of "aggressiveness" of a an adaptive jitter buffer and may be estimated. A value of 0 indicates that the adjustment time is unknown for this implementation.

4.7.7 Jitter Buffer Parameters

jitter buffer nominal size in frames (JB nominal): 8 bits
This is the current nominal fill point within the jitter buffer, which corresponds to the nominal jitter buffer delay for packets that arrive exactly on time. This parameter MUST be provided for both fixed and adaptive jitter buffer implementations.

jitter buffer maximum size in frames (JB maximum): 8 bits
This is the current maximum jitter buffer level corresponding to the earliest arriving packet that would not be discarded. In simple queue implementations this may correspond to the nominal size. In adaptive jitter buffer implementations this value may dynamically vary up to JB abs max (see below). This parameter MUST be provided for both fixed and adaptive jitter buffer implementations.

jitter buffer absolute maximum size in frames (JB abs max): 8 bits
This is the absolute maximum size that the adaptive jitter

buffer can reach under worst case jitter conditions. This parameter MUST be provided for adaptive jitter buffer implementations and its value MUST be set to JB maximum for fixed jitter buffer implementations.

5. IANA Considerations

This document defines a new RTP packet type, the extended report (XR) type, within the existing Internet Assigned Numbers Authority (IANA) registry of RTP RTCP Control Packet Types. This document also defines a new IANA registry: the registry of RTP XR Block Types. Within this new registry, this document defines an initial set of seven block types and describes how the remaining types are to be allocated.

5.1 XR Packet Type

The IANA SHALL register the RTP extended report (XR) packet defined by this document as packet type 207 in the registry of RTP RTCP Control Packet types (PT).

5.2 RTP XR Block Type Registry

The IANA SHALL create the RTP XR Block Type Registry to cover the name space of the extended report block type (BT) field specified in Section 3 of this document. The BT field contains eight bits, allowing 256 values. The IANA SHALL manage the RTP XR Block Type Registry according to the Specification Required policy of RFC 2434 [6]. Future specifications SHOULD attribute block type values in strict numeric order following the values attributed in this document:

BT	name
1	Loss RLE Report Block
2	Duplicate RLE Report Block
3	Timestamp Report Block
4	Statistics Summary Report Block
5	Receiver Timestamp Report Block
6	DLRR Report Block
7	VoIP Metrics Report Block

Furthermore, future specifications SHOULD avoid the values 0 and 255. Doing so facilitates packet validity checking, since all-zeros and all-ones are values that might commonly be found in ill-formed packets.

6. Security Considerations

This document extends the RTCP reporting mechanism. The security considerations that apply to RTCP reports also apply to XR reports. This section details the additional security considerations that apply to the extensions.

The extensions introduce heightened confidentiality concerns. Standard RTCP reports contain a limited number of summary statistics. The information contained in XR reports is both more detailed and more extensive (covering a larger number of parameters). The per-packet information contained in Loss RLE, Duplicate RLE, and Timestamp Report Blocks facilitates MINC inference of multicast distribution trees for RTP data packets, and inference of link characteristics such as loss and delay. This inference reveals information that might otherwise be considered confidential to autonomous system administrators. The VoIP Metrics Report Block provides information on the quality of ongoing voice calls. Though such information might be carried in application specific format in standard RTP sessions, making it available in a standard format here makes it more available to potential eavesdroppers.

No new mechanisms are introduced in this document to ensure confidentiality. Standard encryption procedures can be used when confidentiality is a concern to end hosts. Autonomous system administrators concerned about the loss of confidentiality regarding their networks can encrypt traffic, or filter it to exclude RTCP packets containing the XR report blocks concerned.

Any encryption or filtering of XR report blocks entails a loss of monitoring information to third parties. For example, a network that establishes a tunnel to encrypt VoIP Report Blocks denies that information to the service providers traversed by the tunnel. The service providers cannot then monitor or respond to the quality of the VoIP calls that they carry, potentially creating problems for the network's users. As a default, XR packets SHOULD NOT be encrypted or filtered.

The extensions also make certain denial of service attacks easier. This is because of the potential to create RTCP packets much larger than average with the per packet reporting capabilities of the Loss RLE, Duplicate RLE, and Timestamp Report Blocks. Because of the automatic bandwidth adjustment mechanisms in RTCP, if some session participants are sending large RTCP packets, all participants will see their RTCP reporting intervals lengthened, meaning they will be able to report less frequently. To limit the effects of large packets, even in the absence of denial of service attacks, applications SHOULD limit the size of XR report blocks and employ the

thinning techniques described in this document in order to fit reports into the space available.

A. Algorithms

A.1 Sequence Number Interpretation

This the algorithm suggested by Section 4.1 for keeping track of the sequence numbers from a given sender. It implements the accounting practice required for the generation of Loss RLE Report Blocks.

This algorithm keeps track of 16 bit sequence numbers by translating them into a 32 bit sequence number space. The first packet received from a source is considered to have arrived roughly in the middle of that space. Each packet that follows is placed either ahead or behind the prior one in this 32 bit space, depending upon which choice would place it closer (or, in the event of a tie, which choice would not require a rollover in the 16 bit sequence number).

```
// The reference sequence number is an extended sequence number
// that serves as the basis for determining whether a new 16 bit
// sequence number comes earlier or later in the 32 bit sequence
// space.
u_int32 _src_ref_seq;
bool    _uninitialized_src_ref_seq;

// Place seq into a 32-bit sequence number space based upon a
// heuristic for its most likely location.
u_int32 extend_seq(const u_int16 seq) {

        u_int32 extended_seq, seq_a, seq_b, diff_a, diff_b;
        if(_uninitialized_src_ref_seq) {

                // This is the first sequence number received.  Place
                // it in the middle of the extended sequence number
                // space.
                _src_ref_seq                = seq | 0x80000000u;
                _uninitialized_src_ref_seq  = false;
                extended_seq                = _src_ref_seq;
        }
        else {

                // Prior sequence numbers have been received.
                // Propose two candidates for the extended sequence
                // number: seq_a is without wraparound, seq_b with
                // wraparound.
                seq_a = seq | (_src_ref_seq & 0xFFFF0000u);
                if(_src_ref_seq < seq_a) {
```

```
                        seq_b  = seq_a - 0x00010000u;
                        diff_a = seq_a - _src_ref_seq;
                        diff_b = _src_ref_seq - seq_b;
                }
                else {
                        seq_b  = seq_a + 0x00010000u;
                        diff_a = _src_ref_seq - seq_a;
                        diff_b = seq_b - _src_ref_seq;
                }

                // Choose the closer candidate.  If they are equally
                // close, the choice is somewhat arbitrary: we choose
                // the candidate for which no rollover is necessary.
                if(diff_a < diff_b) {
                        extended_seq = seq_a;
                }
                else {
                        extended_seq = seq_b;
                }

                // Set the reference sequence number to be this most
                // recently-received sequence number.
                _src_ref_seq = extended_seq;
        }

        // Return our best guess for a 32-bit sequence number that
        // corresponds to the 16-bit number we were given.
        return extended_seq;
}
```

A.2 Example Burst Packet Loss Calculation.

This is an algorithm for measuring the burst characteristics for the VoIP Metrics Report Block (Section 4.7). It is reproduced from ETSI TS 101 329-5 [2]. The algorithm is event driven and hence extremely computationally efficient.

Given the following definition of states:

state 1 = received a packet during a gap
state 2 = received a packet during a burst
state 3 = lost a packet during a burst
state 4 = lost an isolated packet during a gap

The "c" variables below correspond to state transition counts, i.e. c14 is the transition from state 1 to state 4. It is possible to

infer one of a pair of state transition counts to an accuracy of 1 which is generally sufficient for this application.

"pkt" is the count of packets received since the last packet was declared lost or discarded and "lost" is the number of packets lost within the current burst. "packet_lost" and "packet_discarded" are Boolean variables that indicate if the event that resulted in this function being invoked was a lost or discarded packet.

```
if(packet_lost) {
        loss_count++;
}
if(packet_discarded) {
        discard_count++;
}
if(!packet_lost && !packet_discarded) {
        pkt++;
}
else {
        if(pkt >= gmin) {
                if(lost == 1) {
                        c14++;
                }
                else {
                        c13++;
                }
                lost = 1;
                c11 += pkt;
        }
        else {
                lost++;
                if(pkt == 0) {
                        c33++;
                }
                else {
                        c23++;
                        c22 += (pkt - 1);
                }
        }
        pkt = 0;
}
```

At each reporting interval the burst and gap metrics can be calculated as follows.

```
// Calculate additional transition counts.
c31 = c13;
c32 = c23;
```

```
   ctotal = c11 + c14 + c13 + c22 + c23 + c31 + c32 + c33;

   // Calculate burst and densities.
   p32 = c32 / (c31 + c32 + c33);
   if((c22 + c23) < 1) {
           p23 = 1;
   }
   else {
           p23 = 1 - c22/(c22 + c23);
   }
   burst_density = 256 * p23 / (p23 + p32);
   gap_density = 256 * c14 / (c11 + c14);

   // Calculate burst and gap durations in ms
   m = frameDuration_in_ms * framesPerRTPPkt;
   gap_length = (c11 + c14 + c13) * m / c13;
   burst_length = ctotal * m / c13 - lgap;

   /* calculate loss and discard densities */
   loss_density = 256 * loss_count / ctotal;
   discard_density = 256 * discard_count / ctotal;
```

Intellectual Property

Full Copyright Statement

Acknowledgments

We thank the following people: Colin Perkins, Steve Casner, and Henning Schulzrinne for their considered guidance; Sue Moon for helping foster collaboration between the authors; Magnus Westerlund for his detailed comments; Mounir Benzaid for drawing our attention to the reporting needs of MLDA; and Dorgham Sisalem and Adam Wolisz for encouraging us to incorporate MLDA block types.

Contributors

The following people are the authors of this document:

Kevin Almeroth, UCSB
Ramon Caceres, ShieldIP
Alan Clark, Telchemy
Robert Cole, AT&T Labs
Nick Duffield, AT&T Labs-Research
Timur Friedman, Paris 6
Kaynam Hedayat, Brix Networks
Kamil Sarac, UT Dallas

The principal people to contact regarding the individual report blocks described in this document are as follows:

sec.	report block	principal contributors
4.1	Loss RLE Report Block	Friedman, Caceres, Duffield
4.2	Duplicate RLE Report Block	Friedman, Caceres, Duffield
4.3	Timestamp Report Block	Friedman, Caceres, Duffield
4.4	Statistics Summary Report Block	Almeroth, Sarac
4.5	Receiver Timestamp Report Block	Friedman
4.6	DLRR Report Block	Friedman
4.7	VoIP Metrics Report Block	Clark, Cole, Hedayat

Authors' Addresses

Kevin Almeroth <almeroth@cs.ucsb.edu>
Department of Computer Science
University of California
Santa Barbara, CA 93106 USA

Ramon Caceres <ramon@shieldip.com>
ShieldIP, Inc.
11 West 42nd Street, 31st Floor
New York, NY 10036 USA

Alan Clark <alan@telchemy.com>
Telchemy Incorporated
3360 Martins Farm Road, Suite 200
Suwanee, GA 30024 USA
Tel: +1 770 614 6944
Fax: +1 770 614 3951

Robert Cole <rgcole@att.com>
AT&T Labs
330 Saint Johns Street,
2nd Floor
Havre de Grace, MD 21078 USA
Tel: +1 410 939 8732
Fax: +1 410 939 8732

Nick Duffield <duffield@research.att.com>
AT&T Labs-Research
180 Park Avenue, P.O. Box 971
Florham Park, NJ 07932-0971 USA
Tel: +1 973 360 8726
Fax: +1 973 360 8050

Timur Friedman <timur.friedman@lip6.fr>
Universite Pierre et Marie Curie (Paris 6)
Laboratoire LiP6-CNRS
8, rue du Capitaine Scott
75015 PARIS France
Tel: +33 1 44 27 71 06
Fax: +33 1 44 27 74 95

Kaynam Hedayat <khedayat@brixnet.com>
Brix Networks
285 Mill Road
Chelmsford, MA 01824 USA
Tel: +1 978 367 5600
Fax: +1 978 367 5700

Kamil Sarac <ksarac@utdallas.edu>
Department of Computer Science (ES 4.207)
Eric Jonsson School of Engineering & Computer Science
University of Texas at Dallas
Richardson, TX 75083-0688 USA
Tel: +1 972 883 2337
Fax: +1 972 883 2349

References

The references are divided into normative references and non-normative references.

Normative References

[1] S. Bradner, "Key words for use in RFCs to indicate requirement levels," BCP 14, RFC 2119, IETF, March 1997.

[2] ETSI, "Quality of Service (QoS) measurement methodologies," ETSI TS 101 329-5 V1.1.1 (2000-11), November 2000.

[3] R. Hovey and S. Bradner, "The Organizations Involved in the IETF Standards Process," best current practice BCP 11, RFC 2028, IETF, October 1996.

[4] ITU-T, "The E-Model, a computational model for use in transmission planning," Recommendation G.107 (05/00), May 2000.

[5] J. Reynolds and J. Postel, "Assigned Numbers," standard STD 2, RFC 1700, IETF, October 1994.

[6] T. Narten and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs," best current practice BCP 26, RFC

2434, IETF, October 1998.

[7] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," RFC 1889, IETF, February 1996.

Non-Normative References

[8] A. Adams, T. Bu, R. Caceres, N. G. Duffield, T. Friedman, J. Horowitz, F. Lo Presti, S. B. Moon, V. Paxson, and D. Towsley, "The Use of End-to-End Multicast Measurements for Characterizing Internal Network Behavior," IEEE Communications Magazine, May 2000.

[9] R. Caceres, N. G. Duffield, and T. Friedman, "Impromptu measurement infrastructures using RTP," Proc. IEEE Infocom 2002.

[10] A. D. Clark, "Modeling the Effects of Burst Packet Loss and Recency on Subjective Voice Quality," Proc. IP Telephony Workshop 2001.

[11] D. Sisalem and A. Wolisz, "MLDA: A TCP-friendly Congestion Control Framework for Heterogeneous Multicast Environments", Proc. IWQoS 2000.