Network Working Group
Request for Comments: 3611
Category: Standards Track

T. Friedman, Ed.
Paris 6
R. Caceres, Ed.
IBM Research
A. Clark, Ed.
Telchemy
November 2003

RTP Control Protocol Extended Reports (RTCP XR)

Status of this Memo

Copyright Notice

Abstract

This document defines the Extended Report (XR) packet type for the RTP Control Protocol (RTCP), and defines how the use of XR packets can be signaled by an application if it employs the Session Description Protocol (SDP). 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) [9], and defines how the use of XR packets can be signaled by an application if it employs the Session Description Protocol (SDP) [4]. 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 sections that follow the Introduction. Section 2 defines the XR packet as consisting of an eight octet header followed by a series of components called report blocks. Section 3 defines the common format, or framework, consisting of a type and a length field, required for all report blocks. Section 4 defines several specific report block types. Other block types can be defined in future documents as the need arises.

The report block types defined in this document fall into three categories. The first category consists of packet-by-packet reports on received or lost RTP packets. Reports in the second category convey reference time information between RTP participants. In the third category, reports convey metrics relating to packet receipts, that are summary in nature but that are more detailed, or of a different type, than that conveyed in existing RTCP packets.

All told, seven report block formats are defined by this document. Of these, three are packet-by-packet block types:

- Loss RLE Report Block (Section 4.1): Run length encoding of reports concerning the losses and receipts of RTP packets.

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

- Packet Receipt Times Report Block (Section 4.3): A list of reception timestamps of RTP packets.

There are two reference time related block types:

- Receiver Reference Time Report Block (Section 4.4): Receiver-end wallclock timestamps. Together with the DLRR Report Block mentioned next, these allow non-senders to calculate round-trip times.

- DLRR Report Block (Section 4.5): The delay since the last Receiver Reference Time Report Block was received. An RTP data sender that receives a Receiver Reference Time Report Block can respond with a DLRR Report Block, in much the same way as, in the mechanism already defined for RTCP [9, Section 6.3.1], an RTP data receiver that receives a sender's NTP timestamp can respond by filling in the DLSR field of an RTCP reception report block.

Finally, this document defines two summary metric block types:

- Statistics Summary Report Block (Section 4.6): Statistics on RTP packet sequence numbers, losses, duplicates, jitter, and TTL or Hop Limit values.

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

Before proceeding to the XR packet and report block definitions, this document provides an applicability statement (Section 1.1) that describes the contexts in which these report blocks can be used. It also defines (Section 1.2) the normative use of key words, such as MUST and SHOULD, as they are employed in this document.

Following the definitions of the various report blocks, this document describes how applications that employ SDP can signal their use (Section 5). The document concludes with a discussion (Section 6) of numbering considerations for the Internet Assigned Numbers Authority (IANA), of security considerations (Section 7), and with appendices that provide examples of how to implement algorithms discussed in the text.

1.1. Applicability

The XR packets are useful across multiple applications, and for that reason are not defined as profile-specific extensions to RTCP sender or Receiver Reports [9, Section 6.4.3]. Nonetheless, they are not of use in all contexts. In particular, the VoIP metrics report block (Section 4.7) is specific to voice applications, though it can be employed over a wide variety of such applications.

The VoIP metrics report block can be applied to any one-to-one or one-to-many voice application for which the use of RTP and RTCP is specified. The use of conversational metrics (Section 4.7.5), including the R factor (as described by the E Model defined in [3]) and the mean opinion score for conversational quality (MOS-CQ), in applications other than simple two party calls is not defined; hence, these metrics should be identified as unavailable in multicast conferencing applications.

The packet-by-packet report block types, Loss RLE (Section 4.1), Duplicate RLE (Section 4.2), and Packet Receipt Times (Section 4.3), have been defined with network tomography applications, such as multicast inference of network characteristics (MINC) [11], in mind. MINC requires detailed packet receipt traces from multicast session receivers in order to infer the gross structure of the multicast distribution tree and the parameters, such as loss rates and delays, that apply to paths between the branching points of that tree.

Any real time multicast multimedia application can use the packet-by-packet report block types. Such an application could employ a MINC inference subsystem that would provide it with multicast tree topology information. One potential use of such a subsystem would be for the identification of high loss regions in the multicast tree and the identification of multicast session participants well situated to provide retransmissions of lost packets.

Detailed packet-by-packet reports do not necessarily have to consume disproportionate bandwidth with respect to other RTCP packets. An application can cap the size of these blocks. A mechanism called "thinning" is provided for these report blocks, and can be used to ensure that they adhere to a size limit by restricting the number of packets reported upon within any sequence number interval. The rationale for, and use of this mechanism is described in [13]. Furthermore, applications might not require report blocks from all receivers in order to answer such important questions as where in the multicast tree there are paths that exceed a defined loss rate threshold. Intelligent decisions regarding which receivers send these report blocks can further restrict the portion of RTCP bandwidth that they consume.

The packet-by-packet report blocks can also be used by dedicated network monitoring applications. For such an application, it might be appropriate to allow more than 5% of RTP data bandwidth to be used for RTCP packets, thus allowing proportionately larger and more detailed report blocks.

Nothing in the packet-by-packet block types restricts their use to multicast applications. In particular, they could be used for network tomography similar to MINC, but using striped unicast packets instead. In addition, if it were found useful, they could be used for applications limited to two participants.

One use to which the packet-by-packet reports are not immediately suited is for data packet acknowledgments as part of a packet retransmission mechanism. The reason is that the packet accounting technique suggested for these blocks differs from the packet accounting normally employed by RTP. In order to favor measurement

applications, an effort is made to interpret as little as possible at the data receiver, and leave the interpretation as much as possible to participants that receive the report blocks. Thus, for example, a packet with an anomalous SSRC ID or an anomalous sequence number might be excluded by normal RTP accounting, but would be reported upon for network monitoring purposes.

The Statistics Summary Report Block (Section 4.6) has also been defined with network monitoring in mind. This block type can be used equally well for reporting on unicast and multicast packet reception.

The reference time related block types were conceived for receiver-based TCP-friendly multicast congestion control [18]. By allowing data receivers to calculate their round trip times to senders, they help the receivers estimate the downstream bandwidth they should request. Note that if every receiver is to send Receiver Reference Time Report Blocks (Section 4.4), a sender might potentially send a number of DLRR Report Blocks (Section 4.5) equal to the number of receivers whose RTCP packets have arrived at the sender within its reporting interval. As the number of participants in a multicast session increases, an application should use discretion regarding which participants send these blocks, and how frequently.

XR packets supplement the existing RTCP packets, and may be stacked with other RTCP packets to form compound RTCP packets [9, Section 6]. The introduction of XR packets into a session in no way changes the rules governing the calculation of the RTCP reporting interval [9, Section 6.2]. As XR packets are RTCP packets, they count as such for bandwidth calculations. As a result, the addition of extended reporting information may tend to increase the average RTCP packet size, and thus the average reporting interval. This increase may be limited by limiting the size of XR packets.

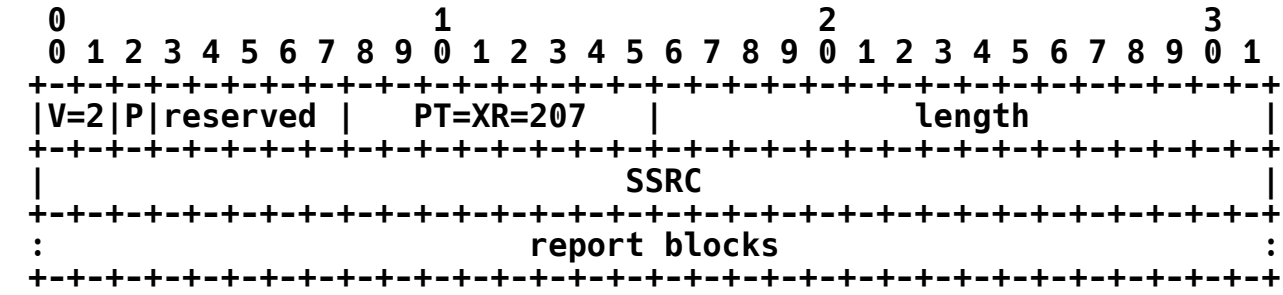
The SDP signaling defined for XR packets in this document (Section 5) was done so with three use scenarios in mind: a Real Time Streaming Protocol (RTSP) controlled streaming application, a one-to-many multicast multimedia application such as a course lecture with enhanced feedback, and a Session Initiation Protocol (SIP) controlled conversational session involving two parties. Applications that employ SDP are free to use additional SDP signaling for cases not covered here. In addition, applications are free to use signaling mechanisms other than SDP.

1.2. 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 BCP 14, RFC 2119 [1] and indicate requirement levels for compliance with this specification.

2. XR Packet Format

An XR packet consists of a header of two 32-bit words, followed by a number, possibly zero, of extended report blocks. This type of packet is laid out in a manner consistent with other RTCP packets, as concerns the essential version, packet type, and length information. XR packets are thus backwards compatible with RTCP receiver implementations that do not recognize them, but that ought to be able to parse past them using the length information. A padding field and an SSRC field are also provided in the same locations that they appear in other RTCP packets, for simplicity. The format is as follows:



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 6.1.

length: 16 bits
As described for the RTCP Sender Report (SR) packet (see Section 6.4.1 of the RTP specification [9]). 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.

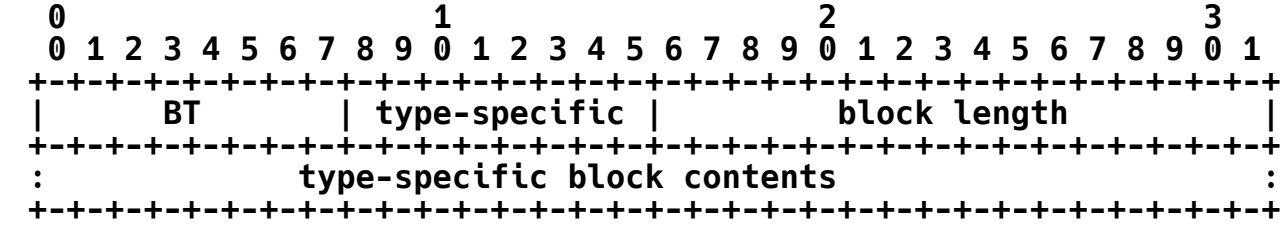
report blocks: variable length.
Zero or more extended report blocks. In keeping with the extended report block framework defined below, each block MUST consist of one or more 32-bit words.

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 6.2.

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 block contents field.

type-specific block contents: 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 reporting upon received packet losses and duplicates, packet reception times, receiver reference time information, receiver inter-report delays, detailed reception statistics, and voice over IP (VoIP) metrics. An implementation SHOULD ignore incoming blocks with types not relevant or unknown to it. Additional block types MUST be registered with the Internet Assigned Numbers Authority (IANA) [16], as described in Section 6.2.

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) [11]. 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 [13].

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 [9] 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 RTP data packet 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 of identifying 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 was 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

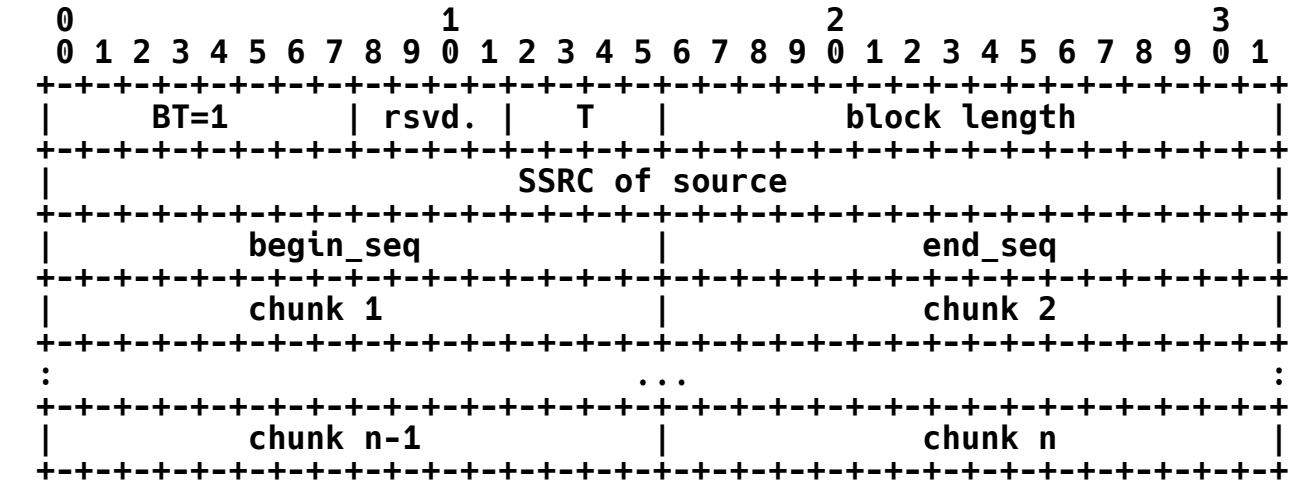
This could be encoded as follows:

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.

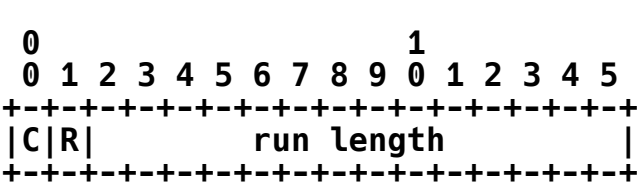
SSRC of source: 32 bits
The SSRC of the RTP data packet source being reported upon by this report block.

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 left most bit of the chunk determines its type: 0 for run length and 1 for bit vector.

4.1.1. Run Length Chunk

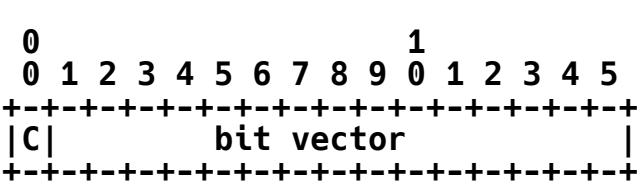


chunk type (C): 1 bit
A zero identifies this as a run length chunk.

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

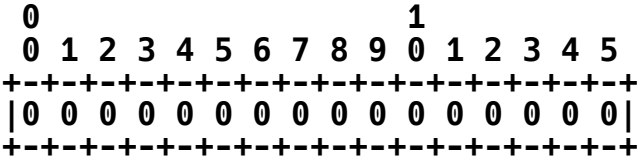


chunk type (C): 1 bit
A one identifies this as a bit vector chunk.

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

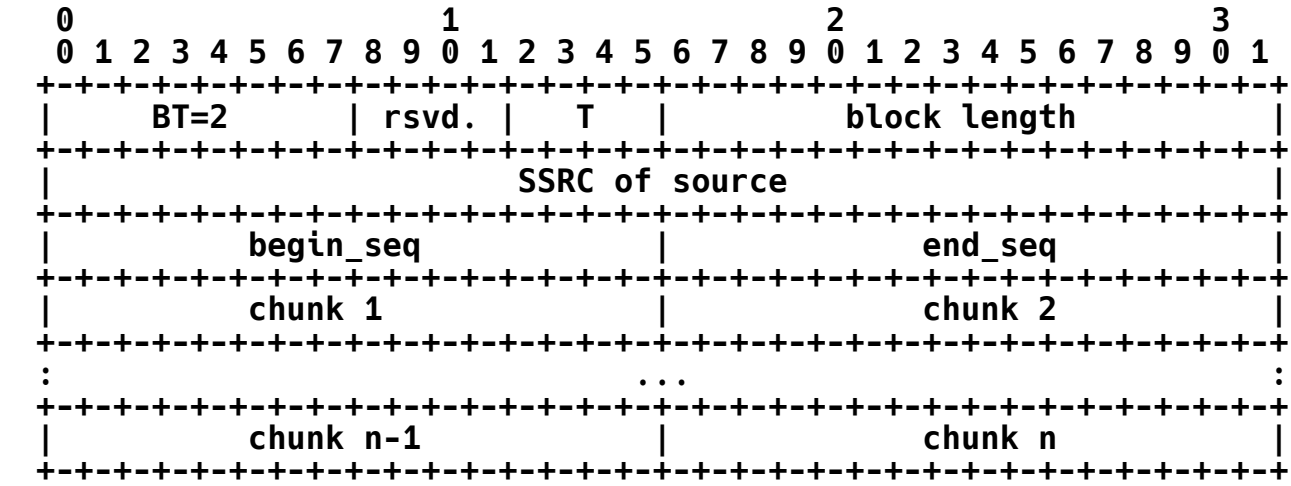
This block type permits per-sequence-number reports on duplicates in a source's RTP packet stream. Such information can be used for network diagnosis, and provide an alternative to packet losses as a basis for multicast tree topology inference.

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 a 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.

SSRC of source: 32 bits
As defined in Section 4.1.

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. Packet Receipt Times Report Block

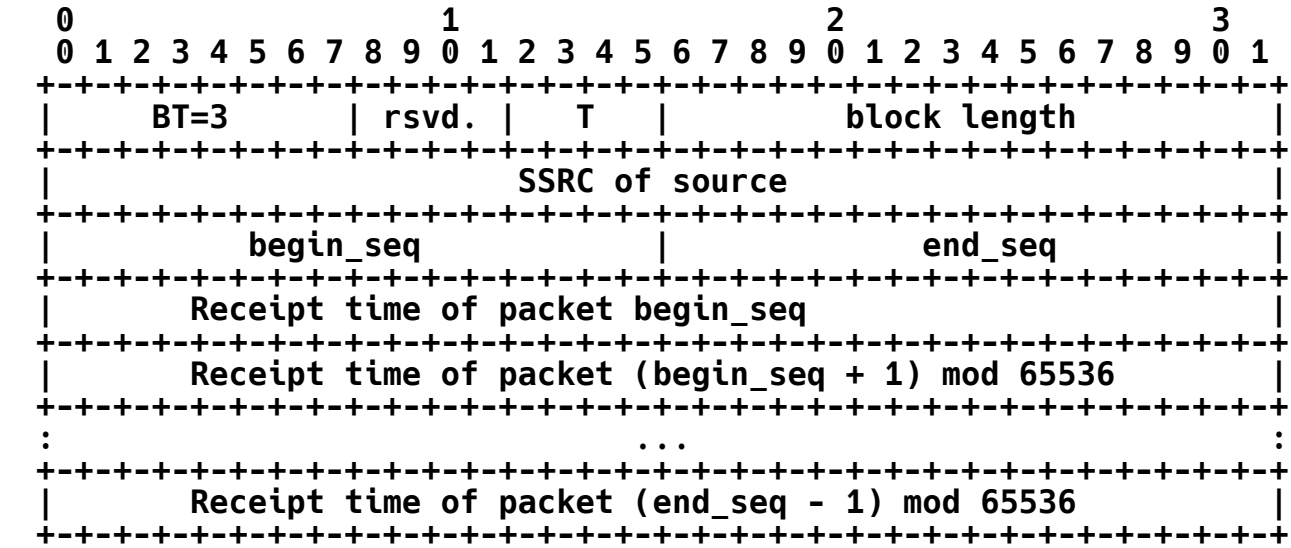
This block type permits per-sequence-number reports on packet receipt times for a given source's RTP packet stream. Such information can be used for MINC inference of the topology of the multicast tree used to distribute the source's RTP packets, and of the delays along the links within that tree. It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

Packet receipt times are expressed in the same units as in the RTP timestamps of data packets. This is so that, for each packet, one can establish both the send time and the receipt time in comparable terms. Note, however, that as an RTP sender ordinarily initializes its time to a value chosen at random, there can be no expectation that reported send and receipt times will differ by an amount equal to the one-way delay between sender and receiver. The reported times can nonetheless be useful for the purposes mentioned above.

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 receipt times 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 receipt times, any time other than the earliest MUST NOT be reported. This is to ensure consistency among reports.

Times reported in RTP timestamp format 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 Packet Receipt Times Report Blocks.

The Packet Receipt Times Report Block has the following format:



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

rsvd.: 4 bits
This field is reserved for future definition. In the absence of such a 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.

SSRC of source: 32 bits
As defined in Section 4.1.

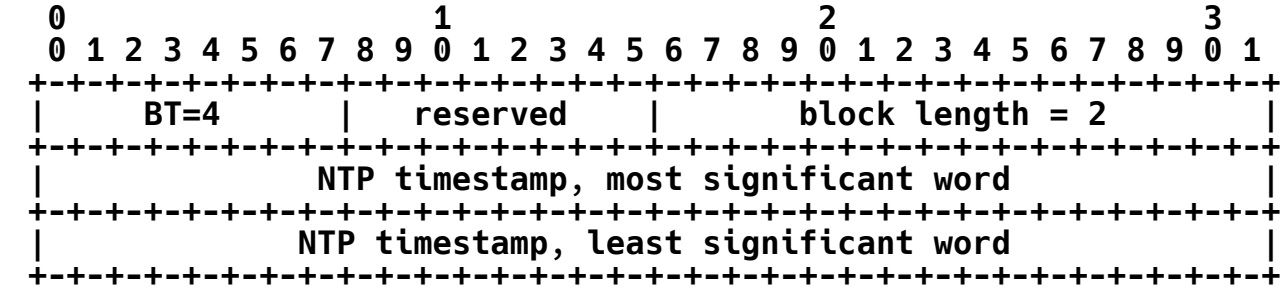
begin_seq: 16 bits
As defined in Section 4.1.

end_seq: 16 bits
As defined in Section 4.1.

Packet i receipt time: 32 bits
The receipt time of the packet with sequence number i at the receiver. The modular arithmetic shown in the packet format diagram is to allow for sequence number rollover. It is preferable for the time value 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 receipt times should reflect the actual time as closely as possible. For a session, if the RTP timestamp is chosen at random, the first receipt time value SHOULD also be chosen at random, and subsequent timestamps offset from this value. On the other hand, if the RTP timestamp is meant to reflect the reference time at the sender, then the receipt time SHOULD be as close as possible to the reference time at the receiver.

4.4. Receiver Reference Time 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 [9, Sec. 6.3.1]. A non-sender may estimate its round trip time (RTT) to other participants, as proposed in [18], by sending this report block and receiving DLRR Report Blocks (see next section) in reply.



block type (BT): 8 bits
A Receiver Reference Time Report Block is identified by the constant 4.

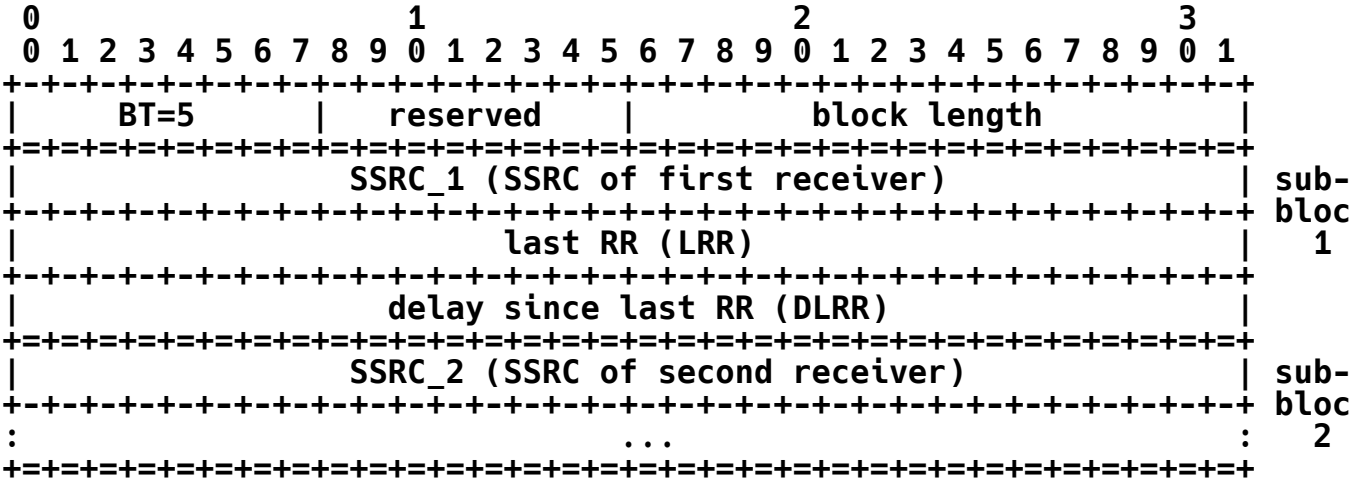
reserved: 8 bits
This field is reserved for future definition. In the absence of such definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

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.

4.5. DLRR Report Block

This block extends RTCP's delay since the last Sender Report (DLSR) mechanism [9, Sec. 6.3.1] so that non-senders may also calculate round trip times, as proposed in [18]. It is termed DLRR for delay since the 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 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
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 Reference Time Report Block from participant SSRC_n. If no such block has been received, the field is set to zero.

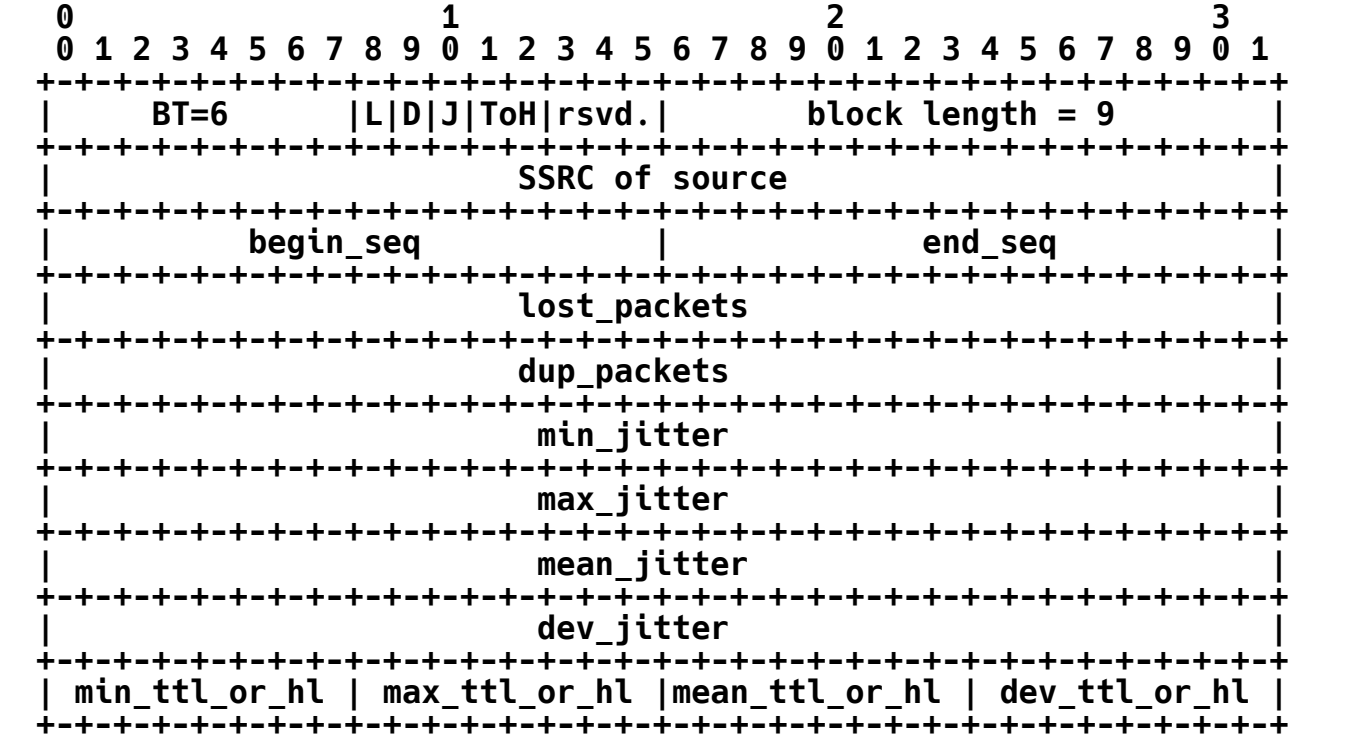
delay since last RR (DLRR): 32 bits
The delay, expressed in units of 1/65536 seconds, between receiving the last Receiver Reference Time Report Block from participant SSRC_n and sending this DLRR Report Block. If a Receiver Reference Time Report Block has yet to be received from SSRC_n, the DLRR field 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-LRR using the last RR timestamp (LRR) field, and then subtracting this field to leave the round-trip propagation delay as A-LRR-DLRR. This is illustrated in [9, Fig. 2].

4.6. Statistics Summary Report Block

This block reports statistics beyond the information carried in the standard RTCP packet format, but is not as finely grained as that carried in the report blocks previously described. Information is recorded about lost packets, duplicate packets, jitter measurements, and TTL or Hop Limit values. Such information can be useful for network management.

The report block contents are dependent upon a series of flag bits 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 present, but are 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 6.

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, mean_jitter, and dev_jitter fields all contain reports, 0 if none of them do.

TTL or Hop Limit flag (ToH): 2 bits
This field is set to 0 if none of the fields min_ttl_or_hl, max_ttl_or_hl, mean_ttl_or_hl, or dev_ttl_or_hl contain reports. If the field is non-zero, then all of these fields contain reports. The value 1 signifies that they report on IPv4 TTL values. The value 2 signifies that they report on

IPv6 Hop Limit values. The value 3 is undefined and MUST NOT be used.

rsvd.: 3 bits
This field is reserved for future definition. In the absence of such a 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.

SSRC of source: 32 bits
As defined in Section 4.1.

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.

mean_jitter: 32 bits
The mean relative transit time between each two packet series in the above sequence number interval, rounded to the nearest value expressible as an RTP timestamp.

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_or_hl: 8 bits
The minimum TTL or Hop Limit value of data packets in the sequence number range.

max_ttl_or_hl: 8 bits
The maximum TTL or Hop Limit value of data packets in the sequence number range.

mean_ttl_or_hl: 8 bits
The mean TTL or Hop Limit value of data packets in the sequence number range, rounded to the nearest integer.

dev_ttl_or_hl: 8 bits
The standard deviation of TTL or Hop Limit values of data packets in the sequence number range.

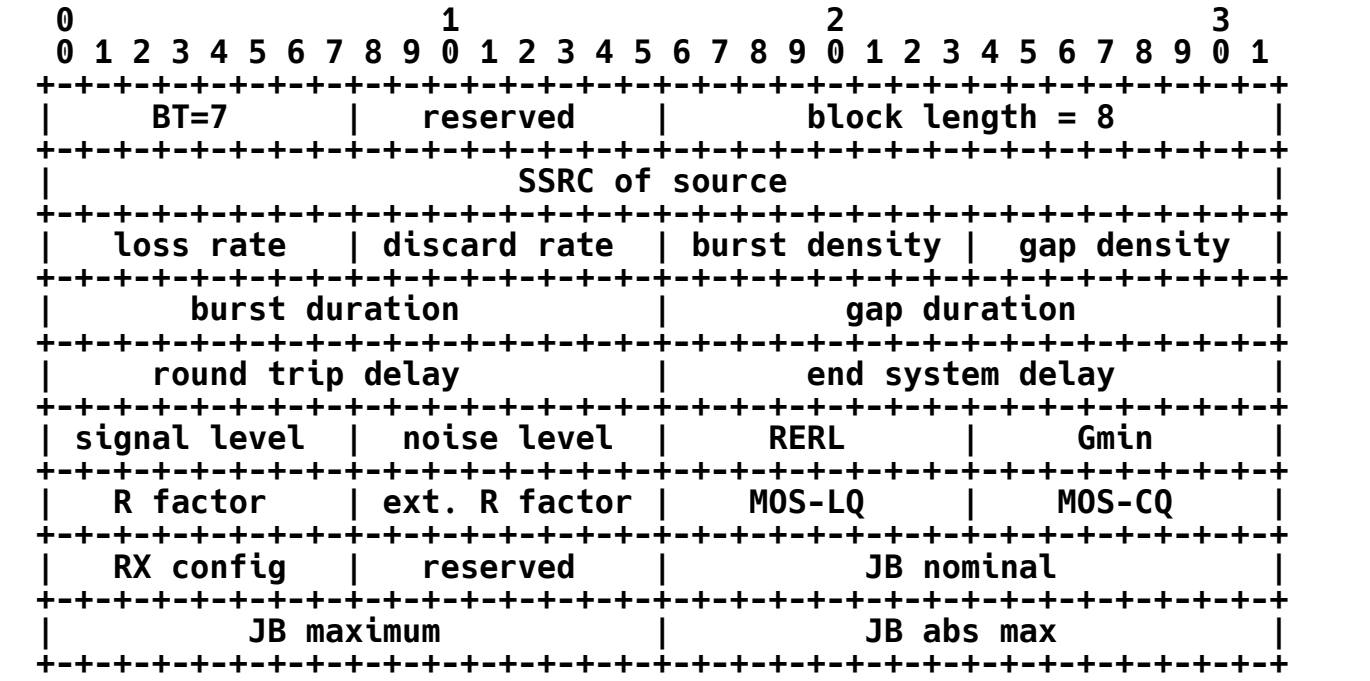
4.7. VoIP Metrics Report Block

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 [14]. Following a Gilbert-Elliott model [3], 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 RTP 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 [6,3]) and mean opinion scores (MOS scores).

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 a definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

block length: 16 bits
The constant 8, in accordance with the definition of this field in Section 3.

SSRC of source: 32 bits
As defined in Section 4.1.

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, and MUST be set to zero if no packets have been received.

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 is a period during which a high proportion of packets are either lost or discarded due to late arrival. A burst is defined, in terms of a value Gmin, as the longest sequence that (a) starts with a lost or discarded packet, (b) does not contain any occurrences of Gmin or more consecutively received (and not discarded) packets, and (c) ends with a lost or discarded packet.

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 rate 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 packet, 0 a lost packet, and X a discarded packet in the following pattern covering 64 packets:

```
   1111011111111111111111X111X10111101111111111111111X111111111
   |---------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 rate              12, which corresponds to 5%
   discard rate           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. This field MUST be populated and MUST be set to zero if no packets have been received.

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. This field MUST be populated and MUST be set to zero if no packets have been received.

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 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 MAY be measured using RTCP, the DLRR method defined in Section 4.5 of this document, where it is necessary to convert the units of measurement from NTP timestamp values to milliseconds, or other approaches. If RTCP is used, then the reported delay 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; however, this field 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 sum of the total sample accumulation and encoding delay associated with the sending direction and the jitter buffer, decoding, and playout buffer delay associated with the receiving direction. This delay MAY be estimated or measured. This value SHOULD be provided in all VoIP metrics reports. If an implementation is unable to provide the data, the value 0 MUST be used.

Note that the one way symmetric VoIP segment delay may be calculated from the round trip and end system delays is 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 0 dBm0 reference [10], 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 Log10 (rms talkspurt power (mW))

  A value of 127 indicates that this parameter is unavailable. Typical values should generally be 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 0 dBm0 reference, 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 value may be measured directly by the VoIP end system's echo canceller or may be estimated by adding the echo return loss (ERL) and echo return loss enhancement (ERLE) values reported by the echo canceller.

  RERL(dB) = ERL (dB) + ERLE (dB)

In the case of a VoIP gateway, the source of echo is typically line echo that occurs at 2-4 wire conversion points in the network. This 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 acoustic coupling between handset speaker and microphone or 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 loss due to acoustic feedback.

Examples:

- 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.

- 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.

- IP phone with conventional handset. Acoustic coupling from handset speaker to microphone (terminal coupling loss) is typically 40 dB. RERL = 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 using a format such as RTCP XR.

Echo related information may not be available in all VoIP end systems. As echo does have a significant effect on conversational quality, it is recommended that estimated values for echo return loss and terminal coupling loss be provided (if sensible estimates can be reasonably determined).

Typical values for end systems are given below to provide guidance:

- IP Phone with handset: typically 45 dB.
- PC softphone or speakerphone: extremely variable, consider reporting "undefined" (127).
- IP gateway with line echo canceller: typically has ERL and ERLE available.
- IP gateway without line echo canceller: frequently a source of echo related problems, consider reporting either a low value (12 dB) or "undefined" (127).

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, they 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 [6] and ETSI TS 101 329-5 [3].

A value of 127 indicates that this parameter is unavailable. Values other than 127 and the valid range defined above MUST not be sent and MUST be ignored by the receiving system.

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 [6] and ETSI TS 101 329-5 [3], 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. Values other than 127 and the valid range defined above MUST not be sent and MUST be ignored by the receiving system.

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. Values other than 127 and the valid range defined above MUST not be sent and MUST be ignored by the receiving system.

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 [6] or ETSI TS 101 329-5 [3] 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.

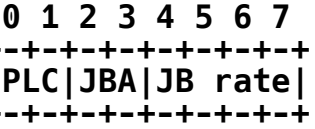
A value of 127 indicates that this parameter is unavailable. Values other than 127 and the valid range defined above MUST not be sent and MUST be ignored by the receiving system.

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 part of a burst or gap in later reports. ETSI TS 101 329-5 [3] 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, MUST be provided, and MUST remain constant across VoIP Metrics report blocks for the duration of the RTP session.

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 approach is typically able to conceal isolated lost packets at low packet loss rates. When PLC = 10, then an enhanced interpolation algorithm is being used; algorithms of this type are able to conceal high packet loss rates effectively. When PLC = 01, then silence is being 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 an adaptive jitter buffer and may be estimated. A value of 0 indicates that the adjustment time is unknown for this implementation.

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

4.7.7. Jitter Buffer Parameters

The values reported in these fields SHOULD be the most recently obtained values at the time of reporting.

jitter buffer nominal delay (JB nominal): 16 bits
This is the current nominal jitter buffer delay in milliseconds, 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 delay (JB maximum): 16 bits
This is the current maximum jitter buffer delay in milliseconds which corresponds 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 delay (JB abs max): 16 bits
This is the absolute maximum delay in milliseconds that the adaptive jitter buffer can reach under worst case conditions. If this value exceeds 65535 milliseconds, then this field SHALL convey the value 65535. 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. SDP Signaling

This section defines Session Description Protocol (SDP) [4] signaling for XR blocks that can be employed by applications that utilize SDP. This signaling is defined to be used either by applications that implement the SDP Offer/Answer model [8] or by applications that use SDP to describe media and transport configurations in connection

with such protocols as the Session Announcement Protocol (SAP) [15] or the Real Time Streaming Protocol (RTSP) [17]. There exist other potential signaling methods that are not defined here.

The XR blocks MAY be used without prior signaling. This is consistent with the rules governing other RTCP packet types, as described in [9]. An example in which signaling would not be used is an application that always requires the use of one or more XR blocks. However, for applications that are configured at session initiation, the use of some type of signaling is recommended.

Note that, although the use of SDP signaling for XR blocks may be optional, if used, it MUST be used as defined here. If SDP signaling is used in an environment where XR blocks are only implemented by some fraction of the participants, the ones not implementing the XR blocks will ignore the SDP attribute.

5.1. The SDP Attribute

This section defines one new SDP attribute "rtcp-xr" that can be used to signal participants in a media session that they should use the specified XR blocks. This attribute can be easily extended in the future with new parameters to cover any new report blocks.

The RTCP XR blocks SDP attribute is defined below in Augmented Backus-Naur Form (ABNF) [2]. It is both a session and a media level attribute. When specified at session level, it applies to all media level blocks in the session. Any media level specification MUST replace a session level specification, if one is present, for that media block.

```
 rtcp-xr-attrib = "a=" "rtcp-xr" ":" [xr-format *(SP xr-format)] CRLF

  xr-format = pkt-loss-rle
            / pkt-dup-rle
            / pkt-rcpt-times
            / rcvr-rtt
            / stat-summary
            / voip-metrics
            / format-ext

  pkt-loss-rle   = "pkt-loss-rle" ["=" max-size]
  pkt-dup-rle    = "pkt-dup-rle" ["=" max-size]
  pkt-rcpt-times = "pkt-rcpt-times" ["=" max-size]
  rcvr-rtt       = "rcvr-rtt" "=" rcvr-rtt-mode [":" max-size]
  rcvr-rtt-mode  = "all"
                 / "sender"
  stat-summary   = "stat-summary" ["=" stat-flag *("," stat-flag)]
```

```
   stat-flag      = "loss"
                  / "dup"
                  / "jitt"
                  / "TTL"
                  / "HL"
   voip-metrics   = "voip-metrics"
   max-size       = 1*DIGIT ; maximum block size in octets
   DIGIT          = %x30-39
   format-ext     = non-ws-string

   non-ws-string  = 1*(%x21-FF)
   CRLF           = %d13.10
```

The "rtcp-xr" attribute contains zero, one, or more XR block related parameters. Each parameter signals functionality for an XR block, or a group of XR blocks. The attribute is extensible so that parameters can be defined for any future XR block (and a parameter should be defined for every future block).

Each "rtcp-xr" parameter belongs to one of two categories. The first category, the unilateral parameters, are for report blocks that simply report on the RTP stream and related metrics. The second category, collaborative parameters, are for XR blocks that involve actions by more than one party in order to carry out their functions.

Round trip time (RTT) measurement is an example of collaborative functionality. An RTP data packet receiver sends a Receiver Reference Time Report Block (Section 4.4). A participant that receives this block sends a DLRR Report Block (Section 4.5) in response, allowing the receiver to calculate its RTT to that participant. As this example illustrates, collaborative functionality may be implemented by two or more different XR blocks. The collaborative functionality of several XR blocks may be governed by a single "rtcp-xr" parameter.

For the unilateral category, this document defines the following parameters. The parameter names and their corresponding XR formats are as follows:

Parameter name	XR block (block type and name)
pkt-loss-rle	1 Loss RLE Report Block
pkt-dup-rle	2 Duplicate RLE Report Block
pkt-rcpt-times	3 Packet Receipt Times Report Block
stat-summary	6 Statistics Summary Report Block
voip-metrics	7 VoIP Metrics Report Block

The "pkt-loss-rle", "pkt-dup-rle", and "pkt-rcpt-times" parameters MAY specify an integer value. This value indicates the largest size the whole report block SHOULD have in octets. This shall be seen as an indication that thinning shall be applied if necessary to meet the target size.

The "stat-summary" parameter contains a list indicating which fields SHOULD be included in the Statistics Summary report blocks that are sent. The list is a comma separated list, containing one or more field indicators. The space character (0x20) SHALL NOT be present within the list. Field indicators represent the flags defined in Section 4.6. The field indicators and their respective flags are as follows:

Indicator	Flag
loss	loss report flag (L)
dup	duplicate report flag (D)
jitt	jitter flag (J)
TTL	TTL or Hop Limit flag (ToH)
HL	TTL or Hop Limit flag (ToH)

For "loss", "dup", and "jitt", the presence of the indicator indicates that the corresponding flag should be set to 1 in the Statistics Summary report blocks that are sent. The presence of "TTL" indicates that the corresponding flag should be set to 1. The presence of "HL" indicates that the corresponding flag should be set to 2. The indicators "TTL" and "HL" MUST NOT be signaled together.

Blocks in the collaborative category are classified as initiator blocks or response blocks. Signaling SHOULD indicate which participants are required to respond to the initiator block. A party that wishes to receive response blocks from those participants can trigger this by sending an initiator block.

The collaborative category currently consists only of one functionality, namely the RTT measurement mechanism for RTP data receivers. The collective functionality of the Receiver Reference Time Report Block and DLRR Report Block is represented by the "rcvr-rtt" parameter. This parameter takes as its arguments a mode value and, optionally, a maximum size for the DLRR report block. The mode value "all" indicates that both RTP data senders and data receivers MAY send DLRR blocks, while the mode value "sender" indicates that only active RTP senders MAY send DLRR blocks, i.e., non RTP senders SHALL NOT send DLRR blocks. If a maximum size in octets is included, any DLRR Report Blocks that are sent SHALL NOT exceed the specified size. If size limitations mean that a DLRR Report Block sender cannot report in one block upon all participants from which it has

received a Receiver Reference Time Report Block then it SHOULD report on participants in a round robin fashion across several report intervals.

The "rtcp-xr" attributes parameter list MAY be empty. This is useful in cases in which an application needs to signal that it understands the SDP signaling but does not wish to avail itself of XR functionality. For example, an application in a SIP controlled session could signal that it wishes to stop using all XR blocks by removing all applicable SDP parameters in a re-INVITE message that it sends. If XR blocks are not to be used at all from the beginning of a session, it is RECOMMENDED that the "rtcp-xr" attribute not be supplied at all.

When the "rtcp-xr" attribute is present, participants SHOULD NOT send XR blocks other than the ones indicated by the parameters. This means that inclusion of a "rtcp-xr" attribute without any parameters tells a participant that it SHOULD NOT send any XR blocks at all. The purpose is to conserve bandwidth. This is especially important when collaborative parameters are applied to a large multicast group: the sending of an initiator block could potentially trigger responses from all participants. There are, however, contexts in which it makes sense to send an XR block in the absence of a parameter signaling its use. For instance, an application might be designed so as to send certain report blocks without negotiation, while using SDP signaling to negotiate the use of other blocks.

5.2. Usage in Offer/Answer

In the Offer/Answer context [8], the interpretation of SDP signaling for XR packets depends upon the direction attribute that is signaled: "recvonly", "sendrecv", or "sendonly" [4]. If no direction attribute is supplied, then "sendrecv" is assumed. This section applies only to unicast media streams, except where noted. Discussion of unilateral parameters is followed by discussion of collaborative parameters in this section.

For "sendonly" and "sendrecv" media stream offers that specify unilateral "rtcp-xr" attribute parameters, the answerer SHOULD send the corresponding XR blocks. For "sendrecv" offers, the answerer MAY include the "rtcp-xr" attribute in its response, and specify any unilateral parameters in order to request that the offerer send the corresponding XR blocks. The offerer SHOULD send these blocks.

For "recvonly" media stream offers, the offerer's use of the "rtcp-xr" attribute in connection with unilateral parameters indicates that the offerer is capable of sending the corresponding XR blocks. If

the answerer responds with an "rtcp-xr" attribute, the offerer SHOULD send XR blocks for each specified unilateral parameter that was in its offer.

For multicast media streams, the inclusion of an "rtcp-xr" attribute with unilateral parameters means that every media recipient SHOULD send the corresponding XR blocks.

An SDP offer with a collaborative parameter declares the offerer capable of receiving the corresponding initiator and replying with the appropriate responses. For example, an offer that specifies the "rcvr-rtt" parameter means that the offerer is prepared to receive Receiver Reference Time Report Blocks and to send DLRR Report Blocks. An offer of a collaborative parameter means that the answerer MAY send the initiator, and, having received the initiator, the offerer SHOULD send the responses.

There are exceptions to the rule that an offerer of a collaborative parameter should send responses. For instance, the collaborative parameter might specify a mode that excludes the offerer; or congestion control or maximum transmission unit considerations might militate against the offerer's response.

By including a collaborative parameter in its answer, the answerer declares its ability to receive initiators and to send responses. The offerer MAY then send initiators, to which the answerer SHOULD reply with responses. As for the offer of a collaborative parameter, there are exceptions to the rule that the answerer should reply.

When making an SDP offer of a collaborative parameter for a multicast media stream, the offerer SHOULD specify which participants are to respond to a received initiator. A participant that is not specified SHOULD NOT send responses. Otherwise, undue bandwidth might be consumed. The offer indicates that each participant that is specified SHOULD respond if it receives an initiator. It also indicates that a specified participant MAY send an initiator block.

An SDP answer for a multicast media stream SHOULD include all collaborative parameters that are present in the offer and that are supported by the answerer. It SHOULD NOT include any collaborative parameter that is absent from the offer.

If a participant receives an SDP offer and understands the "rtcp-xr" attribute but does not wish to implement XR functionality offered, its answer SHOULD include an "rtcp-xr" attribute without parameters. By doing so, the party declares that, at a minimum, is capable of understanding the signaling.

5.3. Usage Outside of Offer/Answer

SDP can be employed outside of the Offer/Answer context, for instance for multimedia sessions that are announced through the Session Announcement Protocol (SAP) [15], or streamed through the Real Time Streaming Protocol (RTSP) [17]. The signaling model is simpler, as the sender does not negotiate parameters, but the functionality expected from specifying the "rtcp-xr" attribute is the same as in Offer/Answer.

When a unilateral parameter is specified for the "rtcp-xr" attribute associated with a media stream, the receiver of that stream SHOULD send the corresponding XR block. When a collaborative parameter is specified, only the participants indicated by the mode value in the collaborative parameter are concerned. Each such participant that receives an initiator block SHOULD send the corresponding response block. Each such participant MAY also send initiator blocks.

6. IANA Considerations

This document defines a new RTCP 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 RTCP 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.

Further, this document defines a new SDP attribute, "rtcp-xr", within the existing IANA registry of SDP Parameters. It defines a new IANA registry, the registry of RTCP XR SDP Parameters, and an initial set of six parameters, and describes how additional parameters are to be allocated.

6.1. XR Packet Type

The XR packet type defined by this document is registered with the IANA as packet type 207 in the registry of RTP RTCP Control Packet types (PT).

6.2. RTCP XR Block Type Registry

This document creates an IANA registry called the RTCP XR Block Type Registry to cover the name space of the Extended Report block type (BT) field specified in Section 3. The BT field contains eight bits, allowing 256 values. The RTCP XR Block Type Registry is to be managed by the IANA according to the Specification Required policy of

RFC 2434 [7]. 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	Packet Receipt Times Report Block
4	Receiver Reference Time Report Block
5	DLRR Report Block
6	Statistics Summary Report Block
7	VoIP Metrics Report Block

The BT value 255 is reserved for future extensions.

Furthermore, future specifications SHOULD avoid the value 0. Doing so facilitates packet validity checking, since an all-zeros field might commonly be found in an ill-formed packet.

Any registration MUST contain the following information:

- Contact information of the one doing the registration, including at least name, address, and email.

- The format of the block type being registered, consistent with the extended report block format described in Section 3.

- A description of what the block type represents and how it shall be interpreted, detailing this information for each of its fields.

6.3. The "rtcp-xr" SDP Attribute

The SDP attribute "rtcp-xr" defined by this document is registered with the IANA registry of SDP Parameters as follows:

SDP Attribute ("att-field"):

Attribute name:	rtcp-xr
Long form:	RTP Control Protocol Extended Report Parameters
Type of name:	att-field
Type of attribute:	session and media level
Subject to charset:	no
Purpose:	see Section 5 of this document
Reference:	this document
Values:	see this document and registrations below

The attribute has an extensible parameter field and therefore a registry for these parameters is required. This document creates an IANA registry called the RTCP XR SDP Parameters Registry. It contains the six parameters defined in Section 5.1: "pkt-loss-rle", "pkt-dup-rle", "pkt-rcpt-times", "stat-summary", "voip-metrics", and "recv-rtt".

Additional parameters are to be added to this registry in accordance with the Specification Required policy of RFC 2434 [7]. Any registration MUST contain the following information:

- Contact information of the one doing the registration, including at least name, address, and email.

- An Augmented Backus-Naur Form (ABNF) [2] definition of the parameter, in accordance with the "format-ext" definition of Section 5.1.

- A description of what the parameter represents and how it shall be interpreted, both normally and in Offer/Answer.

7. Security Considerations

This document extends the RTCP reporting mechanism. The security considerations that apply to RTCP reports [9, Appendix B] 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 report blocks and the VoIP Metrics Report Block provide examples.

The per-packet information contained in Loss RLE, Duplicate RLE, and Packet Receipt Times Report Blocks facilitates multicast inference of network characteristics (MINC) [11]. Such inference can reveal the gross topology of a multicast distribution tree, as well as parameters, such as the loss rates and delays, along paths between branching points in that tree. Such information might be considered sensitive 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 an 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. Encryption procedures, such as those being suggested for a Secure RTCP (SRTCP) [12] at the time that this document was written, can be used when confidentiality is a concern to end hosts. Given that RTCP traffic can be encrypted by the end hosts, autonomous systems must be prepared for the fact that certain aspects of their network topology can be revealed.

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 place an upper limit on the size of the XR report blocks they employ. The "thinning" techniques described in Section 4.1 permit the packet-by-packet report blocks to adhere to a predefined size limit.

A. Algorithms

A.1. Sequence Number Interpretation

This is 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 of 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). The algorithm, which has been verified against a working implementation for correctness, is reproduced from ETSI TS 101 329-5 [3]. The algorithm, as described here, takes precedence over any change that might eventually be made to the algorithm in future ETSI documents.

This 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 rates */
   loss_rate = 256 * loss_count / ctotal;
   discard_rate = 256 * discard_count / ctotal;
```

Intellectual Property Notice

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; Mounir Benzaid for drawing our attention to the reporting needs of MLDA; Dorgham Sisalem and Adam Wolisz for encouraging us to incorporate MLDA block types; and Jose Rey for valuable review of the SDP Signaling section.

Contributors

The following people are the authors of this document:

Kevin Almeroth, UCSB
Ramon Caceres, IBM Research
Alan Clark, Telchemy
Robert G. Cole, JHU Applied Physics Laboratory
Nick Duffield, AT&T Labs-Research
Timur Friedman, Paris 6
Kaynam Hedayat, Brix Networks
Kamil Sarac, UT Dallas
Magnus Westerlund, Ericsson

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	Packet Receipt Times Report Block	Friedman, Caceres, Duffield
4.4	Receiver Reference Time Report Block	Friedman
4.5	DLRR Report Block	Friedman
4.6	Statistics Summary Report Block	Almeroth, Sarac
4.7	VoIP Metrics Report Block	Clark, Cole, Hedayat

The principal person to contact regarding the SDP signaling described in this document is Magnus Westerlund.

References

Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[2] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

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

[4] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

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

[6] ITU-T, "The E-Model, a computational model for use in transmission planning", Recommendation G.107, January 2003.

[7] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[8] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with the Session Description Protocol (SDP)", RFC 3264, June 2002.

[9] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.

[10] TIA/EIA-810-A Transmission Requirements for Narrowband Voice over IP and Voice over PCM Digital Wireline Telephones, December 2000.

Informative References

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

[12] Baugher, McGrew, Oran, Blom, Carrara, Naslund and Norrman, "The Secure Real-time Transport Protocol", Work in Progress.

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

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

[15] Handley, M., Perkins, C. and E. Whelan, "Session Announcement Protocol", RFC 2974, October 2000.

[16] Reynolds, J., Ed., "Assigned Numbers: RFC 1700 is Replaced by an On-line Database", RFC 3232, January 2002.

[17] Schulzrinne, H., Rao, A. and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

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

Authors' Addresses

Kevin Almeroth
Department of Computer Science
University of California
Santa Barbara, CA 93106 USA

EMail: almeroth@cs.ucsb.edu

Ramon Caceres
IBM Research
19 Skyline Drive
Hawthorne, NY 10532 USA

EMail: caceres@watson.ibm.com

Alan Clark
Telchemy Incorporated
3360 Martins Farm Road, Suite 200
Suwanee, GA 30024 USA

Phone: +1 770 614 6944
Fax: +1 770 614 3951
EMail: alan@telchemy.com

Robert G. Cole
Johns Hopkins University Applied Physics Laboratory
MP2-S170
11100 Johns Hopkins Road
Laurel, MD 20723-6099 USA

Phone: +1 443 778 6951
EMail: robert.cole@jhuapl.edu

Nick Duffield
AT&T Labs-Research
180 Park Avenue, P.O. Box 971
Florham Park, NJ 07932-0971 USA

Phone: +1 973 360 8726
Fax: +1 973 360 8050
EMail: duffield@research.att.com

Timur Friedman
Universite Pierre et Marie Curie (Paris 6)
Laboratoire LiP6-CNRS
8, rue du Capitaine Scott
75015 PARIS France

Phone: +33 1 44 27 71 06
Fax: +33 1 44 27 74 95
EMail: timur.friedman@lip6.fr

Kaynam Hedayat
Brix Networks
285 Mill Road
Chelmsford, MA 01824 USA

Phone: +1 978 367 5600
Fax: +1 978 367 5700
EMail: khedayat@brixnet.com

Kamil Sarac
Department of Computer Science (ES 4.207)
Eric Jonsson School of Engineering & Computer Science
University of Texas at Dallas
Richardson, TX 75083-0688 USA

Phone: +1 972 883 2337
Fax: +1 972 883 2349
EMail: ksarac@utdallas.edu

Magnus Westerlund
Ericsson Research
Ericsson AB
SE-164 80 Stockholm Sweden

Phone: +46 8 404 82 87
Fax: +46 8 757 55 50
EMail: magnus.westerlund@ericsson.com

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.