Audio/Video Transport Working Group
Internet Draft
June 25, 1999
Expires December 1999
draft-wing-avt-tcrtp-00.txt

Tmima Koren
Patrick Ruddy
Bruce Thompson
Alex Tweedly
Dan Wing
Cisco Systems

Tunneled multiplexed Compressed RTP ("TCRTP")

Status of this memo

Copyright Notice

Abstract

This document describes a mechanism which improves the end-to-end bandwidth utilization of RTP streams over an IP network by compressing the UDP and RTP headers and allowing the packets of multiple RTP streams to be multiplexed into one IP packet.

This new protocol is useful in low-bandwidth environments, such as dialup modems, ISDN, cellular, and G.Lite which wish to reduce the encapsulation overhead of normal RTP packets through the entire network. The protocol provides mechanisms to ensure synchronization at certain points to reduce the requirement for the receiver to communicate synchronization loss to the sender.

1. Introduction

This document describes a new protocol which compresses RTP [RTP] streams. The new protocol is based on RFC2508 [CRTP] and RFC2507 [IPHCOMP].

Readers should be familiar with RFC2508 [CRTP] and RFC2507 [IPHCOMP].

1.1. Background

Existing methods for compression of RTP [RTP] streams, such as [CRTP], are per-hop instead of end-to-end which places a processing burden on ingress routers and, if the bandwidth savings are desired in the network core, an additional processing burden on core routers.

[CRTP] includes features such as negative acknowledgements when the decompressor becomes unsynchronized with the compressor. [CRTP] was designed to operate hop-by-hop, but Negative acknowledgements are unsuitable as the primary mechanism to re-synchronize the decompressor in an end-to-end protocol because the round-trip time is longer than the useful life of the data being sent with RTP.

1.2. Overview

[CRTP] relies on a number of link-level protocol (such as PPP) features including: packet length indication, packet type indication, good error detection, lack of reordering.

The protocol described by this document is an application-level protocol, not a strict link-layer protocol. The end-to-end tunnel is built using IP, with a new protocol number. Multiple packets with compressed IP/UDP/RTP headers are carried by this, each preceded by a type and length indication.

This scheme transports multiple independent UDP/RTP streams between a pair of IP endpoints, i.e. the streams retain independent UDP source and destination ports, and RTP timestamps and sequence numbers.

TCRTP capability notification can be out-of-band, using the same mechanism to establish an RTP stream -- H.245 [H.245], SIP [SIP], SGCP [SGCP], or similar protocols.

This document assumes the reader is familiar with the protocol described in RFC2508 [CRTP]. Major differences from [CRTP] are:

* Delivery between compressor and decompressor is provided by an IP tunnel, using a new protocol number, rather than a link-level protocol;
* Multiple packets with compressed IP/UDP/RTP headers can be carried in a single IP packet;
* Explicit type and length fields are provided for each subpacket;
* The same compressed header types as RFC2508 are supported, with the addition of a new CRTPX type;

1.3. Advantages and Disadvantages of TCRTP

Advantages

a. TCRTP saves bandwidth over RTP, even when TCRTP is not multiplexing;
b. When multiplexing, TCRTP does not require the RTP sequence number, SSRC, or any other RTP field to be related to the RTP fields in the other RTP streams in the same multiplexed packet;
c. Unmultiplexed TCRTP incurs 4-7 bytes of header overhead, versus 20 for normal UDP+RTP;
d. A multiplexed TCRTP stream saves 40 bytes of header overhead (IP+UDP+RTP) per multiplexed packet;
e. Compared to CRTP, TCRTP works end-to-end instead of only one router hop;
f. TCRTP retains the IP source and destination, and UDP source and destination information in each sub-payload, which allows for:
    * easier dispatch to other processors in a distributed processor environment,
    * distributed compression and decompression in a distributed system environment,
    * better functionality with NAT [NAT],
    * multiplexing data to different destinations;
h. TCRTP leverages CRTP;
i. TCRTP can multiplex UDP as well as RTP streams.

Disadvantages

a. Each TCRTP packet isn't self-contained and loss of 'significant' packets or several successive packets can cause the TCRTP decompressor to temporarily lose synchronization.

1.4. Requirements Notation

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 RFC2119 [KEYW].

2. Encapsulation Format

The TCRTP packet uses a new IP protocol. This protocol will be named TCRTP and is not yet registered with IANA.

A TCRTP packet consists of:

1. an IP header, with the new TCRTP protocol type.
2. a set of packets with compressed IP/UDP/RTP headers, those headers replaced with a TCRTP header of variable length as described below.

Every TCRTP header starts with type and length fields. A detailed description follows:

Element	Bytes	Description
Type/CID/length	1	3 bits Type (see below) 1 bit CID-length indicator 1 reserved bit 3 bits high-order payload length
Length	1	8 bits low-order payload length
RTP Timestamp	4	RTP Timestamp (only present if Type = CRTPX)
RTP Sequence	2	RTP Sequence (only present if Type = CRTPX)
RTP Type	1	RTP Type (only present if type = CRTPX)
RTP DeltaT	1	RTP Delta time (only present if type = CRTPX)
Context ID	1/2	Unique per-source context value
MSTI/Sequence	1	Changed-field bitmask plus 4 bit sequence number (see RFC2508)
Checksum	0/2	Optional Checksum
delta fields	0-3	encoded delta values, which are present if the associated bit is "1" in MSTI, above
RTP Payload	varies	

Multiple TCRTP packets can be concatenated together in one IP packet. The IP 'length' field can be compared to the TCRTP length field to determine if multiple TCRTP packets are contained in one IP packet.

2.1. Description of Elements

2.1.1. Type/CID/Length Field

Type/CID/Length - this 8-bit field is divided as follows:

```
bit 0,1,2  = TCRTP Type (see below for values)
bit 3      = length of Context-ID.  0=one byte, 1=two bytes
bit 4      = reserved.  must be 0.
bit 5,6,7  = high-order length bits
```

There are six values for the 3-bit Type field, all of which are from RFC2508 [CRTP], except CRTPX which is a new Type field defined in this document.

```
0b000  reserved
0b001  FH        Full Header
0b010  CUDP      Compressed UDP
0b011  CNTCP     Compressed Non-TCP
0b100  CRTP      Compressed RTP
0b101  CRTPX     Compressed RTP with Extra Fields
0b110  CS        Context State
0b111  reserved
```

2.1.2. Length Field

The length indicates the length of the payload (not including the Type and Length bytes themselves).

The length of the payload is expressed in 11 bits. The lower 8 of these bits are in the Length field, and the upper 3 bits are in the Type/CID/Length field.

The 11 bit length field limits RTP payload length to 2048 bytes, but this should cover nearly all cases, and in any case, header compression is of little worth on long packets.

2.1.3. RTP Timestamp Field

This field is taken from the RTP header and placed in the TCRTP payload.

2.1.4. RTP Sequence Field

This field is taken from the RTP header and placed in the TCRTP payload.

2.1.5. RTP Type Field

This field is taken from the RTP header and placed in the TCRTP payload.

2.1.6. RTP DeltaT Field

This field is the first-order time increment and is used by the decompressor to re-compute the RTP timestamp.

This field is inherited from RFC2508 [CRTP].

2.1.7. Context ID Field

This field defines the unique per-destination stream. This value is chosen by the sender and MUST be unique per each IP destination.

If the associated bit in the Type field is 0, the Context-ID is one byte long; if 1, the Context-ID is two bytes long.

This field is inherited from RFC2508 [CRTP].

2.1.8. MSTI/Sequence Field

The MSTI bits indicate if the 'delta fields' are present.

The Sequence bitfield is incremented sequentially for each TCRTP packet transmitted and is used by the decompressor to determine if a packet was lost or delivered out-of-sequence. It is used in conjunction with the 'twice algorithm', described below.

Both the MSTI and Sequence are inherited from RFC2508 [CRTP].

2.1.9. Checksum Field

This is the UDP checksum of what would have been the originally-sent UDP packet, had there been no TCRTP compression or multiplexing.

The checksum field is necessary to verify the payload was not corrupted and is especially useful with the 'twice algorithm', described below, to verify the 'twice algorithm' has re-synchronized the decompressor.

The presence of this field is REQUIRED if the UDP checksum was present in the Full Header (FH) synchronization packet for this Context-ID. Note that the compressor can freely choose to provide checksums on some streams and no checksums on other streams, even in the same multiplexed packet -- however, the existence of a checksum must agree with the original synchronizing FH packet.

This field is inherited from RFC2508 [CRTP].

2.1.10. Delta Fields

These fields are only present if their associated bits are turned on in the MSTI field (described above). See RFC2508 [CRTP] for a description of these fields.

This field is inherited from RFC2508 [CRTP].

2.1.11. RTP Payload Field

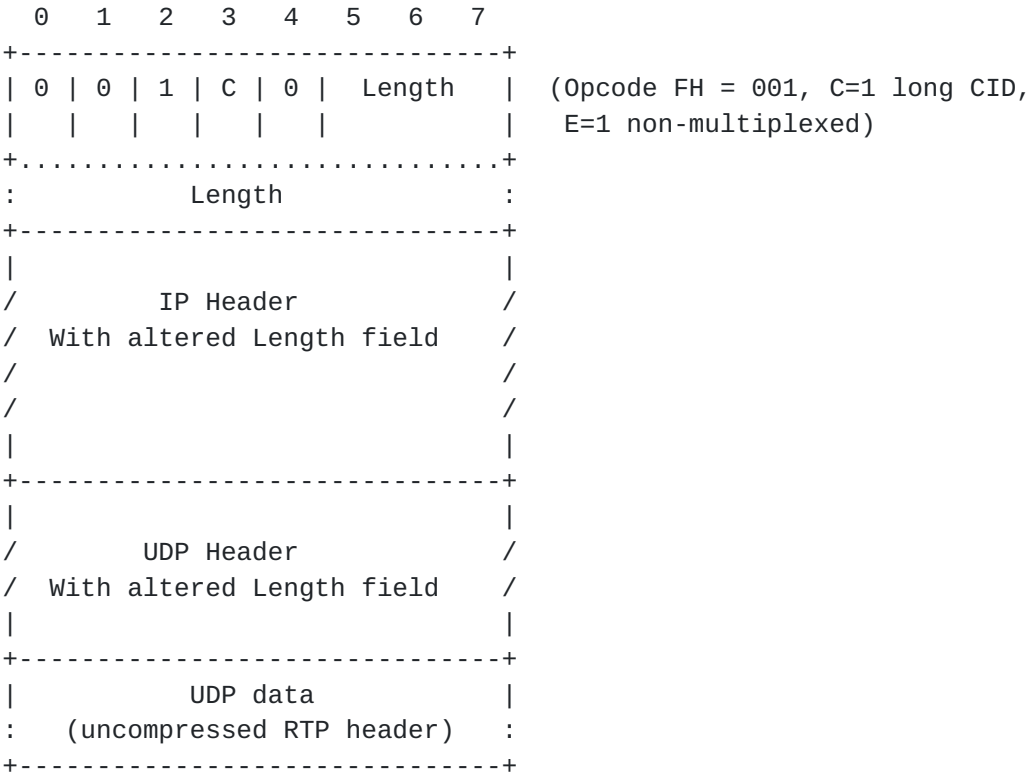
This is the RTP payload itself.

2.2. Payload Formats

The six payload types are diagrammed below. In all cases the 'delta fields' are shown, but note that the 'delta fields' are actually only present if their associated MSTI bits are set. See RFC2508 [CRTP] for details on the 'delta fields' and the MSTI bits.

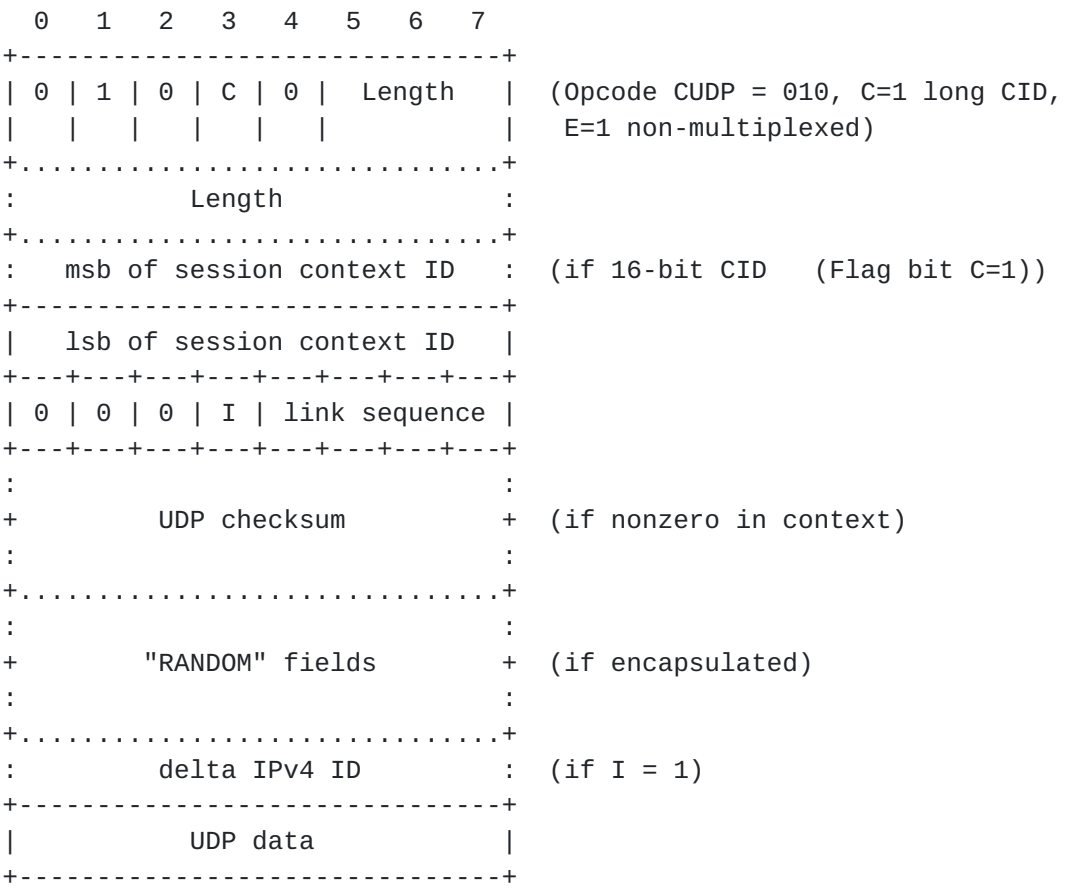
2.2.1. Full-Header (FH) Payload Format

This is contained in the payload of an IP packet:



2.2.2. Compressed UDP (CUDP) Payload Format

This is contained in the payload of an IP packet:



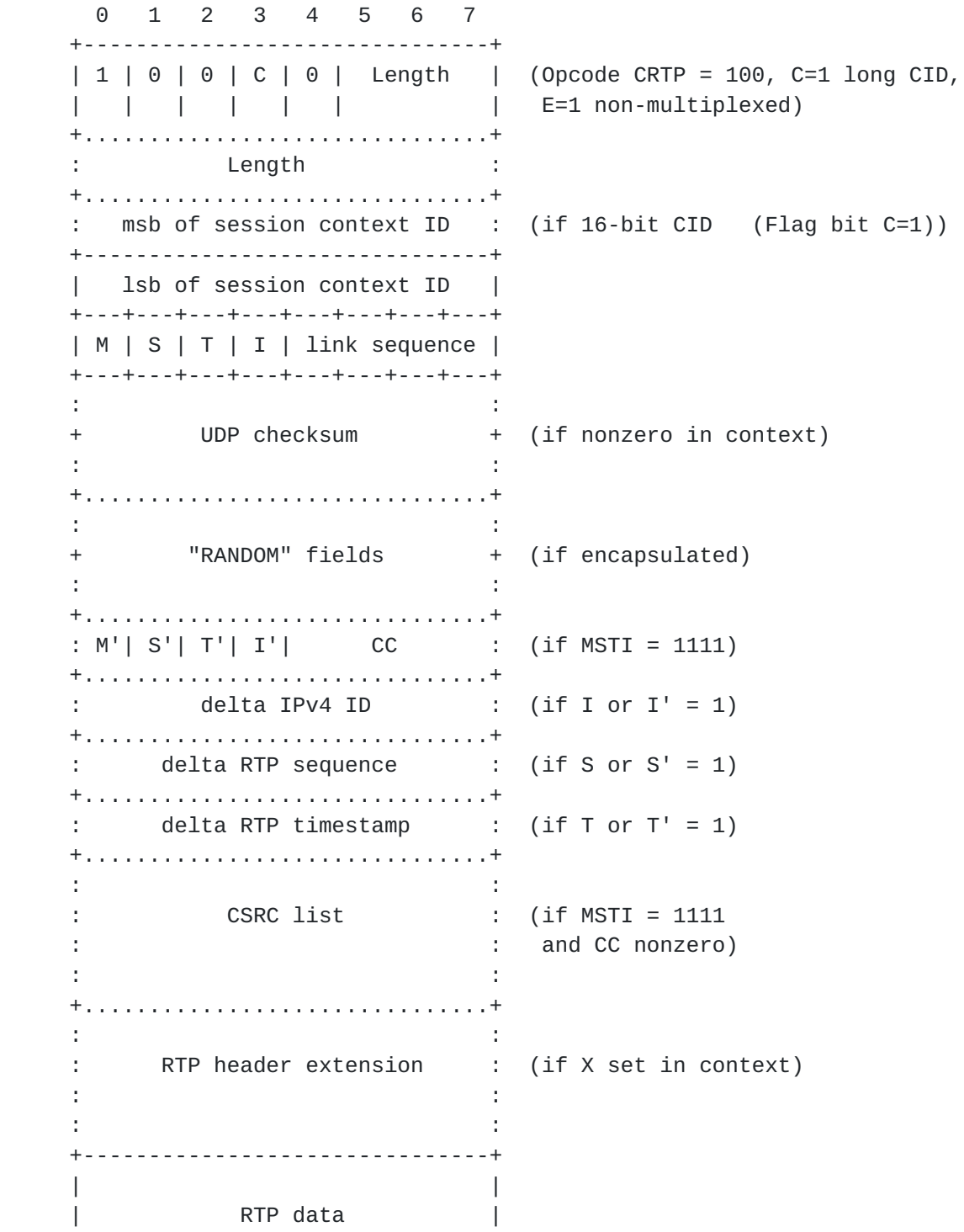
2.2.3. Compressed (CNTCP) Non-TCP Payload Format

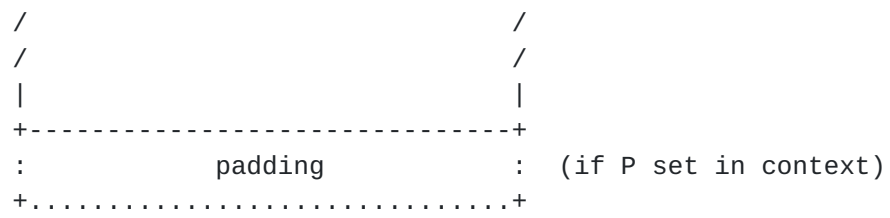
This is the Compressed UDP (CUDP) format for IPv6 and is identical to CUDP, described above.

2.2.4. Compressed RTP (CRTP) Payload Format

This is the 'typical' payload for most data, with the exceptions for significant events as described in section 3.

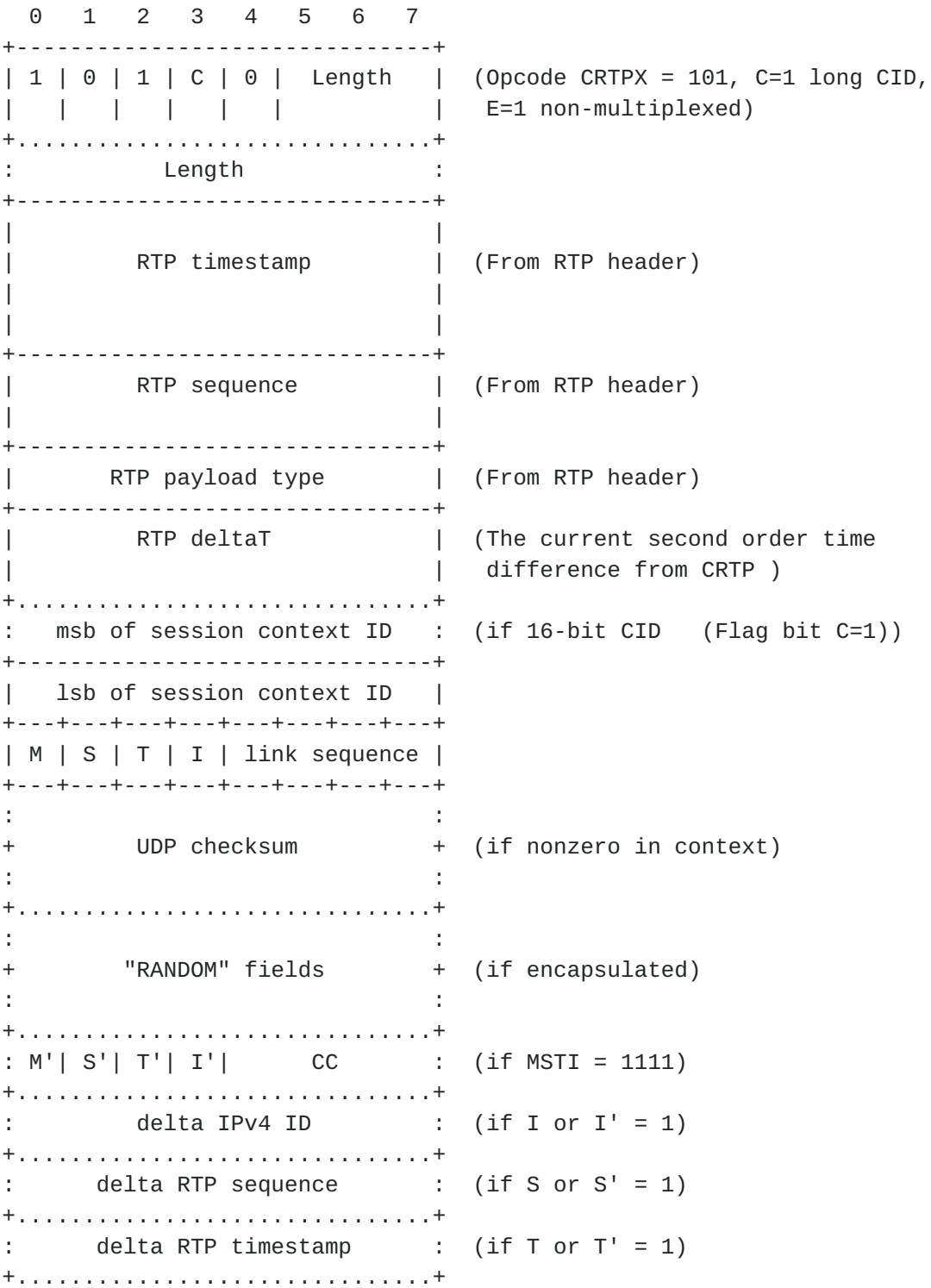
This is contained in the payload of an IP packet.

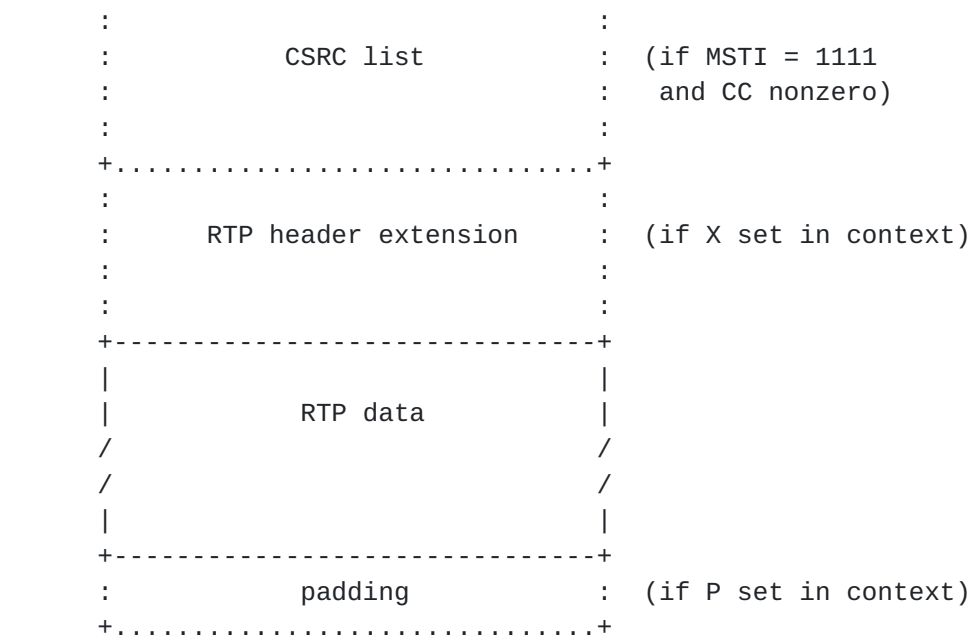



padding
(if P set in context)

2.2.5. Compressed RTP with Extra Fields (CRTPX)

The payload used to re-establish the full state of the receiver is typically as follows. This would only be sent when a major event occurs.

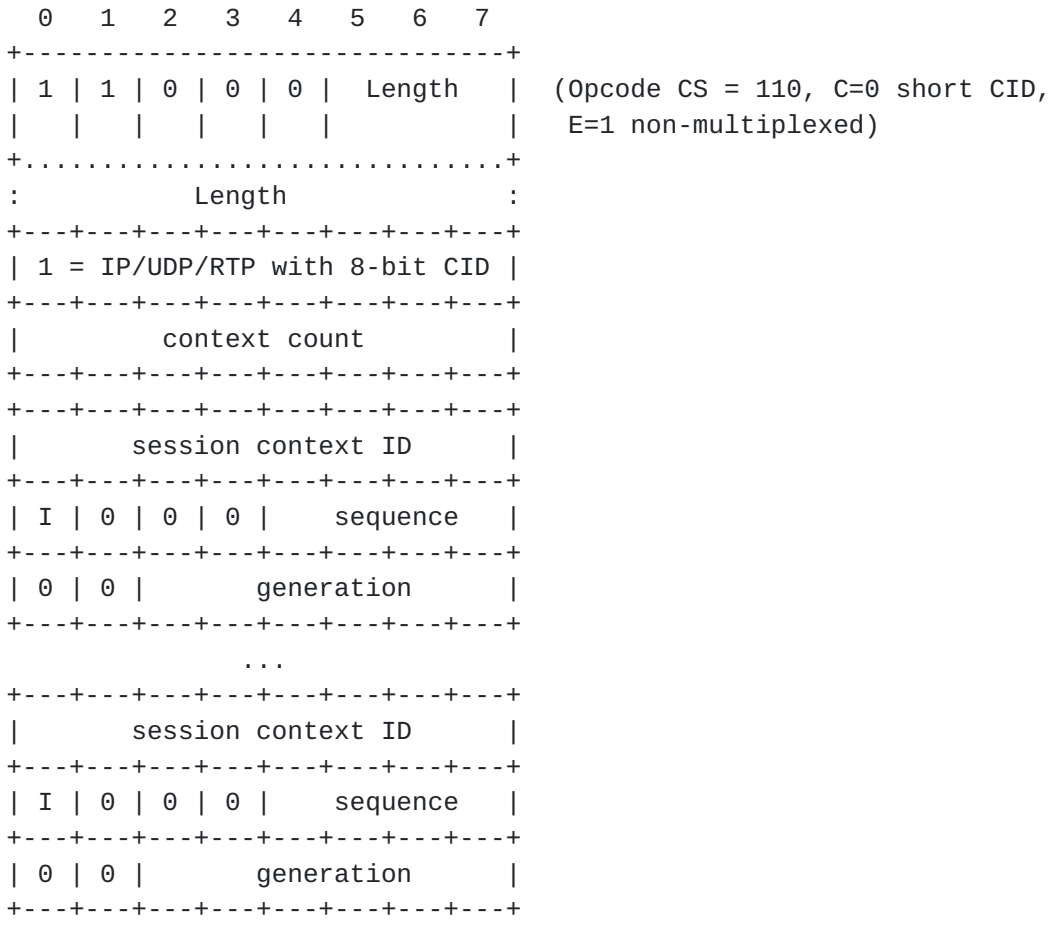



CSRC list
(if MSTI = 1111 and CC nonzero)
RTP header extension
(if X set in context)
RTP data
padding
(if P set in context)

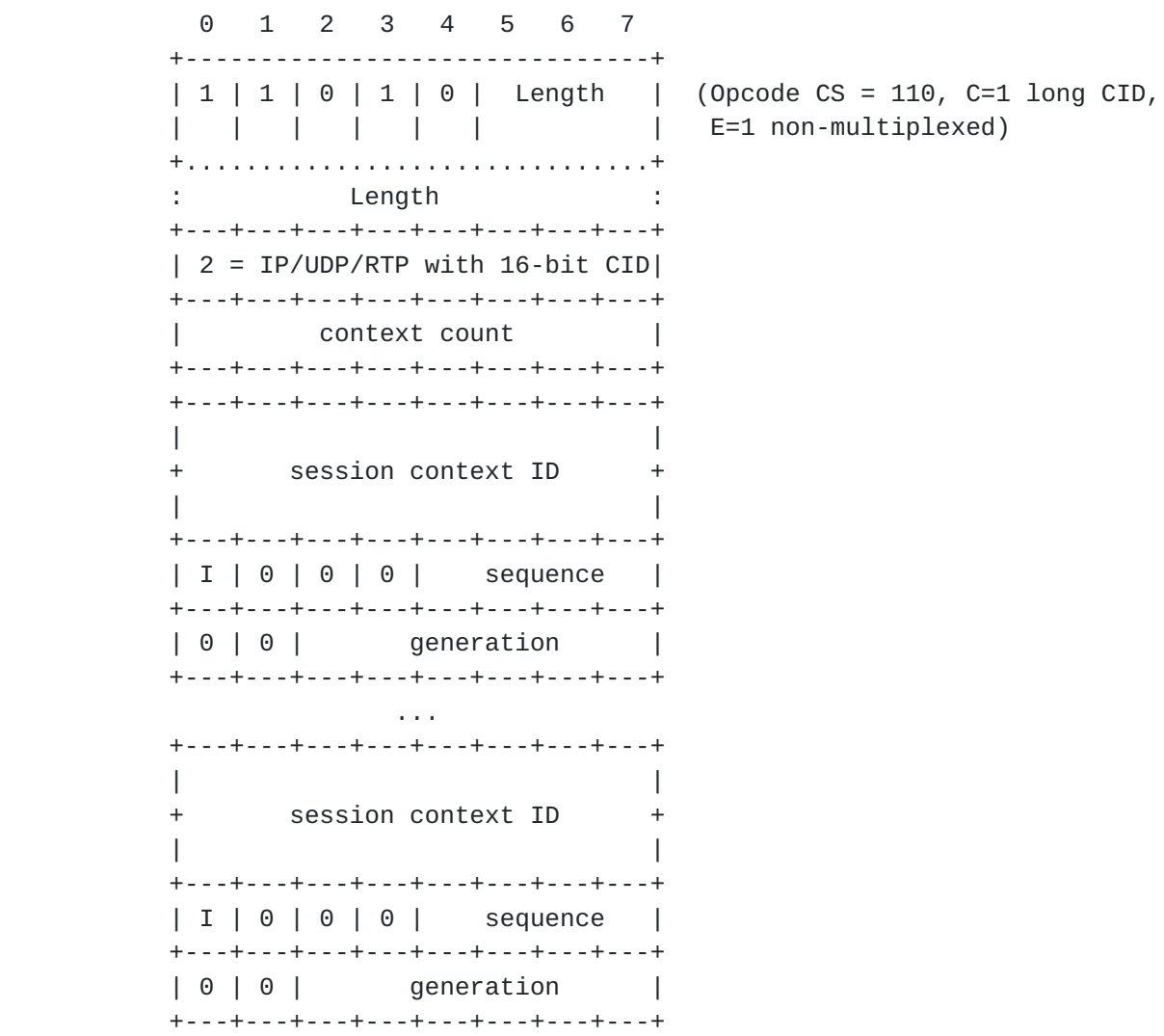
2.2.6. Context State (CS) Payload Format

There are two CS payload formats: one with long CID, one with short CID. Both are diagrammed below.

CS payload format with short CID:



CS payload format with long CID:



3. Operation of the Protocol

Generally, the operation of TCRTP is similar to RFC2508 [CRTP].

The Full Header packet shall consist of the full IP/UDP/RTP packet covering the CRTP Header and RTP Payload sections.

Compression of non-RTP UDP Packets: As with CRTP, TCRTP can also be used to compress non-RTP UDP packets: in this case the encoder sends only Compressed UDP (CUDP) packets for that stream.

3.1. Differences from RFC2508 behavior

As TCRTP operates end-to-end instead of hop-by-hop, some changes to the behavior described in RFC2508 [CRTP] are necessary. These

behavior exceptions are described below.

3.1.1. Changes to RTP stream

If the sending TCRTP determines there are any significant changes to the RTP fields that might cause the decompressor to lose synchronization if the significant change was lost (as is the case at the start of a voice spurt, for example) the sending TCRTP MAY ensure that the next N packets are all able to be fully and independently reconstructed by the decompressor -- for example, by sending FH, CUDP or CRTPX as appropriate.

If subsequent changes occur before N packets have been generated, the count to N is reset.

The transmission of non-compressed data during a significant event is useful for the TCRTP receiver so that it can recover in the event of a packet loss or corruption. This will allow the decompressor to recover from the loss of any of the N packets which will be sent. To actually cause the TCRTP receiver to lose state, all N packets would have to be dropped or corrupted.

The value of N is implementation-dependent. Higher values of N consume more bandwidth, while lower values increase the risk of the TCRTP decompressor losing synchronization.

When the decompressor detects lost or out-of-order packets, it should continue attempting to decode subsequent packets using the 'twice' algorithm instead of simply invalidating the stream as specified in RFC2508 [CRTP].

3.1.2. Dealing with lost or out-of-order packets

In the "normal" case where a long series of CRTP packets are sent (i.e. without FH, CUDP or CRTPX), the "twice" algorithm (described below) may allow recovery from loss of up to 16 consecutive packets, or out-of-order reception of more than 16 packets.

It may be also useful for the sender to occasionally send a CRTPX packet to ensure the decompressor is not out of synchronization due to packet loss.

3.1.3. Decompressor out-of-synchronization indication

If the decompressor loses synchronization and can no longer decode packets, even using the 'twice' algorithm (described below), the decompressor should send a Context State packet to the sender.

Upon receipt of a Context State packet, the sender MAY send a FH packet followed by a count of N-1 CRTPX packets.

Due to the round-trip delay between the sender and receiver this mechanism should not be relied upon as the only mechanism for the decompressor to recover when it has lost synchronization. The other mechanisms described in this section are usually preferred.

3.2. Description of the "'Twice' Algorithm"

From RFC2508 [CRTP], section 3.3.5:

> In the case where UDP checksums are being transmitted, the decompressor MAY attempt to use the "twice" algorithm described in section 10.1 of RFC2507 [IPHCOMP].
>
> In this algorithm, the delta is applied more than once on the assumption that the delta may have been the same on the missing packet(s) and the one subsequently received. Success is indicated by a checksum match. For the scheme defined here, the difference in the 4-bit sequence number tells number of times the delta must be applied. Note, however, that there is a nontrivial risk of an incorrect positive indication. It may be advisable to request a FULL_HEADER or COMPRESSED_NON_TCP packet even if the "twice" algorithm succeeds.

3.3. Transmitting CRTPX instead of FH or CUDP

RFC2508 [CRTP] defines FH and CUDP as the mechanisms to synchronize the CRTP receiver. TCRTP allows FH and CUDP to continue to perform this synchronization function, but also defines a new type, CRTPX, which does not reset some fields.

CRTPX is superior to FH and CUDP because:

* A CRTPX packet is shorter than a FH or a CUDP packet;
* FH and CUDP have the side effect of causing the Delta-T and the IP Identification fields to be reset.

4. Multiplexing

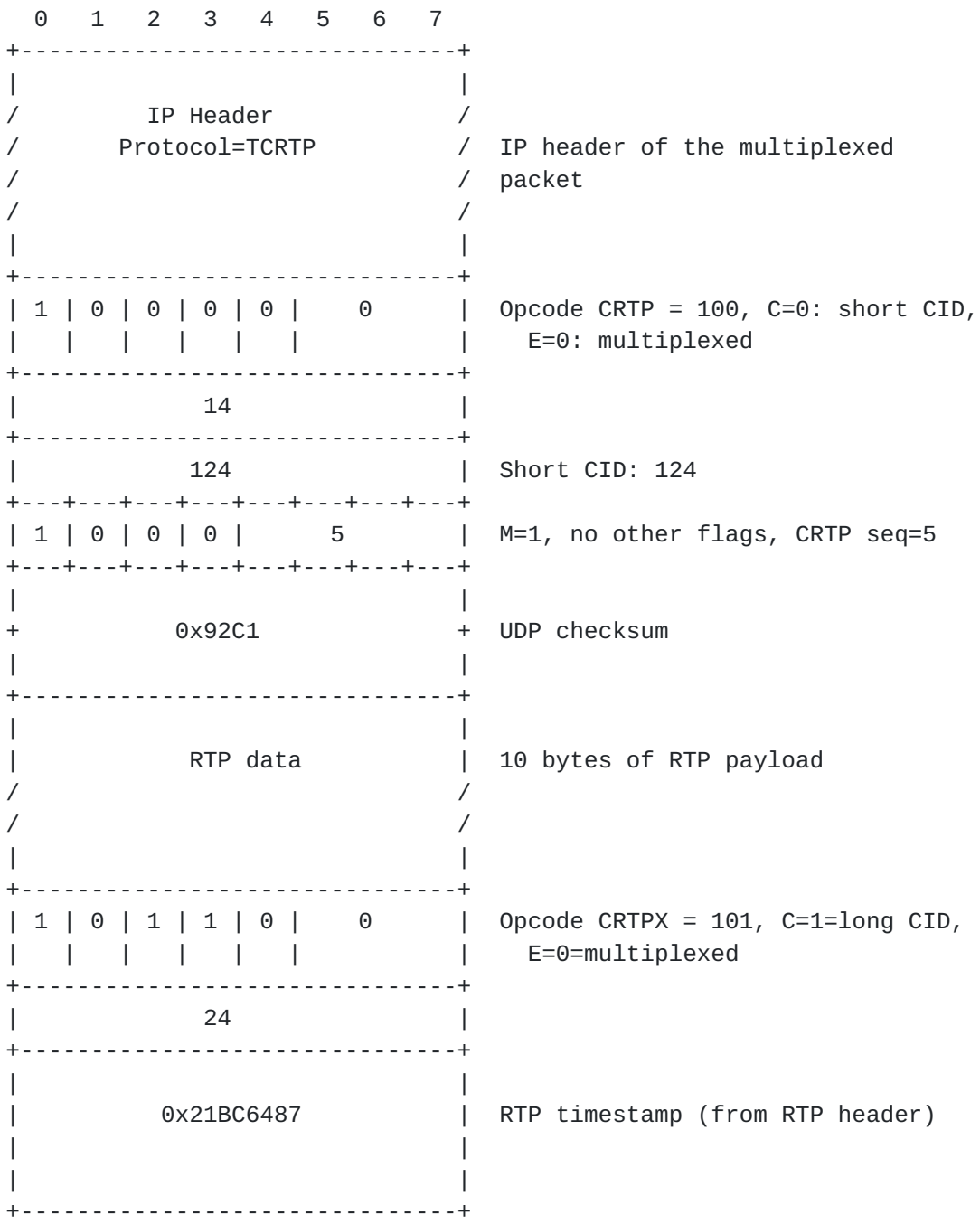
Multiple TCRTP payloads may be concatenated together when they originate from the same source host and are being sent to the same destination host. Each TCRTP payload includes its own TCRTP-type and TCRTP-length fields. The IP length is used to determine if multiple TCRTP payloads are present. The length of each TCRTP payload is used to determine the location of subsequent TCRTP payloads.

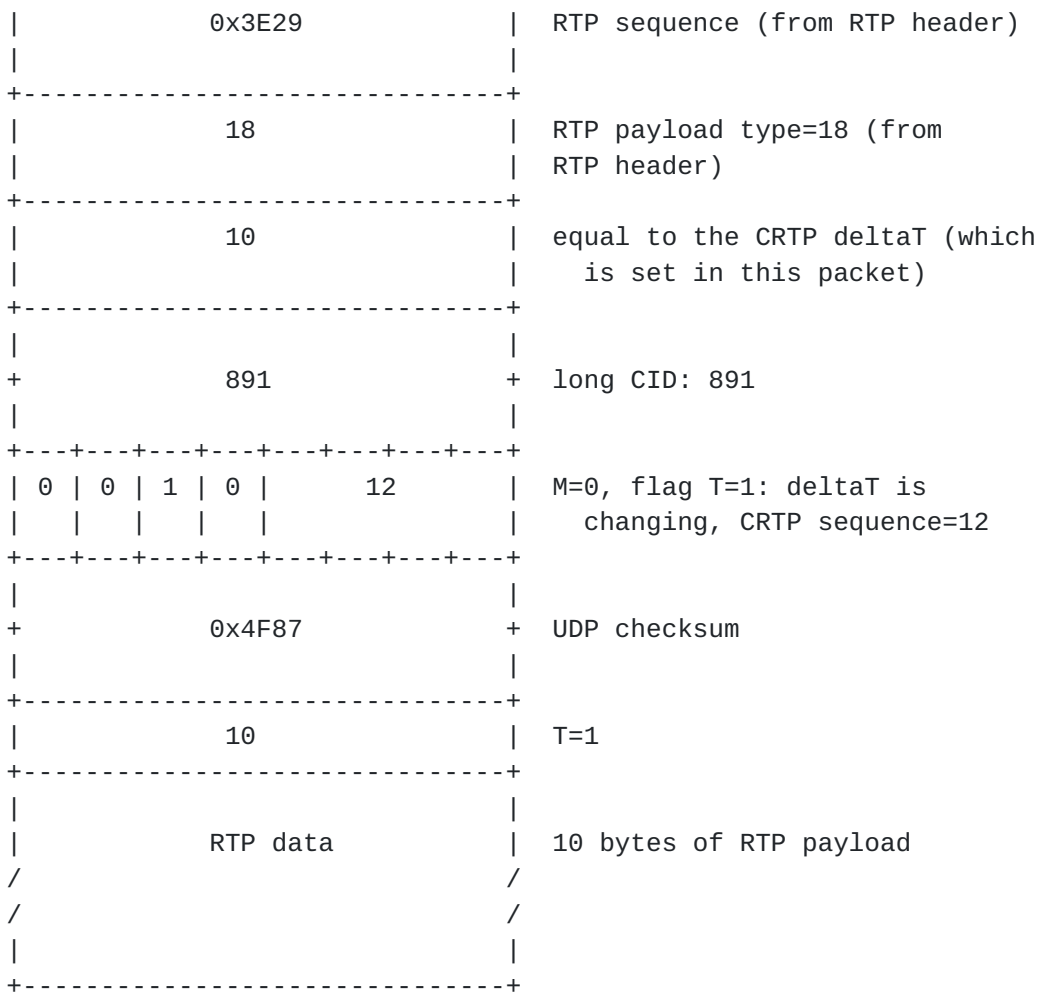
An implementation SHOULD be able to multiplex multiple streams when

sending, and MUST be able to de-multiplex multiple streams when receiving.

4.1. Multiplexed Packet Example

Example of a multiplexed TCRTP packet that contains 2 TCRTP payloads:
packet 1: CRTP packet for CID 124: no special change in state
packet 2: CRTPX packet for CID 891: new deltaT is set, CRTPX fields are present





5. Suggested Mechanisms to Indicate Support of TCRTP

Two mechanisms have been suggested to indicate support of TCRTP between two endpoints:

1. TCRTP negotiation between two hosts
2. Change session announcement protocol to advertise TCRTP as well as RTP.

Method (1) does not require changes to existing session announcement protocols, and could be implemented with little or even no modifications to existing RTP implementations. Its disadvantage is sending unsolicited probes to determine if TCRTP is supported by the remote end and the associated setup delay.

Method (2) requires changes to existing session announcement protocols. Its advantage is faster setup than method (1).

6. New IP Protocol Number

TCRTP requires a new IP protocol number. A formal request to IANA for a new IP protocol number assignment will be made.

7. Security Considerations

This section describes security considerations of TCRTP.

7.1. RTP

Security considerations that apply to RTP and RTP streams also apply to TCRTP streams. These include sniffing packets to observe or listen to communications between two parties.

8. Acknowledgements

The authors would like to thank the authors of RFC2508, Stephen Casner and Van Jacobson, and the authors of RFC2507, Mikael Degermark, Bjorn Nordgren, and Stephen Pink.

The authors would also like to thank Dana Blair, Francois Le Faucheur, Tim Gleeson, Matt Madison, Hussein Salama, Mallik Tatipamula, Mike Thomas, and Herb Wildfeuer.

9. References

[CRTP] S. Casner, V. Jacobson, "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links", RFC2508, February 1999.

[H.245] ITU-T Recommendation H.245 (1998), "Control of communications between Visual Telephone Systems and Terminal Equipment".

[IPHCOMP] M. Degermark, B. Nordgren, S. Pink, "IP Header Compression", RFC2507, February 1999.

[KEYW] S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", RFC2119, BCP 14, March 1997.

[RTP] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC1889, January 1996.

[SDP] M. Handley, V. Jacobson, "SDP: Session Description Protocol", RFC2327, April 1998.

[SGCP] M. Arango, C. Huitema, "Simple Gateway Control Protocol (SGCP)", Internet Draft, Work In Progress, draft-huitema-sgcp-v1-02.txt, Expired.

[SIP] M. Handley, H. Schulzrinne, E. Schooler, J. Rosenberg, "SIP: Session Initiation Protocol", RFC2543, March 1999.

10. Authors' Addresses

Tmima Koren
170 West Tasman Drive
San Jose, CA 95134-1706
United States of America

Phone: +1 408 527 6169
Email: tmima@cisco.com

Patrick Ruddy
3rd Floor, 96 Commercial Street
Edinburgh
EH6 6LX
Scotland

Phone: +44 131 561 3608
Email: pruddy@cisco.com

Bruce Thompson
170 West Tasman Drive
San Jose, CA 95134-1706
United States of America

Phone: +1 408 527 0446
Email: brucet@cisco.com

Alex Tweedly
3 The Square, Stockley Park
Uxbridge, Middlesex
UB11 1BN
United Kingdom

Phone: +44 181 756 8694
Email: agt@cisco.com

Dan Wing
170 West Tasman Drive
San Jose, CA 95134-1706
United States of America

Phone: +1 408 525 5314
Email: dwing@cisco.com

11. Copyright