AVTCore
Internet-Draft
Obsoletes: RFC5285 (if approved)
Intended status: Standards Track
Expires: November 12, 2016

R. Even, Ed.
Huawei Technologies
D. Singer
Apple, Inc.
H. Desineni

May 11, 2016

A General Mechanism for RTP Header Extensions
draft-ietf-avtcore-rfc5285-bis-02.txt

Abstract

This document provides a general mechanism to use the header extension feature of RTP (the Real-Time Transport Protocol). It provides the option to use a small number of small extensions in each RTP packet, where the universe of possible extensions is large and registration is de-centralized. The actual extensions in use in a session are signaled in the setup information for that session.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on November 12, 2016.

Table of Contents

1. Introduction

The RTP specification [RFC3550] provides a capability to extend the RTP header. It defines the header extension format and rules for its use in Section 5.3.1. The existing header extension method permits at most one extension per RTP packet, identified by a 16-bit identifier and a 16-bit length field specifying the length of the header extension in 32-bit words.

This mechanism has two conspicuous drawbacks. First, it permits only one header extension in a single RTP packet. Second, the specification gives no guidance as to how the 16-bit header extension identifiers are allocated to avoid collisions.

This specification removes the first drawback by defining a backward-compatible and extensible means to carry multiple header extension elements in a single RTP packet. It removes the second drawback by defining that these extension elements are named by URIs, defining an IANA registry for extension elements defined in IETF specifications,

and a Session Description Protocol (SDP) method for mapping between the naming URIs and the identifier values carried in the RTP packets.

This header extension applies to RTP/AVP (the Audio/Visual Profile) and its extensions.

This document removes a limitation from RFC5285 that did not allow sending both one byte and two bytes header extensions in the same RTP stream

2. 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].

3. Design Goals

The goal of this design is to provide a simple mechanism whereby multiple identified extensions can be used in RTP packets, without the need for formal registration of those extensions but nonetheless avoiding collision.

This mechanism provides an alternative to the practice of burying associated metadata into the media format bit stream. This has often been done in media data sent over fixed-bandwidth channels. Once this is done, a decoder for the specific media format needs to extract the metadata. Also, depending on the media format, the metadata can be added at the time of encoding the media so that the bit-rate used for the metadata is taken into account. But the metadata can be unknown at that time. Inserting metadata at a later time can cause a decode and re-encode to meet bit-rate requirements.

In some cases, a more appropriate, higher-level mechanism can be available, and if so, it can be used. For cases where a higher-level mechanism is not available, it is better to provide a mechanism at the RTP level than have the metadata be tied to a specific form of media data.

4. Packet Design

4.1. General

The following design is fit into the "header extension" of the RTP extension, as described above.

The presence and format of this header extension and its contents are negotiated or defined out-of-band, such as through signaling (see

below for SDP signaling). The value defined for an RTP extension (defined below for the one-byte and two-byte header forms) is only an architectural constant (e.g., for use by network analyzers); it is the negotiation/definition (e.g., in SDP) that is the definitive indication that this header extension is present.

This specification updates the requirement from the RTP specification that the header extension "is designed so that the header extension MAY be ignored". To be specific, header extensions using this specification SHOULD be used for data that can safely be ignored by the recipient without affecting interoperability, there can be essential header extensions for interoperability and intermediaries SHOULD NOT remove such header extensions. Note that the support of header extension as specified in this recommendation is negotiated. RTP Header extensions MUST NOT be used when the presence of the extension has changed the form or nature of the rest of the packet in a way that is not compatible with the way the stream is signaled (e.g., as defined by the payload type). Valid examples might include metadata that is additional to the usual RTP information, e.g. Audio level from Client to mixer [RFC6464].

The RTP header extension is formed as a sequence of extension elements, with possible padding. Each extension element has a local identifier and a length. The local identifiers MAY be mapped to a larger namespace in the negotiation (e.g., session signaling).

As is good network practice, data SHOULD only be transmitted when needed. The RTP header extension SHOULD only be present in a packet if that packet also contains one or more extension elements, as defined here. An extension element SHOULD only be present in a packet when needed; the signaling setup of extension elements indicates only that those elements can be present in some packets, not that they are in fact present in all (or indeed, any) packets.

Each extension element in a packet has a local identifier (ID) and a length. The local identifiers present in the stream MUST have been negotiated or defined out-of-band. There are no static allocations of local identifiers. Each distinct extension MUST have a unique ID. The value 0 is reserved for padding and MUST NOT be used as a local identifier.

There are two variants of the extension: one-byte and two-byte headers. Since it is expected that (a) the number of extensions in any given RTP session is small and (b) the extensions themselves are small, the one-byte header form is preferred and MUST be supported by all receivers. A stream MUST contain only one-byte or two-byte headers unless it is known that all recipients support mixing, either by offer/answer negotiation (see section 6) or by out-of-band

knowledge. One-byte and two-byte headers MUST NOT be mixed in a single RTP packet. Transmitters SHOULD NOT use the two-byte form when all extensions are small enough for the one-byte header form. A transmitter MAY be aware that an intermediary may add RTP header extensions in this case, the transmitter SHOULD use two-byte form.

A sequence of extension elements, possibly with padding, forms the header extension defined in the RTP specification. There are as many extension elements as fit into the length as indicated in the RTP header extension length. Since this length is signaled in full 32-bit words, padding bytes are used to pad to a 32-bit boundary. The entire extension is parsed byte-by-byte to find each extension element (no alignment is needed), and parsing stops at the earlier of the end of the entire header extension, or in one-byte headers only case, on encountering an identifier with the reserved value of 15.

In both forms, padding bytes have the value of 0 (zero). They MAY be placed between extension elements, if desired for alignment, or after the last extension element, if needed for padding. A padding byte does not supply the ID of an element, nor the length field. When a padding byte is found, it is ignored and the parser moves on to interpreting the next byte.

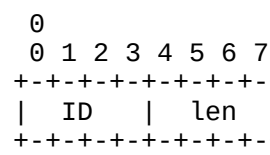
Note carefully that the one-byte header form allows for data lengths between 1 and 16 bytes, by adding 1 to the signaled length value (thus, 0 in the length field indicates 1 byte of data follows). This allows for the important case of 16-byte payloads. This addition is not performed for the two-byte headers, where the length field signals data lengths between 0 and 255 bytes.

Use of RTP header extensions will reduce the efficiency of RTP header compression, since the header extension will be sent uncompressed unless the RTP header compression module is updated to recognize the extension header. If header extensions are present in some packets, but not in others, this can also reduce compression efficiency by requiring an update to the fixed header to be conveyed when header extensions start or stop being sent. The interactions of the RTP header extension and header compression is explored further in [RFC2508] and [RFC3095].

4.2. One-Byte Header

In the one-byte header form of extensions, the 16-bit value REQUIRED by the RTP specification for a header extension, labeled in the RTP specification as "defined by profile", MUST have the fixed bit pattern 0xBEDE (the first version of this specification was written on the feast day of the Venerable Bede).

Each extension element MUST starts with a byte containing an ID and a length:

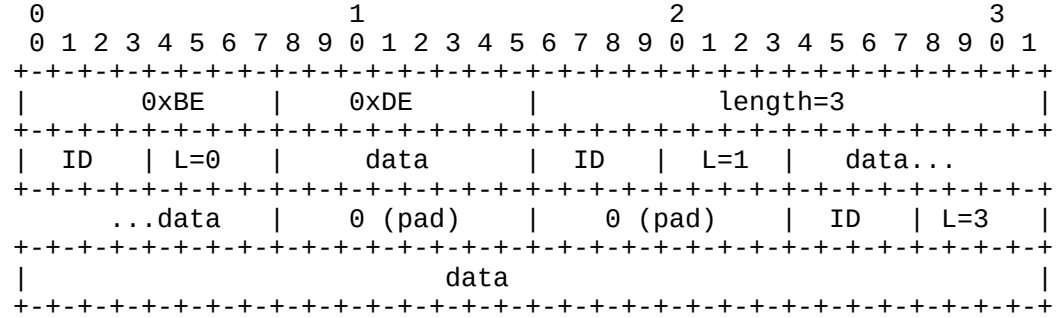


The 4-bit ID is the local identifier of this element in the range 1-14 inclusive. In the signaling section, this is referred to as the valid range.

The local identifier value 15 is reserved for future extension and MUST NOT be used as an identifier. If the ID value 15 is encountered, its length field MUST be ignored, processing of the entire extension MUST terminate at that point, and only the extension elements present prior to the element with ID 15 SHOULD be considered.

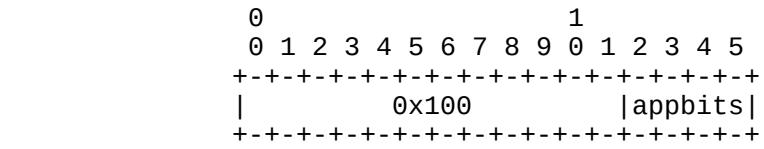
The 4-bit length is the number minus one of data bytes of this header extension element following the one-byte header. Therefore, the value zero in this field indicates that one byte of data follows, and a value of 15 (the maximum) indicates element data of 16 bytes. (This permits carriage of 16-byte values, which is a common length of labels and identifiers, while losing the possibility of zero-length values -- which would often be padded anyway.)

An example header extension, with three extension elements, and some padding follows:



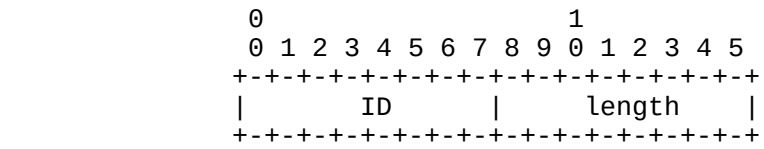
4.3. Two-Byte Header

In the two-byte header form, the 16-bit value defined by the RTP specification for a header extension, labeled in the RTP specification as "defined by profile", is defined as shown below.



The appbits field is 4 bits that are application-dependent and MAY be defined to be any value or meaning, and are outside the scope of this specification. For the purposes of signaling, this field is treated as a special extension value assigned to the local identifier 256. If no extension has been specified through configuration or signaling for this local identifier value 256, the appbits field SHOULD be set to all 0s by the sender and MUST be ignored by the receiver.

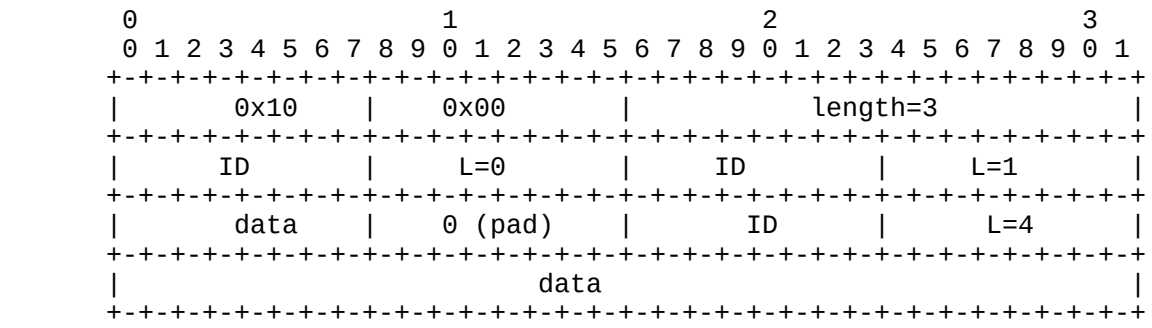
Each extension element starts with a byte containing an ID and a byte containing a length:



The 8-bit ID is the local identifier of this element in the range 1-255 inclusive. In the signaling section, the range 1-256 is referred to as the valid range, with the values 1-255 referring to extension elements, and the value 256 referring to the 4-bit field ’appbits’ (above).

The 8-bit length field is the length of extension data in bytes not including the ID and length fields. The value zero indicates there is no data following.

An example header extension, with three extension elements, and some padding follows:



5. SDP Signaling Design

The indication of the presence of this extension, and the mapping of local identifiers used in the header extension to a larger namespace, MUST be performed out-of-band, for example, as part of a SIP offer/answer exchange using SDP. This section defines such signaling in SDP.

A usable mapping MUST use IDs in the valid range, and each ID in this range MUST be used only once for each media (or only once if the mappings are session level). Mappings that do not conform to these rules MAY be presented, for instance, during offer/answer negotiation as described in the next section, but remapping to conformant values is necessary before they can be applied.

Each extension is named by a URI. That URI MUST be absolute, and precisely identifies the format and meaning of the extension. URIs that contain a domain name SHOULD also contain a month-date in the form mmyyyy. The definition of the element and assignment of the URI MUST have been authorized by the owner of the domain name on or very close to that date. (This avoids problems when domain names change ownership.) If the resource or document defines several extensions, then the URI MUST identify the actual extension in use, e.g., using a fragment or query identifier (characters after a '#' or '?' in the URI).

Rationale: the use of URIs provides for a large, unallocated space, and gives documentation on the extension. The URIs do not have to be de-referencable, in order to permit confidential or experimental use, and to cover the case when extensions continue to be used after the organization that defined them ceases to exist.

An extension URI with the same attributes MUST NOT appear more than once applying to the same stream, i.e., at session level or in the declarations for a single stream at media level. (The same extension can, of course, be used for several streams, and can appear differently parameterized for the same stream.)

For extensions defined in RFCs, the URI used SHOULD be a URN starting "urn:ietf:params:rtp-hdrext:" and followed by a registered, descriptive name.

The registration requirements are detailed in the IANA Considerations section, below.

An example (this is only an example), where 'avt-example-metadata' is the hypothetical name of a header extension, might be:

urn:ietf:params:rtp-hdrext:avt-example-metadata

An example name not from the IETF (this is only an example) might be:

http://example.com/082005/ext.htm#example-metadata

The mapping MAY be provided per media stream (in the media-level section(s) of SDP, i.e., after an "m=" line) or globally for all streams (i.e., before the first "m=" line, at session level). The definitions MUST be either all session level or all media level; it is not permitted to mix the two styles. In addition, as noted above, the IDs used MUST be unique for each stream type for a given media, or for the session for session-level declarations.

Each local identifier potentially used in the stream is mapped to a string using an attribute of the form:

a=extmap:<value>["/"<direction>] <URI> <extensionattributes>

where <URI> is a URI, as above, <value> is the local identifier (ID) of this extension and is an integer in the valid range inclusive (0 is reserved for padding in both forms, and 15 is reserved in the one-byte header form, as noted above), and <direction> is one of "sendonly", "recvonly", "sendrecv", or "inactive" (without the quotes).

The formal BNF syntax is presented in a later section of this specification.

Example:

a=extmap:1 http://example.com/082005/ext.htm#ttime

a=extmap:2/sendrecv http://example.com/082005/ext.htm#xmeta short

When SDP signaling is used for the RTP session, it is the presence of the 'extmap' attribute(s) that is diagnostic that this style of header extensions is used, not the magic number indicated above.

6. SDP Signaling for support of mixed one byte and two bytes header extensions.

In order to allow for backward interoperability with systems that do not support mixing of one byte and two bytes header extensions this document defines the "a=extmap-allow-mixed" Session Description Protocol (SDP) [RFC4566] attribute to indicate if the participant is capable of supporting this new mode. The attribute takes no value. This attribute can be used at the session or media levels. A participant that proposes the use of this mode SHALL itself support the reception of mixed one byte and two bytes header extensions.

The negotiation for mixed one byte and two bytes extension MUST be negotiated in offer/answer [RFC3264]. In the absence of negotiation using offer/answer, mixed headers MUST NOT occur unless the transmitter has some (out of band) knowledge that all potential recipients support this mode.

The formal definition of this attribute is:

Name: extmap-allow-mixed

Value:

Usage Level: session, media

Charset Dependent: no

Example:

a=extmap-allow-mixed

When doing SDP Offer/Answer [RFC3264] an offering client that wishes to use both one and two bytes extensions MUST include the attribute "a= extmap-allow-mixed " in the SDP offer. If "a= extmap-allow-mixed " is present in the offer SDP, the answerer that supports this mode and wishes to use it SHALL include the "a=extmap-allow-mixed " attribute in the answer. In cases the answer has been excluded, neither clients SHALL use mixed one bytes and two bytes extensions in the same RTP stream.

7. Offer/Answer

The simple signaling described above MAY be enhanced in an offer/answer context, to permit:

- asymmetric behavior (extensions sent in only one direction),
- the offer of mutually exclusive alternatives, or
- the offer of more extensions than can be sent in a single session.

A direction attribute MAY be included in an extmap; without it, the direction implicitly inherits, of course, from the stream direction, or is "sendrecv" for session-level attributes or extensions of "inactive" streams. The direction MUST be one of "sendonly", "recvonly", "sendrecv", or "inactive". A "sendonly" direction indicates an ability to send; a "recvonly" direction indicates a desire to receive; a "sendrecv" direction indicates both. An "inactive" direction indicates neither, but later re-negotiation MAY make an extension active.

Extensions, with their directions, MAY be signaled for an "inactive" stream. It is an error to use an extension direction incompatible with the stream direction (e.g., a "sendonly" attribute for a "recvonly" stream).

If an offer or answer contains session-level mappings (and hence no media-level mappings), and different behavior is desired for each stream, then the entire set of extension map declarations MAY be moved into the media-level section(s) of the SDP. (Note that this specification does not permit mixing global and local declarations, to make identifier management easier.)

If an extension map is offered as "sendrecv", explicitly or implicitly, and asymmetric behavior is desired, the SDP MAY be modified to modify or add direction qualifiers for that extension.

If an extension is marked as "sendonly" and the answerer desires to receive it, the extension MUST be marked as "recvonly" in the SDP answer. An answerer that has no desire to receive the extension or does not understand the extension SHOULD remove it from the SDP answer.

If an extension is marked as "recvonly" and the answerer desires to send it, the extension MUST be marked as "sendonly" in the SDP answer. An answerer that has no desire to, or is unable to, send the extension SHOULD remove it from the SDP answer.

Local identifiers in the valid range inclusive in an offer or answer MUST NOT be used more than once per media section (including the session-level section). A session update MAY change the direction qualifiers of extensions under use. A session update MAY add or remove extension(s). Identifiers values in the valid range MUST NOT be altered (remapped).

Note that, under this rule, the same local identifier cannot be used for two extensions for the same media, even when one is "sendonly" and the other "recvonly", as it would then be impossible to make either of them sendrecv (since re-numbering is not permitted either).

If a party wishes to offer mutually exclusive alternatives, then multiple extensions with the same identifier in the (unusable) range 4096-4351 MAY be offered; the answerer SHOULD select at most one of the offered extensions with the same identifier, and remap it to a free identifier in the valid range, for that extension to be usable.

Similarly, if more extensions are offered than can be fit in the valid range, identifiers in the range 4096-4351 MAY be offered; the answerer SHOULD choose those that are desired, and remap them to a free identifier in the valid range.

It is always allowed to place the offered identifier value "as is" in the SDP answer (for example, due to lack of a free identifier value in the valid range). Extensions with an identifier outside the valid range MUST NOT, of course, be used. If needed, the offerer or answerer can update the session to make space for such an extension.

Rationale: the range 4096-4351 for these negotiation identifiers is deliberately restricted to allow expansion of the range of valid identifiers in future.

Either party MAY include extensions in the stream other than those negotiated, or those negotiated as "inactive", for example, for the benefit of intermediate nodes. Only extensions that appeared with an identifier in the valid range in SDP originated by the sender can be sent.

Example (port numbers, RTP profiles, payload IDs and rtpmaps, etc. all omitted for brevity):

The offer:

```
a=extmap:1 URI-toffset
a=extmap:14 URI-obscure
a=extmap:4096 URI-gps-string
a=extmap:4096 URI-gps-binary
a=extmap:4097 URI-frametype
m=video
a=sendrecv
m=audio
a=sendrecv
```

The answerer is interested in receiving GPS in string format only on video, but cannot send GPS at all. It is not interested in transmission offsets on audio, and does not understand the URI-obscure extension. It therefore moves the extensions from session level to media level, and adjusts the declarations:

```
m=video
a=sendrecv
a=extmap:1 URI-toffset
a=extmap:2/recvonly URI-gps-string
a=extmap:3 URI-frametype
m=audio
a=sendrecv
a=extmap:1/sendonly URI-toffset
```

8. BNF Syntax

The syntax definition below uses ABNF according to [RFC5234]. The syntax element 'URI' is defined in [RFC3986] (only absolute URIs are permitted here). The syntax element 'extmap' is an attribute as defined in [RFC4566], i.e., "a=" precedes the extmap definition. Specific extensionattributes are defined by the specification that defines a specific extension name; there can be several.

```
extmap = mapentry SP extensionname [SP extensionattributes]

extensionname = URI

direction = "sendonly" / "recvonly" / "sendrecv" / "inactive"

mapentry = "extmap:" 1*5DIGIT ["/" direction]

extensionattributes = byte-string

URI = <Defined in RFC 3986>

byte-string = <Defined in RFC 4566>

SP = <Defined in RFC 5234>

DIGIT = <Defined in RFC 5234>
```

9. Security Considerations

This document defines only a place to transmit information; the security implications of each of the extensions MUST be discussed with those extensions.

Header extensions have the same security coverage as the RTP header itself. When Secure Real-time Transport Protocol (SRTP) [RFC3711] is used to protect RTP sessions, the RTP payload can be both encrypted and integrity protected, while the RTP header is either unprotected or integrity protected. RTP header extensions can carry sensitive information for which participants in multimedia sessions want confidentiality. RFC6904 [RFC6904] provides a mechanism, extending the mechanisms of SRTP, to selectively encrypt RTP header extensions in SRTP.

10. IANA Considerations

This document updates the IANA consideration to reference this document and adds a new SDP attribute in section 10.3

Note to IANA : change RFCxxxx to this RFC number and remove the note.

10.1. Identifier Space for IANA to Manage

The mapping from the naming URI form to a reference to a specification is managed by IANA. Insertion into this registry is under the requirements of "Expert Review" as defined in [RFC5226].

The IANA will also maintain a server that contains all of the registered elements in a publicly accessible space.

Here is the formal declaration to comply with the IETF URN Sub-namespace specification [RFC3553].

- Registry name: RTP Compact Header Extensions
- Specification: RFC 5285 and RFCs updating RFC 5285.
- Information required:

  A. The desired extension naming URI

  B. A formal reference to the publicly available specification

  C. A short phrase describing the function of the extension

  D. Contact information for the organization or person making the registration

  For extensions defined in RFCs, the URI SHOULD be of the form urn:ietf:params:rtp-hdrext:, and the formal reference is the RFC number of the RFC documenting the extension.

- Review process: Expert review is REQUIRED. The expert review SHOULD check the following requirements:

  1. that the specification is publicly available;

  2. that the extension complies with the requirements of RTP and this specification, for extensions (notably, that the stream is still decodable if the extension is ignored or not recognized);

  3. that the extension specification is technically consistent (in itself and with RTP), complete, and comprehensible;

  4. that the extension does not duplicate functionality in existing IETF specifications (including RTP itself), or other extensions already registered;

  5. that the specification contains a security analysis regarding the content of the header extension;

  6. that the extension is generally applicable, for example point-to-multipoint safe, and the specification correctly describes limitations if they exist; and

7. that the suggested naming URI form is appropriately chosen and unique.

- Size and format of entries: a mapping from a naming URI string to a formal reference to a publicly available specification, with a descriptive phrase and contact information.

- Initial assignments: none.

10.2. Registration of the SDP extmap Attribute

This section contains the information requested by [RFC4566] for an SDP attribute.

- contact name, email address, and telephone number:

  D. Singer
  singer@apple.com
  +1 408-974-3162

- attribute name (as it will appear in SDP): extmap

- long-form attribute name in English: generic header extension map definition

- type of attribute (session level, media level, or both): both

- whether the attribute value is subject to the charset attribute: not subject to the charset attribute

- a one-paragraph explanation of the purpose of the attribute: This attribute defines the mapping from the extension numbers used in packet headers into extension names as documented in specifications and appropriately registered.

- a specification of appropriate attribute values for this attribute: see RFC 5285.

10.3. Registration of the SDP Attribute

The IANA is requested to register one new SDP attribute:

```
SDP Attribute ("att-field"):
    Attribute name:     extmap-allow-mixed
    Long form:          One and Two bytes mixed mode
    Type of name:       att-field
    Type of attribute:  Media or session level
    Subject to charset: No
    Purpose:            Negotiate the use of One and Two bytes
                        in the same RTP stream.
    Reference:          [RFCXXXX]
    Values:             None
```

11. Acknowledgments

Both Brian Link and John Lazzaro provided helpful comments on an initial draft of this document. Colin Perkins was helpful in reviewing and dealing with the details. The use of URNs for IETF-defined extensions was suggested by Jonathan Lennox, and Pete Cordell was instrumental in improving the padding wording. Dave Oran provided feedback and text in the review. Mike Dolan contributed the two-byte header form. Magnus Westerlund and Tom Taylor were instrumental in managing the registration text.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2508] Casner, S. and V. Jacobson, "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links", RFC 2508, February 1999.

[RFC3095] Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H., Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T., and H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, July 2001.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3553] Mealling, M., Masinter, L., Hardie, T., and G. Klyne, "An IETF URN Sub-namespace for Registered Protocol Parameters", BCP 73, RFC 3553, June 2003.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC6904] Lennox, J., "Encryption of Header Extensions in the Secure Real-time Transport Protocol (SRTP)", RFC 6904, DOI 10.17487/RFC6904, April 2013, <http://www.rfc-editor.org/info/rfc6904>.

12.2. Informative References

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[RFC6464] Lennox, J., Ed., Ivov, E., and E. Marocco, "A Real-time Transport Protocol (RTP) Header Extension for Client-to-Mixer Audio Level Indication", RFC 6464, DOI 10.17487/RFC6464, December 2011, <http://www.rfc-editor.org/info/rfc6464>.

Authors' Addresses

Roni Even (editor)
Huawei Technologies
Shabazi 12A
Tel Aviv
Israel

Email: Roni.even@mail01.huawei.com

David Singer
Apple, Inc.
1 Infinite Loop
Cupertino, CA 95014
USA

Phone: +1 408 996 1010
Email: singer@apple.com
URI: http://www.apple.com/quicktime

Harikishan Desineni
10001 Pacific Heights Blvd
San Diego, CA 92121
USA

Phone: +1 858 845 8996
Email: hdesinen@quicinc.com

AVTCORE Working Group
Internet-Draft
Updates: 3550 (if approved)
Intended status: Standards Track
Expires: December 13, 2016

C. Perkins
University of Glasgow
V. Singh
callstats.io

June 11, 2016

Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions
draft-ietf-avtcore-rtp-circuit-breakers-16

Abstract

The Real-time Transport Protocol (RTP) is widely used in telephony, video conferencing, and telepresence applications. Such applications are often run on best-effort UDP/IP networks. If congestion control is not implemented in these applications, then network congestion can lead to uncontrolled packet loss, and a resulting deterioration of the user's multimedia experience. The congestion control algorithm acts as a safety measure, stopping RTP flows from using excessive resources, and protecting the network from overload. At the time of this writing, however, while there are several proprietary solutions, there is no standard algorithm for congestion control of interactive RTP flows.

This document does not propose a congestion control algorithm. It instead defines a minimal set of RTP circuit breakers: conditions under which an RTP sender needs to stop transmitting media data, to protect the network from excessive congestion. It is expected that, in the absence of long-lived excessive congestion, RTP applications running on best-effort IP networks will be able to operate without triggering these circuit breakers. To avoid triggering the RTP circuit breaker, any standards-track congestion control algorithms defined for RTP will need to operate within the envelope set by these RTP circuit breaker algorithms.

Status of This Memo

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 13, 2016.

Table of Contents

1. Introduction

The Real-time Transport Protocol (RTP) [RFC3550] is widely used in voice-over-IP, video teleconferencing, and telepresence systems. Many of these systems run over best-effort UDP/IP networks, and can

suffer from packet loss and increased latency if network congestion occurs. Designing effective RTP congestion control algorithms, to adapt the transmission of RTP-based media to match the available network capacity, while also maintaining the user experience, is a difficult but important problem. Many such congestion control and media adaptation algorithms have been proposed, but to date there is no consensus on the correct approach, or even that a single standard algorithm is desirable.

This memo does not attempt to propose a new RTP congestion control algorithm. Instead, we propose a small set of RTP circuit breakers: mechanisms that terminate RTP flows in conditions under which there is general agreement that serious network congestion is occurring. The RTP circuit breakers proposed in this memo are a specific instance of the general class of network transport circuit breakers [I-D.ietf-tsvwg-circuit-breaker], designed to act as a protection mechanism of last resort to avoid persistent excessive congestion. To avoid triggering the RTP circuit breaker, any standards-track congestion control algorithms defined for RTP will need to operate within the envelope set by the RTP circuit breaker algorithms defined by this memo.

2. Background

We consider congestion control for unicast RTP traffic flows. This is the problem of adapting the transmission of an audio/visual data flow, encapsulated within an RTP transport session, from one sender to one receiver, so that it does not use more capacity than is available along the network path. Such adaptation needs to be done in a way that limits the disruption to the user experience caused by both packet loss and excessive rate changes. Congestion control for multicast flows is outside the scope of this memo. Multicast traffic needs different solutions, since the available capacity estimator for a group of receivers will differ from that for a single receiver, and because multicast congestion control has to consider issues of fairness across groups of receivers that do not apply to unicast flows.

Congestion control for unicast RTP traffic can be implemented in one of two places in the protocol stack. One approach is to run the RTP traffic over a congestion controlled transport protocol, for example over TCP, and to adapt the media encoding to match the dictates of the transport-layer congestion control algorithm. This is safe for the network, but can be suboptimal for the media quality unless the transport protocol is designed to support real-time media flows. We do not consider this class of applications further in this memo, as their network safety is guaranteed by the underlying transport.

Alternatively, RTP flows can be run over a non-congestion controlled transport protocol, for example UDP, performing rate adaptation at the application layer based on RTP Control Protocol (RTCP) feedback. With a well-designed, network-aware, application, this allows highly effective media quality adaptation, but there is potential to cause persistent congestion in the network if the application does not adapt its sending rate in a timely and effective manner. We consider this class of applications in this memo.

Congestion control relies on monitoring the delivery of a media flow, and responding to adapt the transmission of that flow when there are signs that the network path is congested. Network congestion can be detected in one of three ways: 1) a receiver can infer the onset of congestion by observing an increase in one-way delay caused by queue build-up within the network; 2) if Explicit Congestion Notification (ECN) [RFC3168] is supported, the network can signal the presence of congestion by marking packets using ECN Congestion Experienced (CE) marks (this could potentially be augmented by mechanisms such as ConEX [RFC7713], or other future protocol extensions for network signalling of congestion); or 3) in the extreme case, congestion will cause packet loss that can be detected by observing a gap in the received RTP sequence numbers.

Once the onset of congestion is observed, the receiver has to send feedback to the sender to indicate that the transmission rate needs to be reduced. How the sender reduces the transmission rate is highly dependent on the media codec being used, and is outside the scope of this memo.

There are several ways in which a receiver can send feedback to a media sender within the RTP framework:

- The base RTP specification [RFC3550] defines RTCP Reception Report (RR) packets to convey reception quality feedback information, and Sender Report (SR) packets to convey information about the media transmission. RTCP SR packets contain data that can be used to reconstruct media timing at a receiver, along with a count of the total number of octets and packets sent. RTCP RR packets report on the fraction of packets lost in the last reporting interval, the cumulative number of packets lost, the highest sequence number received, and the inter-arrival jitter. The RTCP RR packets also contain timing information that allows the sender to estimate the network round trip time (RTT) to the receivers. RTCP reports are sent periodically, with the reporting interval being determined by the number of SSRCs used in the session and a configured session bandwidth estimate (the number of synchronisation sources (SSRCs) used is usually two in a unicast session, one for each participant, but can be greater if the participants send multiple

media streams). The interval between reports sent from each receiver tends to be on the order of a few seconds on average, although it varies with the session bandwidth, and sub-second reporting intervals are possible in high bandwidth sessions, and it is randomised to avoid synchronisation of reports from multiple receivers. RTCP RR packets allow a receiver to report ongoing network congestion to the sender. However, if a receiver detects the onset of congestion part way through a reporting interval, the base RTP specification contains no provision for sending the RTCP RR packet early, and the receiver has to wait until the next scheduled reporting interval.

- The RTCP Extended Reports (XR) [RFC3611] allow reporting of more complex and sophisticated reception quality metrics, but do not change the RTCP timing rules. RTCP extended reports of potential interest for congestion control purposes are the extended packet loss, discard, and burst metrics [RFC3611], [RFC7002], [RFC7097], [RFC7003], [RFC6958]; and the extended delay metrics [RFC6843], [RFC6798]. Other RTCP Extended Reports that could be helpful for congestion control purposes might be developed in future.

- Rapid feedback about the occurrence of congestion events can be achieved using the Extended RTP Profile for RTCP-Based Feedback (RTP/AVPF) [RFC4585] (or its secure variant, RTP/SAVPF [RFC5124]) in place of the RTP/AVP profile [RFC3551]. This modifies the RTCP timing rules to allow RTCP reports to be sent early, in some cases immediately, provided the RTCP transmission rate keeps within its bandwidth allocation. It also defines transport-layer feedback messages, including negative acknowledgements (NACKs), that can be used to report on specific congestion events. RTP Codec Control Messages [RFC5104] extend the RTP/AVPF profile with additional feedback messages that can be used to influence that way in which rate adaptation occurs, but do not further change the dynamics of how rapidly feedback can be sent. Use of the RTP/AVPF profile is dependent on signalling.

- Finally, Explicit Congestion Notification (ECN) for RTP over UDP [RFC6679] can be used to provide feedback on the number of packets that received an ECN Congestion Experienced (CE) mark. This RTCP extension builds on the RTP/AVPF profile to allow rapid congestion feedback when ECN is supported.

In addition to these mechanisms for providing feedback, the sender can include an RTP header extension in each packet to record packet transmission times [RFC5450]. Accurate transmission timestamps can be helpful for estimating queuing delays, to get an early indication of the onset of congestion.

Taken together, these various mechanisms allow receivers to provide feedback on the senders when congestion events occur, with varying degrees of timeliness and accuracy. The key distinction is between systems that use only the basic RTCP mechanisms, without RTP/AVPF rapid feedback, and those that use the RTP/AVPF extensions to respond to congestion more rapidly.

3. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119]. This interpretation of these key words applies only when written in ALL CAPS. Mixed- or lower-case uses of these key words are not to be interpreted as carrying special significance in this memo.

The definition of the RTP circuit breaker is specified in terms of the following variables:

- Td is the deterministic RTCP reporting interval, as defined in Section 6.3.1 of [RFC3550].

- Tdr is the sender's estimate of the deterministic RTCP reporting interval, Td, calculated by a receiver of the data it is sending. Tdr is not known at the sender, but can be estimated by executing the algorithm in Section 6.2 of [RFC3550] using the average RTCP packet size seen at the sender, the number of members reported in the receiver's SR/RR report blocks, and whether the receiver is sending SR or RR packets. Tdr is recalculated when each new RTCP SR/RR report is received, but the media timeout circuit breaker (see Section 4.2) is only reconsidered when Tdr increases.

- Tr is the network round-trip time, calculated by the sender using the algorithm in Section 6.4.1 of [RFC3550] and smoothed using an exponentially weighted moving average as Tr = (0.8 * Tr) + (0.2 * Tr_new) where Tr_new is the latest RTT estimate obtained from an RTCP report. The weight is chosen so old estimates decay over k intervals.

- k is the non-reporting threshold (see Section 4.2).

- Tf is the media framing interval at the sender. For applications sending at a constant frame rate, Tf is the inter-frame interval. For applications that switch between a small set of possible frame rates, for example when sending speech with comfort noise, where comfort noise frames are sent less often than speech frames, Tf is set to the longest of the inter-frame intervals of the different frame rates. For applications that send periodic frames but

dynamically vary their frame rate, Tf is set to the largest inter-frame interval used in the last 10 seconds. For applications that send less than one frame every 10 seconds, or that have no concept of periodic frames (e.g., text conversation [RFC4103], or pointer events [RFC2862]), Tf is set to the time interval since the previous frame when each frame is sent.

- G is the frame group size. That is, the number of frames that are coded together based on a particular sending rate setting. If the codec used by the sender can change its rate on each frame, G = 1; otherwise G is set to the number of frames before the codec can adjust to the new rate. For codecs that have the concept of a group-of-pictures (GoP), G is likely the GoP length.

- T_rr_interval is the minimal interval between RTCP reports, as defined in Section 3.4 of [RFC4585]; it is only meaningful for implementations of RTP/AVPF profile [RFC4585] or the RTP/SAVPF profile [RFC5124].

- X is the estimated throughput a TCP connection would achieve over a path, in bytes per second.

- s is the size of RTP packets being sent, in bytes. If the RTP packets being sent vary in size, then the average size over the packet comprising the last 4 * G frames MUST be used (this is intended to be comparable to the four loss intervals used in [RFC5348]).

- p is the loss event rate, between 0.0 and 1.0, that would be seen by a TCP connection over a particular path. When used in the RTP congestion circuit breaker, this is approximated as described in Section 4.3.

- t_RTO is the retransmission timeout value that would be used by a TCP connection over a particular path, in seconds. This MUST be approximated using t_RTO = 4 * Tr when used as part of the RTP congestion circuit breaker.

- b is the number of packets that are acknowledged by a single TCP acknowledgement. Following [RFC5348], it is RECOMMENDED that the value b = 1 is used as part of the RTP congestion circuit breaker.

4. RTP Circuit Breakers for Systems Using the RTP/AVP Profile

The feedback mechanisms defined in [RFC3550] and available under the RTP/AVP profile [RFC3551] are the minimum that can be assumed for a baseline circuit breaker mechanism that is suitable for all unicast applications of RTP. Accordingly, for an RTP circuit breaker to be

useful, it needs to be able to detect that an RTP flow is causing excessive congestion using only basic RTCP features, without needing RTCP XR feedback or the RTP/AVPF profile for rapid RTCP reports.

RTCP is a fundamental part of the RTP protocol, and the mechanisms described here rely on the implementation of RTCP. Implementations that claim to support RTP, but that do not implement RTCP, will be unable to use the circuit breaker mechanisms described in this memo. Such implementations SHOULD NOT be used on networks that might be subject to congestion unless equivalent mechanisms are defined using some non-RTCP feedback channel to report congestion and signal circuit breaker conditions.

The RTCP timeout circuit breaker (Section 4.1) will trigger if an implementation of this memo attempts to interwork with an endpoint that does not support RTCP. Implementations that sometimes need to interwork with endpoints that do not support RTCP need to disable the RTP circuit breakers if they don’t receive some confirmation via signalling that the remote endpoint implements RTCP (the presence of an SDP "a=rtcp:" attribute in an answer might be such an indication). The RTP Circuit Breaker SHOULD NOT be disabled on networks that might be subject to congestion, unless equivalent mechanisms are defined using some non-RTCP feedback channel to report congestion and signal circuit breaker conditions [I-D.ietf-tsvwg-circuit-breaker].

Three potential congestion signals are available from the basic RTCP SR/RR packets and are reported for each SSRC in the RTP session:

1. The sender can estimate the network round-trip time once per RTCP reporting interval, based on the contents and timing of RTCP SR and RR packets.

2. Receivers report a jitter estimate (the statistical variance of the RTP data packet inter-arrival time) calculated over the RTCP reporting interval. Due to the nature of the jitter calculation ([RFC3550], section 6.4.4), the jitter is only meaningful for RTP flows that send a single data packet for each RTP timestamp value (i.e., audio flows, or video flows where each packet comprises one video frame).

3. Receivers report the fraction of RTP data packets lost during the RTCP reporting interval, and the cumulative number of RTP packets lost over the entire RTP session.

These congestion signals limit the possible circuit breakers, since they give only limited visibility into the behaviour of the network.

RTT estimates are widely used in congestion control algorithms, as a proxy for queuing delay measures in delay-based congestion control or to determine connection timeouts. RTT estimates derived from RTCP SR and RR packets sent according to the RTP/AVP timing rules are too infrequent to be useful for congestion control, and don't give enough information to distinguish a delay change due to routing updates from queuing delay caused by congestion. Accordingly, we cannot use the RTT estimate alone as an RTP circuit breaker.

Increased jitter can be a signal of transient network congestion, but in the highly aggregated form reported in RTCP RR packets, it offers insufficient information to estimate the extent or persistence of congestion. Jitter reports are a useful early warning of potential network congestion, but provide an insufficiently strong signal to be used as a circuit breaker.

The remaining congestion signals are the packet loss fraction and the cumulative number of packets lost. If considered carefully, and over an appropriate time frame to distinguish transient problems from long term issues [I-D.ietf-tsvwg-circuit-breaker], these can be effective indicators that persistent excessive congestion is occurring in networks where packet loss is primarily due to queue overflows, although loss caused by non-congestive packet corruption can distort the result in some networks. TCP congestion control [RFC5681] intentionally tries to fill the router queues, and uses the resulting packet loss as congestion feedback. An RTP flow competing with TCP traffic will therefore expect to see a non-zero packet loss fraction, and some variation in queuing latency, in normal operation when sharing a path with other flows, that needs to be accounted for when determining the circuit breaker threshold [I-D.ietf-tsvwg-circuit-breaker]. This behaviour of TCP is reflected in the congestion circuit breaker below, and will affect the design of any RTP congestion control protocol.

Two packet loss regimes can be observed: 1) RTCP RR packets show a non-zero packet loss fraction, while the extended highest sequence number received continues to increment; and 2) RR packets show a loss fraction of zero, but the extended highest sequence number received does not increment even though the sender has been transmitting RTP data packets. The former corresponds to the TCP congestion avoidance state, and indicates a congested path that is still delivering data; the latter corresponds to a TCP timeout, and is most likely due to a path failure. A third condition is that data is being sent but no RTCP feedback is received at all, corresponding to a failure of the reverse path. We derive circuit breaker conditions for these loss regimes in the following.

4.1. RTP/AVP Circuit Breaker #1: RTCP Timeout

An RTCP timeout can occur when RTP data packets are being sent, but there are no RTCP reports returned from the receiver. This is either due to a failure of the receiver to send RTCP reports, or a failure of the return path that is preventing those RTCP reporting from being delivered. In either case, it is not safe to continue transmission, since the sender has no way of knowing if it is causing congestion.

An RTP sender that has not received any RTCP SR or RTCP RR packets reporting on the SSRC it is using, for a time period of at least three times its deterministic RTCP reporting interval, Td, without the randomization factor, and using the fixed minimum interval of Tmin=5 seconds, SHOULD cease transmission (see Section 4.5). The rationale for this choice of timeout is as described in Section 6.2 of [RFC3550] ("so that implementations which do not use the reduced value for transmitting RTCP packets are not timed out by other participants prematurely"), as updated by Section 6.1.4 of [I-D.ietf-avtcore-rtp-multi-stream] to account for the use of the RTP/AVPF profile [RFC4585] or the RTP/SAVPF profile [RFC5124].

To reduce the risk of premature timeout, implementations SHOULD NOT configure the RTCP bandwidth such that Td is larger than 5 seconds. Similarly, implementations that use the RTP/AVPF profile [RFC4585] or the RTP/SAVPF profile [RFC5124] SHOULD NOT configure T_rr_interval to values larger than 4 seconds (the reduced limit for T_rr_interval follows Section 6.1.3 of [I-D.ietf-avtcore-rtp-multi-stream]).

The choice of three RTCP reporting intervals as the timeout is made following Section 6.3.5 of RFC 3550 [RFC3550]. This specifies that participants in an RTP session will timeout and remove an RTP sender from the list of active RTP senders if no RTP data packets have been received from that RTP sender within the last two RTCP reporting intervals. Using a timeout of three RTCP reporting intervals is therefore large enough that the other participants will have timed out the sender if a network problem stops the data packets it is sending from reaching the receivers, even allowing for loss of some RTCP packets.

If a sender is transmitting a large number of RTP media streams, such that the corresponding RTCP SR or RR packets are too large to fit into the network MTU, the receiver will generate RTCP SR or RR packets in a round-robin manner. In this case, the sender SHOULD treat receipt of an RTCP SR or RR packet corresponding to any SSRC it sent on the same 5-tuple of source and destination IP address, port, and protocol, as an indication that the receiver and return path are working, preventing the RTCP timeout circuit breaker from triggering.

4.2. RTP/AVP Circuit Breaker #2: Media Timeout

If RTP data packets are being sent, but the RTCP SR or RR packets reporting on that SSRC indicate a non-increasing extended highest sequence number received, this is an indication that those RTP data packets are not reaching the receiver. This could be a short-term issue affecting only a few RTP packets, perhaps caused by a slow to open firewall or a transient connectivity problem, but if the issue persists, it is a sign of a more ongoing and significant problem (a "media timeout").

The time needed to declare a media timeout depends on the parameters Tdr, Tr, Tf, and on the non-reporting threshold k. The value of k is chosen so that when Tdr is large compared to Tr and Tf, receipt of at least k RTCP reports with non-increasing extended highest sequence number received gives reasonable assurance that the forward path has failed, and that the RTP data packets have not been lost by chance. The RECOMMENDED value for k is 5 reports.

When Tdr < Tf, then RTP data packets are being sent at a rate less than one per RTCP reporting interval of the receiver, so the extended highest sequence number received can be expected to be non-increasing for some receiver RTCP reporting intervals. Similarly, when Tdr < Tr, some receiver RTCP reporting intervals might pass before the RTP data packets arrive at the receiver, also leading to reports where the extended highest sequence number received is non-increasing. Both issues require the media timeout interval to be scaled relative to the threshold, k.

The media timeout RTP circuit breaker is therefore as follows. When starting sending, calculate MEDIA_TIMEOUT using:

```
MEDIA_TIMEOUT = ceil(k * max(Tf, Tr, Tdr) / Tdr)
```

When a sender receives an RTCP packet that indicates reception of the media it has been sending, then it cancels the media timeout circuit breaker. If it is still sending, then it MUST calculate a new value for MEDIA_TIMEOUT, and set a new media timeout circuit breaker.

If a sender receives an RTCP packet indicating that its media was not received, it MUST calculate a new value for MEDIA_TIMEOUT. If the new value is larger than the previous, it replaces MEDIA_TIMEOUT with the new value, extending the media timeout circuit breaker; otherwise it keeps the original value of MEDIA_TIMEOUT. This process is known as reconsidering the media timeout circuit breaker.

If MEDIA_TIMEOUT consecutive RTCP packets are received indicating that the media being sent was not received, and the media timeout

circuit breaker has not been cancelled, then the media timeout circuit breaker triggers. When the media timeout circuit breaker triggers, the sender SHOULD cease transmission (see Section 4.5).

When stopping sending an RTP stream, a sender MUST cancel the corresponding media timeout circuit breaker.

4.3. RTP/AVP Circuit Breaker #3: Congestion

If RTP data packets are being sent, and the corresponding RTCP SR or RR packets show non-zero packet loss fraction and increasing extended highest sequence number received, then those RTP data packets are arriving at the receiver, but some degree of congestion is occurring. The RTP/AVP profile [RFC3551] states that:

> If best-effort service is being used, RTP receivers SHOULD monitor packet loss to ensure that the packet loss rate is within acceptable parameters. Packet loss is considered acceptable if a TCP flow across the same network path and experiencing the same network conditions would achieve an average throughput, measured on a reasonable time scale, that is not less than the throughput the RTP flow is achieving. This condition can be satisfied by implementing congestion control mechanisms to adapt the transmission rate (or the number of layers subscribed for a layered multicast session), or by arranging for a receiver to leave the session if the loss rate is unacceptably high.
>
> The comparison to TCP cannot be specified exactly, but is intended as an "order-of-magnitude" comparison in time scale and throughput. The time scale on which TCP throughput is measured is the round-trip time of the connection. In essence, this requirement states that it is not acceptable to deploy an application (using RTP or any other transport protocol) on the best-effort Internet which consumes bandwidth arbitrarily and does not compete fairly with TCP within an order of magnitude.

The phase "order of magnitude" in the above means within a factor of ten, approximately. In order to implement this, it is necessary to estimate the throughput a bulk TCP connection would achieve over the path. For a long-lived TCP Reno connection, it has been shown that the TCP throughput, X, in bytes per second, can be estimated using [Padhye]:

$$X = \frac{s}{Tr*\sqrt{2*b*p/3}+(t\_RTO * (3*\sqrt{3*b*p/8}) * p * (1+32*p*p))}$$

This is the same approach to estimated TCP throughput that is used in [RFC5348]. Under conditions of low packet loss the second term on the denominator is small, so this formula can be approximated with reasonable accuracy as follows [Mathis]:

$$X = \frac{s}{Tr*\sqrt{2*b*p/3}}$$

It is RECOMMENDED that this simplified throughput equation be used, since the reduction in accuracy is small, and it is much simpler to calculate than the full equation. Measurements have shown that the simplified TCP throughput equation is effective as an RTP circuit breaker for multimedia flows sent to hosts on residential networks using ADSL and cable modem links [Singh]. The data shows that the full TCP throughput equation tends to be more sensitive to packet loss and triggers the RTP circuit breaker earlier than the simplified equation. Implementations that desire this extra sensitivity MAY use the full TCP throughput equation in the RTP circuit breaker. Initial measurements in LTE networks have shown that the extra sensitivity is helpful in that environment, with the full TCP throughput equation giving a more balanced circuit breaker response than the simplified TCP equation [Sarker]; other networks might see similar behaviour.

No matter what TCP throughput equation is chosen, two parameters need to be estimated and reported to the sender in order to calculate the throughput: the round trip time, Tr, and the loss event rate, p (the packet size, s, is known to the sender). The round trip time can be estimated from RTCP SR and RR packets. This is done too infrequently for accurate statistics, but is the best that can be done with the standard RTCP mechanisms.

Report blocks in RTCP SR or RR packets contain the packet loss fraction, rather than the loss event rate, so p cannot be reported (TCP typically treats the loss of multiple packets within a single RTT as one loss event, but RTCP RR packets report the overall fraction of packets lost, and does not report when the packet losses occurred). Using the loss fraction in place of the loss event rate can overestimate the loss. We believe that this overestimate will not be significant, given that we are only interested in order of magnitude comparison ([Floyd] section 3.2.1 shows that the difference is small for steady-state conditions and random loss, but using the loss fraction is more conservative in the case of bursty loss).

The congestion circuit breaker is therefore: when a sender that is transmitting at least one RTP packet every max(Tdr, Tr) seconds receives an RTCP SR or RR packet that contains a report block for an SSRC it is using, the sender MUST record the value of the fraction

lost field from the report block, and the time since the last report block was received, for that SSRC. If more than CB_INTERVAL (see below) report blocks have been received for that SSRC, the sender MUST calculate the average fraction lost over the last CB_INTERVAL reporting intervals, and then estimate the TCP throughput that would be achieved over the path using the chosen TCP throughput equation and the measured values of the round-trip time, Tr, the loss event rate, p (approximated by the average fraction lost, as is described below), and the packet size, s. The estimate of the TCP throughput, X, is then compared with the actual sending rate of the RTP stream. If the actual sending rate of the RTP stream is more than 10 * X, then the congestion circuit breaker is triggered.

The average fraction lost is calculated based on the sum, over the last CB_INTERVAL reporting intervals, of the fraction lost in each reporting interval multiplied by the duration of the corresponding reporting interval, divided by the total duration of the last CB_INTERVAL reporting intervals. The CB_INTERVAL parameter is set to:

```
CB_INTERVAL =
   ceil(3*min(max(10*G*Tf, 10*Tr, 3*Tdr), max(15, 3*Td))/(3*Tdr))
```

The parameters that feed into CB_INTERVAL are chosen to give the congestion control algorithm time to react to congestion. They give at least three RTCP reports, ten round trip times, and ten groups of frames to adjust the rate to reduce the congestion to a reasonable level. It is expected that a responsive congestion control algorithm will begin to respond with the next group of frames after it receives indication of congestion, so CB_INTERVAL ought to be a much longer interval than the congestion response.

If the RTP/AVPF profile [RFC4585] or the RTP/SAVPF [RFC5124] is used, and the T_rr_interval parameter is used to reduce the frequency of regular RTCP reports, then the value Tdr in the above expression for the CB_INTERVAL parameter MUST be replaced by max(T_rr_interval, Tdr).

The CB_INTERVAL parameter is calculated on joining the session, and recalculated on receipt of each RTCP packet, after checking whether the media timeout circuit breaker or the congestion circuit breaker has been triggered.

To ensure a timely response to persistent congestion, implementations SHOULD NOT configure the RTCP bandwidth such that Tdr is larger than 5 seconds. Similarly, implementations that use the RTP/AVPF profile [RFC4585] or the RTP/SAVPF profile [RFC5124] SHOULD NOT configure T_rr_interval to values larger than 4 seconds (the reduced limit for

T_rr_interval follows Section 6.1.3 of [I-D.ietf-avtcore-rtp-multi-stream]).

The rationale for enforcing a minimum sending rate below which the congestion circuit breaker will not trigger is to avoid spurious circuit breaker triggers when the number of packets sent per RTCP reporting interval is small, and hence the fraction lost samples are subject to measurement artefacts. The bound of at least one packet every max(Tdr, Tr) seconds is derived from the one packet per RTT minimum sending rate of TCP [RFC5405], adapted for use with RTP where the RTCP reporting interval is decoupled from the network RTT.

When the congestion circuit breaker is triggered, the sender SHOULD cease transmission (see Section 4.5). However, if the sender is able to reduce its sending rate by a factor of (approximately) ten, then it MAY first reduce its sending rate by this factor (or some larger amount) to see if that resolves the congestion. If the sending rate is reduced in this way and the congestion circuit breaker triggers again after the next CB_INTERVAL RTCP reporting intervals, the sender MUST then cease transmission. An example of such a rate reduction might be a video conferencing system that backs off to sending audio only, before completely dropping the call. If such a reduction in sending rate resolves the congestion problem, the sender MAY gradually increase the rate at which it sends data after a reasonable amount of time has passed, provided it takes care not to cause the problem to recur ("reasonable" is intentionally not defined here, since it depends on the application, media codec, and congestion control algorithm).

The RTCP reporting interval of the media sender does not affect how quickly congestion circuit breaker can trigger. The timing is based on the RTCP reporting interval of the receiver that generates the SR/RR packets from which the loss rate and RTT estimate are derived (note that RTCP requires all participants in a session to have similar reporting intervals, else the participant timeout rules in [RFC3550] will not work, so this interval is likely similar to that of the sender). If the incoming RTCP SR or RR packets are using a reduced minimum RTCP reporting interval (as specified in Section 6.2 of RFC 3550 [RFC3550] or the RTP/AVPF profile [RFC4585]), then that reduced RTCP reporting interval is used when determining if the circuit breaker is triggered.

If there are more media streams that can be reported in a single RTCP SR or RR packet, or if the size of a complete RTCP SR or RR packet exceeds the network MTU, then the receiver will report on a subset of sources in each reporting interval, with the subsets selected round-robin across multiple intervals so that all sources are eventually reported [RFC3550]. When generating such round-robin RTCP reports,

priority SHOULD be given to reports on sources that have high packet loss rates, to ensure that senders are aware of network congestion they are causing (this is an update to [RFC3550]).

4.4. RTP/AVP Circuit Breaker #4: Media Usability

Applications that use RTP are generally tolerant to some amount of packet loss. How much packet loss can be tolerated will depend on the application, media codec, and the amount of error correction and packet loss concealment that is applied. There is an upper bound on the amount of loss that can be corrected, however, beyond which the media becomes unusable. Similarly, many applications have some upper bound on the media capture to play-out latency that can be tolerated before the application becomes unusable. The latency bound will depend on the application, but typical values can range from the order of a few hundred milliseconds for voice telephony and interactive conferencing applications, up to several seconds for some video-on-demand systems.

As a final circuit breaker, RTP senders SHOULD monitor the reported packet loss and delay to estimate whether the media is likely to be suitable for the intended purpose. If the packet loss rate and/or latency is such that the media has become unusable, and has remained unusable for a significant time period, then the application SHOULD cease transmission. Similarly, receivers SHOULD monitor the quality of the media they receive, and if the quality is unusable for a significant time period, they SHOULD terminate the session. This memo intentionally does not define a bound on the packet loss rate or latency that will result in unusable media, as these are highly application dependent. Similarly, the time period that is considered significant is application dependent, but is likely on the order of seconds, or tens of seconds.

Sending media that suffers from such high packet loss or latency that it is unusable at the receiver is both wasteful of resources, and of no benefit to the user of the application. It also is highly likely to be congesting the network, and disrupting other applications. As such, the congestion circuit breaker will almost certainly trigger to stop flows where the media would be unusable due to high packet loss or latency. However, in pathological scenarios where the congestion circuit breaker does not stop the flow, it is desirable to prevent the application sending unnecessary traffic that might disrupt other uses of the network. The role of the media usability circuit breaker is to protect the network in such cases.

4.5. Ceasing Transmission

What it means to cease transmission depends on the application. The intention is that the application will stop sending RTP data packets on a particular 5-tuple (transport protocol, source and destination ports, source and destination IP addresses), until whatever network problem that triggered the RTP circuit breaker has dissipated. This could mean stopping a single RTP flow, or it could mean that multiple bundled RTP flows are stopped. RTP flows halted by the circuit breaker SHOULD NOT be restarted automatically unless the sender has received information that the congestion has dissipated, or can reasonably be expected to have dissipated. What could trigger this expectation is necessarily application dependent, but could be, for example, an indication that a competing flow has finished and freed up some capacity, or for an application running on a mobile device, that the device moved to a new location so the flow would traverse a different path if it were restarted. Ideally, a human user will be involved in the decision to try to restart the flow, since that user will eventually give up if the flows repeatedly trigger the circuit breaker. This will help avoid problems with automatic redial systems from congesting the network.

It is recognised that the RTP implementation in some systems might not be able to determine if a flow set-up request was initiated by a human user, or automatically by some scripted higher-level component of the system. These implementations MUST rate limit attempts to restart a flow on the same 5-tuple as used by a flow that triggered the circuit breaker, so that the reaction to a triggered circuit breaker lasts for at least the triggering interval [I-D.ietf-tsvwg-circuit-breaker].

The RTP circuit breaker will only trigger, and cease transmission, for media flows subject to long-term persistent congestion. Such flows are likely to have poor quality and usability for some time before the circuit breaker triggers. Implementations can monitor RTCP Reception Report blocks being returned for their media flows, and might find it beneficial to use this information to provide a user interface cue that problems are occurring, in advance of the circuit breaker triggering.

5. RTP Circuit Breakers and the RTP/AVPF and RTP/SAVPF Profiles

Use of the Extended RTP Profile for RTCP-based Feedback (RTP/AVPF) [RFC4585] allows receivers to send early RTCP reports in some cases, to inform the sender about particular events in the media stream. There are several use cases for such early RTCP reports, including providing rapid feedback to a sender about the onset of congestion. The RTP/SAVPF Profile [RFC5124] is a secure variant of the RTP/AVPF

profile, that is treated the same in the context of the RTP circuit breaker. These feedback profiles are often used with non-compound RTCP reports [RFC5506] to reduce the reporting overhead.

Receiving rapid feedback about congestion events potentially allows congestion control algorithms to be more responsive, and to better adapt the media transmission to the limitations of the network. It is expected that many RTP congestion control algorithms will adopt the RTP/AVPF profile or the RTP/SAVPF profile for this reason, defining new transport layer feedback reports that suit their requirements. Since these reports are not yet defined, and likely very specific to the details of the congestion control algorithm chosen, they cannot be used as part of the generic RTP circuit breaker.

Reduced-size RTCP reports sent under the RTP/AVPF early feedback rules that do not contain an RTCP SR or RR packet MUST be ignored by the congestion circuit breaker (they do not contain the information needed by the congestion circuit breaker algorithm), but MUST be counted as received packets for the RTCP timeout circuit breaker. Reduced-size RTCP reports sent under the RTP/AVPF early feedback rules that contain RTCP SR or RR packets MUST be processed by the congestion circuit breaker as if they were sent as regular RTCP reports, and counted towards the circuit breaker conditions specified in Section 4 of this memo. This will potentially make the RTP circuit breaker trigger earlier than it would if the RTP/AVPF profile was not used.

When using ECN with RTP (see Section 7), early RTCP feedback packets can contain ECN feedback reports. The count of ECN-CE marked packets contained in those ECN feedback reports is counted towards the number of lost packets reported if the ECN Feedback Report is sent in a compound RTCP packet along with an RTCP SR/RR report packet. Reports of ECN-CE packets sent as reduced-size RTCP ECN feedback packets without an RTCP SR/RR packet MUST be ignored.

These rules are intended to allow the use of low-overhead RTP/AVPF feedback for generic NACK messages without triggering the RTP circuit breaker. This is expected to make such feedback suitable for RTP congestion control algorithms that need to quickly report loss events in between regular RTCP reports. The reaction to reduced-size RTCP SR/RR packets is to allow such algorithms to send feedback that can trigger the circuit breaker, when desired.

The RTP/AVPF and RTP/SAVPF profiles include the T_rr_interval parameter that can be used to adjust the regular RTCP reporting interval. The use of the T_rr_interval parameter changes the behaviour of the RTP circuit breaker, as described in Section 4.

6. Impact of RTCP Extended Reports (XR)

RTCP Extended Report (XR) blocks provide additional reception quality metrics, but do not change the RTCP timing rules. Some of the RTCP XR blocks provide information that might be useful for congestion control purposes, others provide non-congestion-related metrics. With the exception of RTCP XR ECN Summary Reports (see Section 7), the presence of RTCP XR blocks in a compound RTCP packet does not affect the RTP circuit breaker algorithm. For consistency and ease of implementation, only the reception report blocks contained in RTCP SR packets, RTCP RR packets, or RTCP XR ECN Summary Report packets, are used by the RTP circuit breaker algorithm.

7. Impact of Explicit Congestion Notification (ECN)

The use of ECN for RTP flows does not affect the RTCP timeout circuit breaker (Section 4.1) or the media timeout circuit breaker (Section 4.2), since these are both connectivity checks that simply determinate if any packets are being received.

There is no consensus on what would be the correct response of the congestion circuit breaker (Section 4.3) to ECN-CE marked packets. The guidelines in [RFC3168] and [RFC6679] are that the response to receipt of an ECN-CE marked packet needs to be essentially the same as the response to a lost packet for congestion control purposes. Since the RTP congestion circuit breaker responds to the same congestion signals, this suggests that it ought to consider ECN-CE marked packets as lost packets when calculating the TCP throughput estimate to determine if the congestion circuit breaker triggers.

More recent work, however, has suggested that the response to an ECN-CE mark ought to be less severe than the response to packet loss. For example, the TCP ABE proposal [I-D.khademi-tcpm-alternativebackoff-ecn] makes the argument that TCP congestion control ought to back-off less in response to an ECN-CE mark than to packet loss, because networks that generate ECN-CE marks tend to use AQM schemes with much smaller buffers. For RTP congestion control, both NADA [I-D.ietf-rmcat-nada] and SCREAM [I-D.ietf-rmcat-scream-cc] suggest responding differently to ECN-CE marked packets than to lost packets, for quality of experience reasons, but make different proposals for how the response ought to change. Such proposals would imply that a different circuit breaker threshold be used for congestion signalled by ECN-CE marks than for congestion signalled by packet loss, but unfortunately they offer no clear guidance on how the threshold ought to be changed.

Finally, there are suggestions that forthcoming AQM proposals [I-D.briscoe-aqm-dualq-coupled] might mark packets with ECN-CE in a

significantly more aggressive manner that at present. Any such deployment would likely be incompatible with deployed TCP implementations, so is not a short-term issue, but would require significant changes to the congestion circuit breaker response.

Given the above issues, implementations MAY ignore ECN-CE marks when determining if the congestion circuit breaker triggers, since excessive persistent congestion will eventually lead to packet loss that will trigger the circuit breaker. Doing this will protect the network from congestion collapse, but might result in sub-optimal user experience for competing flows that share the bottleneck queue, since that queue will be driven to overflow, inducing high latency. If this is a concern, the only current guidance is for implementations to treat ECN-CE marked packets as equivalent to lost packets, whilst being aware that this might trigger the circuit breaker prematurely in future, depending on how AQM and ECN deployment evolves. Developers that implement a circuit breaker based on ECN-CE marks will need to track future developments in AQM standards and deployed ECN marking behaviour, and ensure their implementations are updated to match.

For the media usability circuit breaker (Section 4.4), ECN-CE marked packets arrive at the receiver, and if they arrive in time, they will be decoded and rendered as normal. Accordingly, receipt of such packets ought not affect the usability of the media, and the arrival of RTCP feedback indicating their receipt is not expected to impact the operation of the media usability circuit breaker.

8. Impact of Bundled Media and Layered Coding

The RTP circuit breaker operates on a per-RTP session basis. An RTP sender that participates in several RTP sessions MUST treat each RTP session independently with regards to the RTP circuit breaker.

An RTP sender can generate several media streams within a single RTP session, with each stream using a different SSRC. This can happen if bundled media are in use, when using simulcast, or when using layered media coding. By default, each SSRC will be treated independently by the RTP circuit breaker. However, the sender MAY choose to treat the flows (or a subset thereof) as a group, such that a circuit breaker trigger for one flow applies to the group of flows as a whole, and either causes the entire group to cease transmission, or the sending rate of the group to reduce by a factor of ten, depending on the RTP circuit breaker triggered. Grouping flows in this way is expected to be especially useful for layered flows sent using multiple SSRCs, as it allows the layered flow to react as a whole, ceasing transmission on the enhancement layers first to reduce sending rate if necessary, rather than treating each layer independently. Care needs to be

taken if the different media streams sent on a single transport layer flow use different DSCP values [RFC7657], [I-D.ietf-tsvwg-rtcweb-qos], since congestion could be experienced differently depending on the DSCP marking. Accordingly, RTP media streams with different DSCP values SHOULD NOT be considered as a group when evaluating the RTP Circuit Breaker conditions.

9. Security Considerations

The security considerations of [RFC3550] apply.

If the RTP/AVPF profile is used to provide rapid RTCP feedback, the security considerations of [RFC4585] apply. If ECN feedback for RTP over UDP/IP is used, the security considerations of [RFC6679] apply.

If non-authenticated RTCP reports are used, an on-path attacker can trivially generate fake RTCP packets that indicate high packet loss rates, causing the circuit breaker to trigger and disrupt an RTP session. This is somewhat more difficult for an off-path attacker, due to the need to guess the randomly chosen RTP SSRC value and the RTP sequence number. This attack can be avoided if RTCP packets are authenticated; authentication options are discussed in [RFC7201].

Timely operation of the RTP circuit breaker depends on the choice of RTCP reporting interval. If the receiver has a reporting interval that is overly long, then the responsiveness of the circuit breaker decreases. In the limit, the RTP circuit breaker can be disabled for all practical purposes by configuring an RTCP reporting interval that is many minutes duration. This issue is not specific to the circuit breaker: long RTCP reporting intervals also prevent reception quality reports, feedback messages, codec control messages, etc., from being used. Implementations are expected to impose an upper limit on the RTCP reporting interval they are willing to negotiate (based on the session bandwidth and RTCP bandwidth fraction) when using the RTP circuit breaker, as discussed in Section 4.3.

10. IANA Considerations

There are no actions for IANA.

11. Acknowledgements

The authors would like to thank Bernard Aboba, Harald Alvestrand, Ben Campbell, Alissa Cooper, Spencer Dawkins, Gorry Fairhurst, Stephen Farrell, Nazila Fough, Kevin Gross, Cullen Jennings, Randell Jesup, Mirja Kuehlewind, Jonathan Lennox, Matt Mathis, Stephen McQuistin, Simon Perreault, Eric Rescorla, Abheek Saha, Meral Shirazipour, Fabio Verdicchio, and Magnus Westerlund for their valuable feedback.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <http://www.rfc-editor.org/info/rfc3550>.

[RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, DOI 10.17487/RFC3551, July 2003, <http://www.rfc-editor.org/info/rfc3551>.

[RFC3611] Friedman, T., Ed., Caceres, R., Ed., and A. Clark, Ed., "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, DOI 10.17487/RFC3611, November 2003, <http://www.rfc-editor.org/info/rfc3611>.

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, DOI 10.17487/RFC4585, July 2006, <http://www.rfc-editor.org/info/rfc4585>.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, DOI 10.17487/RFC5348, September 2008, <http://www.rfc-editor.org/info/rfc5348>.

[RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, DOI 10.17487/RFC6679, August 2012, <http://www.rfc-editor.org/info/rfc6679>.

12.2. Informative References

[Floyd] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "Equation-Based Congestion Control for Unicast Applications", Proceedings of the ACM SIGCOMM conference, 2000, DOI 10.1145/347059.347397, August 2000.

[I-D.briscoe-aqm-dualq-coupled]
Schepper, K., Briscoe, B., Bondarenko, O., and I. Tsang, "DualQ Coupled AQM for Low Latency, Low Loss and Scalable Throughput", draft-briscoe-aqm-dualq-coupled-01 (work in progress), March 2016.

[I-D.ietf-avtcore-rtp-multi-stream]
Lennox, J., Westerlund, M., Wu, Q., and C. Perkins, "Sending Multiple RTP Streams in a Single RTP Session", draft-ietf-avtcore-rtp-multi-stream-11 (work in progress), December 2015.

[I-D.ietf-rmcat-nada]
Zhu, X., Pan, R., Ramalho, M., Cruz, S., Jones, P., Fu, J., D'Aronco, S., and C. Ganzhorn, "NADA: A Unified Congestion Control Scheme for Real-Time Media", draft-ietf-rmcat-nada-02 (work in progress), March 2016.

[I-D.ietf-rmcat-scream-cc]
Johansson, I. and Z. Sarker, "Self-Clocked Rate Adaptation for Multimedia", draft-ietf-rmcat-scream-cc-04 (work in progress), June 2016.

[I-D.ietf-tsvwg-circuit-breaker]
Fairhurst, G., "Network Transport Circuit Breakers", draft-ietf-tsvwg-circuit-breaker-15 (work in progress), April 2016.

[I-D.ietf-tsvwg-rtcweb-qos]
Jones, P., Dhesikan, S., Jennings, C., and D. Druta, "DSCP Packet Markings for WebRTC QoS", draft-ietf-tsvwg-rtcweb-qos-17 (work in progress), May 2016.

[I-D.khademi-tcpm-alternativebackoff-ecn]
Khademi, N., Welzl, M., Armitage, G., and G. Fairhurst, "TCP Alternative Backoff with ECN (ABE)", draft-khademi-tcpm-alternativebackoff-ecn-00 (work in progress), May 2016.

[Mathis] Mathis, M., Semke, J., Mahdavi, J., and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm", ACM SIGCOMM Computer Communication Review 27(3), DOI 10.1145/263932.264023, July 1997.

[Padhye] Padhye, J., Firoiu, V., Towsley, D., and J. Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation", Proceedings of the ACM SIGCOMM conference, 1998, DOI 10.1145/285237.285291, August 1998.

[RFC2862] Civanlar, M. and G. Cash, "RTP Payload Format for Real-Time Pointers", RFC 2862, DOI 10.17487/RFC2862, June 2000, <http://www.rfc-editor.org/info/rfc2862>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <http://www.rfc-editor.org/info/rfc3168>.

[RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, DOI 10.17487/RFC4103, June 2005, <http://www.rfc-editor.org/info/rfc4103>.

[RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <http://www.rfc-editor.org/info/rfc5104>.

[RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <http://www.rfc-editor.org/info/rfc5124>.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", BCP 145, RFC 5405, DOI 10.17487/RFC5405, November 2008, <http://www.rfc-editor.org/info/rfc5405>.

[RFC5450] Singer, D. and H. Desineni, "Transmission Time Offsets in RTP Streams", RFC 5450, DOI 10.17487/RFC5450, March 2009, <http://www.rfc-editor.org/info/rfc5450>.

[RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, DOI 10.17487/RFC5506, April 2009, <http://www.rfc-editor.org/info/rfc5506>.

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, DOI 10.17487/RFC5681, September 2009, <http://www.rfc-editor.org/info/rfc5681>.

[RFC6798] Clark, A. and Q. Wu, "RTP Control Protocol (RTCP) Extended Report (XR) Block for Packet Delay Variation Metric Reporting", RFC 6798, DOI 10.17487/RFC6798, November 2012, <http://www.rfc-editor.org/info/rfc6798>.

[RFC6843] Clark, A., Gross, K., and Q. Wu, "RTP Control Protocol (RTCP) Extended Report (XR) Block for Delay Metric Reporting", RFC 6843, DOI 10.17487/RFC6843, January 2013, <http://www.rfc-editor.org/info/rfc6843>.

[RFC6958] Clark, A., Zhang, S., Zhao, J., and Q. Wu, Ed., "RTP Control Protocol (RTCP) Extended Report (XR) Block for Burst/Gap Loss Metric Reporting", RFC 6958, DOI 10.17487/RFC6958, May 2013, <http://www.rfc-editor.org/info/rfc6958>.

[RFC7002] Clark, A., Zorn, G., and Q. Wu, "RTP Control Protocol (RTCP) Extended Report (XR) Block for Discard Count Metric Reporting", RFC 7002, DOI 10.17487/RFC7002, September 2013, <http://www.rfc-editor.org/info/rfc7002>.

[RFC7003] Clark, A., Huang, R., and Q. Wu, Ed., "RTP Control Protocol (RTCP) Extended Report (XR) Block for Burst/Gap Discard Metric Reporting", RFC 7003, DOI 10.17487/RFC7003, September 2013, <http://www.rfc-editor.org/info/rfc7003>.

[RFC7097] Ott, J., Singh, V., Ed., and I. Curcio, "RTP Control Protocol (RTCP) Extended Report (XR) for RLE of Discarded Packets", RFC 7097, DOI 10.17487/RFC7097, January 2014, <http://www.rfc-editor.org/info/rfc7097>.

[RFC7201] Westerlund, M. and C. Perkins, "Options for Securing RTP Sessions", RFC 7201, DOI 10.17487/RFC7201, April 2014, <http://www.rfc-editor.org/info/rfc7201>.

[RFC7657] Black, D., Ed. and P. Jones, "Differentiated Services (Diffserv) and Real-Time Communication", RFC 7657, DOI 10.17487/RFC7657, November 2015, <http://www.rfc-editor.org/info/rfc7657>.

[RFC7713] Mathis, M. and B. Briscoe, "Congestion Exposure (ConEx) Concepts, Abstract Mechanism, and Requirements", RFC 7713, DOI 10.17487/RFC7713, December 2015, <http://www.rfc-editor.org/info/rfc7713>.

[Sarker] Sarker, Z., Singh, V., and C. Perkins, "An Evaluation of RTP Circuit Breaker Performance on LTE Networks", Proceedings of the IEEE Infocom workshop on Communication and Networking Techniques for Contemporary Video, 2014, April 2014.

[Singh] Singh, V., McQuistin, S., Ellis, M., and C. Perkins, "Circuit Breakers for Multimedia Congestion Control", Proceedings of the International Packet Video Workshop, 2013, DOI 10.1109/PV.2013.6691439, December 2013.

Authors' Addresses

Colin Perkins
University of Glasgow
School of Computing Science
Glasgow G12 8QQ
United Kingdom

Email: csp@csperkins.org

Varun Singh
Nemu Dialogue Systems Oy
Runeberginkatu 4c A 4
Helsinki 00100
Finland

Email: varun.singh@iki.fi
URI: http://www.callstats.io/

PAYLOAD
Internet-Draft
Intended status: Standards Track
Expires: September 22, 2016

V. Singh
callstats.io
A. Begen
Networked Media
M. Zanaty
Cisco
G. Mandyam
Qualcomm Innovation Center

March 21, 2016

RTP Payload Format for Flexible Forward Error Correction (FEC) draft-ietf-payload-flexible-fec-scheme-02

Abstract

This document defines new RTP payload formats for the Forward Error Correction (FEC) packets that are generated by the non-interleaved and interleaved parity codes from a source media encapsulated in RTP. These parity codes are systematic codes, where a number of repair symbols are generated from a set of source symbols. These repair symbols are sent in a repair flow separate from the source flow that carries the source symbols. The non-interleaved and interleaved parity codes which are defined in this specification offer a good protection against random and bursty packet losses, respectively, at a cost of decent complexity. Moreover, alternate FEC codes may be used with the payload formats presented. The RTP payload formats that are defined in this document address the scalability issues experienced with the earlier specifications including RFC 2733, RFC 5109 and SMPTE 2022-1, and offer several improvements. Due to these changes, the new payload formats are not backward compatible with the earlier specifications, but endpoints that do not implement the scheme can still work by simply ignoring the FEC packets.

Status of This Memo

This Internet-Draft will expire on September 22, 2016.

Table of Contents

1. Introduction

This document defines new RTP payload formats for the Forward Error Correction (FEC) that is generated by the non-interleaved and interleaved parity codes from a source media encapsulated in RTP [RFC3550]. These payload formats may also be used for other types of FEC codes. The type of the source media protected by these parity codes can be audio, video, text or application. The FEC data are generated according to the media type parameters, which are communicated out-of-band (e.g., in SDP). Furthermore, the associations or relationships between the source and repair flows may be communicated in-band or out-of-band. Situations where adaptivitiy of FEC parameters is desired, the endpoint can use the in-band mechanism, whereas when the FEC parameters are fixed, the endpoint may prefer to negotiate them out-of-band.

The repair packets proposed in this document protect the source stream packets that belong to the same RTP session.

1.1. Parity Codes

Both the non-interleaved and interleaved parity codes use the eXclusive OR (XOR) operation to generate the repair symbols. In a nutshell, the following steps take place:

1. The sender determines a set of source packets to be protected by FEC based on the media type parameters.

2. The sender applies the XOR operation on the source symbols to generate the required number of repair symbols.

3. The sender packetizes the repair symbols and sends the repair packet(s) along with the source packets to the receiver(s) (in different flows). The repair packets may be sent proactively or on-demand.

Note that the source and repair packets belong to different source and repair flows, and the sender must provide a way for the receivers to demultiplex them, even in the case they are sent in the same 5-tuple (i.e., same source/destination address/port with UDP). This is required to offer backward compatibility for endpoints that do not understand the FEC packets (See Section 4). At the receiver side, if all of the source packets are successfully received, there is no need for FEC recovery and the repair packets are discarded. However, if there are missing source packets, the repair packets can be used to recover the missing information. Figure 1 and Figure 2 describe example block diagrams for the systematic parity FEC encoder and decoder, respectively.

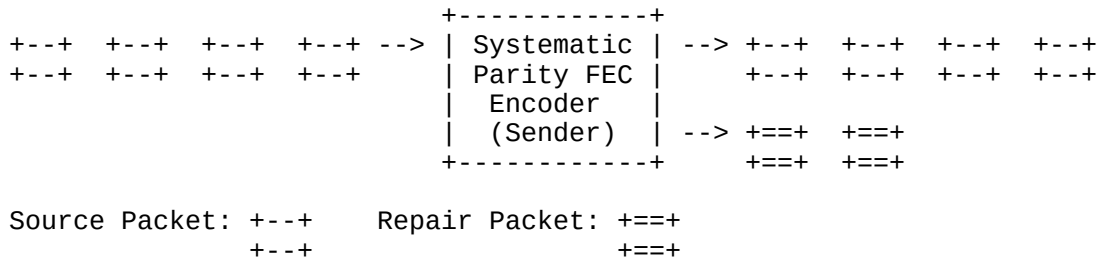


Figure 1: Block diagram for systematic parity FEC encoder

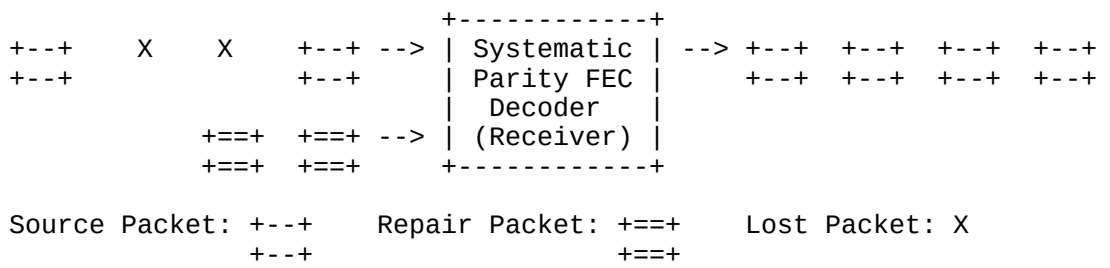


Figure 2: Block diagram for systematic parity FEC decoder

In Figure 2, it is clear that the FEC packets have to be received by the endpoint within a certain amount of time for the FEC recovery process to be useful. In this document, we refer to the time that spans a FEC block, which consists of the source packets and the corresponding repair packets, as the repair window. At the receiver side, the FEC decoder should wait at least for the duration of the repair window after getting the first packet in a FEC block, to allow all the repair packets to arrive. (The waiting time can be adjusted if there are missing packets at the beginning of the FEC block.) The FEC decoder can start decoding the already received packets sooner; however, it should not register a FEC decoding failure until it waits at least for the duration of the repair window.

Suppose that we have a group of D x L source packets that have sequence numbers starting from 1 running to D x L, and a repair packet is generated by applying the XOR operation to every L consecutive packets as sketched in Figure 3. This process is referred to as 1-D non-interleaved FEC protection. As a result of this process, D repair packets are generated, which we refer to as non-interleaved (or row) FEC packets.

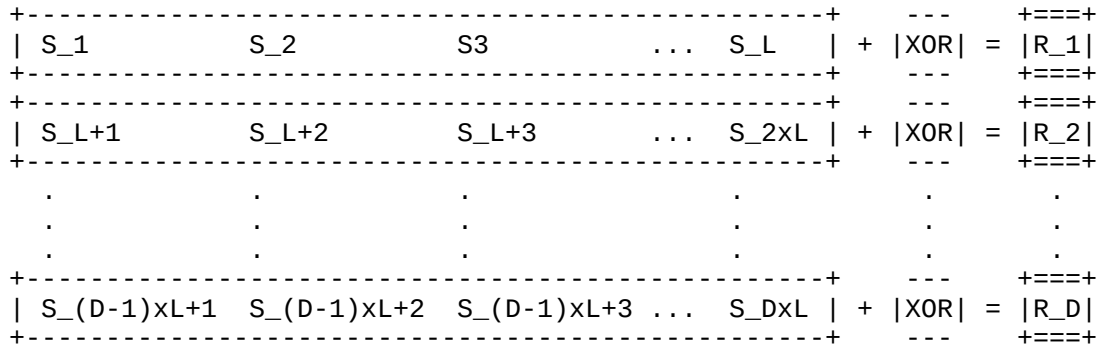


Figure 3: Generating non-interleaved (row) FEC packets

If we apply the XOR operation to the group of the source packets whose sequence numbers are L apart from each other, as sketched in Figure 4. In this case the endpoint generates L repair packets. This process is referred to as 1-D interleaved FEC protection, and the resulting L repair packets are referred to as interleaved (or column) FEC packets.

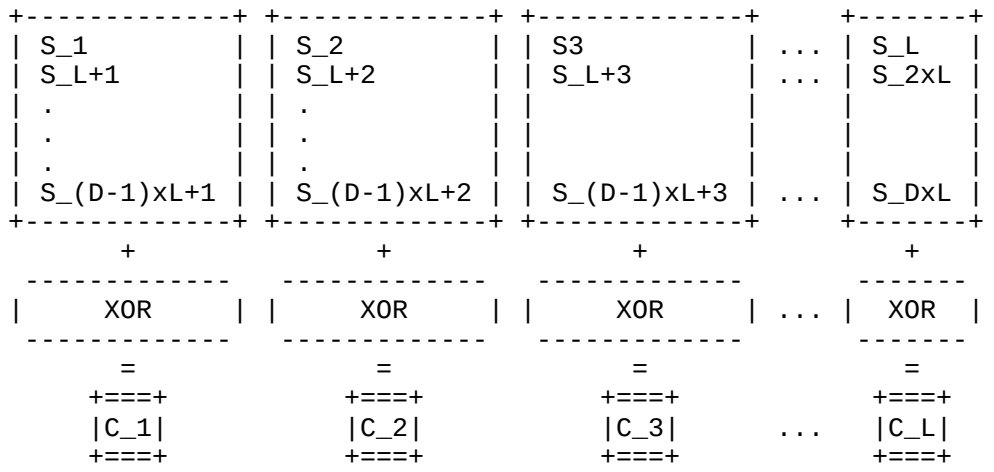


Figure 4: Generating interleaved (column) FEC packets

1.1.1. Use Cases for 1-D FEC Protection

We generate one non-interleaved repair packet out of L consecutive source packets or one interleaved repair packet out of D non-consecutive source packets. Regardless of whether the repair packet is a non-interleaved or an interleaved one, it can provide a full recovery of the missing information if there is only one packet missing among the corresponding source packets. This implies that 1-D non-interleaved FEC protection performs better when the source packets are randomly lost. However, if the packet losses occur in bursts, 1-D interleaved FEC protection performs better provided that L is chosen large enough, i.e., L-packet duration is not shorter than the observed burst duration. If the sender generates non-interleaved FEC packets and a burst loss hits the source packets, the repair operation fails. This is illustrated in Figure 5.

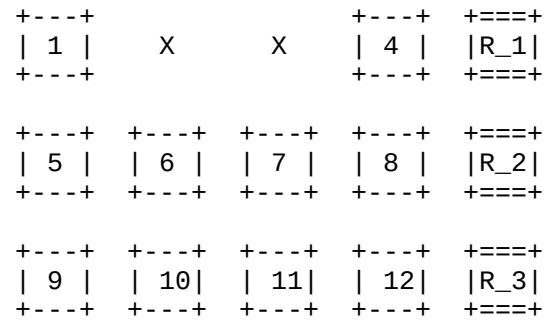


Figure 5: Example scenario where 1-D non-interleaved FEC protection fails error recovery (Burst Loss)

The sender may generate interleaved FEC packets to combat with the bursty packet losses. However, two or more random packet losses may hit the source and repair packets in the same column. In that case, the repair operation fails as well. This is illustrated in Figure 6. Note that it is possible that two burst losses may occur back-to-back, in which case interleaved FEC packets may still fail to recover the lost data.

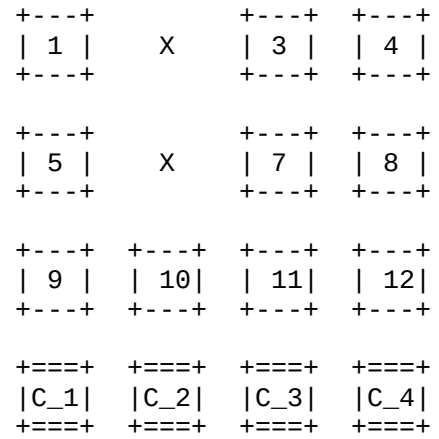


Figure 6: Example scenario where 1-D interleaved FEC protection fails error recovery (Periodic Loss)

1.1.2. Use Cases for 2-D Parity FEC Protection

In networks where the source packets are lost both randomly and in bursts, the sender ought to generate both non-interleaved and interleaved FEC packets. This type of FEC protection is known as 2-D parity FEC protection. At the expense of generating more FEC packets, thus increasing the FEC overhead, 2-D FEC provides superior protection against mixed loss patterns. However, it is still possible for 2-D parity FEC protection to fail to recover all of the lost source packets if a particular loss pattern occurs. An example scenario is illustrated in Figure 7.

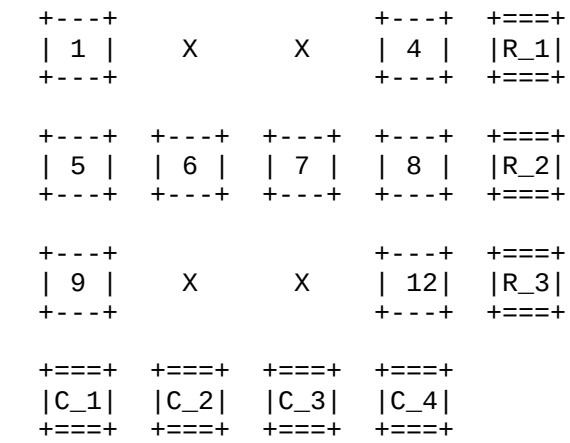


Figure 7: Example scenario #1 where 2-D parity FEC protection fails error recovery

2-D parity FEC protection also fails when at least two rows are missing a source and the FEC packet and the missing source packets (in at least two rows) are aligned in the same column. An example loss pattern is sketched in Figure 8. Similarly, 2-D parity FEC protection cannot repair all missing source packets when at least two columns are missing a source and the FEC packet and the missing source packets (in at least two columns) are aligned in the same row.

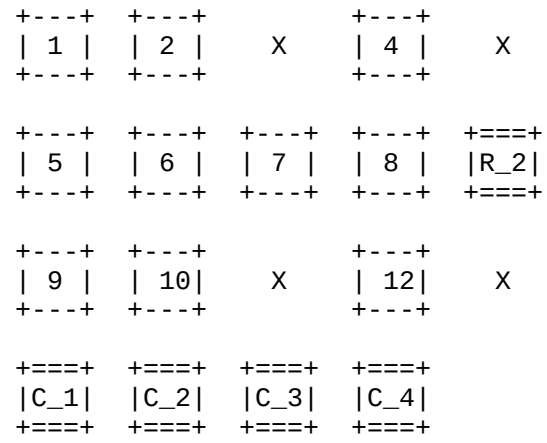


Figure 8: Example scenario #2 where 2-D parity FEC protection fails error recovery

1.1.3. Overhead Computation

The overhead is defined as the ratio of the number of bytes belonging to the repair packets to the number of bytes belonging to the protected source packets.

Generally, repair packets are larger in size compared to the source packets. Also, not all the source packets are necessarily equal in size. However, if we assume that each repair packet carries an equal number of bytes carried by a source packet, we can compute the overhead for different FEC protection methods as follows:

- 1-D Non-interleaved FEC Protection: Overhead = 1/L
- 1-D Interleaved FEC Protection: Overhead = 1/D
- 2-D Parity FEC Protection: Overhead = 1/L + 1/D

where L and D are the number of columns and rows in the source block, respectively.

2. 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].

3. Definitions and Notations

3.1. Definitions

This document uses a number of definitions from [RFC6363].

3.2. Notations

- L: Number of columns of the source block.
- D: Number of rows of the source block.
- bitmask: Run-length encoding of packets protected by a FEC packet. If the bit i in the mask is set to 1, the source packet number N + i is protected by this FEC packet. Here, N is the sequence number base, which is indicated in the FEC packet as well.

4. Packet Formats

This section defines the formats of the source and repair packets.

4.1. Source Packets

The source packets MUST contain the information that identifies the source block and the position within the source block occupied by the packet. Since the source packets that are carried within an RTP stream already contain unique sequence numbers in their RTP headers [RFC3550], we can identify the source packets in a straightforward manner and there is no need to append additional field(s). The primary advantage of not modifying the source packets in any way is that it provides backward compatibility for the receivers that do not support FEC at all. In multicast scenarios, this backward compatibility becomes quite useful as it allows the non-FEC-capable and FEC-capable receivers to receive and interpret the same source packets sent in the same multicast session.

4.2. Repair Packets

The repair packets MUST contain information that identifies the source block they pertain to and the relationship between the contained repair symbols and the original source block. For this purpose, we use the RTP header of the repair packets as well as another header within the RTP payload, which we refer to as the FEC header, as shown in Figure 9.

Note that all the source stream packets that are protected by a particular FEC packet need to be in the same RTP session.

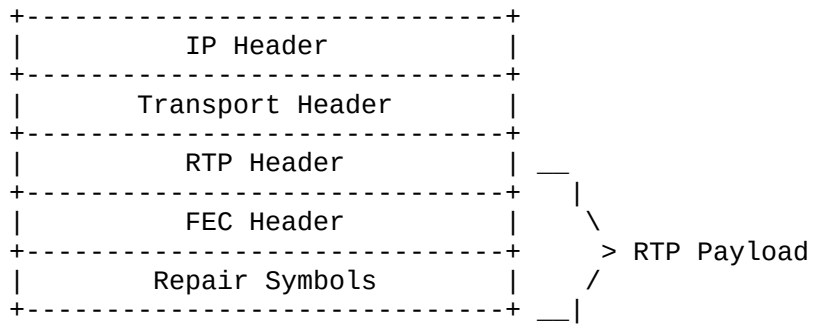


Figure 9: Format of repair packets

The RTP header is formatted according to [RFC3550] with some further clarifications listed below:

- Marker (M) Bit: This bit is not used for this payload type, and SHALL be set to 0.

- Payload Type: The (dynamic) payload type for the repair packets is determined through out-of-band means. Note that this document registers new payload formats for the repair packets (Refer to Section 5 for details). According to [RFC3550], an RTP receiver that cannot recognize a payload type must discard it. This provides backward compatibility. If a non-FEC-capable receiver receives a repair packet, it will not recognize the payload type, and hence, will discard the repair packet.

- Sequence Number (SN): The sequence number has the standard definition. It MUST be one higher than the sequence number in the previously transmitted repair packet. The initial value of the sequence number SHOULD be random (unpredictable, based on [RFC3550]).

- Timestamp (TS): The timestamp SHALL be set to a time corresponding to the repair packet's transmission time. Note that the timestamp value has no use in the actual FEC protection process and is usually useful for jitter calculations.

- Synchronization Source (SSRC): The SSRC value SHALL be randomly assigned as suggested by [RFC3550]. This allows the sender to multiplex the source and repair flows on the same port, or multiplex multiple repair flows on a single port. The repair flows SHOULD use the RTCP CNAME field to associate themselves with the source flow.

In some networks, the RTP Source, which produces the source packets and the FEC Source, which generates the repair packets from the source packets may not be the same host. In such scenarios, using the same CNAME for the source and repair flows means that the RTP Source and the FEC Source MUST share the same CNAME (for this specific source-repair flow association). A common CNAME may be produced based on an algorithm that is known both to the RTP and FEC Source [RFC7022]. This usage is compliant with [RFC3550].

Note that due to the randomness of the SSRC assignments, there is a possibility of SSRC collision. In such cases, the collisions MUST be resolved as described in [RFC3550].

The format of the FEC header is shown in Figure 10.

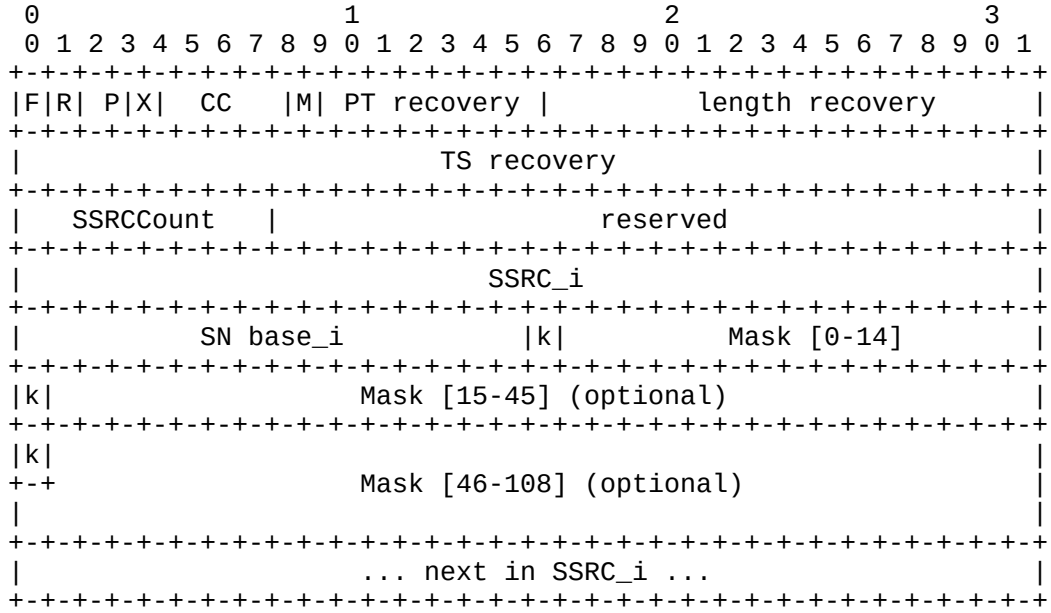


Figure 10: Format of the FEC header

The FEC header consists of the following fields:

o The F field (1 bit) indicates the type of the mask. Namely:

F bit	Use
0	flexible mask
1	packets indicated by offset M and N

Figure 11: F-bit values

- The R bit MUST be set to 1 to indicate a retransmission packet, and MUST be set to 0 for repair packets.
- The P, X, CC, M and PT recovery fields are used to determine the corresponding fields of the recovered packets.
- The Length recovery (16 bits) field is used to determine the length of the recovered packets.
- The TS recovery (32 bits) field is used to determine the timestamp of the recovered packets.
- The SSRC count (8 bits) field describes the number of SSRCs protected by the FEC packet. 0 is not a valid value, and the packet MUST be ignored.
- The Reserved (24 bits) field are reserved for future use. It MUST be set to zero by senders and ignored by receivers (see [RFC6709], Section 4.2).
- The SSRC_i (32 bits) field describes the SSRC of the packets protected by this particular FEC packet. If a FEC packet contains protects multiple SSRCs (indicated by the SSRC Count > 1), there will be multiple blocks of data containing the SSRC, SN base and Mask fields.
- The SN base_i (16 bits) field indicates the lowest sequence number, taking wrap around into account, of the source packets for a particular SSSRC (indicated in SSRC_i) protected by this repair packet.
- If the F-bit is set to 0, it represents that the source packets of all the SSRCs protected by this particular repair packet are indicated by using a flexible bitmask. Mask is a run-length encoding of packets for a particular SSRC_i protected by the FEC packet. Where a bit j set to 1 indicates that the source packet with sequence number (SN base_i + j + 1) is protected by this FEC packet.

o The k-bit in the bitmasks indicates if it is 15-, 46-, or a 109-bitmask. k=0 denotes that there is one more k-bit set, and k=1 denotes that it is the last block of bit mask. While parsing the header, the current count of the number of k-bit gives the size of the bit mask v = count(k). Size of next bitmask = 2^(v+3)-1.

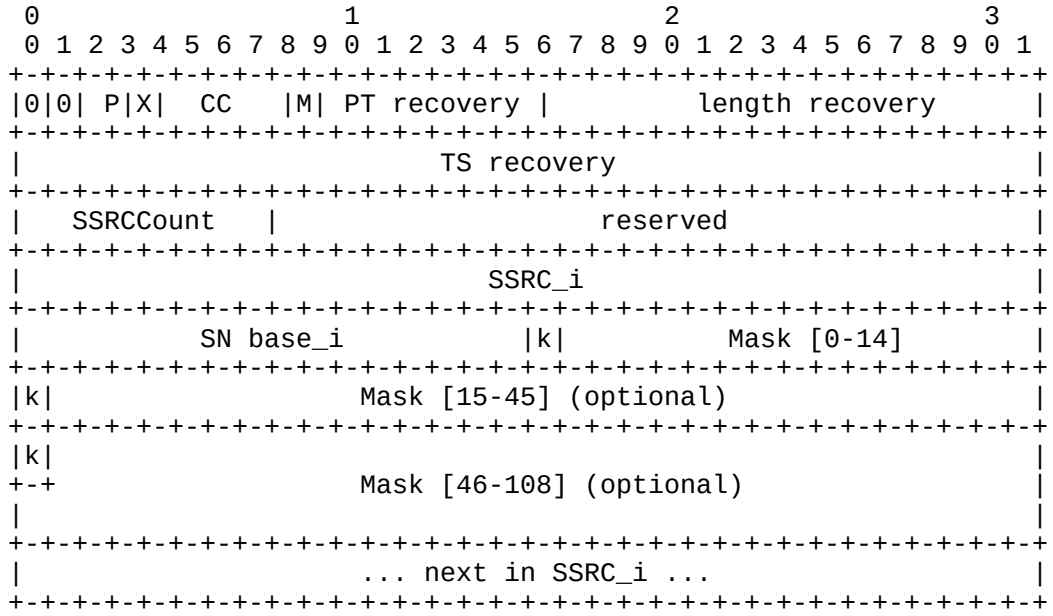


Figure 12: Protocol format for F=0

o If the F-bit is set to 1, it represents that the source packets of all the SSRCs protected by this particular repair packet are indicated by using fixed offsets.

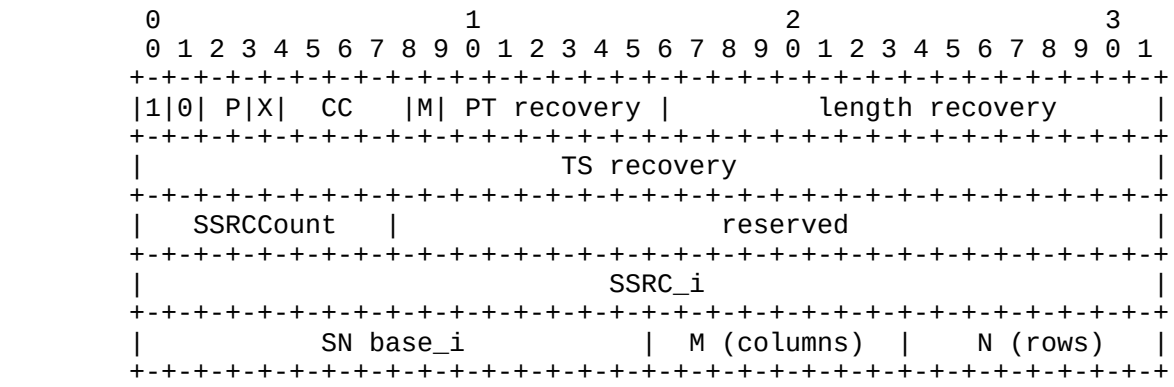


Figure 13: Protocol format for F=1

Consequently, the following conditions occur for M and N values:

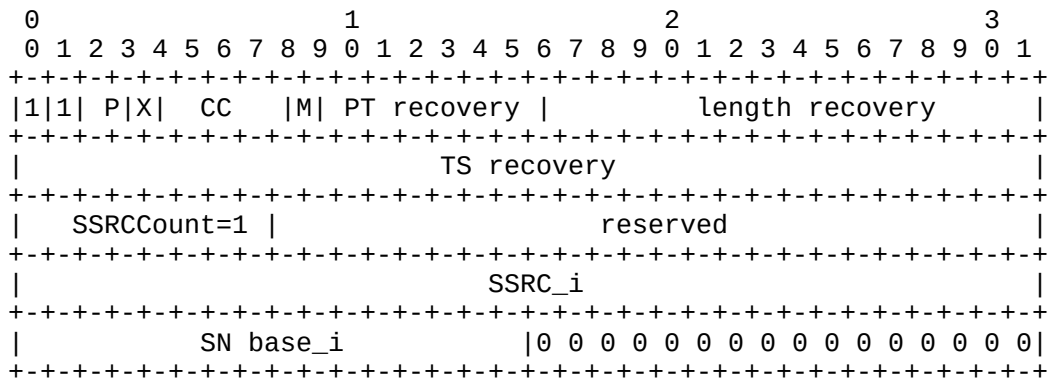
If M>0, N=0, is Row FEC, and no column FEC will follow
Hence, FEC = SN, SN+1, SN+2, ... , SN+(M-1), SN+M.

If M>0, N=1, is Row FEC, and column FEC will follow.
Hence, FEC = SN, SN+1, SN+2, ... , SN+(M-1), SN+M.
and more to come

If M>0, N>1, indicates column FEC of every M packet
in a group of N packets starting at SN base.
Hence, FEC = SN+(Mx0), SN+(Mx1), ... , SN+(MxN).

Figure 14: Interpreting the M and N field values

By setting F to 1, R=1, SSRC count to 1, M=0, and N=0, the FEC protects only one packet, i.e., the FEC payload carries just the packet indicated by SN Base_i, which is effectively retransmitting the packet.



Editor’s note: this can surely be optimized.

Figure 15: Protocol format for Retransmission

The details on setting the fields in the FEC header are provided in Section 6.2.

It should be noted that a mask-based approach (similar to the ones specified in [RFC2733] and [RFC5109]) may not be very efficient to indicate which source packets in the current source block are associated with a given repair packet. In particular, for the applications that would like to use large source block sizes, the size of the mask that is required to describe the source-repair packet associations may be prohibitively large. The 8-bit fields proposed in [SMPTE2022-1] indicate a systematized approach. Instead the approach in this document uses the 8-bit fields to indicate packet offsets protected by the FEC packet. The approach in [SMPTE2022-1] is inherently more efficient for regular patterns, it does not provide flexibility to represent other protection patterns (e.g., staircase).

5. Payload Format Parameters

This section provides the media subtype registration for the non-interleaved and interleaved parity FEC. The parameters that are required to configure the FEC encoding and decoding operations are also defined in this section. If no specific FEC code is specified in the subtype, then the FEC code defaults to the parity code defined in this specification.

5.1. Media Type Registration - Parity Codes

This registration is done using the template defined in [RFC6838] and following the guidance provided in [RFC3555].

Note to the RFC Editor: In the following sections, please replace "XXXX" with the number of this document prior to publication as an RFC.

5.1.1. Registration of audio/flexfec

Type name: audio

Subtype name: flexfec

Required parameters:

- rate: The RTP timestamp (clock) rate. The rate SHALL be larger than 1000 Hz to provide sufficient resolution to RTCP operations. However, it is RECOMMENDED to select the rate that matches the rate of the protected source RTP stream.

- repair-window: The time that spans the source packets and the corresponding repair packets. The size of the repair window is specified in microseconds.

Optional parameters:

- L: indicates the number of columns of the source block that are protected by this FEC block and it applies to all the source SSRCs. L is a positive integer.

- D: indicates the number of rows of the source block that are protected by this FEC block and it applies to all the source SSRCs. D is a positive integer.

- ToP: indicates the type of protection applied by the sender: 0 for 1-D interleaved FEC protection, 1 for 1-D non-interleaved FEC protection, and 2 for 2-D parity FEC protection. The ToP value of 3 is reserved for future uses.

Encoding considerations: This media type is framed (See Section 4.8 in the template document [RFC6838]) and contains binary data.

Security considerations: See Section 9 of [RFCXXXX].

Interoperability considerations: None.

Published specification: [RFCXXXX].

Applications that use this media type: Multimedia applications that want to improve resiliency against packet loss by sending redundant data in addition to the source media.

Fragment identifier considerations: None.

Additional information: None.

Person & email address to contact for further information: Varun Singh <varun@callstats.io> and IETF Audio/Video Transport Payloads Working Group.

Intended usage: COMMON.

Restriction on usage: This media type depends on RTP framing, and hence, is only defined for transport via RTP [RFC3550].

Author: Varun Singh <varun@callstats.io>.

Change controller: IETF Audio/Video Transport Working Group delegated from the IESG.

Provisional registration? (standards tree only): Yes.

5.1.2. Registration of video/flexfec

Type name: video

Subtype name: flexfec

Required parameters:

- rate: The RTP timestamp (clock) rate. The rate SHALL be larger than 1000 Hz to provide sufficient resolution to RTCP operations. However, it is RECOMMENDED to select the rate that matches the rate of the protected source RTP stream.

- repair-window: The time that spans the source packets and the corresponding repair packets. The size of the repair window is specified in microseconds.

Optional parameters:

- L: indicates the number of columns of the source block that are protected by this FEC block and it applies to all the source SSRCs. L is a positive integer.

- o D: indicates the number of rows of the source block that are protected by this FEC block and it applies to all the source SSRCs. D is a positive integer.

- o ToP: indicates the type of protection applied by the sender: 0 for 1-D interleaved FEC protection, 1 for 1-D non-interleaved FEC protection, and 2 for 2-D parity FEC protection. The ToP value of 3 is reserved for future uses.

Encoding considerations: This media type is framed (See Section 4.8 in the template document [RFC6838]) and contains binary data.

Security considerations: See Section 9 of [RFCXXXX].

Interoperability considerations: None.

Published specification: [RFCXXXX].

Applications that use this media type: Multimedia applications that want to improve resiliency against packet loss by sending redundant data in addition to the source media.

Fragment identifier considerations: None.

Additional information: None.

Person & email address to contact for further information: Varun Singh <varun@callstats.io> and IETF Audio/Video Transport Payloads Working Group.

Intended usage: COMMON.

Restriction on usage: This media type depends on RTP framing, and hence, is only defined for transport via RTP [RFC3550].

Author: Varun Singh <varun@callstats.io>.

Change controller: IETF Audio/Video Transport Working Group delegated from the IESG.

Provisional registration? (standards tree only): Yes.

5.1.3. Registration of text/flexfec

Type name: text

Subtype name: flexfec

Required parameters:

- rate: The RTP timestamp (clock) rate. The rate SHALL be larger than 1000 Hz to provide sufficient resolution to RTCP operations. However, it is RECOMMENDED to select the rate that matches the rate of the protected source RTP stream.

- repair-window: The time that spans the source packets and the corresponding repair packets. The size of the repair window is specified in microseconds.

Optional parameters:

- L: indicates the number of columns of the source block that are protected by this FEC block and it applies to all the source SSRCs. L is a positive integer.

- D: indicates the number of rows of the source block that are protected by this FEC block and it applies to all the source SSRCs. D is a positive integer.

- ToP: indicates the type of protection applied by the sender: 0 for 1-D interleaved FEC protection, 1 for 1-D non-interleaved FEC protection, and 2 for 2-D parity FEC protection. The ToP value of 3 is reserved for future uses.

Encoding considerations: This media type is framed (See Section 4.8 in the template document [RFC6838]) and contains binary data.

Security considerations: See Section 9 of [RFCXXXX].

Interoperability considerations: None.

Published specification: [RFCXXXX].

Applications that use this media type: Multimedia applications that want to improve resiliency against packet loss by sending redundant data in addition to the source media.

Fragment identifier considerations: None.

Additional information: None.

Person & email address to contact for further information: Varun Singh <vvarun@callstats.io> and IETF Audio/Video Transport Payloads Working Group.

Intended usage: COMMON.

Restriction on usage: This media type depends on RTP framing, and hence, is only defined for transport via RTP [RFC3550].

Author: Varun Singh <varun@callstats.io>.

Change controller: IETF Audio/Video Transport Working Group delegated from the IESG.

Provisional registration? (standards tree only): Yes.

5.1.4. Registration of application/flexfec

Type name: application

Subtype name: flexfec

Required parameters:

- rate: The RTP timestamp (clock) rate. The rate SHALL be larger than 1000 Hz to provide sufficient resolution to RTCP operations. However, it is RECOMMENDED to select the rate that matches the rate of the protected source RTP stream.

- repair-window: The time that spans the source packets and the corresponding repair packets. The size of the repair window is specified in microseconds.

Optional parameters:

- L: indicates the number of columns of the source block that are protected by this FEC block and it applies to all the source SSRCs. L is a positive integer.

- D: indicates the number of rows of the source block that are protected by this FEC block and it applies to all the source SSRCs. D is a positive integer.

- ToP: indicates the type of protection applied by the sender: 0 for 1-D interleaved FEC protection, 1 for 1-D non-interleaved FEC protection, and 2 for 2-D parity FEC protection. The ToP value of 3 is reserved for future uses.

Encoding considerations: This media type is framed (See Section 4.8 in the template document [RFC6838]) and contains binary data.

Security considerations: See Section 9 of [RFCXXXX].

Interoperability considerations: None.

Published specification: [RFCXXXX].

Applications that use this media type: Multimedia applications that want to improve resiliency against packet loss by sending redundant data in addition to the source media.

Fragment identifier considerations: None.

Additional information: None.

Person & email address to contact for further information: Varun Singh <varun@callstats.io> and IETF Audio/Video Transport Payloads Working Group.

Intended usage: COMMON.

Restriction on usage: This media type depends on RTP framing, and hence, is only defined for transport via RTP [RFC3550].

Author: Varun Singh <varun@callstats.io>.

Change controller: IETF Audio/Video Transport Working Group delegated from the IESG.

Provisional registration? (standards tree only): Yes.

5.2. Media Type Registration - Non-Parity Codes

This registration is done using the template defined in [RFC6838] and following the guidance provided in [RFC3555]. The media type registration follows the "flexfec-XXXX" paradigm, with the Raptor code provided here. Only the application media type is required, as it is assumed the existing source payload registration types are still applicable. Other FEC codes with specified RTP media types can be defined in a similar manner.

Note to the RFC Editor: In the following sections, please replace "XXXX" with the number of this document prior to publication as an RFC.

5.2.1. Registration of application/flexfec-raptor

Type name: application

Subtype name: flexfec-raptor

Required parameters:

See Sec. 6.1.1 of [RFC6682].

Optional parameters:

See Sec. 6.1.1 of [RFC6682].

Encoding considerations: This media type is framed (See Section 4.8 in the template document [RFC6838]) and contains binary data.

Security considerations: See Section 9 of [RFCXXXX].

Interoperability considerations: None.

Published specification: [RFCXXXX].

Applications that use this media type: Multimedia applications that want to improve resiliency against packet loss by sending redundant data in addition to the source media.

Fragment identifier considerations: None.

Additional information: None.

Person & email address to contact for further information: Varun Singh <varun@callstats.io> and IETF Audio/Video Transport Payloads Working Group.

Intended usage: COMMON.

Restriction on usage: This media type depends on RTP framing, and hence, is only defined for transport via RTP [RFC3550].

Author: Varun Singh <varun@callstats.io>.

Change controller: IETF Audio/Video Transport Working Group delegated from the IESG.

Provisional registration? (standards tree only): Yes.

5.3. Mapping to SDP Parameters

Applications that are using RTP transport commonly use Session Description Protocol (SDP) [RFC4566] to describe their RTP sessions. The information that is used to specify the media types in an RTP session has specific mappings to the fields in an SDP description. In this section, we provide these mappings for the media subtypes registered by this document. Note that if an application does not use SDP to describe the RTP sessions, an appropriate mapping must be

defined and used to specify the media types and their parameters for the control/description protocol employed by the application.

The mapping of the media type specification for "non-interleaved-parityfec" and "interleaved-parityfec" and their parameters in SDP is as follows:

- The media type (e.g., "application") goes into the "m=" line as the media name.
- The media subtype goes into the "a=rtpmap" line as the encoding name. The RTP clock rate parameter ("rate") also goes into the "a=rtpmap" line as the clock rate.
- The remaining required payload-format-specific parameters go into the "a=fmtp" line by copying them directly from the media type string as a semicolon-separated list of parameter=value pairs.

SDP examples are provided in Section 7.

5.3.1. Offer-Answer Model Considerations

When offering 1-D interleaved parity FEC over RTP using SDP in an Offer/Answer model [RFC3264], the following considerations apply:

- Each combination of the L and D parameters produces a different FEC data and is not compatible with any other combination. A sender application may desire to offer multiple offers with different sets of L and D values as long as the parameter values are valid. The receiver SHOULD normally choose the offer that has a sufficient amount of interleaving. If multiple such offers exist, the receiver may choose the offer that has the lowest overhead or the one that requires the smallest amount of buffering. The selection depends on the application requirements.
- The value for the repair-window parameter depends on the L and D values and cannot be chosen arbitrarily. More specifically, L and D values determine the lower limit for the repair-window size. The upper limit of the repair-window size does not depend on the L and D values.
- Although combinations with the same L and D values but with different repair-window sizes produce the same FEC data, such combinations are still considered different offers. The size of the repair-window is related to the maximum delay between the transmission of a source packet and the associated repair packet. This directly impacts the buffering requirement on the receiver side and the receiver must consider this when choosing an offer.

- There are no optional format parameters defined for this payload. Any unknown option in the offer MUST be ignored and deleted from the answer. If FEC is not desired by the receiver, it can be deleted from the answer.

5.3.2. Declarative Considerations

In declarative usage, like SDP in the Real-time Streaming Protocol (RTSP) [RFC2326] or the Session Announcement Protocol (SAP) [RFC2974], the following considerations apply:

- The payload format configuration parameters are all declarative and a participant MUST use the configuration that is provided for the session.

- More than one configuration may be provided (if desired) by declaring multiple RTP payload types. In that case, the receivers should choose the repair flow that is best for them.

6. Protection and Recovery Procedures - Parity Codes

This section provides a complete specification of the 1-D and 2-D parity codes and their RTP payload formats.

6.1. Overview

The following sections specify the steps involved in generating the repair packets and reconstructing the missing source packets from the repair packets.

6.2. Repair Packet Construction

The RTP header of a repair packet is formed based on the guidelines given in Section 4.2.

The FEC header includes 12 octets (or upto 28 octets when the longer optional masks are used). It is constructed by applying the XOR operation on the bit strings that are generated from the individual source packets protected by this particular repair packet. The set of the source packets that are associated with a given repair packet can be computed by the formula given in Section 6.3.1.

The bit string is formed for each source packet by concatenating the following fields together in the order specified:

- The first 64 bits of the RTP header (64 bits).

- Unsigned network-ordered 16-bit representation of the source packet length in bytes minus 12 (for the fixed RTP header), i.e., the sum of the lengths of all the following if present: the CSRC list, extension header, RTP payload and RTP padding (16 bits).

By applying the parity operation on the bit strings produced from the source packets, we generate the FEC bit string. The FEC header is generated from the FEC bit string as follows:

- The first (most significant) 2 bits in the FEC bit string are skipped. The MSK bits in the FEC header are set to the appropriate value, i.e., it depends on the chosen bitmask length.
- The next bit in the FEC bit string is written into the P recovery bit in the FEC header.
- The next bit in the FEC bit string is written into the X recovery bit in the FEC header.
- The next 4 bits of the FEC bit string are written into the CC recovery field in the FEC header.
- The next bit is written into the M recovery bit in the FEC header.
- The next 7 bits of the FEC bit string are written into the PT recovery field in the FEC header.
- The next 16 bits are skipped.
- The next 32 bits of the FEC bit string are written into the TS recovery field in the FEC header.
- The next 16 bits are written into the length recovery field in the FEC header.
- Depending on the chosen MSK value, the bit mask of appropriate length will be set to the appropriate values.

As described in Section 4.2, the SN base field of the FEC header MUST be set to the lowest sequence number of the source packets protected by this repair packet. When MSK represents a bitmask (MSK=00,01,10), the SN base field corresponds to the lowest sequence number indicated in the bitmask. When MSK=11, the following considerations apply: 1) for the interleaved FEC packets, this corresponds to the lowest sequence number of the source packets that forms the column, 2) for the non-interleaved FEC packets, the SN base field MUST be set to the lowest sequence number of the source packets that forms the row.

The repair packet payload consists of the bits that are generated by applying the XOR operation on the payloads of the source RTP packets. If the payload lengths of the source packets are not equal, each shorter packet MUST be padded to the length of the longest packet by adding octet 0's at the end.

Due to this possible padding and mandatory FEC header, a repair packet has a larger size than the source packets it protects. This may cause problems if the resulting repair packet size exceeds the Maximum Transmission Unit (MTU) size of the path over which the repair flow is sent.

6.3. Source Packet Reconstruction

This section describes the recovery procedures that are required to reconstruct the missing source packets. The recovery process has two steps. In the first step, the FEC decoder determines which source and repair packets should be used in order to recover a missing packet. In the second step, the decoder recovers the missing packet, which consists of an RTP header and RTP payload.

In the following, we describe the RECOMMENDED algorithms for the first and second steps. Based on the implementation, different algorithms MAY be adopted. However, the end result MUST be identical to the one produced by the algorithms described below.

Note that the same algorithms are used by the 1-D parity codes, regardless of whether the FEC protection is applied over a column or a row. The 2-D parity codes, on the other hand, usually require multiple iterations of the procedures described here. This iterative decoding algorithm is further explained in Section 6.3.4.

6.3.1. Associating the Source and Repair Packets

We denote the set of the source packets associated with repair packet p* by set T(p*). Note that in a source block whose size is L columns by D rows, set T includes D source packets plus one repair packet for the FEC protection applied over a column, and L source packets plus one repair packet for the FEC protection applied over a row. Recall that 1-D interleaved and non-interleaved FEC protection can fully recover the missing information if there is only one source packet missing in set T. If there are more than one source packets missing in set T, 1-D FEC protection will not work.

6.3.1.1. Signaled in SDP

The first step is associating the source and repair packets. If the endpoint relies entirely on out-of-band signaling (MSK=11, and M=N=0), then this information may be inferred from the media type parameters specified in the SDP description. Furthermore, the payload type field in the RTP header, assists the receiver distinguish an interleaved or non-interleaved FEC packet.

Mathematically, for any received repair packet, p*, we can determine the sequence numbers of the source packets that are protected by this repair packet as follows:

$$p^*\_snb + i * X\_1 \text{ (modulo 65536)}$$

where p*_snb denotes the value in the SN base field of p*'s FEC header, X_1 is set to L and 1 for the interleaved and non-interleaved FEC packets, respectively, and

$$0 <= i < X\_2$$

where X_2 is set to D and L for the interleaved and non-interleaved FEC packets, respectively.

6.3.1.2. Using bitmasks

When using fixed size bitmasks (16-, 48-, 112-bits), the SN base field in the FEC header indicates the lowest sequence number of the source packets that forms the FEC packet. Finally, the bits maked by "1" in the bitmask are offsets from the SN base and make up the rest of the packets protected by the FEC packet. The bitmasks are able to represent arbitrary protection patterns, for example, 1-D interleaved, 1-D non-interleaved, 2-D, staircase.

6.3.1.3. Using M and N Offsets

When value of M is non-zero, the 8-bit fields indicate the offset of packets protected by an interleaved (N>0) or non-interleaved (N=0) FEC packet. Using a combination of interleaved and non-interleaved FEC packets can form 2-D protection patterns.

Mathematically, for any received repair packet, p*, we can determine the sequence numbers of the source packets that are protected by this repair packet are as follows:

```
When N = 0:
  p*_snb, p*_snb+1,..., p*_snb+(M-1), p*_snb+M
When N > 0:
  p*_snb, p*_snb+(Mx1), p*_snb+(Mx2),..., p*_snb+(Mx(N-1)), p*_snb+(MxN)
```

6.3.2. Recovering the RTP Header

For a given set T, the procedure for the recovery of the RTP header of the missing packet, whose sequence number is denoted by SEQNUM, is as follows:

1. For each of the source packets that are successfully received in T, compute the 80-bit string by concatenating the first 64 bits of their RTP header and the unsigned network-ordered 16-bit representation of their length in bytes minus 12.

2. For the repair packet in T, compute the FEC bit string from the first 80 bits of the FEC header.

3. Calculate the recovered bit string as the XOR of the bit strings generated from all source packets in T and the FEC bit string generated from the repair packet in T.

4. Create a new packet with the standard 12-byte RTP header and no payload.

5. Set the version of the new packet to 2. Skip the first 2 bits in the recovered bit string.

6. Set the Padding bit in the new packet to the next bit in the recovered bit string.

7. Set the Extension bit in the new packet to the next bit in the recovered bit string.

8. Set the CC field to the next 4 bits in the recovered bit string.

9. Set the Marker bit in the new packet to the next bit in the recovered bit string.

10. Set the Payload type in the new packet to the next 7 bits in the recovered bit string.

11. Set the SN field in the new packet to SEQNUM. Skip the next 16 bits in the recovered bit string.

12. Set the TS field in the new packet to the next 32 bits in the recovered bit string.

13. Take the next 16 bits of the recovered bit string and set the new variable Y to whatever unsigned integer this represents (assuming network order). Convert Y to host order. Y represents the length of the new packet in bytes minus 12 (for the fixed RTP header), i.e., the sum of the lengths of all the following if present: the CSRC list, header extension, RTP payload and RTP padding.

14. Set the SSRC of the new packet to the SSRC of the source RTP stream.

This procedure recovers the header of an RTP packet up to (and including) the SSRC field.

6.3.3. Recovering the RTP Payload

Following the recovery of the RTP header, the procedure for the recovery of the RTP payload is as follows:

1. Append Y bytes to the new packet.

2. For each of the source packets that are successfully received in T, compute the bit string from the Y octets of data starting with the 13th octet of the packet. If any of the bit strings generated from the source packets has a length shorter than Y, pad them to that length. The padding of octet 0 MUST be added at the end of the bit string. Note that the information of the first 8 octets are protected by the FEC header.

3. For the repair packet in T, compute the FEC bit string from the repair packet payload, i.e., the Y octets of data following the FEC header. Note that the FEC header may be 12, 16, 32 octets depending on the length of the bitmask.

4. Calculate the recovered bit string as the XOR of the bit strings generated from all source packets in T and the FEC bit string generated from the repair packet in T.

5. Append the recovered bit string (Y octets) to the new packet generated in Section 6.3.2.

6.3.4. Iterative Decoding Algorithm for the 2-D Parity FEC Protection

In 2-D parity FEC protection, the sender generates both non-interleaved and interleaved FEC packets to combat with the mixed loss patterns (random and bursty). At the receiver side, these FEC packets are used iteratively to overcome the shortcomings of the 1-D

non-interleaved/interleaved FEC protection and improve the chances of full error recovery.

The iterative decoding algorithm runs as follows:

1. Set num_recovered_until_this_iteration to zero

2. Set num_recovered_so_far to zero

3. Recover as many source packets as possible by using the non-interleaved FEC packets as outlined in Section 6.3.2 and Section 6.3.3, and increase the value of num_recovered_so_far by the number of recovered source packets.

4. Recover as many source packets as possible by using the interleaved FEC packets as outlined in Section 6.3.2 and Section 6.3.3, and increase the value of num_recovered_so_far by the number of recovered source packets.

5. If num_recovered_so_far > num_recovered_until_this_iteration
   ---num_recovered_until_this_iteration = num_recovered_so_far
   ---Go to step 3
   Else
   ---Terminate

The algorithm terminates either when all missing source packets are fully recovered or when there are still remaining missing source packets but the FEC packets are not able to recover any more source packets. For the example scenarios when the 2-D parity FEC protection fails full recovery, refer to Section 1.1.2. Upon termination, variable num_recovered_so_far has a value equal to the total number of recovered source packets.

Example:

Suppose that the receiver experienced the loss pattern sketched in Figure 16.

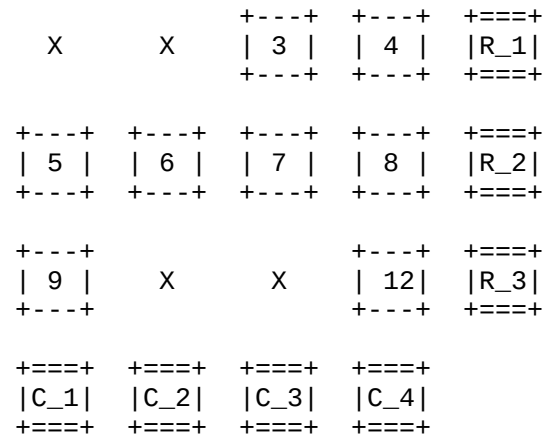


Figure 16: Example loss pattern for the iterative decoding algorithm

The receiver executes the iterative decoding algorithm and recovers source packets #1 and #11 in the first iteration. The resulting pattern is sketched in Figure 17.

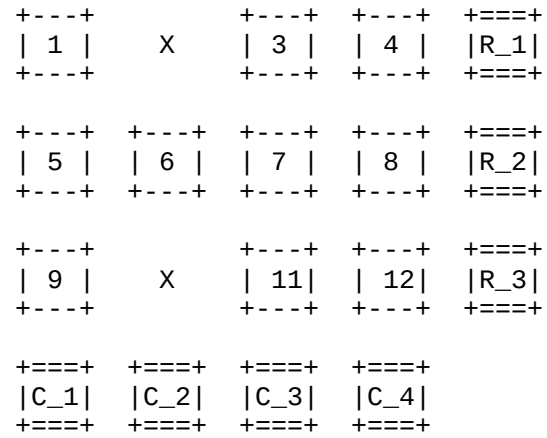


Figure 17: The resulting pattern after the first iteration

Since the if condition holds true, the receiver runs a new iteration. In the second iteration, source packets #2 and #10 are recovered, resulting in a full recovery as sketched in Figure 18.

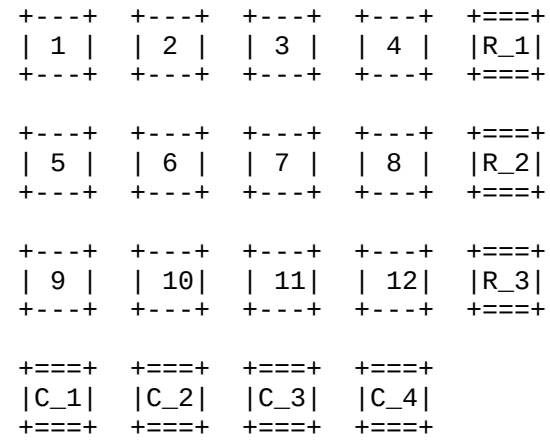


Figure 18: The resulting pattern after the second iteration

7. SDP Examples

This section provides two SDP [RFC4566] examples. The examples use the FEC grouping semantics defined in [RFC5956].

7.1. Example SDP for Flexible FEC Protection with in-band SSRC mapping

In this example, we have one source video stream and one FEC repair stream. The source and repair streams are multiplexed on different SSRCs. The repair window is set to 200 ms.

```
v=0
o=mo 1122334455 1122334466 IN IP4 fec.example.com
s=FlexFEC minimal SDP signalling Example
t=0 0
m=video 30000 RTP/AVP 96 98
c=IN IP4 143.163.151.157
a=rtpmap:96 VP8/90000
a=rtpmap:98 flexfec/90000
a=fmtp:98; repair-window=200ms
```

7.2. Example SDP for Flex FEC Protection with explicit signalling in the SDP

In this example, we have one source video stream (ssrc:1234) and one FEC repair streams (ssrc:2345). We form one FEC group with the "a=ssrc-group:FEC-FR 1234 2345" line. The source and repair streams

are multiplexed on different SSRCs. The repair window is set to 200 ms.

```
v=0
o=ali 1122334455 1122334466 IN IP4 fec.example.com
s=2-D Parity FEC with no in band signalling Example
t=0 0
m=video 30000 RTP/AVP 100 110
c=IN IP4 233.252.0.1/127
a=rtpmap:100 MP2T/90000
a=rtpmap:110 flexfec/90000
a=fmtp:110 L:5; D:10; ToP:2; repair-window:200000
a=ssrc:1234
a=ssrc:2345
a=ssrc-group:FEC-FR 1234 2345
```

8. Congestion Control Considerations

FEC is an effective approach to provide applications resiliency against packet losses. However, in networks where the congestion is a major contributor to the packet loss, the potential impacts of using FEC SHOULD be considered carefully before injecting the repair flows into the network. In particular, in bandwidth-limited networks, FEC repair flows may consume most or all of the available bandwidth and consequently may congest the network. In such cases, the applications MUST NOT arbitrarily increase the amount of FEC protection since doing so may lead to a congestion collapse. If desired, stronger FEC protection MAY be applied only after the source rate has been reduced [I-D.singh-rmcat-adaptive-fec].

In a network-friendly implementation, an application SHOULD NOT send/receive FEC repair flows if it knows that sending/receiving those FEC repair flows would not help at all in recovering the missing packets. However, it MAY still continue to use FEC if considered for bandwidth estimation instead of speculatively probe for additional capacity [Holmer13][Nagy14]. It is RECOMMENDED that the amount of FEC protection is adjusted dynamically based on the packet loss rate observed by the applications.

In multicast scenarios, it may be difficult to optimize the FEC protection per receiver. If there is a large variation among the levels of FEC protection needed by different receivers, it is RECOMMENDED that the sender offers multiple repair flows with different levels of FEC protection and the receivers join the corresponding multicast sessions to receive the repair flow(s) that is best for them.

Editor's note: Additional congestion control considerations regarding the use of 2-D parity codes should be added here.

9. Security Considerations

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550] and in any applicable RTP profile. The main security considerations for the RTP packet carrying the RTP payload format defined within this memo are confidentiality, integrity and source authenticity. Confidentiality is achieved by encrypting the RTP payload. Integrity of the RTP packets is achieved through a suitable cryptographic integrity protection mechanism. Such a cryptographic system may also allow the authentication of the source of the payload. A suitable security mechanism for this RTP payload format should provide confidentiality, integrity protection, and at least source authentication capable of determining if an RTP packet is from a member of the RTP session.

Note that the appropriate mechanism to provide security to RTP and payloads following this memo may vary. It is dependent on the application, transport and signaling protocol employed. Therefore, a single mechanism is not sufficient, although if suitable, using the Secure Real-time Transport Protocol (SRTP) [RFC3711] is recommended. Other mechanisms that may be used are IPsec [RFC4301] and Transport Layer Security (TLS) [RFC5246] (RTP over TCP); other alternatives may exist.

10. IANA Considerations

New media subtypes are subject to IANA registration. For the registration of the payload formats and their parameters introduced in this document, refer to Section 5.

11. Acknowledgments

Some parts of this document are borrowed from [RFC5109]. Thus, the author would like to thank the editor of [RFC5109] and those who contributed to [RFC5109].

12. Change Log

Note to the RFC-Editor: please remove this section prior to publication as an RFC.

12.1. draft-ietf-payload-flexible-fec-scheme-02

FEC packet format changed as per discussions in IETF94, Tokyo.

Added section on non-parity codes.

Registration of application/flexfec-raptor.

12.2. draft-ietf-payload-flexible-fec-scheme-01

FEC packet format changed as per discussions in IETF93, Prague.

Replaced non-interleaved-parityfec and interleaved-parity-fec with flexfec.

SDP simplified for the case when association to RTP is made in the FEC header and not in the SDP.

12.3. draft-ietf-payload-flexible-fec-scheme-00

Initial WG version, based on draft-singh-payload-1d2d-parity-scheme-00.

12.4. draft-singh-payload-1d2d-parity-scheme-00

This is the initial version, which is based on draft-ietf-fecframe-1d2d-parity-scheme-00. The following are the major changes compared to that document:

- Updated packet format with 16-, 48-, 112- bitmask.
- Updated the sections on: repair packet construction, source packet construction.
- Updated the media type registration and aligned to RFC6838.

12.5. draft-ietf-fecframe-1d2d-parity-scheme-00

- Some details were added regarding the use of CNAME field.
- Offer-Answer and Declarative Considerations sections have been completed.
- Security Considerations section has been completed.
- The timestamp field definition has changed.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <http://www.rfc-editor.org/info/rfc3264>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <http://www.rfc-editor.org/info/rfc3550>.

[RFC3555] Casner, S. and P. Hoschka, "MIME Type Registration of RTP Payload Formats", RFC 3555, DOI 10.17487/RFC3555, July 2003, <http://www.rfc-editor.org/info/rfc3555>.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <http://www.rfc-editor.org/info/rfc4566>.

[RFC5956] Begen, A., "Forward Error Correction Grouping Semantics in the Session Description Protocol", RFC 5956, DOI 10.17487/RFC5956, September 2010, <http://www.rfc-editor.org/info/rfc5956>.

[RFC6363] Watson, M., Begen, A., and V. Roca, "Forward Error Correction (FEC) Framework", RFC 6363, DOI 10.17487/RFC6363, October 2011, <http://www.rfc-editor.org/info/rfc6363>.

[RFC6682] Watson, M., Stockhammer, T., and M. Luby, "RTP Payload Format for Raptor Forward Error Correction (FEC)", RFC 6682, DOI 10.17487/RFC6682, August 2012, <http://www.rfc-editor.org/info/rfc6682>.

[RFC6709] Carpenter, B., Aboba, B., Ed., and S. Cheshire, "Design Considerations for Protocol Extensions", RFC 6709, DOI 10.17487/RFC6709, September 2012, <http://www.rfc-editor.org/info/rfc6709>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <http://www.rfc-editor.org/info/rfc6838>.

[RFC7022] Begen, A., Perkins, C., Wing, D., and E. Rescorla, "Guidelines for Choosing RTP Control Protocol (RTCP) Canonical Names (CNAMEs)", RFC 7022, DOI 10.17487/RFC7022, September 2013, <http://www.rfc-editor.org/info/rfc7022>.

13.2. Informative References

[Holmer13] Holmer, S., Shemer, M., and M. Paniconi, "Handling Packet Loss in WebRTC", Proc. of IEEE International Conference on Image Processing (ICIP 2013) , 9 2013.

[I-D.singh-rmcat-adaptive-fec] Varun, V., Nagy, M., Ott, J., and L. Eggert, "Congestion Control Using FEC for Conversational Media", draft-singh-rmcat-adaptive-fec-03 (work in progress), March 2016.

[Nagy14] Nagy, M., Singh, V., Ott, J., and L. Eggert, "Congestion Control using FEC for Conversational Multimedia Communication", Proc. of 5th ACM Internation Conference on Multimedia Systems (MMSys 2014) , 3 2014.

[RFC2326] Schulzrinne, H., Rao, A., and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, DOI 10.17487/RFC2326, April 1998, <http://www.rfc-editor.org/info/rfc2326>.

[RFC2733] Rosenberg, J. and H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction", RFC 2733, DOI 10.17487/RFC2733, December 1999, <http://www.rfc-editor.org/info/rfc2733>.

[RFC2974] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", RFC 2974, DOI 10.17487/RFC2974, October 2000, <http://www.rfc-editor.org/info/rfc2974>.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <http://www.rfc-editor.org/info/rfc3711>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <http://www.rfc-editor.org/info/rfc4301>.

[RFC5109] Li, A., Ed., "RTP Payload Format for Generic Forward Error Correction", RFC 5109, DOI 10.17487/RFC5109, December 2007, <http://www.rfc-editor.org/info/rfc5109>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[SMPTE2022-1] SMPTE 2022-1-2007, , "Forward Error Correction for Real-Time Video/Audio Transport over IP Networks", 2007.

Authors' Addresses

Varun Singh
Nemu Dialogue Systems Oy
Runeberginkatu 4c A 4
Helsinki 00100
Finland

Email: varun.singh@iki.fi
URI: http://www.callstats.io/

Ali Begen
Networked Media
Konya
Turkey

Email: ali.begen@networked.media

Mo Zanaty
Cisco
Raleigh, NC
USA

Email: mzanaty@cisco.com

Giridhar Mandyam
Qualcomm Innovation Center
5775 Morehouse Drive
San Diego, CA 92121
USA

Phone: +1 858 651 7200
Email: mandyam@quicinc.com

INTERNET-DRAFT
Intended Status: Standard Track
Expires: January 6, 2017

Q. Wei
Y. Jiang
R. Huang
Huawei

July 5, 2016

RTP Payload Format for HTTP Adaptive Streaming
draft-wei-payload-has-over-rtp-00

Abstract

This document introduces a new RTP payload format for encapsulating the HTTP Adaptive Streaming data into RTP, so that current RTP schemes can be leveraged into OTT video delivery services. For example, operators can easily deliver OTT live content through multicast to eliminating the impact of live content consumption peaks.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

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

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

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

Copyright and License Notice

Table of Contents

1 Introduction

Video consumption has exploded over the last few years as more and more consumers are watching live Over-the-Top (OTT) content on smartphones, tablets, PCs and other IP connected devices. Since OTT video services rely on HTTP adaptive streaming (HAS) technology, e.g., DASH and HTTP Live Streaming (HLS), to deliver content, so every time a user requests a piece of content, a stream is sent throughout the entire network. If a significant number of users are requesting content, the operator’s bandwidth is drained. It is usually difficult for operators to predict the popularity of live video content, especially for some major sporting events. Even when a content delivery network (CDN) is involved, the edge network may become congested, and CDN scalability could be a problem. All of this leads to a poor Quality of Experience (QoE) for users.

The most effective solution is to use multicast technology, even for OTT live content delivery. Through multicast technology, operators can stream live content only once in their network, regardless of the number of viewers watching, which eliminates the impact of live content consumption peaks.

This document introduces a new RTP payload format for encapsulating the HAS data into RTP, so that current RTP schemes can be leveraged into OTT video delivery services. For example, operators can easily deliver OTT live content through multicast to eliminating the impact of live content consumption peaks.

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 RFC 2119 [RFC2119].

This document uses the following terms:

3. Existing Technologies

3.1 HTTP Adaptive Streaming

HTTP adaptive streaming has become a popular approach in video commercial deployments. The multimedia content is captured, divided into small segments, and stored on an HTTP server. The consuming user first obtains the manifest file, e.g., the Media Presentation Description (MPD), which describes a manifest of the available segments information, corresponding bitrates, their URL addresses, and other characteristics. Based on this, the consuming user selects

the appropriate encoded alternative and starts streaming of the content by fetching the segments using HTTP GET requests in unicast.

MPEG developed the specification, known as MPEG Dynamic Adaptive Streaming over HTTP (DASH), to standardize the MPD and the segment formats. Other private mechanisms like Apple's HTTP Live Streaming (HLS) [HLS] are also popular.

HAS is a typical client pull model. All the manifest files, HAS segments, and etc., are pulled from the HTTP server one after another by the clients issuing HTTP requests.

HTTP adaptive streaming is very efficient for the usage of Video on Demand (VOD). However, when delivering live content simultaneously to millions of users, this becomes quite a problem. The peak bandwidth in video consumption is simply too much for an operator to handle since each viewer counts as a separate unicast session. As live OTT multi-screen video consumption shows no signs of slowing down, a traditional unicast delivery method is becoming too expensive in terms of bandwidth and investments that must be made to maintain the network. Partnering with a CDN provider only helps optimize the traffic on the backbone for known content. Additional infrastructure investment is still required at the edge of the network to absorb the load, but is too costly of an unertaking and would only be a temporary solution, as there would always be a need for more servers when live OTT consumption increases.

3.2 Multicast Adaptive Bit Rate (Multicast-ABR)

Operators are seeking ways to improve the quality of services available, while also creating more balanced and effective delivery of data to enhance the operators' cost-efficiency, and reduce wastage across increasingly constrained bandwidths. Multicast-ABR, specified in [CableLabs], is one of the innovations.

Multicast-ABR leverages HTTP streaming into multicast by keeping the different alternatives in separate multicast groups, so that smart network nodes or clients are able to select an appropriate rate by joining the correct multicast and delivering these segments to clients. And multicast-ABR uses NACK-Oriented Reliable Multicast (NORM) [RFC5740] to deliver HTTP adaptive streaming data in multicast.

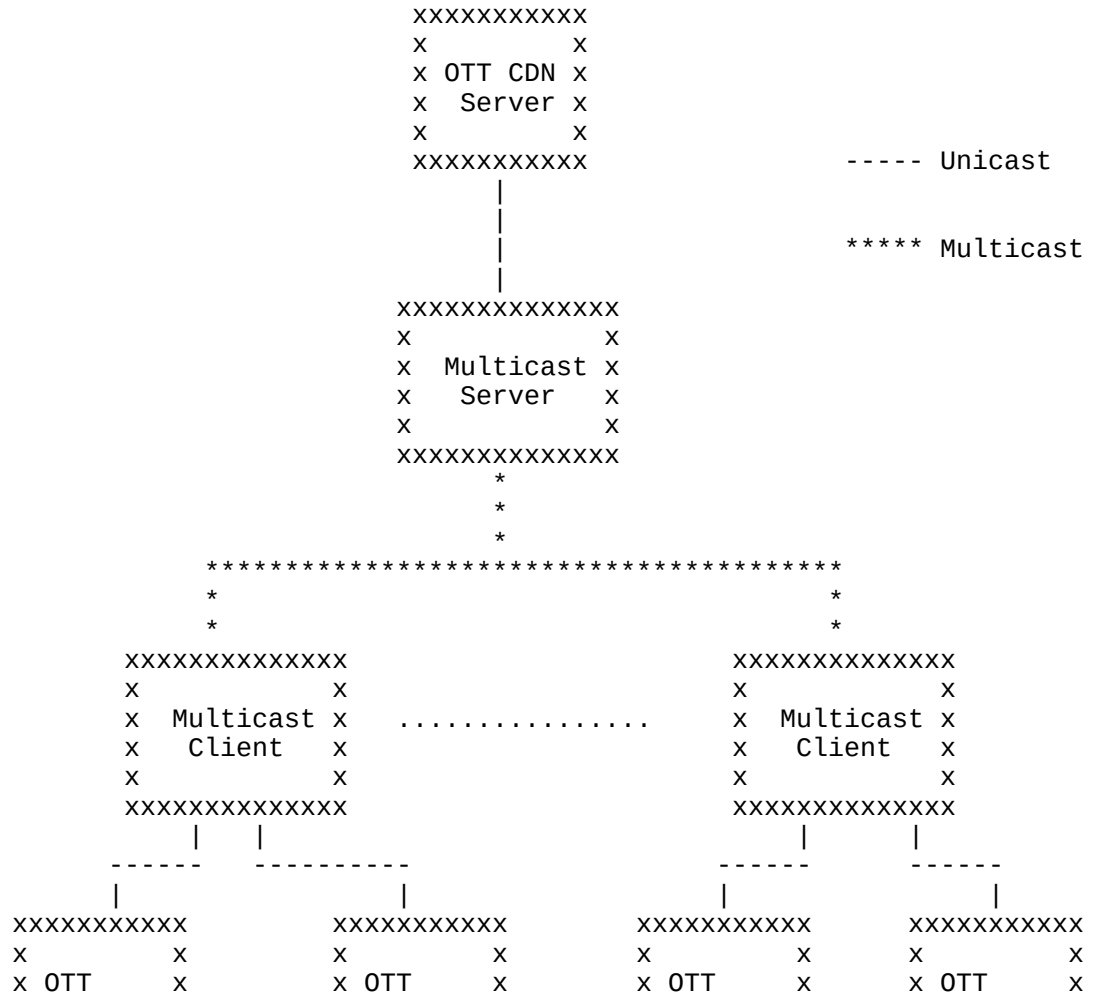
Multicast-ABR is a low cost and easy to deploy solution that allows operators to see multicast gains on all in-home devices leveraging their TV Everywhere infrastructure. However, using NORM to convey HTTP adaptive streaming data has 3 shortcomings: Firstly, NORM has no fast channel change (FCC) mechanisms, like [RFC6285], so that

changing different video resolutions may take some time and cause video frame freeze. Secondly, some telcom operators only have IPTV multicast platform, which may not support NORM protocol. Thirdly, NORM is not aware of the media timing in a way that RTP is as RTP is nature to handle multimedia.

Based on this, using RTP to deliver HTTP adaptive streaming data could be an alternative.

4. Overview of HTTP Adaptive Streaming over RTP

Figure 1 shows the architecture for HTTP adaptive streaming over RTP, which is similar to the ones defined in Multicast-ABR.



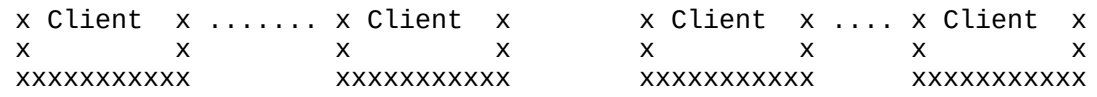


Figure 1: Architecture of HTTP Adaptive Streaming over Mutlicast

In this figure, the multicast server near the head-end taking a standard HAS stream (e.g., a DASH stream) as input and converting it into a multicast channel, so that the live streams can be conveyed in multicast down to its receivers, i.e., multicast clients. Different resolutions or bitrates are kept in different multicast groups.

Multicast server: It is responsible for converting the HAS streams into multicast data, and providing the multicast service to its receivers.

Multicast client: It is responsible for terminating the multicast. It can be a network device, deployed inside the ISP managed network, e.g., the home gateways, broadband network gateways (BNG), or optical line terminals (OLT), which converts the multicast streams back into unicast, so that all the compatible devices in the home network could receive the stream without modifying the end-use applications. Or, it can also be a user device which supports multicast.

As indicated in the figure, HAS over RTP will be used between the multicast server and the multicast clients.

Unlike HTTP, RTP is based on a push model where the server actively and continuously pushes the data to the client without the round trips between the client and server for requests. In essence, the manifest files are unnecessary and the receivers can handle the HAS data without the manifest files when leveraging HAS over RTP. However, Since the multicast client still has the requirement to work as converting multicast to traditional HAS mechanism, the manifest files should be considered in transmission to reduce the workload of the multicast client and traditional HAS client.

There are two ways to send the manifest files from the multicast server to the multicast receivers. One way is to send them out of band, e.g., through HTTP. It is reliable, but requires additional bandwidth and round-trip time. The other way is to send them in the RTP payload, which requires to handle losses and partial receives.

5. HTTP Adaptive Streaming Payload

This section specifies the format of the RTP payload of HTTP adaptive streaming data. The structure of the payload is illustrated as Figure 2. This payload format uses the fields of the header in a manner consistent with that specification.

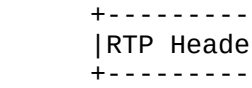


Figure 2: Packet Structure with RTP Header

5.1. RTP Header Definitions

The format of RTP header is specified in [RFC3550] and is shown as Figure 3 for convenience.

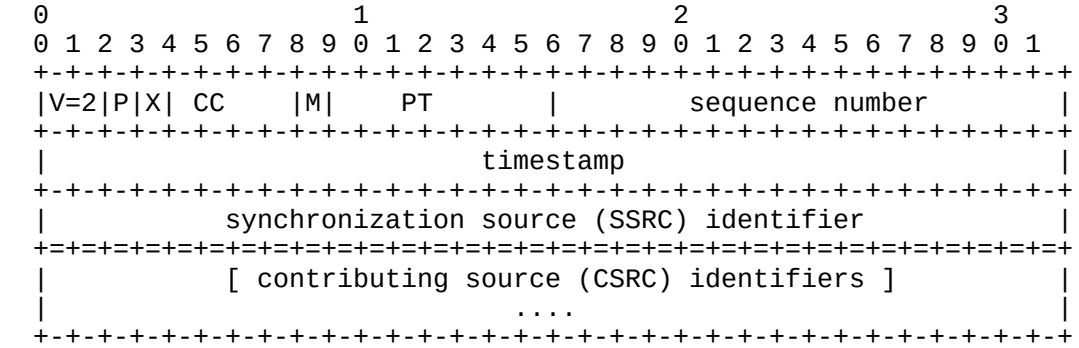


Figure 3: RTP Header Defined in [RFC3550]

The RTP header information to be set according to this RTP payload format is set as followings:

Market bit (M): 1 bits

The market bit set "1" SHALL indicate the last RTP packet of the media segment, carried in the current RTP stream. This is in line with the normal use of the M bit in video formats to allow an efficient playout buffer handling.

Payload Type (PT): 7 bits

The assignment of an RTP payload type for this new packet format is outside of the scope of this document and will not be specified here. The assignment of a payload type has to be performed either through the profile used or in a dynamic way.

Sequence Number (SN): 16 bits

Set and used in accordance with [RFC3550].

Timestamp: 16 bits

The RTP timestamp is set to the sampling timestamp of the content. The clock rate is specified dynamically through non-RTP means. If no clock rate is signaled, 90 kHz MUST be used.

When a media segment or a information list is encapsulated into several RTP packets, each of them shares the same timestamp.

5.2. Payload Definitions

The format of the HAS payload is illustrated in Figure 4.

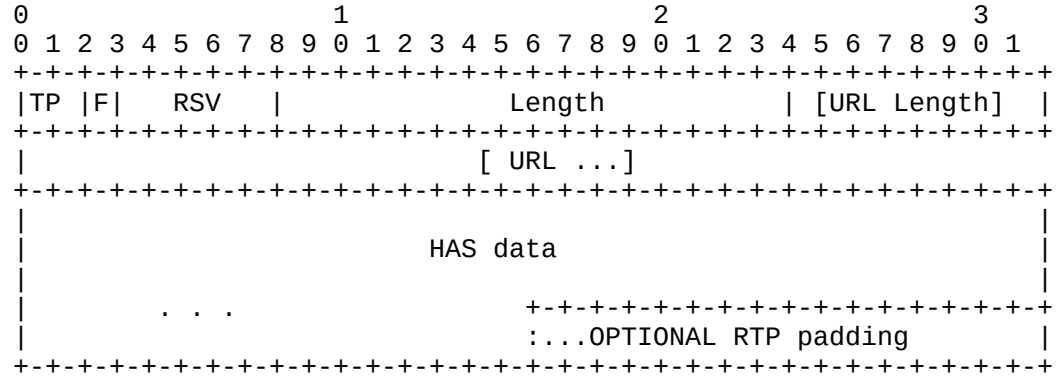


Figure 4: Format for HTTP Adaptive Streaming Payload

TYPE (TP): 2 bits

This field indicates the content type of this payload:

TYPE=0: Manifest - Information about the media, in particular the list of segments making up the media. For example, MPD, Playlist.

TYPE=1: Initial information - Information required to initialize the video decoder. For example, DASH initialization segment. This information is optional.

TYPE=2: Media segment - A part or the whole of the HAS media segment.

TYPE=3: Reserved

When HSPT=0, TP=0 means the payload carries a MPD file; TP=1 means the payload carries an initialization Segment; and TP=2 indicates a media segment content. When HSPT=1, TP=0 means the payload carries Playlist file; TP=2 indicates a media segment content; and TP=1 will never be used.

Fragmentation (F): 1 bit

If the fragmentation is set, it indicates the received packet is a fragment and can not be decoded correctly until all the fragments belong to the same content are received. The fragments belong to the same content are ordered by their sequence numbers, and share the timestamp.

If the fragmentation is set, URL length field and URL field will be omitted.

RSV : 5 bits

These bits are reserved for future use. They MUST be set to zero by senders and ignored by receivers.

Length: 16 bits

The size of the RTP payload in bytes, excluding the RTP header but including the payload header.

URL length: 8 bits

The size of the URL field in bytes, including the URL length. This field only appears when the fragmentation is set.

URL: bits defined by URL length.

This field indicates the URL of the content. Examples would be "/PLTV/88888888/224/3221225484/3221225484.mpd", or "Stream_1_1944000". It facilitates associating the content with HTTP request so that receivers can easily turn it into HAS scheme when receiving it. It is used to relate the RTP packet with corresponding HAS segment specified in the manifest.

5.3. Packetization Consideration

Senders of this payload SHOULD transmit the HAS data, e.g., MPD or segments, by encapsulating the whole HTTP packet so as to reduce

receivers' processing if they wish to convert them into conventional HTTP streaming scheme when forwarding these data to final end users and facilitate the support of different user devices.

This payload format introduces a method to send the manifest or initial information through the media tunnel together with the HAS segments. This is optional. The manifest or initial information can also be sent by out of band methods like HTTP or SDP. Especially when the receiver joins the multicast, it's better to obtain the manifest or initial information by out of band ways in advance. In cases where manifest information or initial information is missing or partially missing when sent in the media tunnel, it is also suggested to retransmit them by reliable ways.

A HAS segment may be much larger than the Maximum Transmission Unit (MTU), which will result in the segmentation of the HAS segment. Basically, it is required to split application data into RTP packets so that each packet is usable, no matter what is lost. So it is suggested to do so if the multicast server has the ability and authorized to access the HAS content. However, when OTT content is encrypted to the multicast server, the frame boundary that can be decoded independently is hardly figured out. Accordingly, one fragment of the HAS segment lost will lead to the whole segment undecodable, and the receivers joining the multicast randomly cannot view the content immediately. In this case, mechanisms like FEC [RFC5109], or retransmission [RFC4585] MUST be used to alleviate packet losses. And FCC [RFC6285] SHOULD be used to ensure users who joins the multicast randomly can view the content immediately.

Another possible way to do smart fragmenting is to extend the manifest files, e.g. MPD, to allow the OTT content providers indicate the fragmentation points where independently decodable application data can be extracted. Thus, the multicast server bridging HAS into RTP would fetch the extended manifest file, then use these hints to determine how to fragment each segment into RTP packets. Since DASH is the standard in MPEG, this method requires the work in MPEG to specify the extended fragmentation points.

6 Payload Format Parameters

This section specifies the media type and the parameters identifying this RTP payload format.

6.1 Media Type Definition

The media subtype for HAS is allocated from the IETF tree.

The receiver MUST ignore any unrecognized parameter.

Type name: HAS

Subtype name: N/A

Required parameter:

has-type: This parameter indicates the HTTP adaptive streaming protocol. The value of has-type MUST be in the range of 0 to 7, inclusive. The detailed value can be seen as following.

HSPT=0: DASH
HSPT=1: Http Live Streaming (HLS)
HSPT=2-6: Reserved
HSPT=7: Profile-specific HTTP adaptive streaming

Optional parameters: TBD

Encoding considerations: This type is only defined for transfer via RTP [RFC3550].

Security considerations: See section 7 of RFCXXXX.

Published specification: N/A

Additional information: None

File extensions: none

Macintosh file type code: none

Object identifier or OID: none

Person & email address to contact for further information: Rachel Huang (rachel.huang@huawei.com)

Intended usage: COMMON

Author: See Authors' Addresses section of RFCxxxx.

Change controller: IETF Audio/Video Transport Payloads working group delegated from the IESG.

6.2 SDP Signaling

TBD.

Negotiation of the new RTP payload is required. Further details will be provided in the next versions.

7. Congestion Control

Current DASH clients do congestion control individually. When using multicast to transport HAS data, it is expected that multicast receivers have the ability to dynamically join the corresponding multicast group based on different network conditions. Multicast receivers share the same stream in one multicast group, but HAS receivers compete each other with different streams, which means the congestion control mechanisms used in HAS don't work in HAS over RTP. A new congestion control mechanism is needed to coordinate the receivers with the same problem, so that they can share the stream to obtain the best quality they can have.

8 Security Considerations

TBD.

9 IANA Considerations

TBD.

10 Acknowledgments

The authors would like to thank the following individuals who help to review this document and provide very valuable comments: Colin Perkins, Roni Even.

11 References

11.1 Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[CableLabs] "IP Multicast Adaptive Bit Rate Architecture Technical Report" http://www.cablelabs.com/wp-content/uploads/specdocs/OC-TR-IP-MULTI-ARCH-V01-1411121.pdf

11.2 Informative References

[RFC5740] Adamson, B., Bormann, C., Handley, M., and J. Macker "NACK-Oriented Reliable Multicast (NORM) Transport Protocol", RFC 5740, November 2009.

[RFC6285] Steeg, B., Begen, A., Caenegem, T., and Z. Vax "Unicast-Based Rapid Acquisition of Multicast RTP Sessions", RFC 6285, June 2011.

[HLS] Pantos, R. and W. May, "HTTP Live Streaming", https://tools.ietf.org/html/draft-pantos-http-live-streaming-19, April 2016.

Authors' Addresses

Qikun Wei
Huawei

Email: weiqikun@huawei.com

Yuping Jiang
Huawei

Email: jiangyuping@huawei.com

Rachel Huang
Huawei

Email: rachel.huang@huawei.com