Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 6, 2015

J. Uberti
Google

March 5, 2015

WebRTC Forward Error Correction Requirements
draft-ietf-rtcweb-fec-01

Abstract

This document provides information and requirements for how Forward Error Correction (FEC) should be used by WebRTC applications.

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 September 6, 2015.

Table of Contents

1. Introduction

In situations where packet loss is high, or perfect media quality is essential, Forward Error Correction (FEC) can be used to proactively recover from packet losses. This specification provides guidance on which FEC mechanisms to use, and how to use them, for WebRTC client implementations.

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 [RFC2119].

3. Types of FEC

By its name, FEC describes the sending of redundant information in an outgoing packet stream so that information can still be recovered even in the face of packet loss. There are multiple ways in which this can be accomplished; this section enumerates the various mechanisms and describes their tradeoffs.

3.1. Separate FEC Stream

This approach, as described in [RFC5956], Section 4.3, sends FEC packets as an independent SSRC-multiplexed stream, with its own SSRC and payload type. While by far the most flexible, each FEC packet will have its own IP+UDP+RTP+FEC header, leading to additional overhead of the FEC stream.

3.2. Redundant Encoding

This approach, as descibed in [RFC2198], allows for redundant data to be piggybacked on an existing primary encoding, all in a single packet. This redundant data may be an exact copy of a previous packet, or for codecs that support variable-bitrate encodings, possibly a smaller, lower-quality representation. In certain cases, the redundant data could include multiple prior packets.

Since there is only a single set of packet headers, this approach allows for a very efficient representation of primary + redundant data. However, this savings is only realized when the data all fits into a single packet (i.e. the size is less than a MTU). As a result, this approach is generally not useful for video content.

3.3. Codec-Specific In-band FEC

Some audio codecs, notably Opus [RFC6716], support their own in-band FEC mechanism, where FEC data is included in the codec payload. In the case of Opus specifically, packets deemed as important are re-encoded at a lower bitrate and added to the subsequent packet, allowing partial recovery of a lost packet. See [RFC6716], Section 2.1.7 for details.

4. FEC for Audio Content

The following section provides guidance on how to best use FEC for transmitting audio data. As indicated in Section 8 below, FEC should only be activated if network conditions warrant it, or upon explicit application request.

4.1. Recommended Mechanism

When using the Opus codec in its default (hybrid) mode, use of the built-in Opus FEC mechanism is RECOMMENDED. This provides reasonable protection of the audio stream against typical losses, with minimal overhead. [TODO: add stats]

When using variable-bitrate codecs without an internal FEC, use of [RFC2198] redundant encoding with a lower-fidelity version of

previous packet(s) is RECOMMENDED. This provides reasonable protection of the payload with moderate overhead.

When using constant-bitrate codecs, e.g. PCMU, use of [RFC2198] redundant encoding MAY be used, but note that this will result in a potentially significant bitrate increase, and that suddenly increasing bitrate to deal with losses from congestion may actually make things worse.

Because of the lower packet rate of audio encodings, usually a single packet per frame, use of a separate FEC stream comes with a higher overhead than other mechanisms, and therefore is NOT RECOMMENDED.

4.2. Negotiating Support

Support for redundant encoding can be indicated by offering "red" as a supported payload type in the offer. Answerers can reject the use of redundant encoding by not including "red" as a supported payload type in the answer.

Support for codec-specific FEC mechanisms are typically indicated via "a=fmtp" parameters. For Opus specifically, this is controlled by the "useinbandfec=1" parameter, as specified in [I-D.ietf-payload-rtp-opus]. These parameters are declarative and can be negotiated separately for either media direction.

5. FEC for Video Content

The following section provides guidance on how to best use FEC for transmitting video data. As indicated in Section 8 below, FEC should only be activated if network conditions warrant it, or upon explicit application request.

5.1. Recommended Mechanism

For video content, use of a separate FEC stream with the RTP payload format described in [I-D.ietf-payload-flexible-fec-scheme] is RECOMMENDED. The receiver can demultiplex the incoming FEC stream by SSRC and correlate it with the primary stream via the ssrc-group mechanism.

Note that this only allows the FEC stream to protect a single primary stream. Support for protecting multiple primary streams with a single FEC stream is complicated by WebRTC's 1-m-line-per-stream policy and requires further study.

5.2. Negotiating Support

To offer support for a separate FEC stream, the offerer MUST offer one of the formats described in [I-D.ietf-payload-flexible-fec-scheme], Section 5.1, as well as a ssrc-group with "FEC-FR" semantics as described in [RFC5956], Section 4.3.

Answerers can reject the use of FEC by not including FEC payloads in the answer.

6. FEC for Application Content

While WebRTC also supports the ability to send generic application data, the fact that the application can control exactly what data to send allows it to monitor packet statistics and perform its own FEC when necessary.

As a result, this document makes no recommendations regarding FEC for the underlying data transport.

7. Implementation Requirements

To support the functionality recommended above, implementations MUST support the redundant encoding mechanism described in [RFC2198] and the FEC mechanism described in [RFC5956] and [I-D.ietf-payload-flexible-fec-scheme].

Implementations MAY support additional FEC mechanisms if desired, e.g. [RFC5109].

8. Adaptive Use of FEC

Since use of FEC causes redundant data to be transmitted, this will lead to less bandwidth available for the primary encoding, when in a bandwidth-constrained environment. Given this, WebRTC implementations SHOULD only transmit FEC data when network conditions indicate that this is advisable (e.g. by monitoring transmit packet loss data from RTCP Receiver Reports), or the application indicates it is willing to pay a quality penalty to proactively avoid losses.

9. Security Considerations

This document makes recommendations regarding which FEC mechanisms to use. The security considerations for each individual mechanism are enumerated in their respective documents.

10. IANA Considerations

This document requires no actions from IANA.

11. Acknowledgements

Several people provided significant input into this document, including Jonathan Lennox, Giri Mandyam, Varun Singh, Tim Terriberry, and Mo Zanaty.

12. References

12.1. Normative References

[I-D.ietf-payload-flexible-fec-scheme]
Singh, V., Begen, A., and M. Zanaty, "RTP Payload Format for Non-Interleaved and Interleaved Parity Forward Error Correction (FEC)", draft-ietf-payload-flexible-fec-scheme-00 (work in progress), February 2015.

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

[RFC2198] Perkins, C., Kouvelas, I., Hodson, O., Hardman, V., Handley, M., Bolot, J., Vega-Garcia, A., and S. Fosse-Parisis, "RTP Payload for Redundant Audio Data", RFC 2198, September 1997.

[RFC5956] Begen, A., "Forward Error Correction Grouping Semantics in the Session Description Protocol", RFC 5956, September 2010.

12.2. Informative References

[I-D.ietf-payload-rtp-opus]
Spittka, J., Vos, K., and J. Valin, "RTP Payload Format for the Opus Speech and Audio Codec", draft-ietf-payload-rtp-opus-08 (work in progress), February 2015.

[RFC5109] Li, A., "RTP Payload Format for Generic Forward Error Correction", RFC 5109, December 2007.

[RFC6716] Valin, JM., Vos, K., and T. Terriberry, "Definition of the Opus Audio Codec", RFC 6716, September 2012.

Appendix A. Change log

Changes in draft -01:

- Tweaked abstract/intro text that was ambiguously normative.
- Removed text on FEC for Opus in CELT mode.
- Changed RFC 2198 recommendation for PCMU to be MAY instead of NOT RECOMMENDED, based on list feedback.
- Explicitly called out application data as something not addressed in this document.
- Updated flexible-fec reference.

Changes in draft -00:

- Initial version, from sidebar conversation at IETF 90.

Author's Address

Justin Uberti
Google
747 6th Ave S
Kirkland, WA 98033
USA

Email: justin@uberti.name