Forward Error Correction (FEC) Framework Extension to Sliding Window Codes

Vincent Roca, Ali Begen

TSVWG V. Roca
Internet-Draft INRIA
Intended status: Standards Track A. Begen
Expires: January 18, 2018 Networked Media
July 17, 2017

Forward Error Correction (FEC) Framework Extension to Sliding Window Codes
draft-ietf-tsvwg-fecframe-ext-00

Abstract

RFC 6363 describes a framework for using Forward Error Correction (FEC) codes with applications in public and private IP networks to provide protection against packet loss. The framework supports applying FEC to arbitrary packet flows over unreliable transport and is primarily intended for real-time, or streaming, media. However FECFRAME as per RFC 6363 is restricted to block FEC codes. The present document extends FECFRAME to support FEC Codes based on a sliding encoding window, in addition to Block FEC Codes, in a backward compatible way. During multicast/broadcast real-time content delivery, the use of sliding window codes significantly improves robustness in harsh environments, with less repair traffic and lower FEC-related added latency.

Table of Contents

1. Introduction

Many applications need to transport a continuous stream of packetized data from a source (sender) to one or more destinations (receivers) over networks that do not provide guaranteed packet delivery. In particular packets may be lost, which is strictly the focus of this document: we assume that transmitted packets are either received without any corruption or totally lost (e.g., because of a congested router, of a poor signal-to-noise ratio in a wireless network, or because the number of bit errors exceeds the correction capabilities of a low-layer error correcting code).

For these use-cases, Forward Error Correction (FEC) applied within the transport or application layer, is an efficient technique to improve packet transmission robustness in presence of packet losses (or "erasures"), without going through packet retransmissions that create a delay often incompatible with real-time constraints. The FEC Building Block defined in [RFC5052] provides a framework for the definition of Content Delivery Protocols (CDPs) that make use of separately defined FEC schemes. Any CDP defined according to the requirements of the FEC Building Block can then easily be used with any FEC scheme that is also defined according to the requirements of the FEC Building Block.

Then FECFRAME [RFC6363] provides a framework to define Content Delivery Protocols (CDPs) that provide FEC protection for arbitrary packet flows over unreliable transports such as UDP. It is primarily intended for real-time or streaming media applications, using broadcast, multicast, or on-demand delivery.

However [RFC6363] only considers block FEC schemes defined in accordance with the FEC Building Block [RFC5052] (e.g., [RFC6681], [RFC6816] or [RFC6865]). These codes require the input flow(s) to be segmented into a sequence of blocks. Then FEC encoding (at a sender or an encoding middlebox) and decoding (at a receiver or a decoding middlebox) are both performed on a per-block basis. This approach has major impacts on FEC encoding and decoding delays. The data packets of continuous media flow(s) can be sent immediately, without delay. But the block creation time, that depends on the number k of source symbols in this block, impacts the FEC encoding delay since encoding requires that all source symbols be known. This block creation time also impacts the decoding delay a receiver will experience in case of erasures, since no repair symbol for the current block can be received before. Therefore a good value for the block size is necessarily a balance between the maximum decoding latency at the receivers (which must be in line with the most stringent real-time requirement of the protected flow(s), hence an incentive to reduce the block size), and the desired robustness against long loss bursts (which increases with the block size, hence an incentive to increase this size).

This document extends [RFC6363] in order to also support FEC codes based on a sliding encoding window (A.K.A. convolutional codes). This encoding window, either of fixed or variable size, slides over the set of source symbols. FEC encoding is launched whenever needed, from the set of source symbols present in the sliding encoding window at that time. This approach significantly reduces FEC-related latency, since repair symbols can be generated and sent on-the-fly, at any time, and can be regularly received by receivers to quickly recover packet losses. Using sliding window FEC codes is therefore highly beneficial to real-time flows, one of the primary targets of FECFRAME. [RLC-ID] provides an example of such FEC Scheme for FECFRAME, built upon the simple sliding window Random Linear Codes (RLC).

This document is fully backward compatible with [RFC6363] that it extends but does not replace. Indeed:

- o this extension does not prevent nor compromize in any way the support of block FEC codes. Both types of codes can nicely co-exist, just like different block FEC schemes can co-exist;

- o any receiver, for instance a legacy receiver that only supports block FEC schemes, can easily identify the FEC scheme used in a FECFRAME session thanks to the associated SDP file and its FEC Encoding ID information (i.e., the "encoding-id=" parameter of a "fec-repair-flow" attribute, [RFC6364]). This mechanism is not specific to this extension but is the basic approach for a FECFRAME receiver to determine whether or not it supports the FEC scheme used in a given FECFRAME session;

This document leverages on [RFC6363] and re-uses its structure. It proposes new sections specific to sliding window FEC codes whenever required. The only exception is Section Section 3 that provides a quick summary of FECFRAME in order to facilitate the understanding of this document to readers not familiar with the concepts and terminology.

2. Definitions and Abbreviations

The following list of definitions and abbreviations is copied from [RFC6363], adding only the Block/sliding window FEC Code and Encoding/Decoding Window definitions:

Application Data Unit (ADU): The unit of source data provided as payload to the transport layer.

ADU Flow: A sequence of ADUs associated with a transport-layer flow identifier (such as the standard 5-tuple {source IP address, source port, destination IP address, destination port, transport protocol}).

AL-FEC: Application-layer Forward Error Correction.

Application Protocol: Control protocol used to establish and control the source flow being protected, e.g., the Real-Time Streaming Protocol (RTSP).

Content Delivery Protocol (CDP): A complete application protocol specification that, through the use of the framework defined in this document, is able to make use of FEC schemes to provide FEC capabilities.

FEC Code: An algorithm for encoding data such that the encoded data flow is resilient to data loss. Note that, in general, FEC codes may also be used to make a data flow resilient to corruption, but that is not considered in this document.

Block FEC Code: An FEC Code that operates in a block manner, i.e., for which the input flow MUST be segmented into a sequence of blocks, FEC encoding and decoding being performed independently on a per-block basis.

Sliding Window (or Convolutional) FEC Code: An FEC Code that can generate repair symbols on-the-fly, at any time, from the set of source symbols present in the sliding encoding window at that time.

FEC Framework: A protocol framework for the definition of Content Delivery Protocols using FEC, such as the framework defined in this document.

FEC Framework Configuration Information: Information that controls the operation of the FEC Framework.

FEC Payload ID: Information that identifies the contents of a packet with respect to the FEC scheme.

FEC Repair Packet: At a sender (respectively, at a receiver), a payload submitted to (respectively, received from) the transport protocol containing one or more repair symbols along with a Repair FEC Payload ID and possibly an RTP header.

FEC Scheme: A specification that defines the additional protocol aspects required to use a particular FEC code with the FEC Framework.

FEC Source Packet: At a sender (respectively, at a receiver), a payload submitted to (respectively, received from) the transport protocol containing an ADU along with an optional Explicit Source FEC Payload ID.

Protection Amount: The relative increase in data sent due to the use of FEC.

Repair Flow: The packet flow carrying FEC data.

Repair FEC Payload ID: A FEC Payload ID specifically for use with repair packets.

Source Flow: The packet flow to which FEC protection is to be applied. A source flow consists of ADUs.

Source FEC Payload ID: A FEC Payload ID specifically for use with source packets.

Source Protocol: A protocol used for the source flow being protected, e.g., RTP.

Transport Protocol: The protocol used for the transport of the source and repair flows, e.g., UDP and the Datagram Congestion Control Protocol (DCCP).

Encoding Window: Set of Source Symbols available at the sender/ coding node that are used to generate a repair symbol, with a Sliding Window FEC Code.

Decoding Window: Set of received or decoded source and repair symbols available at a receiver that are used to decode erased source symbols, with a Sliding Window FEC Code.

Code Rate: The ratio between the number of source symbols and the number of encoding symbols. By definition, the code rate is such that 0 < code rate <= 1. A code rate close to 1 indicates that a small number of repair symbols have been produced during the encoding process.

Encoding Symbol: Unit of data generated by the encoding process. With systematic codes, source symbols are part of the encoding symbols.

Packet Erasure Channel: A communication path where packets are either lost (e.g., by a congested router, or because the number of transmission errors exceeds the correction capabilities of the physical-layer codes) or received. When a packet is received, it is assumed that this packet is not corrupted.

Repair Symbol: Encoding symbol that is not a source symbol.

Source Block: Group of ADUs that are to be FEC protected as a single block. This notion is restricted to Block FEC Codes.

Source Symbol: Unit of data used during the encoding process.

Systematic Code: FEC code in which the source symbols are part of the encoding symbols.

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. Architecture Overview

The architecture of [RFC6363], Section 3, equally applies to this FECFRAME extension and is not repeated here. However we provide hereafter a quick summary to facilitate the understanding of this document to readers not familiar with the concepts and terminology.

```
+----------------------+
|     Application      |
+----------------------+
           |
           | (1) Application Data Units (ADUs)
           |
           v
+----------------------+                           +----------------+
|    FEC Framework     |                           |                |
|                      |-------------------------->|   FEC Scheme   |
|(2) Construct source  |(3) Source Block           |                |
|    blocks            |                           |(4) FEC Encoding|
|(6) Construct FEC     |<--------------------------|                |
|    source and repair |                           |                |
|    packets           |(5) Explicit Source FEC    |                |
+----------------------+    Payload IDs            +----------------+
           |                Repair FEC Payload IDs
           |                Repair symbols
           |
           |(7) FEC source and repair packets
           v
+----------------------+
|   Transport Layer    |
|     (e.g., UDP)      |
+----------------------+
```

Figure 1: FECFRAME architecture at a sender.

The FECFRAME architecture is illustrated in Figure 1 from the sender's point of view, in case of a block FEC Scheme. It shows an application generating an ADU flow (other flows, from other applications, may co-exist). These ADUs, of variable size, must be somehow mapped to source symbols of fixed size. This is the goal of an ADU to symbols mapping process that is FEC Scheme specific (see below). Once the source block is built, taking into account both the FEC Scheme constraints (e.g., in terms of maximum source block size) and the application's flow constraints (e.g., in terms of real-time constraints), the associated source symbols are handed to the FEC Scheme in order to produce an appropriate number of repair symbols. FEC Source Packets (containing ADUs) and FEC Repair Packets (containing one or more repair symbols each) are then generated and sent using UDP (more precisely [RFC6363], Section 7, requires a transport protocol providing an unreliable datagram service, like UDP or DCCP). In practice FEC Source Packets can be sent as soon as available, without having to wait for FEC encoding to take place. In that case a copy of the associated source symbols need to be kept within FECFRAME for future FEC encoding purposes.

At a receiver (not shown), FECFRAME processing operates in a similar way, taking as input the incoming FEC source and repair packets received. In case of FEC source packet losses, the FEC decoding of the associated block may recover all (in case of successful decoding) or a subset potentially empty (otherwise) of the missing source symbols. After source symbol to ADU mapping, when lost ADUs are recovered, they are then assigned to their respective flow (see below). ADUs are returned to the application(s), either in their initial transmission order (in that case ADUs received after an erased one will be delayed until FEC decoding has taken place) or not (in that case each ADU is returned as soon as it is received or recovered), depending on the application requirements.

FECFRAME features two subtle mechanisms:

- ADUs to source symbols mapping: in order to manage variable size ADUs, FECFRAME and FEC Schemes can use small, fixed size symbols and create a mapping between ADUs and symbols. To each ADU this mechanism prepends a length field (plus a flow identifier, see below) and pads the result to a multiple of the symbol size. A small ADU may be mapped to a single source symbol while a large one may be mapped to multiple symbols. The mapping details are FEC Scheme dependant and must be defined in the associated document;

- Assignment of decoded ADUs to flows in multi-flow configurations: when multiple flows are multiplexed over the same FECFRAME instance, a problem is to assign a decoded ADU to the right flow

(UDP port numbers and IP addresses traditionally used to map incoming ADUs to flows are not recovered during FEC decoding). To make it possible, at the FECFRAME sending instance, each ADU is prepended with a flow identifier (1 byte) during the mapping to source symbols (see above). The flow identifiers are also shared between all FECFRAME instances as part of the FEC Framework Configuration Information. This (flow identifier + length + application payload + padding), called ADUI, is then FEC protected. Therefore a decoded ADUI contains enough information to assign the ADU to the right flow.

A few aspects are not covered by FECFRAME, namely:

- [RFC6363] section 8 does not detail any congestion control mechanism, but only provides high level normative requirements;
- the possibility of having feedbacks from receiver(s) is considered out of scope, although such a mechanism may exist within the application (e.g., through RCTP control messages);
- flow adaptation at a FECFRAME sender (e.g., how to set the FEC code rate based on transmission conditions) is not detailed, but it needs to comply with the congestion control normative requirements (see above).

4. Procedural Overview

4.1. General

The general considerations of [RFC6363], Section 4.1, that are specific to block FEC codes are not repeated here.

With a Sliding Window FEC Code, the FEC source packet MUST contain information to identify the position occupied by the ADU within the source flow, in terms specific to the FEC scheme. This information is known as the Source FEC Payload ID, and the FEC scheme is responsible for defining and interpreting it.

With a Sliding Window FEC Code, the FEC repair packets MUST contain information that identifies the relationship between the contained repair payloads and the original source symbols used during encoding. This information is known as the Repair FEC Payload ID, and the FEC scheme is responsible for defining and interpreting it.

The Sender Operation ([RFC6363], Section 4.2.) and Receiver Operation ([RFC6363], Section 4.3) are both specific to block FEC codes and therefore omitted below. The following two sections detail similar operations for Sliding Window FEC codes.

4.2. Sender Operation with Sliding Window FEC Codes

With a Sliding Window FEC scheme, the following operations, illustrated in Figure 2 for the case of UDP repair flows, and in Figure 3 for the case of RTP repair flows, describe a possible way to generate compliant source and repair flows:

1. A new ADU is provided by the application.

2. The FEC Framework communicates this ADU to the FEC scheme.

3. The sliding encoding window is updated by the FEC scheme. The ADU to source symbols mapping as well as the encoding window management details are both the responsibility of the FEC scheme and MUST be detailed there. Appendix A provides some hints on the way it might be performed.

4. The Source FEC Payload ID information of the source packet is determined by the FEC scheme. If required by the FEC scheme, the Source FEC Payload ID is encoded into the Explicit Source FEC Payload ID field and returned to the FEC Framework.

5. The FEC Framework constructs the FEC source packet according to [RFC6363] Figure 6, using the Explicit Source FEC Payload ID provided by the FEC scheme if applicable.

6. The FEC source packet is sent using normal transport-layer procedures. This packet is sent using the same ADU flow identification information as would have been used for the original source packet if the FEC Framework were not present (for example, in the UDP case, the UDP source and destination addresses and ports on the IP datagram carrying the source packet will be the same whether or not the FEC Framework is applied).

7. When the FEC Framework needs to send one or several FEC repair packets (e.g., according to the target Code Rate), it asks the FEC scheme to create one or several repair packet payloads from the current sliding encoding window along with their Repair FEC Payload ID.

8. The Repair FEC Payload IDs and repair packet payloads are provided back by the FEC scheme to the FEC Framework.

9. The FEC Framework constructs FEC repair packets according to [RFC6363] Figure 7, using the FEC Payload IDs and repair packet payloads provided by the FEC scheme.

10. The FEC repair packets are sent using normal transport-layer procedures. The port(s) and multicast group(s) to be used for FEC repair packets are defined in the FEC Framework Configuration Information.

```
+----------------------+
|     Application      |
+----------------------+
           |
           | (1) New Application Data Unit (ADU)
           v
+---------------------+                           +----------------+
|    FEC Framework    |                           |   FEC Scheme   |
|                     |-------------------------->|                |
|                     | (2) New ADU               |(3) Update of   |
|                     |                           |    encoding    |
|                     |<--------------------------|    window      |
|(5) Construct FEC    | (4) Explicit Source       |                |
|    source packet    |     FEC Payload ID(s)     |(7) FEC         |
|                     |<--------------------------|    encoding    |
|(9) Construct FEC    | (8) Repair FEC Payload ID |                |
|    repair packet(s) |     + Repair symbol(s)    +----------------+
+---------------------+
           |
           | (6)  FEC source packet
           | (10) FEC repair packets
           v
+----------------------+
|   Transport Layer    |
|     (e.g., UDP)      |
+----------------------+
```

Figure 2: Sender Operation with Convolutional FEC Codes

```
   +----------------------+
   |     Application      |
   +----------------------+
              |
              | (1) New Application Data Unit (ADU)
              v
   +---------------------+                           +----------------+
   |    FEC Framework    |                           |   FEC Scheme   |
   |                     |-------------------------->|                |
   |                     | (2) New ADU               |(3) Update of   |
   |                     |                           |    encoding    |
   |                     |<--------------------------|    window      |
   |(5) Construct FEC    | (4) Explicit Source       |                |
   |    source packet    |     FEC Payload ID(s)     |(7) FEC         |
   |                     |<--------------------------|    encoding    |
   |(9) Construct FEC    | (8) Repair FEC Payload ID |                |
   |    repair packet(s) |     + Repair symbol(s)    +----------------+
   +---------------------+
        |             |
        |(6) Source   |(10) Repair payloads
        |    packets  |
        |      + -- -- -- -- -+
        |      |     RTP      |
        |      +-- -- -- -- --+
        v             v
   +----------------------+
   |   Transport Layer    |
   |     (e.g., UDP)      |
   +----------------------+
```

Figure 3: Sender Operation with RTP Repair Flows

4.3. Receiver Operation with Sliding Window FEC Codes

With a Sliding Window FEC scheme, the following operations, illustrated in Figure 4 for the case of UDP repair flows, and in Figure 5 for the case of RTP repair flows. The only differences with respect to block FEC codes lie in steps (4) and (5). Therefore this section does not repeat the other steps of [RFC6363], Section 4.3, "Receiver Operation". The new steps (4) and (5) are:

4. The FEC scheme uses the received FEC Payload IDs (and derived FEC Source Payload IDs when the Explicit Source FEC Payload ID field is not used) to insert source and repair packets into the decoding window in the right way. If at least one source packet is missing and at least one repair packet has been received and the rank of the associated linear system permits it, then FEC decoding can be performed in order to recover missing source

payloads. The FEC scheme determines whether source packets have been lost and whether enough repair packets have been received to decode any or all of the missing source payloads.

5. The FEC scheme returns the received and decoded ADUs to the FEC Framework, along with indications of any ADUs that were missing and could not be decoded.

```
   +----------------------+
   |     Application      |
   +----------------------+
              ^
              |(6) ADUs
              |
   +----------------------+                           +----------------+
   |    FEC Framework     |                           |   FEC Scheme   |
   |                      |<--------------------------|                |
   |(2)Extract FEC Payload|(5) ADUs                   |(4) FEC Decoding
   |   IDs and pass IDs & |-------------------------->|                |
   |   payloads to FEC    |(3) Explicit Source FEC    +----------------+
   |   scheme             |            Payload IDs
   +----------------------+    Repair FEC Payload IDs
              ^                Source payloads
              |                Repair payloads
              |(1) FEC source
              |    and repair packets
   +----------------------+
   |   Transport Layer    |
   |     (e.g., UDP)      |
   +----------------------+
```

Figure 4: Receiver Operation with Sliding Window FEC Codes

```
+----------------------+
|     Application      |
+----------------------+
           ^
           |(6) ADUs
           |
+----------------------+                           +----------------+
|    FEC Framework     |                           |   FEC Scheme   |
|                      |<--------------------------|                |
|(2)Extract FEC Payload|(5) ADUs                   |(4) FEC Decoding|
|   IDs and pass IDs & |-------------------------->|                |
|   payloads to FEC    |(3) Explicit Source FEC    +----------------+
|   scheme             |             Payload IDs
+----------------------+    Repair FEC Payload IDs
    ^             ^         Source payloads
    |             |         Repair payloads
    |Source pkts  |Repair payloads
    |             |
+-- |- -- -- -- -- -- -+
|RTP| | RTP Processing |
|   | +-- -- -- --|-- -+
| +-- -- -- -- -- |--+ |
| | RTP Demux        | |
+-- -- -- -- -- -- -- -+
           ^
           |(1) FEC source and repair packets
           |
+----------------------+
|   Transport Layer    |
|     (e.g., UDP)      |
+----------------------+
```

Figure 5: Receiver Operation with RTP Repair Flows

5. Protocol Specification

5.1. General

This section discusses the protocol elements for the FEC Framework specific to Sliding Window FEC schemes. The global formats of source data packets (i.e., [RFC6363], Figure 6) and repair data packets (i.e., [RFC6363], Figures 7 and 8) remain the same with Sliding Window FEC codes. They are not repeated here.

5.2. FEC Framework Configuration Information

The FEC Framework Configuration Information considerations of [RFC6363], Section 5.5, equally applies to this FECFRAME extension and is not repeated here.

5.3. FEC Scheme Requirements

The FEC scheme requirements of [RFC6363], Section 5.6, mostly apply to this FECFRAME extension and are not repeated here. An exception though is the "full specification of the FEC code", item (4), that is specific to block FEC codes. The following item (4) applies instead:

4. A full specification of the Sliding Window FEC code

   This specification MUST precisely define the valid FEC-Scheme-Specific Information values, the valid FEC Payload ID values, and the valid packet payload sizes (where packet payload refers to the space within a packet dedicated to carrying encoding symbols).

   Furthermore, given valid values of the FEC-Scheme-Specific Information, a valid Repair FEC Payload ID value, a valid packet payload size, and a valid encoding window (i.e., a set of source symbols), the specification MUST uniquely define the values of the encoding symbols to be included in the repair packet payload with the given Repair FEC Payload ID value.

Additionally, the FEC scheme associated to a Sliding Window FEC Code:

- MUST define the relationships between ADUs and the associated source symbols (mapping);

- MUST define the management of the encoding window that slides over the set of ADUs. Appendix A provides a non normative example;

- MUST define the management of the decoding window, consisting of a system of linear equations (in case of a linear FEC code);

6. Feedback

The discussion of [RFC6363], Section 6, equally applies to this FECFRAME extension and is not repeated here.

7. Transport Protocols

The discussion of [RFC6363], Section 7, equally applies to this FECFRAME extension and is not repeated here.

8. Congestion Control

The discussion of [RFC6363], Section 8, equally applies to this FECFRAME extension and is not repeated here.

9. Implementation Status

Editor's notes: RFC Editor, please remove this section motivated by RFC 7942 before publishing the RFC. Thanks!

An implementation of FECFRAME extended to Sliding Window codes exists:

- o Organisation: Inria

- o Description: This is an implementation of FECFRAME extended to Sliding Window codes and supporting the RLC FEC Scheme [RLC-ID]. It is based on: (1) a proprietary implementation of FECFRAME, made by Inria and Expway for which interoperability tests have been conducted; and (2) a proprietary implementation of RLC Sliding Window FEC Codes.

- o Maturity: the basic FECFRAME maturity is "production", the FECFRAME extension maturity is "under progress".

- o Coverage: the software implements a subset of [RFC6363], as specialized by the 3GPP eMBMS standard [MBMSTS]. This software also covers the additional features of FECFRAME extended to Sliding Window codes, in particular the RLC FEC Scheme.

- o Lincensing: proprietary.

- o Implementation experience: maximum.

- o Information update date: March 2017.

- o Contact: vincent.roca@inria.fr

10. Security Considerations

This FECFRAME extension does not add any new security consideration. All the considerations of [RFC6363], Section 9, apply to this document as well.

11. Operations and Management Considerations

This FECFRAME extension does not add any new Operations and Management Consideration. All the considerations of [RFC6363], Section 10, apply to this document as well.

12. IANA Considerations

A FEC scheme for use with this FEC Framework is identified via its FEC Encoding ID. It is subject to IANA registration in the "FEC Framework (FECFRAME) FEC Encoding IDs" registry. All the rules of [RFC6363], Section 11, apply and are not repeated here.

13. Acknowledgments

TBD

14. References

14.1. 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>.

[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>.

14.2. Informative References

[MBMSTS] 3GPP, "Multimedia Broadcast/Multicast Service (MBMS); Protocols and codecs", 3GPP TS 26.346, March 2009, <http://ftp.3gpp.org/specs/html-info/26346.htm>.

[RFC5052] Watson, M., Luby, M., and L. Vicisano, "Forward Error Correction (FEC) Building Block", RFC 5052, DOI 10.17487/RFC5052, August 2007, <http://www.rfc-editor.org/info/rfc5052>.

[RFC6364] Begen, A., "Session Description Protocol Elements for the Forward Error Correction (FEC) Framework", RFC 6364, DOI 10.17487/RFC6364, October 2011, <http://www.rfc-editor.org/info/rfc6364>.

[RFC6681] Watson, M., Stockhammer, T., and M. Luby, "Raptor Forward Error Correction (FEC) Schemes for FECFRAME", RFC 6681, DOI 10.17487/RFC6681, August 2012, <http://www.rfc-editor.org/info/rfc6681>.

[RFC6816] Roca, V., Cunche, M., and J. Lacan, "Simple Low-Density Parity Check (LDPC) Staircase Forward Error Correction (FEC) Scheme for FECFRAME", RFC 6816, DOI 10.17487/RFC6816, December 2012, <http://www.rfc-editor.org/info/rfc6816>.

[RFC6865] Roca, V., Cunche, M., Lacan, J., Bouabdallah, A., and K. Matsuzono, "Simple Reed-Solomon Forward Error Correction (FEC) Scheme for FECFRAME", RFC 6865, DOI 10.17487/RFC6865, February 2013, <http://www.rfc-editor.org/info/rfc6865>.

[RLC-ID] Roca, V., "Sliding Window Random Linear Code (RLC) Forward Erasure Correction (FEC) Scheme for FECFRAME", Work in Progress, Transport Area Working Group (TSVWG) draft-roca-tsvwg-rlc-fec-scheme (Work in Progress), June 2017, <https://tools.ietf.org/html/draft-roca-tsvwg-rlc-fec-scheme>.

Appendix A. About Sliding Encoding Window Management (non Normative)

The FEC Framework does not specify the management of the sliding encoding window which is the responsibility of the FEC Scheme. This annex provides a few hints with respect to the management of this encoding window.

Source symbols are added to the sliding encoding window each time a new ADU arrives, where the following information is provided for this ADU by the FEC Framework: a description of the source flow with which the ADU is associated, the ADU itself, and the length of the ADU. This information is sufficient for the FEC scheme to map the ADU to the corresponding source symbols.

Source symbols and the corresponding ADUs are removed from the sliding encoding window, for instance:

- o after a certain delay, when an "old" ADU of a real-time flow times out. The source symbol retention delay in the sliding encoding window should therefore be initialized according to the real-time features of incoming flow(s).

- o once the sliding encoding window has reached its maximum size (there is usually an upper limit to the sliding encoding window size). In that case the oldest symbol is removed each time a new source symbol is added.

Several aspects exist that can impact the sliding encoding window management:

- o at the source flows level: real-time constraints can limit the total time source symbols can remain in the encoding window;

- o at the FEC code level: there may be theoretical or practical limitations (e.g., because of computational complexity) that limit the number of source symbols in the encoding window.

- o at the FEC scheme level: signaling and window management are intrinsically related. For instance, an encoding window composed of a non sequential set of source symbols requires an appropriate signaling to inform a receiver of the composition of the encoding window. On the opposite, an encoding window always composed of a sequential set of source symbols simplifies signaling: providing the identity of the first source symbol plus their number is sufficient.

Authors' Addresses

Vincent Roca
INRIA
Grenoble
France

EMail: vincent.roca@inria.fr

Ali Begen
Networked Media
Konya
Turkey

EMail: ali.begen@networked.media