nwcrg V. Roca, Ed.
Internet-Draft INRIA
Intended status: Informational F. Michel
Expires: September 10, 2020 UCLouvain
I. Swett
Google
M-J. Montpetit
Triangle Video
March 9, 2020

# Sliding Window Random Linear Code (RLC) Forward Erasure Correction (FEC) Schemes for QUIC

draft-roca-nwcrg-rlc-fec-scheme-for-quic-03

## Abstract

This document specifies Sliding Window Random Linear Code (RLC) Forward Erasure Correction (FEC) Schemes for the QUIC transport protocol, in order to recover from packet losses.

## Status of This Memo



## Copyright Notice

Table of Contents



## 1. Introduction

QUIC [QUIC-transport] is a transport protocol that aims at improving network performance by enabling stream multiplexing, partial reliability, and methods of recovery besides retransmission, while also improving security. This document specifies FEC schemes for Sliding Window Random Linear Code (RLC) [RFC8681] to recover from lost packets within a single QUIC stream or across several QUIC streams, compliant with the FEC coding framework for QUIC [Coding4QUIC].

The ability to add FEC coding in QUIC may be beneficial in several situations:

o  for a robust transmission of latency-sensitive traffic, for instance real-time flows, since it enables to recover packet losses independently of the round trip time;





o  for the transmission of contents to a large set of QUIC reception endpoints, since the same repair frame may help recovering several different packet losses at different receivers;

o  for multipath communications, since repair traffic adds diversity.

## 2.  Definitions and Abbreviations

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].

Terms and definitions that apply to coding are available in [nc-taxonomy].  More specifically, this document uses the following definitions:

Packet versus Symbol:  a Packet is the unit of data that is exchanged over the network while a Symbol is the unit of data that is manipulated during the encoding and decoding operations

Source Symbol:  a unit of data originating from the source that is used as input to encoding operations

Repair Symbol:  a unit of data that is the result of a coding operation

This document uses the following abbreviations:

E: size of an encoding symbol (i.e., source or repair symbol), assumed fixed (in bytes)

## 3.  Procedures

This section introduces the procedures that are used by these FEC Schemes.

## 3.1. Source Symbols Mapping

The present FEC Scheme follows the source symbols mapping specified in [Coding4QUIC]. Figure 1 illustrates this mapping.





```
   QUIC packet
            |<E-1-4>|< E-1 >|< E-1 >|< E-1 >|
     +------|----+--|-------|-------|-------|
     |Header| 0  |   Packet Payload         | 4 packet chunks
     +------|----+--|-------|-------|-------|
            /         |          |          \
           v          v          v           v
   +-+--+----+  +-+-------+  +-+-------+  +-+-------+
   |m|pn|chnk|  |m|  chunk|  |m|  chunk|  |m|  chunk| 4 source symbols
   +-+--+----+  +-+-------+  +-+-------+  +-+-------+
   |         |  |         |  |         |  |         |
   |< --E-- >|  |< --E-- >|  |< --E-- >|  |< --E-- >|
```

Figure 1: Example of source symbol mapping, when the E value is relatively small.

## 3.2. Source Symbols identification

Similarly to [RFC8681], the present FEC Scheme assigns a unique identifier (ID) to each produced source symbol. The IDs are assigned to the produced source symbols in the ascending order. The IDs start at MUST start 0 and MUST be contiguous. For any symbol with ID x, the source symbol with ID x+1 is :

- o The source symbol containing the next packet chunk in the same QUIC packet as the source symbol X, if it exists.

- o The source symbol containing the first packet chunk of the next generated FEC-protected QUIC packet

Do we want to authorize a wrapping of the source symbol ID ? It would be a lot easier if wrapping is not permitted.

### 3.3. Pseudo-Random Number Generator (PRNG)

The RLC FEC Schemes defined in this document rely on the TinyMT32 PRNG defined in [RFC8682] along with the two mapping functions to 4-bit and 8-bit unsigned integers defined in [RFC8681].

### 3.4. Coding Coefficients Generation Function

The coding coefficients, used during the encoding process, are generated at the RLC encoder by the generate_coding_coefficients() function each time a new repair symbol needs to be produced. This specification uses the generate_coding_coefficients() defined in [RFC8681].





## 4. Sliding Window RLC FEC Scheme over GF(2^^8)

This fully-specified FEC Scheme defines the Sliding Window Random Linear Codes (RLC) over GF(2^^8).

### 4.1. Formats and Codes

#### 4.1.1. Configuration Information

This section provides the RLC configuration information that needs to be shared during QUIC negotiation between the QUIC sender and receiver endpoints in order to synchronize them.

- FEC Encoding ID (8 bits): the value assigned to this fully specified FEC Scheme MUST be XXXX, as assigned by IANA (Section 6). This FEC Encoding ID is used during the QUIC negotiation to uniquely identify the RLC FEC Scheme for QUIC;

- Encoding symbol size, E (in bytes) (16 bits): a non-negative integer that indicates the size of each source and repair symbol, in bytes. This element is required both by the QUIC sender endpoint (RLC encoder) and the QUIC receiver endpoint(s) (RLC decoder).

TODO: specify exact format, with binary encoding, to be carried within the opaque 32-bit field during negotiation.

### 4.1.2. SRC FPI Frame Format

The RLC FEC Scheme requires explicit signaling of the Source Symbols it transmits. The QUIC packets whose payload is protected by FEC MUST contain an SRC FPI frame with the following format.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            ID of First Source Symbol in Packet (i)         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: SRC FPI frame format.

The SRC FPI frame contains the ID of the first source symbol contained in this packet. Each source symbol contains a packet chunk of E-1 bytes long. If the payload to protect is longer than E-1 bytes, this means that the packet contains several packet chunks. In this case the source symbol ID will increase by exactly one for each additional packet chunk contained in the payload to protect.





Note: This frame is not idempotent. In the current version of QUIC, all the frames are idempotent (but this is not especially required). It would be great to preserve this property (a quick fix would be to add the packet number in the frame, but it takes a lot of space and I don't think it is very useful). Another bad property is that the frames are not independant anymore (several SRC FPI frames contained in the same packet have a confusing meaning). I really think that the correct solution would be a (encrypted) header field but I guess it is more complicated to propose (maybe in v2 we'll have a mechanism to define dynamic encrypted header fields during negotiation).

### 4.1.3. REPAIR Frame Format

The RLC FEC Scheme requires QUIC REPAIR frames to convey enough information.  This section specifies the REPAIR frame format specific to the RLC FEC Scheme.  Note that the notion of REPAIR frame format is equivalent to the notion of Repair FEC Payload ID in [RFC8681].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              ID of First Source Symbol in EW (i)            ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Repair_Key              |              NSS              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  DT   |          NRS          |         Repair Symbols      ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: REPAIR frame format when protecting a single QUIC stream.

More precisely, the REPAIR frame format is composed of the following fields (Figure 3):

ID of the first Source Symbol in the Encoding Window (variable-size field):
   a variable-length integer specifying the ID of the first source symbol in the encoding window.  When a FEC REPAIR frame contains several repair symbols, this value applies to all of them;

Repair_Key (16-bit field):  this unsigned integer is used as a seed by the coefficient generation function (Section 3.4) in order to generate the desired number of coding coefficients.  When a FEC Repair Packet contains several repair symbols, this repair key value is that of the first repair symbol.  The remaining repair keys can be deduced by incrementing by 1 this value, up to a maximum value of 65535 after which it loops back to 0.





Number of Source Symbols in the encoding window, NSS (16-bit field):

   this unsigned integer indicates the number of source symbols in the encoding window when this repair symbol was generated.  When a REPAIR frame contains several repair symbols, the NSS value applies to all of them;

Density Threshold for the coding coefficients, DT (4-bit field):  thi s unsigned integer carries the Density Threshold (DT) used by the coding coefficient generation function Section 3.4. More precisely, it controls the probability of having a non zero coding coefficient, which equals (DT+1) / 16. When a REPAIR frame contains several repair symbols, the DT value applies to all of them;

Number of Repair Symbols contained in the frame, NRS (12-bit field):

this unsigned integer specifies the number of Repair Symbols contained in this REPAIR frame;

Repair Symbols: data for this repair symbol(s). This field is NRS*E bytes long.

#### 4.1.4. Additional Procedures

### 4.2. FEC Code Specification

This RLC FEC Scheme relies on the FEC code specification defined in [RFC8681].

#### 4.2.1. Encoding Side

[RFC8681] high level description of a Sliding Window RLC encoder also applies here to this FEC Scheme.

#### 4.2.2. Decoding Side

[RFC8681] high level description of a Sliding Window RLC decoder also applies here to this FEC Scheme.

## 5. Security Considerations

TBD





## 6. IANA Considerations

This document registers two values in the "QUIC FEC Encoding IDs" registry as follows:

- o XXXX refers to the Sliding Window Random Linear Codes (RLC) over GF(2^^8) FEC Scheme for a Single QUIC Stream, as defined in Section 4 of this document.

## 7. Acknowledgments

TBD

## 8. References

### 8.1. Normative References

[Coding4QUIC]
Swett, I., Montpetit, M-J., Roca, V., and F. Michel, "Coding for QUIC", Work in Progress, NWCRG draft-swett-nwcrg-coding-for-quic (Work in Progress), March 2020, <https://tools.ietf.org/html/draft-swett-nwcrg-coding-for-quic>.

[QUIC-transport]
Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport (Work in Progress) (work in progress), November 2019, <https://datatracker.ietf.org/doc/draft-ietf-quic-transport/>.

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

[RFC8681] Roca, V. and B. Teibi, "Sliding Window Random Linear Code (RLC) Forward Erasure Correction (FEC) Schemes for FECFRAME", RFC 8681, DOI 10.17487/RFC8681, January 2020, <https://www.rfc-editor.org/info/rfc8681>.

[RFC8682] Saito, M., Matsumoto, M., Roca, V., Ed., and E. Baccelli, "TinyMT32 Pseudorandom Number Generator (PRNG)", RFC 8682, DOI 10.17487/RFC8682, January 2020, <https://www.rfc-editor.org/info/rfc8682>.

## 8.2. Informative References

[nc-taxonomy]
Roca (Ed.) et al., V., "Taxonomy of Coding Techniques for Efficient Network Communications", Request For Comments RFC 8406, June 2018, <https://datatracker.ietf.org/doc/draft-irtf-nwcrg-network-coding-taxonomy/>.

## Authors' Addresses

Vincent Roca (editor)
INRIA
Univ. Grenoble Alpes
France

Email: vincent.roca@inria.fr

Francois Michel
UCLouvain
Louvain-la-Neuve
Belgium

Email: francois.michel@uclouvain.be

Ian Swett
Google
Cambridge, MA
US

Email: ianswett@google.com

Marie-Jose Montpetit
Triangle Video
Boston, MA
US

Email: marie@mjmontpetit.com