FecFrame   V. Roca
Internet-Draft   INRIA
Intended status: Experimental   M. Cunche
Expires: April 25, 2011   NICTA
                                                               J. Lacan
                                                          A. Bouabdallah
                                                          ISAE/LAAS-CNRS
                                                            K. Matsuzono
                                                         Keio University
                                                        October 22, 2010

        Simple Reed-Solomon Forward Error Correction (FEC) Scheme for FECFRAME
                       draft-roca-fecframe-simple-rs-01

Abstract

   This document describes a fully-specified simple FEC scheme for Reed-
   Solomon codes over GF($2^{\wedge}m$), with $2 <= m <= 16$, that can be used to
   protect arbitrary media streams along the lines defined by the
   FECFRAME framework. Reed-Solomon codes belong to the class of
   Maximum Distance Separable (MDS) codes which means they offer optimal
   protection against packet erasures. They are also systematic codes,
   which means that the source symbols are part of the encoding symbols.
   The price to pay is a limit on the maximum source block size, on the
   maximum number of encoding symbols, and a computational complexity
   higher than that of LDPC codes for instance.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

   The use of Forward Error Correction (FEC) codes is a classic solution
   to improve the reliability of unicast, multicast and broadcast
   Content Delivery Protocols (CDP) and applications. The
   [FECFRAME-FRAMEWORK] document describes a generic framework to use
   FEC schemes with media delivery applications, and for instance with
   real-time streaming media applications based on the RTP real-time
   protocol. Similarly the [RFC5052] document describes a generic
   framework to use FEC schemes with with objects (e.g., files) delivery
   applications based on the ALC [RFC5775] and NORM [RFC5740] reliable
   multicast transport protocols.

   More specifically, the [RFC5053] and [RFC5170] FEC schemes introduce
   erasure codes based on sparse parity check matrices for object
   delivery protocols like ALC and NORM. These codes are efficient in
   terms of processing but not optimal in terms of erasure recovery
   capabilities when dealing with "small" objects.

   The Reed-Solomon FEC codes described in this document belong to the
   class of Maximum Distance Separable (MDS) codes that are optimal in
   terms of erasure recovery capability. It means that a receiver can
   recover the k source symbols from any set of exactly k encoding
   symbols. These codes are also systematic codes, which means that the
   k source symbols are part of the encoding symbols. However they are
   limited in terms of maximum source block size and number of encoding
   symbols. Since the real-time constraints of media delivery
   applications usually limit the maximum source block size, this is not
   considered to be a major issue in the context of the FEC Framework
   for many (but not necessarily all) use-cases. Additionally, if the
   encoding/decoding complexity is higher with Reed-Solomon codes than
   it is with [RFC5053] or [RFC5170] codes, it remains reasonable for
   most use-cases, even in case of a software codec.

   Many applications dealing with reliable content transmission or
   content storage already rely on packet-based Reed-Solomon erasure
   recovery codes. In particular, many of them use the Reed-Solomon
   codec of Luigi Rizzo [RS-codec] [Rizzo97]. The goal of the present
   document is to specify a simple Reed-Solomon scheme that is
   compatible with this codec.

   More specifically, the [RFC5510] document introduced such Reed-
   Solomon codes and several associated FEC schemes that are compatible
   with the [RFC5052] framework. The present document inherits from
   [RFC5510] the specification of the core Reed-Solomon codes based on
   Vandermonde matrices, and specifies a simple FEC scheme that is
   compatible with the FECFRAME framework [FECFRAME-FRAMEWORK].
   Therefore this document specifies only the information specific to

the FECFRAME context and refers to [RFC5510] for the core
specifications of the codes. To that purpose, the present document
introduces:
o   the Fully-Specified FEC Scheme with FEC Encoding ID XXX that
    specifies a simple way of using of Reed-Solomon codes over
    GF(2^^m), with 2 <= m <= 16, with a simple FEC encoding for
    arbitrary packet flows;

For instance, with this scheme, a set of Application Data Units (or
ADUs) coming from one or several (resp. one) media delivery
applications (e.g., a set of RTP packets), are grouped in an ADU
block and FEC encoded as a whole. With Reed-Solomon codes over
GF(2^^8), there is a strict limit over the number of ADUs that can be
protected together, since the number of encoded symbols, n, must be
inferior or equal to 255. This constraint is relaxed when using a
higher finite field size (m > 8), at the price of an increased
computational complexity.

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].

3. Definitions Notations and Abbreviations

3.1. Definitions

This document uses the following terms and definitions. Some of them
are FEC scheme specific and are in line with [RFC5052]:
Source symbol: unit of data used during the encoding process. In
    this specification, there is always one source symbol per ADU.
Encoding symbol: unit of data generated by the encoding process.
    With systematic codes, source symbols are part of the encoding
    symbols.
Repair symbol: encoding symbol that is not a source symbol.
Code rate: the k/n ratio, i.e., 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.
Systematic code: FEC code in which the source symbols are part of
    the encoding symbols. The Reed-Solomon codes introduced in this
    document are systematic.

Source block: a block of k source symbols that are considered
   together for the encoding.
Packet Erasure Channel: a communication path where packets are
   either dropped (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.

Some of them are FECFRAME framework specific and are in line with
[FECFRAME-FRAMEWORK]:
Application Data Unit (ADU): a unit of data coming from (sender) or
   given to (receiver) the media delivery application. Depending on
   the use-case, an ADU can use an RTP encapsulation. In this
   specification, there is always one source symbol per ADU.
(Source) ADU Flow: a flow of ADUs from a media delivery application
   and to which FEC protection is applied. Depending on the use-
   case, several ADU flows can be protected together by the FECFRAME
   framework.
ADU Block: a set of ADUs that are considered together by the
   FECFRAME instance for the purpose of the FEC scheme. Along with
   the F[], L[], and Pad[] fields, they form the set of source
   symbols over which FEC encoding will be performed.
ADU Information (ADUI): a unit of data constituted by the ADU and
   the associated Flow ID, Length and Padding fields (Section 4.3).
   This is the unit of data that is used as source symbol.
FEC Framework Configuration Information: the FEC scheme specific
   information that enables the synchronization of the FECFRAME
   sender and receiver instances.
FEC Source Packet: a data packet submitted to (sender) or received
   from (receiver) the transport protocol. It contains an ADU along
   with its optional Explicit Source FEC Payload ID.
FEC Repair Packet: a repair packet submitted to (sender) or received
   from (receiver) the transport protocol. It contains a repair
   symbol along with its Repair FEC Payload ID.

The above terminology is illustrated in Figure 1 (sender's point of
view):

```
    +---------------------+
    |      Application     |
    +---------------------+
             |
  ADU flow   |  (1) Application Data Unit (ADU)
             v
    +---------------------+                         +----------------+
    |     FEC Framework    |                         |                |
    |                     |----------------------- >|   FEC Scheme    |
    |(2) Construct an ADU |  (4) Source Symbols for |                |
    |     block           |        this Source Block|(5) Perform FEC |
    |(3) Construct ADU Info|                         |     Encoding   |
    |(7) Construct FEC Src |< ------------------------|                |
    |     Packets and FEC  | (6) Ex src FEC Payload Ids,|              |
    |     Repair Packets   |     Repair FEC Payload Ids,|              |
    +---------------------+     Repair Symbols      +----------------+
         |           |
         |(8) FEC Src |(8') FEC Repair
         |    packets |     packets
         v           v
    +---------------------+
    |   Transport Layer    |
    |     (e.g., UDP )     |
    +---------------------+
```
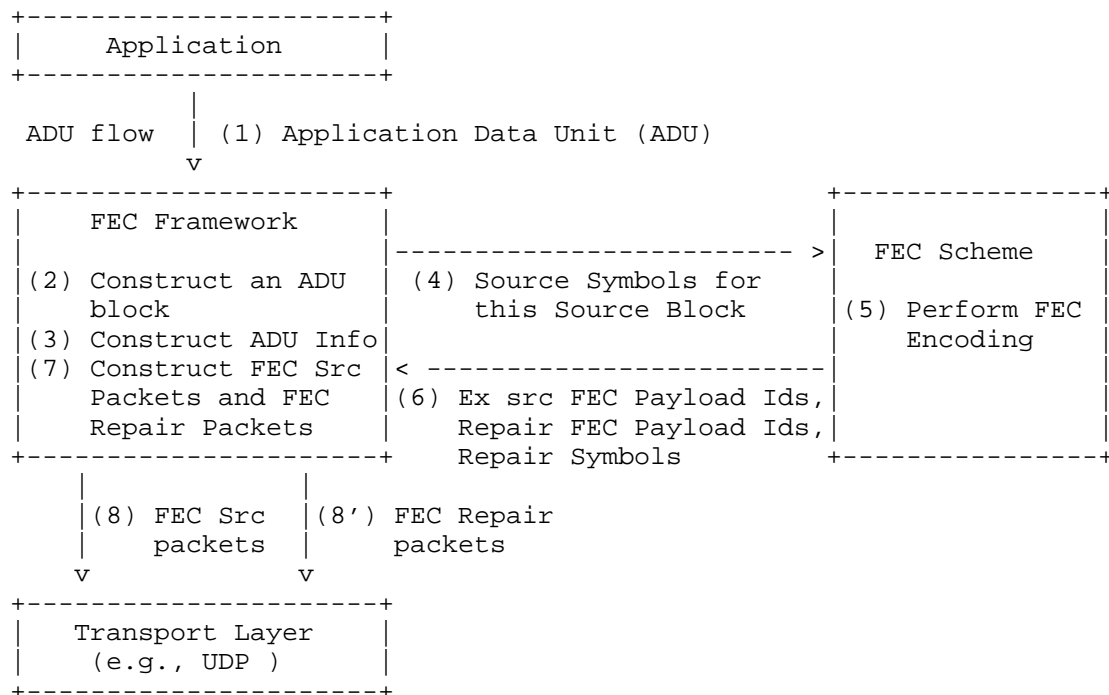
           Figure 1: Terminology used in this document (sender).

3.2. Notations

   This document uses the following notations: Some of them are FEC
   scheme specific:
   k   denotes the number of source symbols in a source block.
   max_k   denotes the maximum number of source symbols for any source
           block.
   n   denotes the number of encoding symbols generated for a source
           block.
   E   denotes the encoding symbol length in bytes.
   GF(q)   denotes a finite field (also known as Galois Field) with q
           elements. We assume that $q = 2^m$ in this document.
   m   defines the length of the elements in the finite field, in
           bits. In this document, m is such that $2 <= m <= 16$.
   q   defines the number of elements in the finite field. We have:
           $q = 2^m$ in this specification.
   CR   denotes the "code rate", i.e., the k/n ratio.

   a^^b   denotes a raised to the power b.

   Some of them are FECFRAME framework specific:
   B   denotes the number of ADUs per ADU block.
   max_B   denotes the maximum number of ADUs for any ADU block.

3.3. Abbreviations

   This document uses the following abbreviations:
   ADU   stands for Application Data Unit.
   ESI   stands for Encoding Symbol ID.
   FEC   stands for Forward Error (or Erasure) Correction code.
   FFCI   stands for FEC Framework Configuration Information.
   RS   stands for Reed-Solomon.
   MDS   stands for Maximum Distance Separable code.

4. Common Procedures Related to the ADU Block and Source Block Creation

   This section introduces the procedures that are used during the ADU
   block and the related source block creation, for the FEC scheme
   considered.

4.1. Restrictions

   This specification has the following restrictions:
   o there MUST be exactly one source symbol per ADUI, and therefore
      per ADU;
   o there MUST be exactly one repair symbol per FEC Repair Packet;
   o there MUST be exactly one source block per ADU block;

4.2. ADU Block Creation

   Several aspects must be considered, that impact the ADU block
   creation:
   o the maximum source block size (k parameter) and number of encoding
      symbols (n parameter), that are constrained by the finite field
      size (m parameter);
   o the potential real-time constraints, that impact the maximum ADU
      block size, since the larger the block size, the larger the
      decoding delay;
   We now detail each of these aspects.

   The finite field size parameter, m, defines the number of non zero
   elements in this field which is equal to: $q - 1 = 2^{\wedge\wedge}m - 1$. This q -
   1 value is also the theoretical maximum number of encoding symbols
   that can be produced for a source block. For instance, when m = 8
   (default) there is a maximum of $2^{\wedge\wedge}8 - 1 = 255$ encoding symbols. So:

k < n <= 255. Given the target FEC code rate (e.g., provided by the
end-user or upper application when starting the FECFRAME framework,
and taking into account the (known or estimated) packet loss rate),
the sender calculates:
    max_k = floor((2^^m - 1) * CR)
This max_k value leaves enough room for the sender to produce the
desired number of repair symbols. Since there is one source symbol
per ADU, max_k is also an upper bound to the maximum number of ADUs
per ADU block.

The source ADU flows usually have real-time constraints. It means
that the maximum number of ADUs of an ADU block must not exceed a
certain threshold since it directly impacts the decoding delay. It
is the role of the developer, who knows the flow real-time features,
to define an appropriate upper bound to the ADU block size, max_rt.

If we take into account these constraints, we find: max_B =
min(max_k, max_rt). Then max_B gives an upper bound to the number of
ADUs that can constitute an ADU block.

4.3. Source Block Creation

In its most general form the FECFRAME framework and the RS FEC scheme
are meant to protect a set of independent flows. Since the flows
have no relationship to one another, the ADU size of each flow can
potentially vary significantly. Even in the special case of a single
flow, the ADU sizes can largely vary (e.g., the various frames of a
"Group of Pictures (GOP) of an H.264 flow). This diversity must be
addressed since the RS FEC scheme requires a constant encoding symbol
size (E parameter) per source block. Since this specification
requires that there is only one source symbol per ADU, E must be
large enough to contain all the ADUs of an ADU block along with their
prepended 3 bytes (see below).

In situations where E is determined per source block (default,
specified by the FCCI/FSSI with S = 0, Section 5.1.1.2), E is equal
to the size of the largest ADU of this source block plus three (for
the prepended 3 bytes, see below). In this case, upon receiving the
first FEC Repair Packet for this source block, since this packet MUST
contain a single repair symbol (Section 5.1.3), a receiver determines
the E parameter used for this source block.

In situations where E is fixed (specified by the FCCI/FSSI with S =
1, Section 5.1.1.2), then E must be greater or equal to the size of
the largest ADU of this source block plus three (for the prepended 3
bytes, see below). If this is not the case, an error is returned.
How to handle this error is use-case specific (e.g., a larger E
parameter may be communicated to the receivers in an updated FFCI

message, using an appropriate mechanism) and is not considered by
this specification.

The ADU block is always encoded as a single source block. There are
a total of B <= max_B ADUs in this ADU block. For the ADU i, with 0
<= i <= B-1, 3 bytes are prepended (Figure 2):
o The first byte, FID[i] (Flow ID), contains the integer identifier
   associated to the source ADU flow to which this ADU belongs to.
   It is assumed that a single byte is sufficient, or said
   differently, that no more than 256 flows will be protected by a
   single instance of the FECFRAME framework.
o The following two bytes, L[i] (Length), contain the length of this
   ADU, in network byte order (i.e., big endian). This length is for
   the ADU itself and does not include the FID[i], L[i], or Pad[i]
   fields.

Then zero padding is added to ADU i (if needed) in field Pad[i], for
alignment purposes up to a size of exactly E bytes. The data unit
resulting from the ADU i and the F[i], L[i] and Pad[i] fields, is
called ADU Information (or ADUI). Each ADUI contributes to exactly
one source symbol to the source block.

```
                       Encoding Symbol Length (E)
 < ------------------------------------------------------------- >
 +----+----+----------------------+----------------------------+
 |F[0]|L[0]|       ADU[0]         |          Pad[0]            |
 +----+----+----------+-----------+----------------------------+
 |F[1]|L[1]| ADU[1]   |                      Pad[1]            |
 +----+----+----------+----------------------------------------+
 |F[2]|L[2]|                    ADU[2]                         |
 +----+----+------+------------------------------------------- +
 |F[3]|L[3]|ADU[3]|                         Pad[3]             |
 +----+----+------+------------------------------------------- +
 _____ _____/
                                  \/
                        simple FEC encoding

 +-------------------------------------------------------------+
 |                         Repair 4                            |
 +-------------------------------------------------------------+
 .                                                             .
 .                                                             .
 +-------------------------------------------------------------+
 |                         Repair 7                            |
 +-------------------------------------------------------------+
```
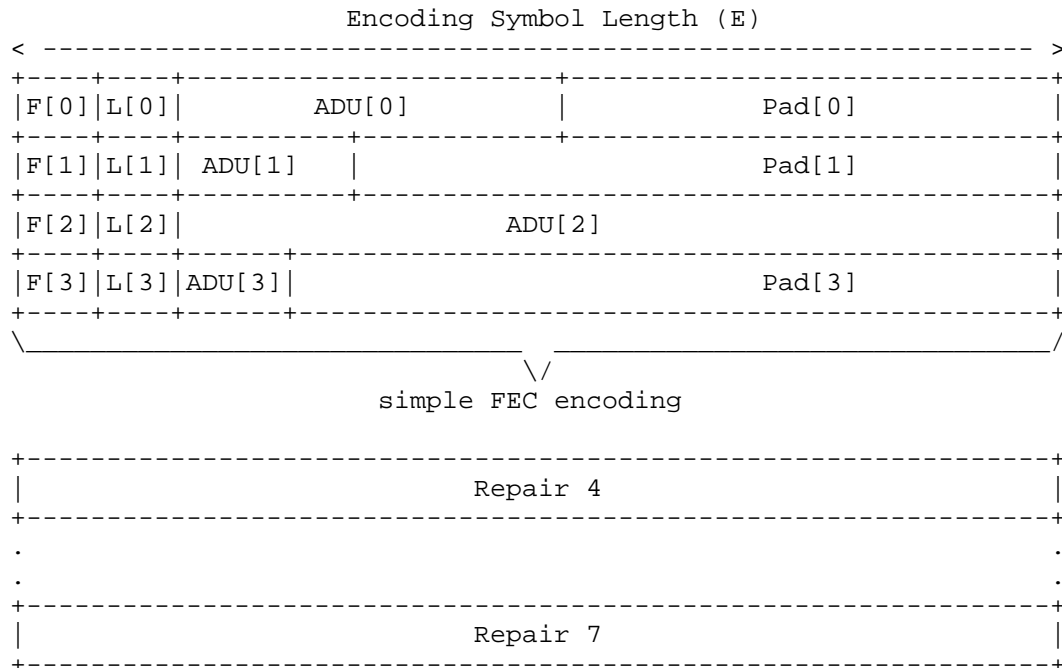
Figure 2: Source block creation, for code rate 1/2 (equal number of
      source and repair symbols, 4 in this example), and S = 0.

Note that neither the initial 3 bytes nor the optional padding are
sent over the network. However, they are considered during FEC
encoding. It means that a receiver who lost a certain FEC source
packet (e.g., the UDP datagram containing this FEC source packet)
will be able to recover the ADUI if FEC decoding succeeds. Thanks to
the initial 3 bytes, this receiver will get rid of the padding (if
any) and identify the corresponding ADU flow.

5. Simple Reed-Solomon FEC Scheme over GF(2^^m) for Arbitrary ADU Flows

   This Fully-Specified FEC Scheme specifies the use of Reed-Solomon
   codes over GF(2^^m), with 2 <= m <= 16, with a simple FEC encoding
   for arbitrary packet flows.

5.1. Formats and Codes

5.1.1. FEC Framework Configuration Information

   The FEC Framework Configuration Information (or FFCI) includes
   information that MUST be communicated between the sender and
   receiver(s). More specifically, it enables the synchronization of
   the FECFRAME sender and receiver instances. It includes both
   mandatory elements and scheme-specific elements, as detailed below.

5.1.1.1. Mandatory Information

   FEC Encoding ID: the value assigned to this fully-specified FEC
      scheme MUST be XXX, as assigned by IANA (Section 7).
   When SDP is used to communicate the FFCI, this FEC Encoding ID is
   carried in the 'encoding-id' parameter.

5.1.1.2. FEC Scheme-Specific Information

   The FEC Scheme Specific Information (FSSI) includes elements that are
   specific to the present FEC scheme. More precisely:
   Encoding symbol length (E): a non-negative integer that indicates
      either the length of each encoding symbol in bytes (strict mode,
      i.e., if S = 1), or the maximum length of any encoding symbol
      (i.e., if S = 0).
   Strict (S) flag: when set to 1 this flag indicates that the E
      parameter is valid for the whole session, unless otherwise
      notified. When set to 0 this flag indicates that the E parameter
      is only the maximum length of each encoding symbol, for the whole
      session, unless otherwise notified.

   m parameter (m): an integer that defines the length of the elements
      in the finite field, in bits. We have: 2 <= m <= 16.
   These elements are required both by the sender (RS encoder) and the
   receiver(s) (RS decoder).

   When SDP is used to communicate the FFCI, this FEC scheme-specific
   information is carried in the 'fssi' parameter in textual
   representation as specified in [SDP_ELEMENTS]. For instance:

   fssi = E:1400,S:0,m:8

   If another mechanism requires the FSSI to be carried as an opaque
   octet string (for instance after a Base64 encoding), the encoding
   format consists of the following 3 octets:
   o Encoding symbol length (E): 16 bit field.
   o Strict (S) flag: 1 bit field.
   o m parameter (m): 7 bit field.

```
 0                   1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Encoding Symbol Length (E)  |S|      m      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
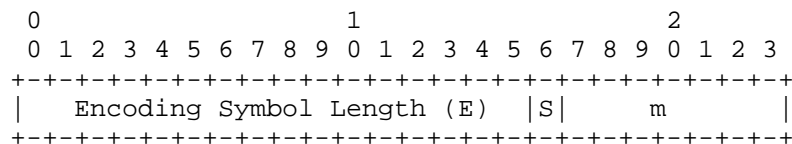
                   Figure 3: FSSI encoding format.

5.1.2. Explicit Source FEC Payload ID

   A FEC source packet MUST contain an Explicit Source FEC Payload ID
   that is appended to the end of the packet as illustrated in Figure 4.

```
+-------------------------------+
|           IP Header           |
+-------------------------------+
|        Transport Header       |
+-------------------------------+
|              ADU              |
+-------------------------------+
| Explicit Source FEC Payload ID |
+-------------------------------+
```
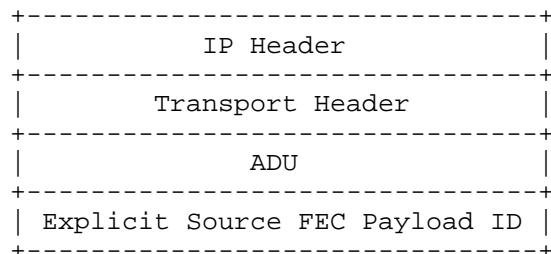
    Figure 4: Structure of a FEC Source Packet with the Explicit Source
                          FEC Payload ID.

   More precisely, the Explicit Source FEC Payload ID is composed of the
   Source Block Number, the Encoding Symbol ID, and the Source Block
   Length. The length of the first two fields depends on the m
   parameter (transmitted separately in the FFCI, Section 5.1.1.2):

Source Block Number (SBN) (32-m bit field): this field identifies
   the source block to which this FEC source packet belongs.
Encoding Symbol ID (ESI) (m bit field): this field identifies the
   source symbol contained in this FEC source packet. This value is
   such that 0 <= ESI <= k - 1 for source symbols.
Source Block Length (k) (16 bit field): this field provides the
   number of source symbols for this source block, i.e., the k
   parameter. If 16 bits are too much when m <= 8, it is needed when
   8 < m <= 16. Therefore we provide a single common format
   regardless of m.
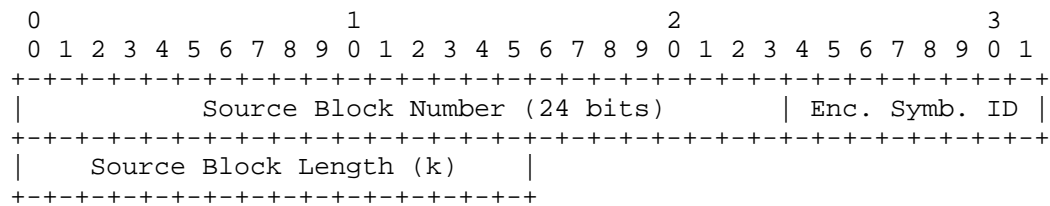
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Source Block Number (24 bits)     | Enc. Symb. ID |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Source Block Length (k)     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: Source FEC Payload ID encoding format for m = 8 (default).


```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Source Block Nb (16 bits)    |   Enc. Symbol ID (16 bits)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Source Block Length (k)     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
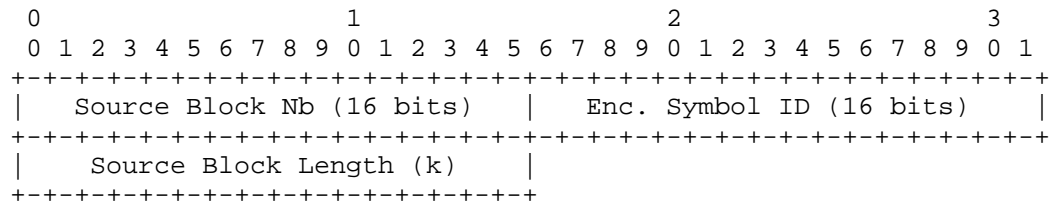
Figure 6: Source FEC Payload ID encoding format for m = 16.

The format of the Source FEC Payload ID for m = 8 and m = 16 are
illustrated in Figure 5 and Figure 6 respectively.

5.1.3. Repair FEC Payload ID

A FEC repair packet MUST contain a Repair FEC Payload ID that is
prepended to the repair symbol(s) as illustrated in Figure 7. There
MUST be a single repair symbol per FEC repair packet.

```
+------------------------------+
|          IP Header           |
+------------------------------+
|       Transport Header       |
+------------------------------+
|     Repair FEC Payload ID    |
+------------------------------+
|        Repair Symbol         |
+------------------------------+
```
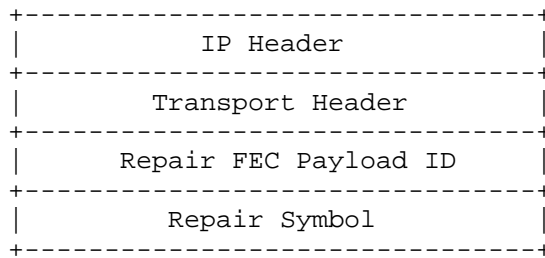
       Figure 7: Structure of a FEC Repair Packet with the Repair FEC
                              Payload ID.

More precisely, the Repair FEC Payload ID is composed of the Source
Block Number, the Encoding Symbol ID, and the Source Block Length.
The length of the first two fields depends on the m parameter
(transmitted separately in the FFCI, Section 5.1.1.2):
Source Block Number (SBN) (32-m bit field): this field identifies
     the source block to which the FEC repair packet belongs.
Encoding Symbol ID (ESI) (m bit field) this field identifies the
     repair symbol contained in this FEC repair packet. This value is
     such that k <= ESI <= n - 1 for repair symbols.
Source Block Length (k) (16 bit field): this field provides the
     number of source symbols for this source block, i.e., the k
     parameter. If 16 bits are too much when m <= 8, it is needed when
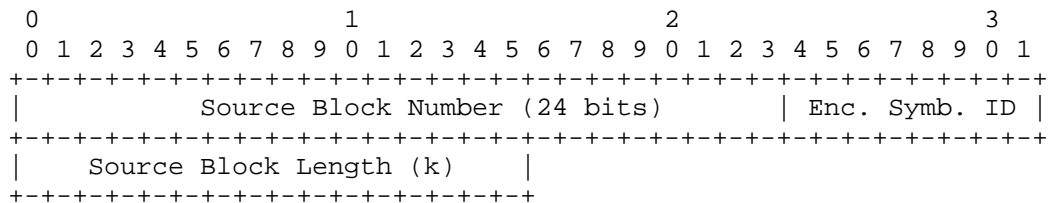   8 < m <= 16. Therefore we provide a single common format
     regardless of m.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Block Number (24 bits)        | Enc. Symb. ID |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Source Block Length (k)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8: Repair FEC Payload ID encoding format for m = 8 (default).


```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Source Block Nb (16 bits)   |   Enc. Symbol ID (16 bits)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Source Block Length (k)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
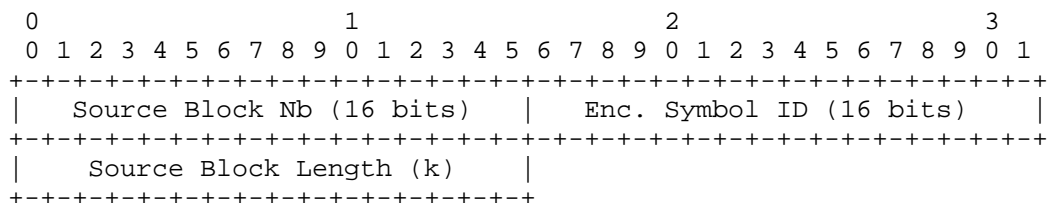
       Figure 9: Repair FEC Payload ID encoding format for m = 16.

The format of the Repair FEC Payload ID for m = 8 and m = 16 are
illustrated in Figure 8 and Figure 9 respectively.

5.2. Procedures

The following procedures apply:
o The source block creation procedures are specified in Section 4.3.
o The SBN value is incremented for each new source block, starting
   at 0 for the first block of the ADU flow. Wrapping to zero will
   happen for long sessions, after value $2^{(32-m)} - 1$.
o The ESI of encoding symbols is managed sequentially, starting at 0
   for the first symbol. The first k values (0 <= ESI <= k - 1)
   identify source symbols, whereas the last n-k values (k <= ESI <=
   n - 1) identify repair symbols.
o The FEC repair packet creation procedures are specified in
   Section 5.1.3.

5.3. FEC Code Specification

The present document inherits from [RFC5510] the specification of the
core Reed-Solomon codes based on Vandermonde matrices for a packet
transmission channel.

6. Security Considerations

6.1. Problem Statement

A content delivery system is potentially subject to many attacks.
Some of them target the network (e.g., to compromise the routing
infrastructure, by compromising the congestion control component),
others target the Content Delivery Protocol (CDP) (e.g., to
compromise its normal behavior), and finally some attacks target the
content itself. Since this document focuses on various FEC schemes,
this section only discusses the additional threats that their use
within the FECFRAME framework can create to an arbitrary CDP.

More specifically, these attacks may have several goals:
o those that are meant to give access to a confidential content
   (e.g., in case of a non-free content),
o those that try to corrupt the ADU Flows being transmitted (e.g.,
   to prevent a receiver from using it),
o and those that try to compromise the receiver's behavior (e.g., by
   making the decoding of an object computationally expensive).
These attacks can be launched either against the data flow itself
(e.g., by sending forged FEC Source/Repair Packets) or against the
FEC parameters that are sent either in-band (e.g., in the Repair FEC
Payload ID) or out-of-band (e.g., in a session description).

6.2. Attacks Against the Data Flow

   First of all, let us consider the attacks against the data flow.

6.2.1. Access to Confidential Contents

   Access control to the ADU Flow being transmitted is typically
   provided by means of encryption. This encryption can be done within
   the content provider itself, by the application (for instance by
   using the Secure Real-time Transport Protocol (SRTP) [RFC3711]), or
   at the Network Layer, on a packet per packet basis when IPSec/ESP is
   used [RFC4303]. If confidentiality is a concern, it is RECOMMENDED
   that one of these solutions be used. Even if we mention these
   attacks here, they are not related nor facilitated by the use of FEC.

6.2.2. Content Corruption

   Protection against corruptions (e.g., after sending forged FEC
   Source/Repair Packets) is achieved by means of a content integrity
   verification/sender authentication scheme. This service is usually
   provided at the packet level. In this case, after removing all
   forged packets, the ADU Flow may be sometimes recovered. Several
   techniques can provide this source authentication/content integrity
   service:
   o at the application level, the Secure Real-time Transport Protocol
      (SRTP) [RFC3711] provides several solutions to authenticate the
      source and check the integrity of RTP and RTCP messages, among
      other services. For instance, associated to the Timed Efficient
      Stream Loss-Tolerant Authentication (TESLA) [RFC4383], SRTP is an
      attractive solution that is robust to losses, provides a true
      authentication/integrity service, and does not create any
      prohibitive processing load or transmission overhead. Yet,
      checking a packet requires a small delay (a second or more) after
      its reception with TESLA. Other building blocks can be used
      within SRTP to provide authentication/content integrity services.
   o at the Network Layer, IPSec/ESP offers (among other services) an
      integrity verification mechanism that can be used to provide
      authentication/content integrity services.

   It is up to the developer and the person in charge of deployment, who
   know the security requirements and features of the target application
   area, to define which solution is the most appropriate. Nonetheless
   it is RECOMMENDED that at least one of these techniques be used.

6.3. Attacks Against the FEC Parameters

   Let us now consider attacks against the FEC parameters included in
   the FFCI that are usually sent out-of-band (e.g., in a session

description). Attacks on these FEC parameters can prevent the
decoding of the associated object. For instance modifying the m
field (when applicable) will lead a receiver to consider a different
code. Modifying the E parameter will lead a receiver to consider bad
Repair Symbols for a received FEC Repair Packet.

It is therefore RECOMMENDED that security measures be taken to
guarantee the FFCI integrity. When the FFCI is sent out-of-band in a
session description, this latter SHOULD be protected, for instance by
digitally signing it.

Attacks are also possible against some FEC parameters included in the
Explicit Source FEC Payload ID and Repair FEC Payload ID. For
instance modifying the Source Block Number of a FEC Source of Repair
Packet will lead a receiver to assign this packet to a wrong block.

It is therefore RECOMMENDED that security measures be taken to
guarantee the Explicit Source FEC Payload ID and Repair FEC Payload
ID integrity. To that purpose, one of the packet-level source
authentication/content integrity techniques of Section 6.2.2 can be
used.

7. IANA Considerations

The FEC Encoding ID value is subject to IANA registration.

TBD

8. Acknowledgments

The authors want to thank Hitoshi Asaeda for his valuable comments.

9. References

9.1. Normative References

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

   [RFC5052] Watson, M., Luby, M., and L. Vicisano, "Forward Error
              Correction (FEC) Building Block", RFC 5052, August 2007.

   [RFC5510] Lacan, J., Roca, V., Peltotalo, J., and S. Peltotalo,
              "Reed-Solomon Forward Error Correction (FEC) Schemes",
              RFC 5510, April 2009.

   [FECFRAME-FRAMEWORK]
             Watson, M., "Forward Error Correction (FEC) Framework",
             draft-ietf-fecframe-framework-10 (Work in Progress),
             September 2010.

   [SDP_ELEMENTS]
             Begen, A., "SDP Elements for FEC Framework",
             draft-ietf-fecframe-sdp-elements-10 (Work in Progress),
             October 2010.

9.2. Informative References

   [RS-codec]
             Rizzo, L., "Reed-Solomon FEC codec (revised version of
             July 2nd, 1998), available at
             http://info.iet.unipi.it/~luigi/vdm98/vdm980702.tgz and
             mirrored at http://planete-bcast.inrialpes.fr/",
             July 1998.

   [Rizzo97] Rizzo, L., "Effective Erasure Codes for Reliable Computer
             Communication Protocols", ACM SIGCOMM Computer
             Communication Review Vol.27, No.2, pp.24-36, April 1997.

   [RFC5170] Roca, V., Neumann, C., and D. Furodet, "Low Density Parity
             Check (LDPC) Forward Error Correction", RFC 5170,
             June 2008.

   [RFC5053] Luby, M., Shokrollahi, A., Watson, M., and T. Stockhammer,
             "Raptor Forward Error Correction Scheme", RFC 5053,
             June 2007.

   [RFC5775] Luby, M., Watson, M., and L. Vicisano, "Asynchronous
             Layered Coding (ALC) Protocol Instantiation", RFC 5775,
             April 2010.

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

   [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)",
             RFC 4303, December 2005.

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

   [RFC4383] Baugher, M. and E. Carrara, "The Use of Timed Efficient
             Stream Loss-Tolerant Authentication (TESLA) in the Secure

                  Real-time Transport Protocol (SRTP)", RFC 4383,
                  February 2006.

Authors' Addresses

   Vincent Roca
   INRIA
   655, av. de l'Europe
   Inovallee; Montbonnot
   ST ISMIER cedex 38334
   France

   Email: vincent.roca@inria.fr
   URI: http://planete.inrialpes.fr/people/roca/

   Mathieu Cunche
   NICTA
   Australia

   Email: mathieu.cunche@nicta.com.au
   URI: http://mathieu.cunche.free.fr/

   Jerome Lacan
   ISAE/LAAS-CNRS
   1, place Emile Blouin
   Toulouse 31056
   France

   Email: jerome.lacan@isae.fr
   URI: http://dmi.ensica.fr/auteur.php3?id_auteur=5

   Amine Bouabdallah
   ISAE/LAAS-CNRS
   1, place Emile Blouin
   Toulouse 31056
   France

   Email: Amine.Bouabdallah@isae.fr
   URI: http://dmi.ensica.fr/

Kazuhisa Matsuzono
Keio University
Graduate School of Media and Governance
5322 Endo
Fujisawa, Kanagawa 252-8520
Japan

Email: kazuhisa@sfc.wide.ad.jp