Simple LDPC-Staircase Forward Error Correction (FEC) Scheme for FECFRAME
                    draft-ietf-fecframe-ldpc-00

Abstract

   This document describes a fully-specified simple FEC scheme for LDPC-
   Staircase codes that can be used to protect media streams along the
   lines defined by the FECFRAME framework. These codes have many
   interesting properties: they are systematic codes, they perform close
   to ideal codes in many use-cases and they also feature very high
   encoding and decoding throughputs. LDPC-Staircase codes are
   therefore a good solution to protect a single high bitrate source
   flow, or to protect globally several mid-rate flows within a single
   FECFRAME instance. They are also a good solution whenever the
   processing load of a software encoder or decoder must be kept to a
   minimum.

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 March 17, 2012.

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 [RFC3453]. The
[RFC6363] 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] (Raptor) and [RFC5170] (LDPC-
Staircase and LDPC-Triangle) FEC schemes introduce erasure codes
based on sparse parity check matrices for object delivery protocols
like ALC and NORM. Similarly, the [RFC5510] document introduces
Reed-Solomon codes based on Vandermonde matrices for the same object
delivery protocols. All these codes are systematic codes, meaning
that the k source symbols are part of the n encoding symbols.
Additionally, the Reed-Solomon FEC codes belong to the class of
Maximum Distance Separable (MDS) codes that are optimal in terms of
erasure recovery capabilities. It means that a receiver can recover
the k source symbols from any set of exactly k encoding symbols out
of n. This is not the case with either Raptor or LDPC-Staircase
codes, and these codes require a certain number of encoding symbols
in excess to k. However, this number is small in practice when an
appropriate decoding scheme is used at the receiver [Cunche08].
Another key difference is the high encoding/decoding complexity of
Reed-Solomon codecs compared to Raptor or LDPC-Staircase codes. A
difference of one or more orders of magnitude or more in terms of
encoding/decoding speed exists between the Reed-Solomon and LDPC-
Staircase software codecs [Cunche08][CunchePHD10]. Finally, Raptor
and LDPC-Staircase codes are large block FEC codes, in the sense of
[RFC3453], since they can efficiently deal with a large number of
source symbols.

The present document focuses on LDPC-Staircase codes, that belong to
the well-known class of "Low Density Parity Check" codes. Because of
their key features, these codes are a good solution in many
situations, as detailed in Section 7.

This documents inherits from [RFC5170] the specifications of the core
LDPC-Staircase codes. Therefore this document specifies only the
information specific to the FECFRAME context and refers to [RFC5170]
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 LDPC-Staircase codes in order to
   protect arbitrary ADU flows.

Finally, publicly available reference implementations of these codes
are available [LDPC-codec] [LDPC-codec-OpenFEC].

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 LDPC-Staircase 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
[RFC6363]:

   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 )      |
   +----------------------+
```
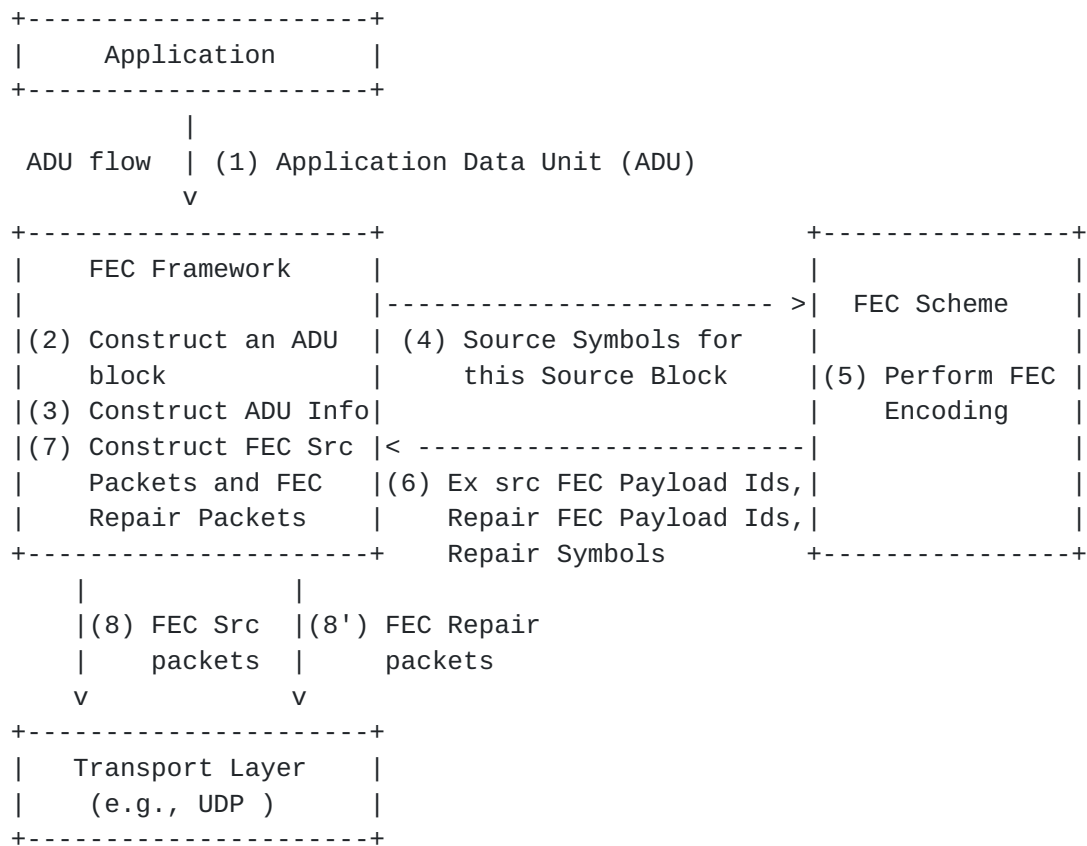
           Figure 1: Terminology used in this document (sender).

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.

CR   denotes the "code rate", i.e., the k/n ratio.

N1   denotes the target number of "1s" per column in the left side
        of the parity check matrix.

N1m3   denotes the value N1 - 3.

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.
FSSI   stands for FEC Scheme Specific Information.
LDPC   stands for Low Density Parity Check.
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;
o the use of the LDPC-Staircase scheme is such that there MUST be
   exactly one encoding symbol per group, i.e., G MUST be equal to 1
   [RFC5170];

4.2. ADU Block Creation

Several aspects must be considered, that impact the ADU block
creation:
o the maximum source block size (max_k 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 maximum source block length in symbols, max_k, depends on several
parameters: the code rate (CR), the Encoding Symbol ID (ESI) field
length in the Explicit Source/Repair FEC Payload ID (16 bits), as
well as possible internal codec limitations. More specifically,
max_k cannot be larger than the following values, derived from the
ESI field size limitation, for a given code rate:

       max1_k = 2^^(16 - ceil(Log2(1/CR)))
    Some common max1_k values are:
    o   CR == 1 (no repair symbol): max1_k = 2^^16 = 65536 symbols
    o   1/2 <= CR < 1: max1_k = 2^^15 = 32,768 symbols
    o   1/4 <= CR < 1/2: max1_k = 2^^14 = 16,384 symbols

    Additionally, a codec MAY impose other limitations on the maximum
    block size, for instance, because of a limited working memory size.
    This decision MUST be clarified at implementation time, when the
    target use-case is known. This results in a max2_k limitation.

    Then, max_k is given by:
       max_k = min(max1_k, max2_k)
    Note that this calculation is only required at the encoder (sender),
    since the actual k parameter (k <= max_k) is communicated to the
    decoder (receiver) through the Explicit Source/Repair FEC Payload ID.

    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 LDPC-
    Staircase 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 LDPC-Staircase 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 FFCI/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 FFCI/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                               |
    +---------------------------------------------------------------+
```

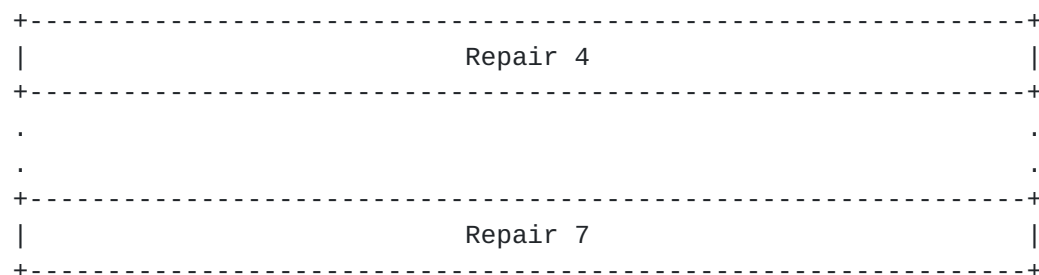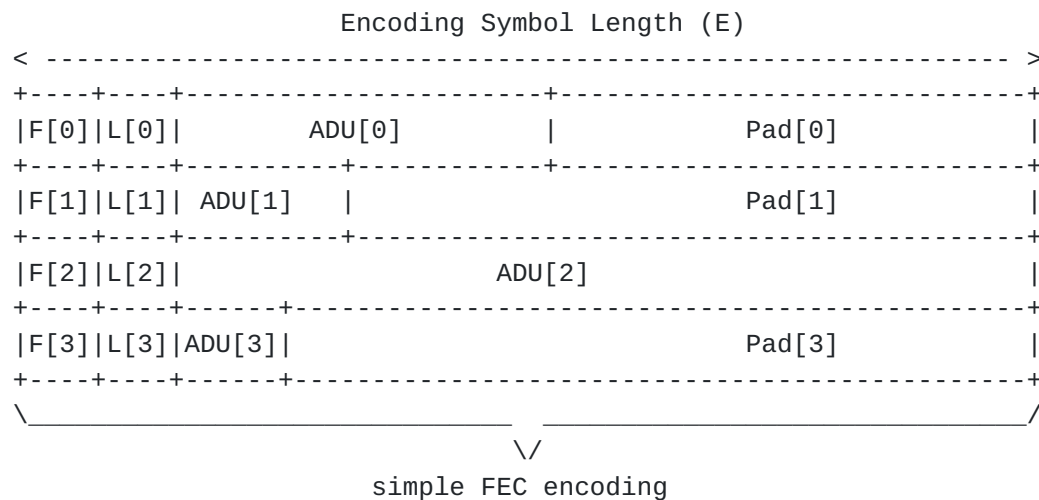   Figure 2: Source block creation, for code rate 1/2 (equal number of
          source and repair symbols, 4 in this example), and S = 0.

   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. LDPC-Staircase FEC Scheme for Arbitrary ADU 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 8).
   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:
   PRNG seed (seed): a non-negative 32 bit integer used as the seed of
      the Pseudo Random Number Generator, as defined in [RFC5170].
   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 the actual encoding symbol length value for each
      block of the session (unless otherwise notified by an updated FFCI
      if this possibility is considered by the use-case or CDP). When
      set to 0 this flag indicates that the E parameter is the maximum
      encoding symbol length value for each block of the session (unless
      otherwise notified by an updated FFCI if this possibility is
      considered by the use-case or CDP).
   N1 minus 3 (n1m3): an integer between 0 (default) and 7, inclusive.
      The number of "1s" per column in the left side of the parity check
      matrix, N1, is then equal to N1m3 + 3, as specified in [RFC5170].
   These elements are required both by the sender (LDPC-Staircase
   encoder) and the receiver(s) (LDPC-Staircase 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 = seed:1234,E:1400,S:0,n1m3:0

   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 7 octets:
   o PRNG seed (seed): 32 bit field.
   o Encoding symbol length (E): 16 bit field.
   o Strict (S) flag: 1 bit field.
   o Reserved: a 4 bit field that MUST be set to zero.
   o N1m3 parameter (n1m3): 3 bit field.
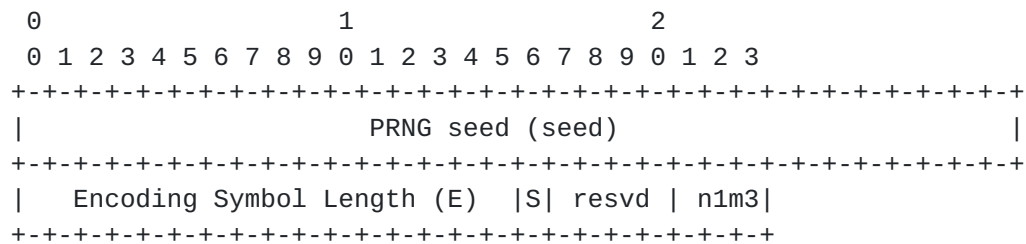
```
 0                   1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    PRNG seed (seed)            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Encoding Symbol Length (E)  |S| resvd | n1m3|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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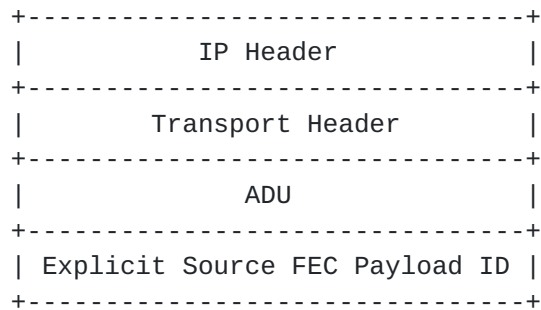
     Figure 4: Structure of a FEC Source Packet with the Explicit Source
                          FEC Payload ID.

   More precisely, the Explicit Source FEC Payload ID is composed of the
   following fields (Figure 5):
   Source Block Number (SBN) (16 bit field): this field identifies the
      source block to which this FEC source packet belongs.
   Encoding Symbol ID (ESI) (16 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.

```
 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 (SBN)  |   Encoding Symbol ID (ESI)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Source Block Length (k)   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
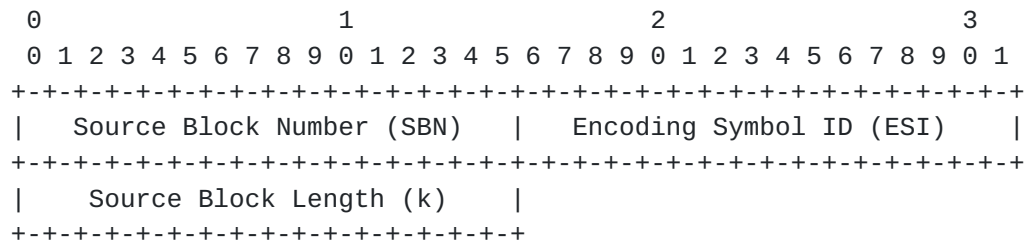
              Figure 5: Source FEC Payload ID encoding format.

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 6. There
   MUST be a single repair symbol per FEC repair packet.

```
+--------------------------------+
|            IP Header           |
+--------------------------------+
|         Transport Header       |
+--------------------------------+
|        Repair FEC Payload ID   |
+--------------------------------+
|            Repair Symbol       |
+--------------------------------+
```

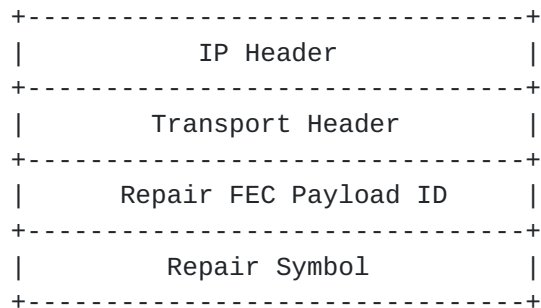   Figure 6: Structure of a FEC Repair Packet with the Repair FEC
                                Payload ID.

   More precisely, the Repair FEC Payload ID is composed of the
   following fields: (Figure 7):
   Source Block Number (SBN) (16 bit field): this field identifies the
      source block to which the FEC repair packet belongs.
   Encoding Symbol ID (ESI) (16 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.
   Number of Encoding Symbols (n) (16 bit field): this field provides
      the number of encoding symbols for this source block, i.e., the n
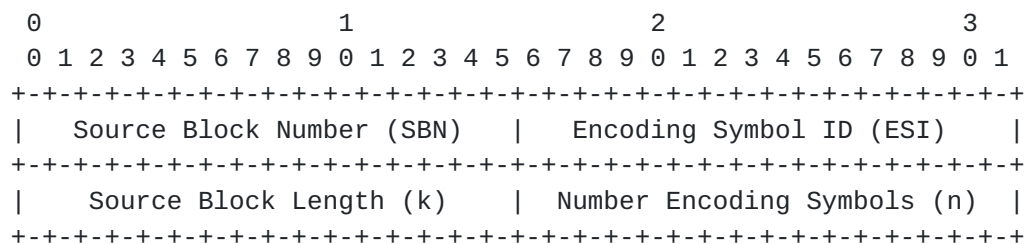   parameter.

```
 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 (SBN)   |   Encoding Symbol ID (ESI)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Source Block Length (k)    |  Number Encoding Symbols (n)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

           Figure 7: Repair FEC Payload ID encoding format.

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^^16 - 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 [RFC5170] the specification of the
core LDPC-Staircase codes for a packet erasure transmission channel.

Because of the requirement to have exactly one encoding symbol per
group, i.e., because G MUST be equal to 1 (Section 4.1), several
parts of [RFC5170] are useless. In particular, this is the case of
Section 5.6. "Identifying the G Symbols of an Encoding Symbol
Group".

6. Security Considerations

The FEC Framework document [RFC6363] provides a comprehensive
analysis of security considerations applicable to FEC schemes.
Therefore the present section follows the security considerations
section of [RFC6363] and only discusses topics that are specific to
the use of LDPC-Staircase codes.

6.1. Attacks Against the Data Flow

6.1.1. Access to Confidential Content

The LDPC-Staircase FEC Scheme specified in this document does not
change the recommendations of [RFC6363]. To summarize, if
confidentiality is a concern, it is RECOMMENDED that one of the
solutions mentioned in [RFC6363] is used, with special considerations
to the way this solution is applied (e.g., before versus after FEC
protection, and within the end-system versus in a middlebox), to the
operational constraints (e.g., performing FEC decoding in a protected
environment may be complicated or even impossible) and to the threat
model.

6.1.2. Content Corruption

   The LDPC-Staircase FEC Scheme specified in this document does not
   change the recommendations of [RFC6363]. To summarize, it is
   RECOMMENDED that one of the solutions mentioned in [RFC6363] is used
   on both the FEC Source and Repair Packets.

6.2. Attacks Against the FEC Parameters

   The FEC Scheme specified in this document defines parameters that can
   be the basis of several attacks. More specifically, the following
   parameters of the FFCI may be modified by an attacker
   (Section 5.1.1.2):
   o FEC Encoding ID: changing this parameter leads the receiver to
      consider a different FEC Scheme, which enables an attacker to
      create a Denial of Service (DoS).
   o Encoding symbol length (E): setting this E parameter to a value
      smaller than the valid one enables an attacker to create a DoS
      since the repair symbols and certain source symbols will be larger
      than E, which is an incoherency for the receiver. Setting this E
      parameter to a value larger than the valid one has similar impacts
      when S=1 since the received repair symbol size will be smaller
      than expected. On the opposite it will not lead to any
      incoherency when S=0 since the actual symbol length value for the
      block is determined by the size of any received repair symbol, as
      long as this value is smaller than E. However setting this E
      parameter to a larger value may have impacts on receivers that
      pre-allocate memory space in advance to store incoming symbols.
   o Strict (S) flag: flipping this S flag from 0 to 1 (i.e., E is now
      considered as a strict value) enables an attacker to mislead the
      receiver if the actual symbol size varies over different source
      blocks. Flipping this S flag from 1 to 0 has no major
      consequences unless the receiver requires to have a fixed E value
      (e.g., because the receiver pre-allocates memory space).
   o N1 minus 3 (n1m3): changing this parameter leads the receiver to
      consider a different code, which enables an attacker to create a
      DoS.

   It is therefore RECOMMENDED that security measures are taken to
   guarantee the FFCI integrity, as specified in [RFC6363]. How to
   achieve this depends on the way the FFCI is communicated from the
   sender to the receiver, which is not specified in this document.

   Similarly, attacks are possible against the Explicit Source FEC
   Payload ID and Repair FEC Payload ID: by modifying the Source Block
   Number (SBN), or the Encoding Symbol ID (ESI), or the Source Block
   Length (k), or the Number Encoding Symbols (n), an attacker can
   easily corrupt the block identified by the SBN. Other consequences,

that are use-case and/or CDP dependant, may also happen. It is
therefore RECOMMENDED that security measures are taken to guarantee
the FEC Source and Repair Packets as stated in [RFC6363].

6.3. When Several Source Flows are to be Protected Together

The LDPC-Staircase FEC Scheme specified in this document does not
change the recommendations of [RFC6363].

6.4. Baseline Secure FEC Framework Operation

The LDPC-Staircase FEC Scheme specified in this document does not
change the recommendations of [RFC6363] concerning the use of the
IPsec/ESP security protocol as a mandatory to implement (but not
mandatory to use) security scheme. This is well suited to situations
where the only insecure domain is the one over which the FEC
Framework operates.

7. Operations and Management Considerations

The FEC Framework document [RFC6363] provides a comprehensive
analysis of operations and management considerations applicable to
FEC schemes. Therefore the present section only discusses topics
that are specific to the use of LDPC-Staircase codes as specified in
this document.

7.1. Operational Recommendations

LDPC-Staircase codes have excellent erasure recovery capabilities
with large source blocks, close to ideal MDS codes. For instance,
with a medium source block size k=1024, CR=2/3, N1=5, G=1, with a
hybrid ITerative/Maximum Likelihood (IT/ML) decoding approach (see
below) and when all symbols are sent in a random order (see below),
the average overhead amounts to 0.64% (corresponding to 6.5 symbols
in addition to k) and receiving 1043 symbols (corresponding to a 1.9%
overhead) is sufficient to reduce the decoding failure probability to
$5.1*10^{-5}$. This is why these codes are a good solution to protect a
single high bitrate source flow as in [Matsuzono10], or to protect
globally several mid-rate source flows within a single FECFRAME
instance: in both cases the source block size can be assumed to be
equal to a few hundreds (or more) source symbols.

LDPC-Staircase codes are also a good solution whenever processing
requirements at a software encoder or decoder must be kept to a
minimum. This is true when the decoder uses an IT decoding
algorithm, or an ML algorithm (we use a Gaussian Elimination as the
ML algorithm) when this latter is carefully implemented and the

source block size kept reasonable, or a mixture of both techniques which is the recommended solution [Cunche08][CunchePHD10]. For instance an average decoding speed between 1.3 Gbps (corresponding to a very bad channel, close to the theoretical decoding limit and requiring an ML decoding) and 4.3 Gbps (corresponding to a medium quality channel where IT decoding is sufficient) are easily achieved with a source block size composed of k=1024 source symbols, a code rate CR=2/3 (i.e., 512 repair symbols), 1024 byte long symbols, G=1, and N1=5, on an Intel Xeon 5120/1.86GHz workstation running Linux/64 bits. Additionally, with a hybrid IT/ML approach, a receiver can decide if and when ML decoding is used, depending on local criteria (e.g., battery or CPU capabilities), independently from other receivers.

As the source block size decreases, the erasure recovery capabilities of LDPC codes in general also decrease. In the case of LDPC-Staircase codes, in order to compensate this phenomenon, it is recommended to increase the N1 parameter (e.g., experiments carried out in [Matsuzono10] use N1=7 if k=170 symbols, and N1=5 otherwise) and to use a hybrid IT/ML decoding approach. For instance, with a small source block size k=256 symbols, CR=2/3, N1=7, and G=1, the average overhead amounts to 0.67% (corresponding to 1.7 symbols in addition to k), and receiving 267 symbols (corresponding to a 4.3% overhead) is sufficient to reduce the decoding failure probability to $1.4*10^{-5}$. Using N1=9 further improves these results if need be, which also enables to use LDPC-Staircase codes with k=100 symbols for instance.

With very small source blocks (e.g., a few tens symbols), using for instance Reed-Solomon codes [SIMPLE_RS] or 2D parity check codes MAY be more appropriate.

The way the FEC Repair Packets are transmitted is of high importance. A good strategy, that works well for any kind of channel loss model, consists in sending FEC Repair Packets in random order (rather than in sequence) while FEC Source Packets are sent first and in sequence. Sending all packets in a random order is another possibility, but it requires that all repair symbols for a source block be produced first, which adds some extra delay at a sender.

8. IANA Considerations

Values of FEC Encoding IDs are subject to IANA registration. [RFC6363] defines general guidelines on IANA considerations. In particular it defines a registry called FEC Framework (FECFRAME) FEC Encoding IDs whose values are granted on an IETF Consensus basis.

This document registers one value in the FEC Framework (FECFRAME) FEC
Encoding IDs registry as follows:
o XXX refers to the Simple LDPC-Staircase [RFC5170] FEC Scheme for
   Arbitrary Packet Flows.

9. Acknowledgments

The authors want to thank K. Matsuzono, J. Detchart and H. Asaeda for
their contributions in evaluating the use of LDPC-Staircase codes in
the context of FECFRAME [Matsuzono10].

10. References

10.1. Normative References

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

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

[RFC6363] Watson, M., Begen, A., and V. Roca, "Forward Error
           Correction (FEC) Framework", RFC 6363, September 2011.

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

10.2. Informative References

[RFC3453] Luby, M., Vicisano, L., Gemmell, J., Rizzo, L., Handley,
           M., and J. Crowcroft, "The Use of Forward Error Correction
           (FEC) in Reliable Multicast", RFC 3453, December 2002.

[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.

[SIMPLE_RS]
           Roca, V., Cunche, M., Lacan, J., Bouabdallah, A., and K.
           Matsuzono, "Simple Reed-Solomon Forward Error Correction

                   (FEC) Scheme for FECFRAME",
                   draft-ietf-fecframe-simple-rs-01 (Work in Progress),
                   September 2011.

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

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

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

   [Cunche08]
                   Cunche, M. and V. Roca, "Optimizing the Error Recovery
                   Capabilities of LDPC-Staircase Codes Featuring a Gaussian
                   Elimination Decoding Scheme", 10th IEEE International
                   Workshop on Signal Processing for Space Communications
                   (SPSC'08), October 2008.

   [CunchePHD10]
                   Cunche, M., "High performances AL-FEC codes for the
                   erasure channel : variation around LDPC codes", PhD
                   dissertation (in
                   French) (http://tel.archives-ouvertes.fr/tel-
                   00451336/en/), June 2010.

   [Matsuzono10]
                   Matsuzono, K., Detchart, J., Cunche, M., Roca, V., and H.
                   Asaeda, "Performance Analysis of a High-Performance Real-
                   Time Application with Several AL-FEC Schemes", 35th Annual
                   IEEE Conference on Local Computer Networks (LCN 2010),
                   October 2010.

   [LDPC-codec]
                   Cunche, M., Roca, V., Neumann, C., and J. Laboure, "LDPC-
                   Staircase/LDPC-Triangle Codec Reference Implementation",
                   INRIA Rhone-Alpes and STMicroelectronics,
                   <http://planete-bcast.inrialpes.fr/>.

   [LDPC-codec-OpenFEC]
                   "The OpenFEC project", <http://openfec.org/>.

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