FEC Framework Working Group	M. Watson	
Internet-Draft	Netflix, Inc.	
Intended status: Standards Track	September 7, 2010	
Expires: March 11, 2011		

TOC

Forward Error Correction (FEC) Framework
draft-ietf-fecframe-framework-10

Abstract

This document 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 Forward Error Correction to arbitrary packet flows over unreliable transport and is primarily intended for real-time, or streaming, media. This framework can be used to define Content Delivery Protocols that provide Forward Error Correction for streaming media delivery or other packet flows. Content Delivery Protocols defined using this framework can support any FEC Scheme (and associated FEC codes) which is compliant with various requirements defined in this document. Thus, Content Delivery Protocols can be defined which are not specific to a particular FEC Scheme and FEC Schemes can be defined which are not specific to a particular Content Delivery Protocol.

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 11, 2011.

Copyright Notice

Table of Contents

1. Introduction

TOC

Many applications have a requirement to transport a continuous stream of packetized data from a source (sender) to one or more destinations (receivers) over networks which do not provide guaranteed packet delivery. Primary examples are real-time, or streaming, media applications such as broadcast, multicast or on-demand audio, video or multimedia.

Forward Error Correction is a well-known technique for improving reliability of packet transmission over networks which do not provide guaranteed packet delivery, especially in multicast and broadcast applications. The FEC Building Block defined in [RFC5052] (Watson, M., Luby, M., and L. Vicisano, “Forward Error Correction (FEC) Building Block,” August 2007.) provides a framework for definition of Content Delivery Protocols (CDPs) for object delivery (including, primarily, file delivery) which 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 which is also defined according to the requirements of the FEC Building Block. (Note that the term "Forward Erasure Correction" is sometimes used, 'erasures' being a type of error in which data is lost and this loss can be detected, rather than being received in corrupted form - the focus of this document is strictly on erasures, however the term Forward Error Correction is more widely used).

This document defines a framework for the definition of CDPs which provide for FEC protection of arbitrary packet flows over unreliable transports such as UDP. As such, this document complements the FEC Building Block of [RFC5052] (Watson, M., Luby, M., and L. Vicisano, “Forward Error Correction (FEC) Building Block,” August 2007.), by providing for the case of arbitrary packet flows over unreliable transport, the same kind of framework as that document provides for object delivery. This document does not define a complete Content Delivery Protocol, but rather defines only those aspects that are expected to be common to all Content Delivery Protocols based on this framework.

This framework does not define how the flows to be protected are determined, nor how the details of the protected flows and the FEC streams which protect them are communicated from sender to receiver. It is expected that any complete Content Delivery Protocol specification which makes use of this framework will address these signalling requirements. However, this document does specify the information which is required by the FEC Framework at the sender and receiver - for example details of the flows to be FEC protected, the flow(s) that will

carry the FEC protection data and an opaque container for FEC-Scheme-specific information.
FEC Schemes designed for use with this framework must fulfil a number of requirements defined in this document. Note that these requirements are different from those defined in [RFC5052] (Watson, M., Luby, M., and L. Vicisano, “Forward Error Correction (FEC) Building Block,” August 2007.) for FEC Schemes for object delivery. However there is a great deal of commonality and FEC Schemes defined for object delivery may be easily adapted for use with the framework defined here.
Since the RTP protocol layer is used over UDP, this framework can be applied to RTP flows as well. FEC repair packets may be sent directly over UDP or over RTP. The latter approach has the advantage that RTP instrumentation, based on RTCP, can be used for the repair flow. Additionally, the post-repair RTCP extended report [RFC5725] (Begen, A., Hsu, D., and M. Lague, “Post-Repair Loss RLE Report Block Type for RTP Control Protocol (RTCP) Extended Reports (XRs),” February 2010.) may be used to obtain information about the loss rate after FEC recovery.
The use of RTP for repair flows is defined for each FEC Scheme by defining an RTP Payload Format for that particular FEC Scheme (possibly in the same document).

---

2. Definitions/Abbreviations

TOC

'ADU Flow' A sequence of ADUs associated with a transport layer flow identifier (such as the standard 5-tuple { Source IP Address, Source Transport Port, Destination IP Address, Destination Transport Port, Transport Protocol } in the case of UDP)

'AL-FEC' Application Layer Forward Error Correction

'Application Data Unit' The unit of source data provided as payload to the transport layer

'Application protocol' Control protocol used to establish and control the source data flow being protected - e.g. RTSP.

'Content Delivery Protocol (CDP)' A complete application protocol specification which, through the use of the framework defined in this document, is able to make use of FEC Schemes to provide Forward Error Correction capabilities

'FEC' Forward Error Correction.

'FEC Code' An algorithm for encoding data such that the encoded data flow is resilient to data loss (Note: in general FEC Codes

may also be used to make a data flow resilient to corruption, but that is not considered here).

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

'FEC Framework Configuration Information' Information which controls the operation of the FEC Framework.

'FEC Payload ID' Information which 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 which defines the additional protocol aspects required to use a particular FEC code with the FEC Framework, or, in the context of RMT, with the RMT FEC Building Block.

'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 Source FEC Payload ID.

'Protection amount' The relative increase in data sent due to the use of FEC.

'Repair data flow' The packet flow or flows carrying forward error correction data

'Repair FEC Payload ID' An FEC Payload ID specifically for use with repair packets.

'Source data flow' The packet flow or flows to which FEC protection is to be applied. A source data flow consists of ADUs.

'Source FEC Payload ID' An FEC Payload ID specifically for use with source packets.

'Source protocol' A protocol used for the source data flow being protected - e.g. RTP.

'Transport protocol' The protocol used for transport of the source and repair data flows - e.g. UDP, DCCP.

The following definitions are aligned with [RFC5052] (Watson, M., Luby, M., and L. Vicisano, “Forward Error Correction (FEC) Building Block,” August 2007.)

'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 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.

'Repair symbol' encoding symbol that is not a source symbol.

'Source Block' group of ADUs which are to be FEC protected as a single block.

'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 Reed-Solomon codes introduced in this document are systematic.

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] (Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” March 1997.).

---

3. Architecture Overview

TOC

The FEC Framework is described in terms of an additional layer between the transport layer (e.g. UDP or DCCP) and protocols running over this transport layer. Examples of such protocols are RTP, RTCP, etc. As such, the data path interface between the FEC Framework and both underlying and overlying layers can be thought of as being the same as the standard interface to the transport layer - i.e. the data exchanged consists of datagram payloads each associated with a single ADU flow identified (in the case of UDP) by the standard 5-tuple { Source IP

Address, Source Transport Port, Destination IP Address, Destination Transport Port, Transport Protocol }. In the case that RTP is used for the repair flows, the source and repair data may be multiplexed using RTP onto a single UDP flow and must consequently be demultiplexed at the receiver. There are various ways in which this multiplexing can be done, for example as described in [RFC4588] (Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R. Hakenberg, “RTP Retransmission Payload Format,” July 2006.).

It is important to understand that the main purpose of the FEC Framework architecture is to allocate functional responsibilities to separately documented components in such a way that specific instances of the components can be combined in different ways to describe different protocols.

The FEC Framework makes use of an FEC Scheme, in a similar sense to that defined in [RFC5052] (Watson, M., Luby, M., and L. Vicisano, “Forward Error Correction (FEC) Building Block,” August 2007.) and uses the terminology of that document. The FEC Scheme defines the FEC encoding and decoding and defines the protocol fields and procedures used to identify packet payload data in the context of the FEC Scheme. The interface between the FEC Framework and an FEC Scheme, which is described in this document, is a logical one, which exists for specification purposes only. At an encoder, the FEC Framework passes ADUs to the FEC Scheme for FEC encoding. The FEC Scheme returns repair symbols with their associated Repair FEC Payload IDs, and in some case Source FEC Payload IDs, depending on the FEC Scheme. At a decoder, the FEC Framework passes transport packet payloads (source and repair) to the FEC Scheme and the FEC Scheme returns additional recovered source packet payloads.

This document defines certain FEC Framework Configuration Information which MUST be available to both sender and receiver(s). For example, this information includes the specification of the ADU flows which are to be FEC protected, specification of the ADU flow(s) which will carry the FEC protection (repair) data and the relationship(s) between these source and repair flows (i.e. which source flow(s) are protected by each repair flow. The FEC Framework Configuration Information also includes information fields which are specific to the FEC Scheme. This information is analogous to the FEC Object Transmission Information defined in [RFC5052] (Watson, M., Luby, M., and L. Vicisano, “Forward Error Correction (FEC) Building Block,” August 2007.).

The FEC Framework does not define how the FEC Framework Configuration Information for the stream is communicated from sender to receiver. This must be defined by any Content Delivery Protocol specification as described in the following sections.

In this architecture we assume that the interface to the transport layer supports the concepts of data units (referred to here as Application Data Units) to be transported and identification of ADU flows on which those data units are transported. Since this is an interface internal to the architecture, we do not specify this interface explicitly. We do require that ADU flows which are distinct

from the transport layer point of view (for example, distinct UDP flows as identified by the UDP source/destination ports/addresses) are also distinct on the interface between the transport layer and the FEC Framework.

As noted above, RTP flows are a specific example of ADU flows which might be protected by the FEC Framework. From the FEC Framework point of view, RTP source flows are ADU flows like any other, with the RTP header included within the ADU.

Depending on the FEC Scheme, RTP may also be used as a transport for repair packet flows. In this case an FEC Scheme must define an RTP Payload Format for the repair data.

The architecture outlined above is illustrated in the Figure 1 (FEC Framework Architecture). In this architecture, two RTP instances are shown, for the source and repair data respectively. This is because the use of RTP for the source data is separate from and independent of the use of RTP for the repair data. The appearance of two RTP instances is more natural when you consider that in many FEC codes, the repair payload contains repair data calculated across the RTP headers of the source packets. Thus a repair packet carried over RTP starts with an RTP header of its own which is followed (after the Repair Payload ID) by repair data containing bytes which protect the source RTP headers (as well as repair data for the source RTP payloads).

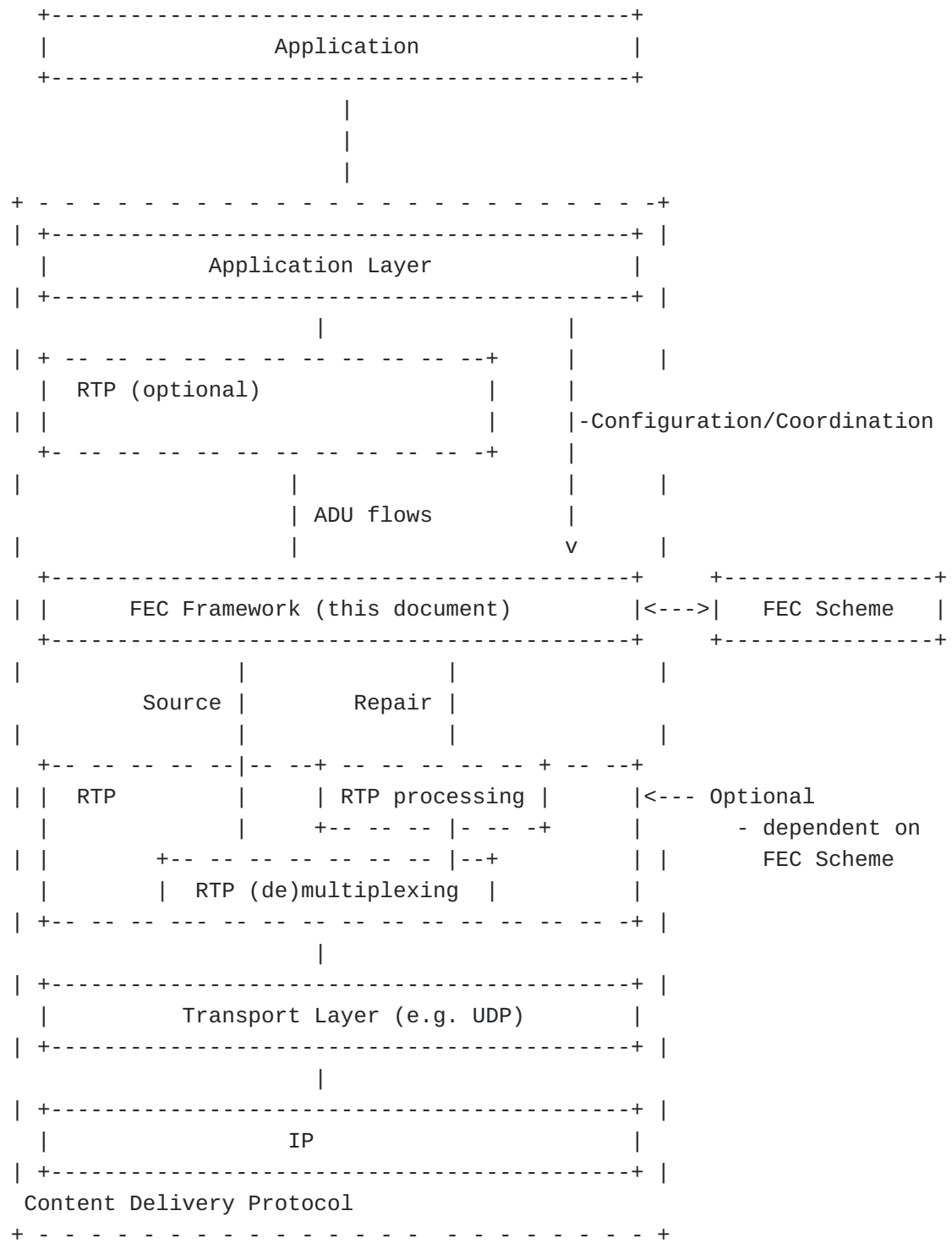

Figure 1: FEC Framework Architecture

The contents of the transport payload for repair packets is fully defined by the FEC Scheme. For a specific FEC Scheme, a means MAY be defined for repair data to be carried over RTP, in which case the

repair packet payload format starts with the RTP header. This corresponds to defining an RTP Payload Format for the specific FEC Scheme. Guidelines for writers of RTP Payload Formats are provided in [RFC2736] (Handley, M. and C. Perkins, “Guidelines for Writers of RTP Payload Format Specifications,” December 1999.).
The use of RTP for repair packets is independent of the protocols used for source packets: if RTP is used for source packets then repair packets may or may not use RTP and vice versa (although it is unlikely that there are useful scenarios where non-RTP source flows are protected by RTP repair flows). FEC Schemes are expected to recover entire transport payloads for recovered source packets in all cases. For example if RTP is used for source flows, the FEC Scheme is expected to recover the entire UDP payload, including the RTP header.

---

4. Procedural overview

TOC

4.1. General

TOC

The mechanism defined in this document does not place any restrictions on the Application Data Units which can be protected together, except that the Application Data Unit is carried over a supported transport protocol (See Section 7 (Transport Protocols)). The data may be from multiple Source Data Flows that are protected jointly. The FEC framework handles the Source Data Flows as a sequence of 'source blocks' each consisting of a set of Application Data Units, possibly from multiple Source Data Flows which are to be protected together. For example, each source block may be constructed from those Application Data Units related to a particular segment in time of the flow.
At the sender, the FEC Framework passes the payloads for a given block to the FEC Scheme for FEC encoding. The FEC Scheme performs the FEC encoding operation and returns the following information:

- optionally, FEC Payload IDs for each of the source payloads (encoded according to an FEC-Scheme-specific format)
- one or more FEC repair packet payloads
- FEC Payload IDs for each of the repair packet payloads (encoded according to an FEC-Scheme-specific format)

The FEC framework then performs two operations: Firstly, it appends the FEC payload IDs, if provided, to each of the Application Data Units, and sends the resulting packets, known as 'FEC source packets', to the

receiver and secondly it places the provided 'FEC repair packet payloads' and corresponding 'FEC Repair Payload IDs' appropriately to construct 'FEC repair packets' and send them to the receiver. Note that FEC repair packets MAY be sent to a different multicast group or groups from the source packets.

This document does not define how the sender determines which Application Data Units are included in which source blocks or the sending order and timing of FEC source and FEC repair packets. A specific Content Delivery Protocol MAY define this mapping or it MAY be left as implementation dependent at the sender. However, a CDP specification MUST define how a receiver determines a minimum length of time that it should wait to receive FEC repair packets for any given source block. FEC Schemes MAY define limitations on this mapping, such as maximum size of source blocks, but SHOULD NOT attempt to define specific mappings. The sequence of operations at the sender is described in more detail in Section 4.2 (Sender Operation).

At the receiver, original Application Data Units are recovered by the FEC Framework directly from any FEC Source Packets received simply by removing the Source FEC Payload ID, if present. The receiver also passes the contents of the received Application Data Units, plus their FEC Payload IDs to the FEC Scheme for possible decoding.

If any Application Data Units related to a given source block have been lost, then the FEC Scheme may perform FEC decoding to recover the missing Application Data Units (assuming sufficient FEC Source and FEC Repair packets related to that source block have been received).

Note that the receiver may need to buffer received source packets to allow time for the FEC Repair packets to arrive and FEC decoding to be performed before some or all of the received or recovered packets are passed to the application. If such a buffer is not provided, then the application must be able to deal with the severe re-ordering of packets that may occur. However, such buffering is Content Delivery Protocol and/or implementation-specific and is not specified here. The receiver operation is described in more detail in Section 4.3 (Receiver Operation)

The FEC Source packets MUST contain information which identifies the source block and the position within the source block (in terms specific to the FEC Scheme) occupied by the Application Data Unit. This information is known as the 'Source FEC Payload ID'. The FEC Scheme is responsible for defining and interpreting this information. This information MAY be encoded into a specific field within the FEC Source packet format defined in this specification, called the Explicit Source FEC Payload ID field. The exact contents and format of the Explicit Source FEC Payload ID field are defined by the FEC Scheme.

Alternatively, the FEC Scheme MAY define how the Source FEC Payload ID is derived from other fields within the source packets. This document defines the way that the Explicit Source FEC Payload ID field is appended to source packets to form FEC Source packets.

The FEC Repair packets MUST contain information which identifies the source block and the relationship between the contained repair payloads

and the original source block. This is known as the 'Repair FEC Payload ID'. This information MUST be encoded into a specific field, the Repair FEC Payload ID field, the contents and format of which are defined by the FEC Scheme.
The FEC Scheme MAY use different FEC Payload ID field formats for FEC Source packets and FEC Repair packets.

4.2. Sender Operation

TOC

It is assumed that the sender has constructed or received original data packets for the session. These may be RTP, RTCP, MIKEY or indeed any other type of packet. The following operations, illustrated in Figure 2 (Sender operation), for the case of UDP repair flows and Figure 3 (Sender operation with RTP repair flows) for the case of RTP repair flows, describe a possible way to generate compliant FEC Source packet and FEC repair packet streams:

1. Application Data Units are provided by the application.

2. A source block is constructed as specified in Section 5.2 (Structure of the source block).

3. The source block is passed to the FEC Scheme for FEC encoding. The Source FEC Payload ID information of each 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.

4. The FEC Scheme performs FEC Encoding, generating repair packet payloads from a source block and a Repair FEC Payload ID field for each repair payload.

5. The Explicit Source FEC Payload IDs (if used), Repair FEC Payload IDs and repair packet payloads are provided back from the FEC Scheme to the FEC Framework.

6. The FEC Framework constructs FEC Source packets according to Section 5.3 (Packet format for FEC Source packets) and FEC Repair packets according to Section 5.4 (Packet Format for FEC Repair packets) using the FEC Payload IDs and repair packet payloads provided by the FEC Scheme.

7. The FEC Source and 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. The FEC Source packets are sent using the same ADU flow identification information as would have been used for

the original source packets 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).

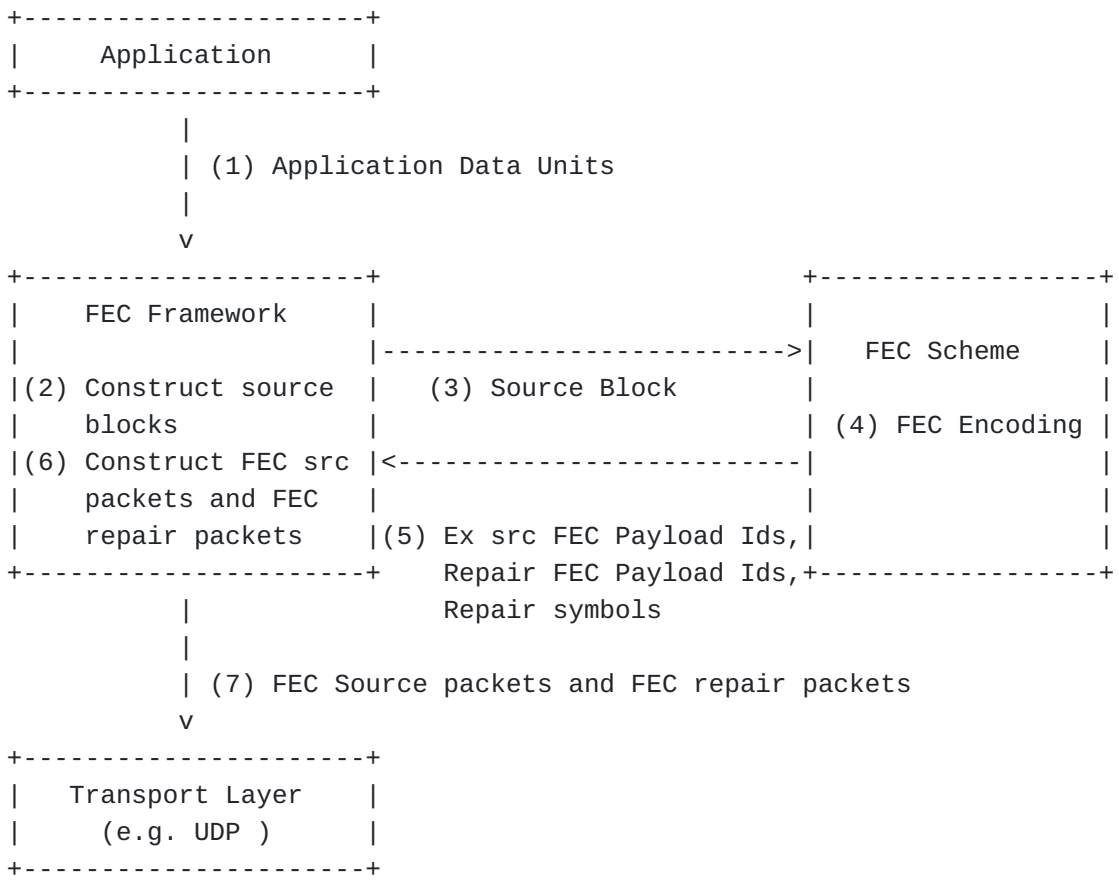

Figure 2: Sender operation

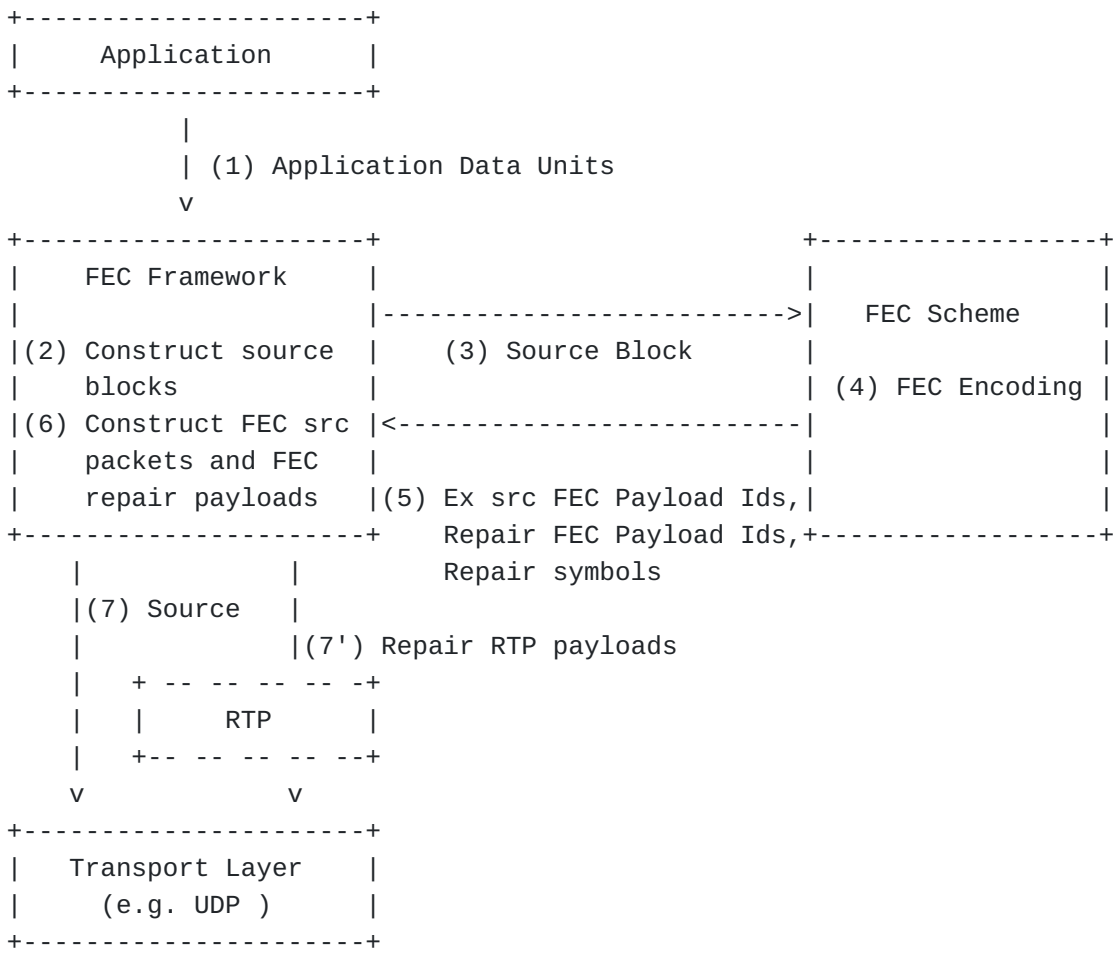

Figure 3: Sender operation with RTP repair flows

4.3. Receiver Operation

TOC

The following describes a possible receiver algorithm, illustrated in Figure 4 (Receiver Operation) and Figure 5 (Receiver Operation) for the case of RTP repair flows, when receiving an FEC source or repair packet:

1. FEC Source Packets and FEC Repair packets are received and passed to the FEC Framework. The type of packet (Source or Repair) and the Source Data Flow to which it belongs (in the case of source packets) is indicated by the ADU flow information which identifies the flow at the transport layer (for example source and destination ports and addresses in the case of UDP).

1a. In the special case that RTP is used for repair packets and source and repair packets are multiplexed onto the same UDP flow, then RTP demultiplexing is required to demultiplex source and repair flows. However, RTP processing is applied only to the repair packets at this stage: source packets continue to be handled as UDP payloads (i.e. including their RTP headers).

2. The FEC Framework extracts the Explicit Source FEC Payload ID field (if present) from FEC Source Packets and the Repair FEC Payload ID from FEC Repair Packets.

3. The Explicit Source FEC Payload IDs (if present), Repair FEC Payload IDs, FEC Source payloads and FEC Repair payloads are passed to the FEC Scheme.

4. The FEC Scheme uses the received FEC Payload IDs (and derived FEC Source Payload IDs in the case that the Explicit Source FEC Payload ID field is not used) to group source and repair packets into source blocks. If at least one source packet is missing from a source block, and at least one repair packet has been received for the same source block then FEC decoding may be performed in order to recover missing source payloads. The FEC Scheme determines whether source packets have been lost and whether enough data for decoding of any or all of the missing source payloads in the source block has been received.

5. The FEC Scheme returns the Application Data Units to the FEC Framework in the form of source blocks containing received and decoded Application Data Units and indications of any Application Data Units which were missing and could not be decoded.

6. The FEC Framework passes the received and recovered Application Data Units to the application.

Note that the description above defines functionality responsibilities but does not imply a specific set of timing relationships. For example, ADUs may be provided to the application as soon as they are received or recovered (and hence potentially out-of-order) or they may be buffered and delivered to the application in-order.

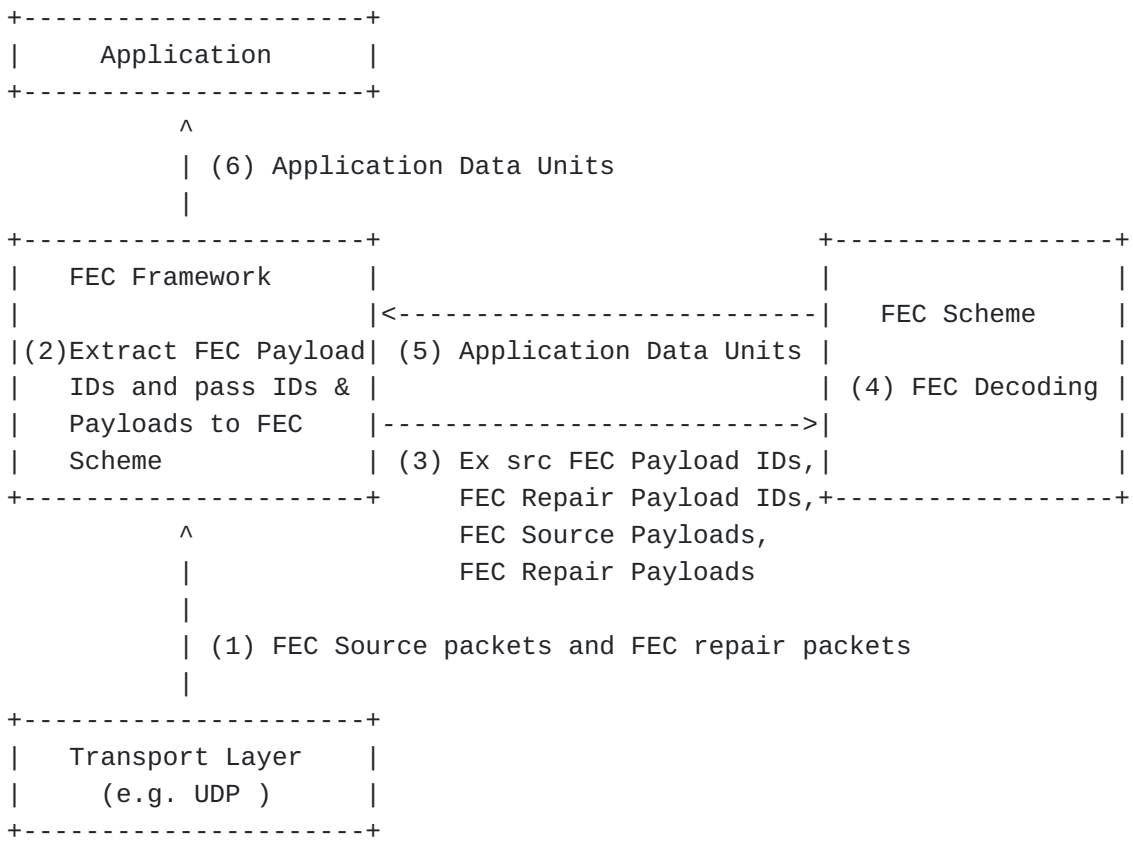

Figure 4: Receiver Operation

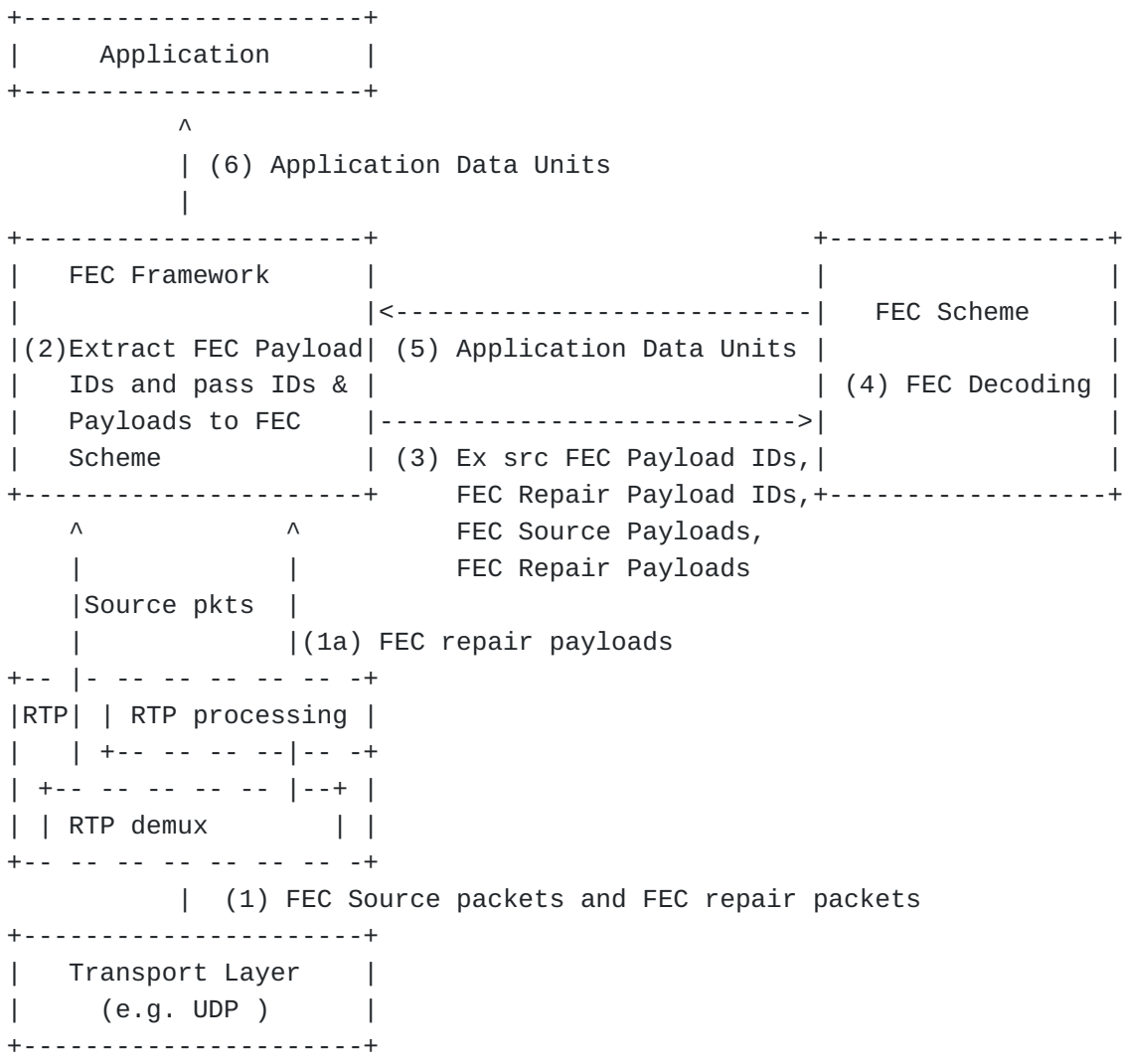

Figure 5: Receiver Operation

Note that the above procedure may result in a situation in which not all ADUs are recovered.
Source packets which are correctly received and those which are reconstructed MAY be delivered to the application out of order and in a different order from the order of arrival at the receiver.
Alternatively, buffering and packet re-ordering MAY be applied to re-order received and reconstructed source packets into the order they were placed into the source block, if that is necessary according to the application.

TOC

5. Protocol Specification

5.1. General

TOC

This section specifies the protocol elements for the FEC Framework. Three components of the protocol are defined in this document and are described in the following sections:

1. Construction of a source block from Application Data Units. The FEC code will be applied to this source block to produce the repair payloads.

2. A format for packets containing source data.

3. A format for packets containing repair data.

The operation of the FEC Framework is governed by certain FEC Framework Configuration Information. This configuration information is also defined in this section. A complete protocol specification that uses this framework MUST specify the means to determine and communicate this information between sender and receiver.

5.2. Structure of the source block

TOC

The FEC Framework and FEC Scheme exchange Application Data Units in the form of source blocks. A source block is generated by the FEC Framework from an ordered sequence of Application Data Units. The allocation of Application Data Units to blocks is dependent on the application. Note that some Application Data Units may not be included in any block. Each Source Block provided to the FEC scheme consists of an ordered sequence of Application Data Units where the following information is provided for each ADU:

*A description of the Source Data Flow with which the Application Data Unit is associated (See 6.5)

*The Application Data Unit itself

*The length of the Application Data Unit

TOC

5.3. Packet format for FEC Source packets

The packet format for FEC Source packets MUST be used to transport the payload of an original source packet. As depicted in Figure 6 (Structure of the FEC packet format for FEC Source packets), it consists of the original packet, optionally followed by the Explicit Source FEC Payload ID field. The FEC Scheme determines whether the Explicit Source FEC Payload ID field is required. This determination is specific to each ADU flow.

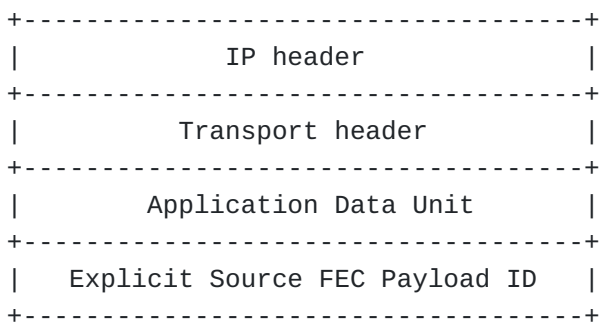

Figure 6: Structure of the FEC packet format for FEC Source packets

The FEC Source packets MUST be sent using the same ADU flow as would have been used for the original source packets if the FEC Framework were not present. The transport payload of the FEC Source packet MUST consist of the Application Data Unit followed by the Explicit Source FEC Payload ID field, if required.
The Explicit Source FEC Payload ID field contains information required to associate the source packet with a source block and for the operation of the FEC algorithm and is defined by the FEC Scheme. The format of the Source FEC Payload ID field is defined by the FEC Scheme. Note that in the case that the FEC Scheme or CDP defines a means to derive the Source FEC Payload ID from other information in the packet (for example a sequence number used by the application protocol), then the Source FEC Payload ID field is not included in the packet. In this case the original source packet and FEC Source Packet are identical.
In applications where avoidance of IP packet fragmentation is a goal, Content Delivery Protocols SHOULD consider the Explicit Source FEC Payload ID size when determining the size of Application Data Units that will be delivered using the FEC Framework. This is because the addition of the Explicit Source FEC Payload ID increases the packet length.
Note: The Explicit Source FEC Payload ID is placed at the end of the packet so that in the case that Robust Header Compression [RFC3095] (Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H.,

Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T., and H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed," July 2001.) or other header compression mechanisms are used and in the case that a ROHC profile is defined for the protocol carried within the transport payload (for example RTP), then ROHC will still be applied for the FEC Source packets.
Applications that may be used with this Framework should consider that FEC Schemes may add this Explicit Source FEC Payload ID and thereby increase the packet size.
In many applications, support for Forward Error Correction is added to a pre-existing protocol and in this case use of the Explicit Source FEC Payload ID may break backwards compatibility, since source packets are modified.

5.3.1. Generic Explicit Source FEC Payload Id

TOC

In order to apply FEC protection using multiple FEC Schemes to a single source flow all schemes must use the same Explicit Source FEC Payload Id format. In order to enable this, it is RECOMMENDED that FEC Schemes support the Generic Explicit Source FEC Payload Id format described below.
The Generic Explicit Source FEC Payload Id has length of 2 bytes and consists of an unsigned packet sequence number in network byte order. The allocation of sequence numbers to packets is independent of any FEC Scheme and of the Source Block construction, except that the use of this sequence number places a constraint on source block construction. Source packets within a given source block MUST have consecutive sequence numbers (where consecutive includes wrap-around from the maximum value which can be represented in 2 bytes - 65535 - to 0). Sequence numbers SHOULD NOT be reused until all values in the sequence number space have been used.

5.4. Packet Format for FEC Repair packets

TOC

The packet format for FEC Repair packets is shown in Figure 7 (Packet format for repair packets). The transport payload consists of a Repair FEC Payload ID field followed by repair data generated in the FEC encoding process.

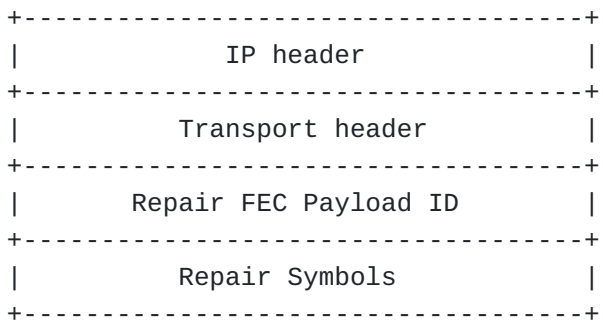

Figure 7: Packet format for repair packets

The Repair FEC Payload ID field contains information required for the operation of the FEC algorithm at the receiver. This information is defined by the FEC Scheme. The format of the Repair FEC Payload ID field is defined by the FEC Scheme.

5.4.1. Packet Format for FEC Repair packets over RTP

TOC

For FEC Schemes which specify the use of RTP for repair packets, the packet format for repair packets includes an RTP header as shown in Figure 8 (Packet format for repair packets).

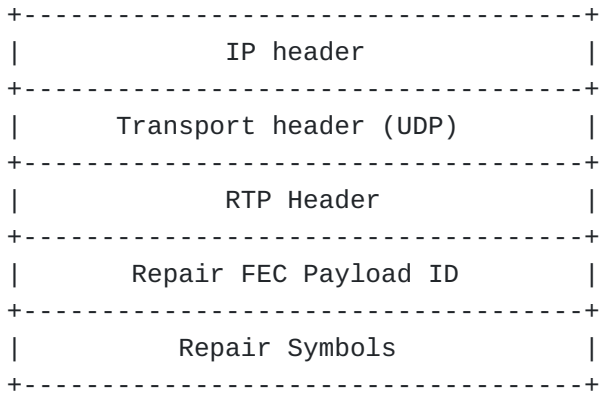

Figure 8: Packet format for repair packets

5.5. FEC Framework Configuration Information

TOC

The FEC Framework Configuration Information is information that the FEC Framework needs in order to apply FEC protection to the ADU flows. A complete Content Delivery Protocol specification that uses the framework specified here MUST include details of how this information is derived and communicated between sender and receiver.

The FEC Framework Configuration Information includes identification of the set of Source Data Flows. For example, in the case of UDP, each Source Data Flow is uniquely identified by a tuple { Source IP Address, Destination IP Address, Source UDP port, Destination UDP port }. Note that in some applications some of these fields may contain wildcards, so that the flow is identified by a subset of the fields and in particular in many applications the limited tuple { Destination IP Address, Destination UDP port } is sufficient.

A single instance of the FEC Framework provides FEC protection for packets of the specified set of Source Data Flows, by means of one or more packet flows consisting of repair packets. The FEC Framework Configuration Information includes, for each instance of the FEC Framework:

1. Identification of the packet flow(s) carrying FEC Repair packets, known as the FEC repair flow(s).

2. For each Source Data Flow protected by the FEC repair flow(s):

   a. Definition of the Source Data Flow carrying source packets (for example, by means of a tuple as described above for UDP).

   b. An integer identifier for this Source Data Flow. This identifier MUST be unique amongst all Source Data Flows which are protected by the same FEC repair flow.

3. The FEC Encoding ID, identifying the FEC Scheme

4. The length of the Explicit Source FEC Payload Id, in bytes

5. Zero or more FEC-Scheme-specific information elements, each consisting of a name and a value where the valid element names and value ranges are defined by the FEC Scheme

Multiple instances of the FEC Framework, with separate and independent FEC Framework Configuration Information, may be present at a sender or receiver. A single instance of the FEC Framework protects packets of the Source Data Flows identified in (2) above i.e. all packets sent on those flows MUST be FEC Source packets as defined in Section 5.3 (Packet format for FEC Source packets). A single Source Data Flow may be protected by multiple instances of the FEC Framework.

The integer flow identifier identified in 2(b) is a "shorthand" to identify source flows between the FEC Framework and the FEC Scheme. The reason for defining this as an integer, and including it in the FEC Framework Configuration Information is so that the FEC Scheme at the sender and receiver may use it to identify the source flow with which a recovered packet is associated. The integer flow identifier may therefore take the place of the complete flow description (e.g. UDP 4-tuple).

Whether and how this flow identifier is used is defined by the FEC Scheme. Since source packets are directly associated with a flow by virtue of their packet headers, this identifier need not be carried in source packets. Since repair packets may provide protection for multiple source flows, repair packets would either not carry the identifier at all or may carry multiple identifiers. However, in any case, the flow identifier associated with a particular source packet may be recovered from the repair packets as part of an FEC decoding operation. Integer flow identifiers SHOULD be allocated starting from zero and increasing by one for each flow.

A single FEC repair flow provides repair packets for a single instance of the FEC Framework. Other packets MUST NOT be sent within this flow i.e. all packets in the FEC repair flow MUST be FEC repair packets as defined in Section 5.4 (Packet Format for FEC Repair packets) and MUST relate to the same FEC Framework instance.

In the case that RTP is used for repair packets, the identification of the repair packet flow MAY also include the RTP Payload Type to be used for repair packets.

FEC Scheme-specific information elements MAY be encoded into a text string for transport within Content Delivery Protocols. See Section 4.5 of [I-D.ietf-fecframe-sdp-elements] (Begen, A., “Session Description Protocol (SDP) Elements for FEC Framework,” August 2010.) for the ABNF [RFC5234] (Crocker, D. and P. Overell, “Augmented BNF for Syntax Specifications: ABNF,” January 2008.) syntax.

5.6. FEC Scheme requirements

TOC

In order to be used with this framework, an FEC Scheme MUST be capable of processing data arranged into blocks of Application Data Units (source blocks).

A specification for a new FEC scheme MUST include the following things:

1. The FEC Encoding ID value that uniquely identifies the FEC scheme. This value MUST be registered with IANA as described in Section 10 (IANA Considerations).

2. The type, semantics and encoding format of the Repair FEC Payload ID.

3. The name, type, semantics and text value encoding rules for zero or more FEC Scheme-specific FEC Framework Configuration Information elements. Names must conform to the `name` production and values encodings to the `value` production defined in Section 5.5 (FEC Framework Configuration Information)

4. A full specification of the FEC code.

   This specification MUST precisely define the valid FEC-Scheme-Specific FEC Framework Configuration 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 symbol bytes).

   Furthermore, given a source block as defined in Section 5.2 (Structure of the source block), valid values of the FEC-Scheme-Specific FEC Framework Configuration Information, a valid Repair FEC Payload ID value and a valid packet payload size, the specification MUST uniquely define the values of the encoding symbol bytes to be included in the repair packet payload of a packet with the given Repair FEC Payload ID value.

   A common and simple way to specify the FEC code to the required level of detail is to provide a precise specification of an encoding algorithm which, given a source block, valid values of the FEC-Scheme-Specific FEC Framework Configuration Information, a valid Repair FEC Payload ID value and a valid packet payload size as input produces the exact value of the encoding symbol bytes as output.

5. A description of practical encoding and decoding algorithms.

   This description need not be to the same level of detail as for the encoding above, however it must be sufficient to demonstrate that encoding and decoding of the code is both possible and practical.

FEC scheme specifications MAY additionally define the following:

1. Type, semantics and encoding format of an Explicit Source FEC Payload ID.

Whenever an FEC scheme specification defines an 'encoding format' for an element, this must be defined in terms of a sequence of bytes which can be embedded within a protocol. The length of the encoding format MUST either be fixed or it must be possible to derive the length from examining the encoded bytes themselves. For example, the initial bytes may include some kind of length indication.
FEC scheme specifications SHOULD use the terminology defined in this document and SHOULD follow the following format:

1. **Introduction**
   <describe the use-cases addressed by this FEC scheme>

2. **Formats and Codes**

   2.1 Source FEC Payload ID(s) <Either, define the type and format of the Explicit Source FEC Payload ID, or define how Source FEC Payload ID information is derived from source packets>

   2.2 Repair FEC Payload Id <Define the type and format of the Repair FEC Payload ID>

   2.3 FEC Framework Configuration Information <Define the names, types and text value encoding formats of the FEC Scheme-specific FEC Framework configuration information elements>

3. **Procedures** <describe any procedures which are specific to this FEC scheme, in particular derivation and interpretation of the fields in the FEC Payload ID and FEC Scheme-specific FEC Framework configuration information.>

4. **FEC code specification** <provide a complete specification of the FEC Code>

Specifications MAY include additional sections, for example, examples. Each FEC scheme MUST be specified independently of all other FEC schemes; for example, in a separate specification or a completely independent section of larger specification (except, of course, a specification of one FEC Scheme may include portions of another by reference).
Where an RTP Payload Format is defined for repair data for a specific FEC Scheme, the RTP Payload Format and the FEC Scheme MAY be specified within the same document.

---

6. Feedback

TOC

Many applications require some kind of feedback on transport performance: how much data arrived at the receiver, at what rate, when etc. When FEC is added to such applications, feedback mechanisms may also need to be enhanced to report on the performance of the FEC (for example how much lost data was recovered by the FEC).
When used to provide instrumentation for engineering purposes, it is important to remember that FEC is generally applied to relatively small blocks of data (in the sense that each block is transmitted over a

relatively small period of time) and so feedback information averaged over longer periods of time than the FEC block transmission time will likely not provide sufficient information for engineering purposes. For example see [RFC5725] (Begen, A., Hsu, D., and M. Lague, “Post-Repair Loss RLE Report Block Type for RTP Control Protocol (RTCP) Extended Reports (XRs),” February 2010.).
Applications which used feedback for congestion control purposes MUST calculate such feedback on the basis of packets received before FEC recovery is applied. If this requirement conflicts with other uses of the feedback information then the application MUST be enhanced to support both information calculated pre- and post- FEC recovery. This is to ensure that congestion control mechanisms operate correctly based on congestion indications received from the network, rather than on post-FEC recovery information which would give an inaccurate picture of congestion conditions.
New applications which require such feedback SHOULD use RTP/RTCP [RFC3550] (Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications,” July 2003.).

---

7. Transport Protocols

TOC

This framework is intended to be used to define Content Delivery Protocols which operate over transport protocols which provide an unreliable datagram service, including in particular the User Datagram Protocol (UDP) and the Datagram Congestion Control Protocol (DCCP).

---

8. Congestion Control

TOC

This section starts with a informative section on the motivation of the normative requirements for congestion control, which are spelled out in Section 8.1 (Normative requirements).

> Informative Note: The enforcement of Congestion Control (CC) principles has gained a lot of momentum in the IETF over the recent years. While the need of CC over the open Internet is unquestioned, and the goal of TCP friendliness is generally agreed for most (but not all) applications, the subject of congestion detection and measurement in heterogeneous networks can hardly be considered as solved. Most congestion control algorithms detect and measure congestion by taking (primarily or exclusively) the packet loss rate into account. This appears to be inappropriate in environments where a large percentage of the packet losses are the result of link-layer errors and independent of the network load. Note that such environments exist in the "open Internet", as well as in "closed" IP

based networks. An example for the former would be the use of IP/UDP/RTP based streaming from an Internet-connected streaming server to a device attached to the Internet using cellular technology.

The authors of this draft are primarily interested in applications where the application reliability requirements and end-to-end reliability of the network differ, such that it warrants higher layer protection of the packet stream - for example due to the presence of unreliable links in the end-to-end path - and where real-time, scalability or other constraints prohibit the use of higher layer (transport or application) feedback. A typical example for such applications is multicast and broadcast streaming or multimedia transmission over heterogeneous networks. In other cases, application reliability requirements may be so high that the required end-to-end reliability is difficult to achieve even over wired networks. Furthermore the end-to-end network reliability may not be known in advance.

This FEC framework is not proposed, nor intended, as a QoS enhancement tool to combat losses resulting from highly congested networks. It should not be used for such purposes.

In order to prevent such mis-use, one approach would be to leave standardisation to bodies most concerned with the problem described above. However, the IETF defines base standards used by several bodies, including DVB, 3GPP, 3GPP2, all of which appear to share the environment and the problem described.

Another approach would be to write a clear applicability statement - for example restricting use of the framework to networks with wireless links. However, there may be applications where the use of FEC may be justified to combat congestion-induced packet losses - particularly in lightly loaded networks, where congestion is the result of relatively rare random peaks in instantaneous traffic load - thereby intentionally violating congestion control principles. One possible example for such an application could be a no-matter-what, brute-force FEC protection of traffic generated as an emergency signal.

We propose a third approach, which is to require at a minimum that the use of this framework with any given application, in any given environment, does not cause congestion issues which the application alone would not itself cause i.e. the use of this framework must not make things worse.

Taking above considerations into account, Section 8.1 (Normative requirements) specifies a small set of constraints for the FEC, which are mandatory for all senders compliant with this FEC framework. Further restrictions may be imposed for certain Content

> Delivery Protocols. In this it follows the spirit of the congestion control section of RTP and its Audio-Visual Profile (RFC3550/STD64 and RFC3551/STD65).
>
> One of the constraints effectively limits the bandwidth for the FEC protected packet stream to be no more than roughly twice as high as the original, non-FEC protected packet stream. This disallows the (static or dynamic) use of excessively strong FEC to combat high packet loss rates, which may otherwise be chosen by naively implemented dynamic FEC-strength selection mechanisms. We acknowledge that there may be a few exotic applications, e.g. IP traffic from space-based senders, or senders in certain hardened military devices, which would warrant a higher FEC strength. However, in this specification we give preference to the overall stability and network friendliness of the average application, and for those a factor of 2 appears to be appropriate.
>
> A second constraint requires that the FEC protected packet stream be in compliance with the congestion control in use for the application and network in question.

8.1. Normative requirements

TOC

The bandwidth of FEC Repair packet flows MUST NOT exceed the bandwidth of the source packet flows being protected. In addition, whenever the source packet flow bandwidth is adapted due to the operation of congestion control mechanisms, the FEC repair packet flow bandwidth MUST be similarly adapted.

---

9. Security Considerations

TOC

The application of FEC protection to a stream does not provide any kind of security protection.
If security services are required for the stream, then they MUST either be applied to the original source data before FEC protection is applied, or to both the source and repair data, after FEC protection has been applied.
If integrity protection is applied to source packets before FEC protection is applied, and no further integrity protection is applied to repair packets, then a denial of service attack is possible if an attacker is in a position to inject fake repair transport payloads. If received by a receiver, such fake repair transport payloads could cause incorrect FEC decoding resulting in incorrect Application Data Units

being passed up to the application protocol. A similar attack may be possible if an attacker is in a position to inject fake FEC Framework Configuration Information or fake FEC Payload IDs. Such incorrect decoded Application Data Units would then be detected by the source integrity protection and discarded, resulting in partial or complete denial of service. Therefore, in such environments, integrity protection MUST also be applied to the FEC repair transport payloads, FEC Framework Configuration Information and FEC Payload IDs, for example using IPsec to integrity protect all packets. Receivers MUST also verify the integrity of source symbols before including the source symbols into the source block for FEC purposes.

It is possible that multiple streams with different confidentiality requirements (for example, the streams may be visible to different sets of users) can be FEC protected by a single repair stream. This scenario is not recommended, since resources will be used to distribute and FEC decode encrypted data which cannot then be decrypted by at least some receivers. However, in this scenario, confidentiality protection MUST be applied before FEC encoding of the streams, otherwise repair transport payload may be used by a receiver to decode unencrypted versions of source streams which they do not have permissions to view.

---

10. IANA Considerations

TOC

FEC Schemes for use with this framework may be identified in protocols using FEC Encoding IDs. Values of FEC Encoding IDs are subject to IANA registration. They are in the registry named "FEC Framework (FECFRAME) FEC Encoding IDs" located at time of publication at <tbd>.

The values that can be assigned within the FEC Framework (FECFRAME) FEC Encoding ID registry are numeric indexes in the range [0, 255], boundaries included. Assignment requests are granted on a "IETF Consensus" basis as defined in[RFC5226] (Narten, T. and H. Alvestrand, “Guidelines for Writing an IANA Considerations Section in RFCs,” May 2008.) . Section 5.6 (FEC Scheme requirements) defines explicit requirements that documents defining new FEC Encoding IDs should meet.

---

11. Acknowledgments

TOC

This document is based in part on [I-D.watson-tsvwg-fec-sf] (Watson, M., “Forward Error Correction (FEC) Streaming Framework,” July 2005.) and so thanks are due to the additional authors of that document, Mike Luby, Magnus Westerlund and Stephan Wenger. That document was in turn based on the FEC streaming protocol defined by 3GPP in [MBMSTS] (3GPP, “Multimedia Broadcast/Multicast Service (MBMS); Protocols and codecs,” April 2005.) and thus thanks are also due to the participants in 3GPP

TSG SA working group 4. Further thanks are due to the members of the FECFRAME working group for their comments and review.

12. References

TOC

12.1. Normative references

TOC

[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).
[RFC3095]	Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H., Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T., and H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed," RFC 3095, July 2001 (TXT).
[RFC5052]	Watson, M., Luby, M., and L. Vicisano, "Forward Error Correction (FEC) Building Block," RFC 5052, August 2007 (TXT).
[RFC3550]	Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," STD 64, RFC 3550, July 2003 (TXT, PS, PDF).
[RFC5226]	Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs," BCP 26, RFC 5226, May 2008 (TXT).
[RFC5234]	Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF," STD 68, RFC 5234, January 2008 (TXT).

12.2. Informative references

TOC

[I-D.watson-tsvwg-fec-sf]	Watson, M., "Forward Error Correction (FEC) Streaming Framework," draft-watson-tsvwg-fec-sf-00 (work in progress), July 2005 (TXT).
[RFC5725]	Begen, A., Hsu, D., and M. Lague, "Post-Repair Loss RLE Report Block Type for RTP Control Protocol (RTCP) Extended Reports (XRs)," RFC 5725, February 2010 (TXT).
[RFC4588]	

	Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R. Hakenberg, “RTP Retransmission Payload Format,” RFC 4588, July 2006 (TXT).
[RFC2736]	Handley, M. and C. Perkins, “Guidelines for Writers of RTP Payload Format Specifications,” BCP 36, RFC 2736, December 1999 (TXT).
[I-D.ietf-fecframe-sdp-elements]	Begen, A., “Session Description Protocol (SDP) Elements for FEC Framework,” draft-ietf-fecframe-sdp-elements-08 (work in progress), August 2010 (TXT).
[MBMSTS]	3GPP, “Multimedia Broadcast/Multicast Service (MBMS); Protocols and codecs,” 3GPP TS 26.346, April 2005.

Author's Address

TOC

	Mark Watson
	Netflix, Inc.
	100 Winchester Circle
	Los Gatos, CA, CA 95032
	U.S.A.
Email:	watsonm@netflix.com