FEC Framework
Internet-Draft
Intended status: Standards Track
Expires: March 4, 2011

S. Galanos
O. Peck
RADVISION
V. Roca
INRIA

August 31, 2010

RTP Payload Format for Reed Solomon FEC
draft-galanos-fecframe-rtp-reedsolomon-02

Abstract

This document defines an RTP payload format for the Forward Error Correction (FEC) that uses Reed-Solomon codes. The format defined by this document enables the protection of source media encapsulated in RTP with one or more repair flows and is based on the FEC framework and the SDP Elements for FEC Framework. The Reed-Solomon codes used in this document belong to the class of Maximum Distance Separable (MDS) codes which means they offer optimal protection against random and bursty packet losses.

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 4, 2011.

Copyright Notice

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

This document defines new RTP payload formats for the Forward Error Correction (FEC) that is generated by the Reed-Solomon code.

By nature, interactive Real-time applications are extremely sensitive to delay and require very low latency. As a result, retransmission of lost packets and using other closed-loop schemes are not valid options while the use of Forward Error Correction (FEC) is an effective approach.

A primary requirement from FEC for real time applications is the ability to correctly recover from both random and bursty packet losses. The Reed-Solomon FEC codes used in this document belong to the class of Maximum Distance Separable (MDS) codes that are optimal in terms of erasure recovery capability for both situations.

The format defined by this document enables the protection of media source flow with one or more repair flows without adding additional information to the source packets. Such behavior reduces the delay presented by any FEC scheme and maintains backwards compatibility with non FEC-enabled receivers.

Number of previous drafts were composed to draw different FEC schemes suitable for different applications. The scheme defined in this draft is designed to compensate a burst of packet loss over RTP networks with minimum delay, which is needed in interactive IP-based applications such as video conferencing.

The method described in this document is generic to all media types and provides the sender with the flexibility of deciding if FEC protection is required and if so, how many protecting packets and how many source packets to use in a block according to network conditions. Furthermore it allows applying unequal error protection that provides different level of protection to different packets. For example, it can be combined with Scalable Video Coding to protect only the base layer packets of the video flow. At the receiver, both the FEC and original media are received. If no media packets are lost, the FEC packets can be ignored. In the event of a loss, the FEC packets can be combined with other received media to recover all or part of the missing media packets.

The Reed-Solomon codes used in this document have been specified in [RFC5510] and are compatible with Luigi Rizzo codec (see [Rizzo97]). This document is compliant with the Forward Error Correction (FEC) Framework (described in [I-D.ietf-fecframe-framework]) and SDP Elements for FEC Framework (described in [I-D.ietf-fecframe-sdp-elements] [RFC4566]). This draft completes

[I-D.roca-fecframe-rs] by defining Reed-Solomon usage for RTP transport ([RFC3550]) and specifies the appropriate media types (see [RFC4288] [RFC3555]).

2. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Definitions, Notations and Abbreviations

This document uses the following definitions and notations. For further definitions that apply to FEC Framework in general, see [I-D.ietf-fecframe-framework].

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.

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

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

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

Systematic code: FEC code in which the source symbols are part of the encoding symbols. The Reed-Solomon codes introduced in this document are systematic.

Source block: a block of k source symbols that are considered together for the encoding.

Packet Erasure Channel: a communication path where packets are either dropped (e.g., by a congested router, or because the number of transmission errors exceeds the correction capabilities of the physical layer codes) or received. When a packet is received, it

is assumed that this packet is not corrupted.

Some of them are FECFRAME framework specific and are in line with [I-D.ietf-fecframe-framework]:

Application Data Unit (ADU): a unit of data coming from (sender) or given to (receiver) the media delivery application. In this document, an ADU MUST use an RTP encapsulation.

(Source) ADU Flow: a flow of ADUs from a media delivery application and to which FEC protection is applied. In this document, there MUST be a single ADU flow per FECFRAME framework instance.

ADU Block: a set of ADUs that are considered together by the FECFRAME instance for the purpose of the FEC scheme.

FEC Framework Configuration Information: the FEC scheme specific information that enables the synchronization of the FECFRAME sender and receiver instances.

FEC Source Packet: an RTP data packet submitted to (sender) or received from (receiver) the transport protocol. In this document, FEC Source Packets and ADU MUST be the same (e.g., for backward compability purposes).

FEC Repair Packet: an RTP repair packet submitted to (sender) or received from (receiver) the transport protocol. It contains a repair symbol along with its Explicit Repair FEC Payload ID.

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_r denotes the number of repair symbols generated for a source block.

n denotes the number of encoding symbols generated for a source block. Therefore: n = k + n_r.

max_n denotes the maximum number of encoding symbols generated for any source block.

S denotes the encoding symbol length in units of m-bit elements. When m = 8, then S and E are equal.

GF(q) denotes a finite field (also known as Galois Field) with q elements. We assume that q = 2^^m in this document.

m defines the length of the elements in the finite field, in bits. In this document, m belongs to {2..16}.

q defines the number of elements in the finite field. We have: q = 2^^m in this specification.

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

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

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 Correction code.

FFCI stands for FEC Framework Configuration Information.

RS stands for Reed-Solomon.

MDS stands for Maximum Distance Separable code.

4. Reed Solomon Codes

The detailed operation and theory behind Reed Solomon codes is out of the scope of this document. In general a Reed Solomon code takes a group of k source symbols and generates n - k repair symbols. A receiver needs to receive any k of the n source or repair symbols in order to recover the remaining n-k symbols. As explained in RFC 5510, the Reed-Solomon algorithm operates over multiple elements each taken from a single source symbol. Symbols are composed of S "m-bit elements" where m is the Galois Field exponent GF(2^m). In the usual case of GF(2^8), elements are bytes, and the size S in terms of elements is of course equal to the symbol size in bytes. The symbol

size can be different in different implementations. Any symbol size can be used in the format offered by this document. However, it is recommended in terms of implementation simplicity to use 8-bits elements. For more information on Reed Solomon codes, the reader is referred to [Rizzo97].

5. Source Block Creation

This draft defines the protection of an RTP source flow using one or more FEC repair flows.

A source block for the Reed-Solomon code contains k source symbols. In the scheme presented by this document, each source symbol contains a single Application Data Unit (ADU, as defined in [I-D.ietf-fecframe-framework]), which is in our case an RTP packet. Therefore a source block contains exactly k RTP packets. The Reed-Solomon code generates n_r = n - k repair symbols that are transmitted using n_r = n - k FEC repair packets. Each FEC repair packet contains a single repair block.

To create a source block the steps outlined below should be followed:

1. Determine the largest RTP packet size (in bytes) of the source block. During this computation, both the RTP header and payload are considered

2. For each ADU of this source block, create a byte array (of size 2 + this largest RTP packet size), as follows:

   A. In the first two bytes, place the unsigned network-ordered 16-bit representation of the RTP packet size in bytes (including RTP header size and payload size)

   B. Append the entire RTP packet including its RTP header

   C. Add zero padding so that the byte array is the size of the largest packet protected by this source block plus two (to consider the initial two bytes). Therefore, the largest packet does not contain padding.

3. Append all the byte arrays one after the other in the following way:

   A. The packets are in an increasing order of the sequence number as it appears in the RTP packet header taking wraparound into account

Figure 1 demonstrates how a source block is created from 4 packets (P1, P2, P3, P4) with different sizes. The largest packet protected in this source block has a size of 5 (L = 5) and therefore P1 and P3 are both padded with zeros to this size. The source block contains the RTP packet size before each packet. (Note that this example is not a binary representation of the source block. The Packet size spans over two bytes as stated above)

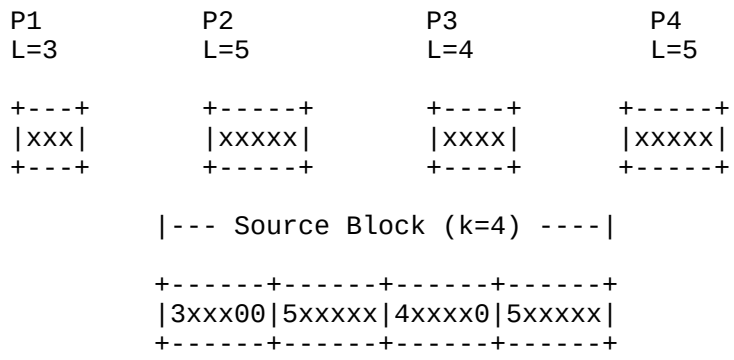

Figure 1: Structure of a Source Block

The FEC Reed-Solomon Scheme gets a source block created from k packets and generates n-k FEC repair packets that protect the entire source block. These packets are then transmitted in the repair flow. Note that source packets padding is done only for FEC packet calculation and the original payloads are transmitted without extra padding.

6. Packet Formats

This section defines the formats of the source and repair packets

6.1. FEC Source Packets

The FEC Framework requires that FEC source packets contain information identifying the source block and the position within the source block occupied by the packet. However, in order to maintain backwards compatibility, the scheme defined by this document enables the receiver to get this information without appending additional information to the source packet. Specifically this information is obtained using the combination of sequence number found in the RTP header and information provided in the FEC header of each FEC repair packet. Such behavior enables both non-FEC-capable and FEC-capable receivers to receive and interpret the same source packets sent in a

multicast session.

6.2. FEC Repair Packets

The FEC repair packets contain information that enables the receiver to reconstruct the source block in the remote end. This is done by using the RTP header of the FEC repair packets as well as another dedicated header that is placed within the RTP payload. This header, referred to as the FEC header, complies with [I-D.ietf-fecframe-framework] (section 6.4.1), as shown in Figure 2.

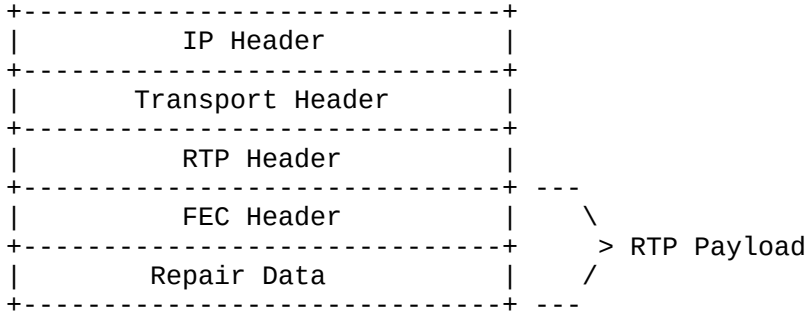

Figure 2: Format of repair packets

6.2.1. RTP header format

The RTP header is formatted according to [RFC3550] with some further clarifications listed below:

- Marker (M) Bit: This bit is not used for this payload type, and is set to 0.

- Payload Type: The (dynamic) payload type for the repair packets is determined through out-of-band means. Note that this document registers new payload formats for the repair packets (Refer to Section 5 for details). According to [RFC3550], an RTP receiver that cannot recognize a payload type must discard it. This provides for backward compatibility. The FEC mechanisms can then be used in a multicast group with mixed FEC-capable and non-FEC-capable receivers. If a non-FEC-capable receiver receives a repair packet, it will not recognize the payload type, and hence, will discard the repair packet. In case more than one repair flow is used, different Payload Types will be used to distinguish between the different flows.

- Sequence Number (SN): The sequence number maintains the standard definition. It is one higher than the sequence number in the previously transmitted repair packet. The initial value of the sequence number is random (unpredictable) [RFC3550].

- Timestamp (TS): The timestamp is set to a time corresponding to the repair packet's transmission time. Note that the timestamp value has no use in the actual FEC protection process and is usually useful for jitter calculations. FEC packets that are the result of the same FEC encoding operation will use the same value as their Timestamp.

- Synchronization Source (SSRC): The SSRC value is randomly assigned as suggested by [RFC3550].

6.2.2. FEC header format

The FEC header includes information that enables the receiver to reconstruct the source block and to identify the FEC repair packets associated with each source block, in their correct order.

The format of the FEC header is shown in figure 3.

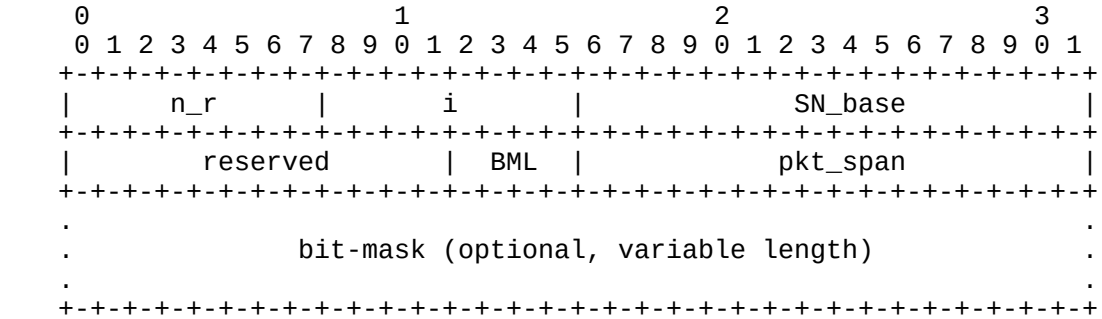

Figure 3: FEC Header Format

The FEC header consists of the following general fields:

- n_r (8-bit field): the number of FEC repair packets used to protect this source block.

- i (8-bit field): the 0-based index in the sequence of n_r FEC repair packets. This index is equal to ESI - k, where ESI is the Encoding Symbol ID of the associated repair symbol.

- SN_base (16-bit field): the lowest RTP sequence number (taking wraparound into account) of the FEC source packets in the

associated source block. This SN_base also identifies the source block. In order to avoid any risk of confusion, two consecutive source blocks MUST use different SN_base values, which is easily verified by the sender (this situation might happen with Reed-Solomon over GF(2^16)).

- o pkt_span (16-bit field): the number of consecutive FEC Source Packets considered. A subset of these FEC Source Packets may be missing, as indicated by the 0 entries of the optional bit-mask.

- o Reserved (12-bit field): reserved for future use. This field MUST be set to zero in this specification.

- o Bit Mask Length, BML (4-bit field): when the pkt_span source packets of the source block don't have consecutive RTP sequence numbers, a bit-mask MUST be used to indicate which packets are protected by this FEC packet. This field indicates the length of the bit-mask in units of 32-bit words, as the following table shows. In any case, only the first pkt_span bits of this bit-field are meaningful, the remaining bits (if any) MUST be set to 0.

- o bit-mask (Optional field, length multiple of 32 bits): When BML is set to a value different than 0000, a bit-mask field is added, whose length in term of number of 32-bit words is indicated by the BML field. The bit-mask indicates which source packets have been considered in the source block ("1" bit valude entry in the bit-field) and which source packets have been ignored ("0" bit valude entry in the bit-field) (usually this happens when a source packet has been erased (lost) before reaching the FECFRAME encoder). The first packet in the bit-mask (corresponding to bit position 0 of the first 32-bit word) corresponds to the source packet whose RTP sequence number is specified in field SN_base.

BML value	bit-mask length (in bits)	bit-mask length (in words)
0	0 (no mask)	0
1	32 bits	1 x 32-bit word
10	64 bits	2 x 32-bit word
11	96 bits	3 x 32-bit word
100	128 bits	4 x 32-bit word
101	160 bits	5 x 32-bit word
110	192 bits	6 x 32-bit word
111	224 bits	7 x 32-bit word
1000	256 bits	8 x 32-bit word
...	...	...
1111	480 bits	15 x 32-bit word

6.2.3. Repair Data Format

The repair data follows the FEC header in the FEC repair packet. It includes the result of the Reed-Solomon encoding over the source block. Note that the first two bytes of the repair data contain the result of the Reed-Solomon encoding over the packet sizes in the source block and that the size of the repair data equals the size of the largest packet protected by this source block plus 2. Therefore, the size of an FEC repair packet (FEC header and data) is larger than the longest source packet. This should be taken under consideration when deciding on the Maximum Transmission Unit (MTU) size used for the source packets.

7. Payload Format Parameters

According to the FEC framework, when RTP is used as a transport for repair packet flows, the scheme must define an RTP Payload Format for the repair data. This section provides the media subtype registration for the Reed-Solomon FEC. The parameters that are required to configure the FEC encoding and decoding operations are also defined in this section.

7.1. Media Type Registration

This registration is done using the template defined in [RFC4288] and following the guidance provided in [RFC3555].

7.1.1. Registration of audio/reed-solomon-fec

Type name: audio

Subtype name: reed-solomon-fec

Required parameters:

- max_n: The upper limit for the sum of source and repair packets that belong to the same FEC block. max_n is a positive integer. The application can change both k and n-k. max_n is the upper limit for n. The value of max_n must be equal to or lower than the codec limitation (2^m).

- repair-window: The time that spans the source packets and the corresponding repair packets. The size of the repair window is specified in microseconds.

- The repair-window impacts the maximum number of source packets in a FEC block at the sender side, and defines the time which the receiver should wait for the repair packets. The repair-window value may be negotiated between the sender and receiver. the details of such negotiation are out-of-scope for this document.

- element-size: a non-negative integer indicating the length of each encoding elements in bits. This value equals to the "m" parameter in the GF (represented by 2^m).

Optional parameters: None.

Encoding considerations: This media type is framed and binary, see section 4.8 in [RFC4288]

Security considerations: Please see security consideration in [I-D.ietf-fecframe-framework]

Interoperability considerations: None.

Published specification: TBD

Applications that use this media type: Multimedia applications that want to improve resiliency against packet loss by sending redundant data in addition to the source media.

Additional information: None.

Magic number(s): none defined

File extension(s): none defined

Macintosh file type code(s): none defined

Person & email address to contact for further information: Sarit Galanos, sarit@radvision.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP [RFC3550]. Transport within other framing protocols is not defined at this time.

7.1.2. Registration of video/reed-solomon-fec

Type name: video

Subtype name: reed-solomon-fec

Required parameters:

- max_n: The upper limit for the sum of source and repair packets that belong to the same FEC block. max_n is a positive integer. The application can change both k and n-k. max_n is the upper limit for n. The value of max_n must be equal to or lower than the codec limitation (2^m).

- repair-window: The time that spans the source packets and the corresponding repair packets. The size of the repair window is specified in microseconds.

- The repair-window impacts the maximum number of source packets in a FEC block at the sender side, and defines the time which the receiver should wait for the repair packets. The repair-window value may be negotiated between the sender and receiver. the details of such negotiation are out-of-scope for this document.

- element-size: a non-negative integer indicating the length of each encoding elements in bits. This value equals to the "m" parameter in the GF (represented by 2^m).

Optional parameters: None.

Encoding considerations: This media type is framed and binary, see section 4.8 in [RFC4288]

Security considerations: Please see security consideration in [I-D.ietf-fecframe-framework]

Interoperability considerations: None.

Published specification: TBD

Applications that use this media type: Multimedia applications that want to improve resiliency against packet loss by sending redundant data in addition to the source media.

Additional information: None.

Magic number(s): none defined

File extension(s): none defined

Macintosh file type code(s): none defined

Person & email address to contact for further information: Sarit Galanos, sarit@radvision.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP [RFC3550]. Transport within other framing protocols is not defined at this time.

7.1.3. Registration of text/reed-solomon-fec

Type name: text

Subtype name: reed-solomon-fec

Required parameters:

- max_n: The upper limit for the sum of source and repair packets that belong to the same FEC block. max_n is a positive integer. The application can change both k and n-k. max_n is the upper limit for n. The value of max_n must be equal to or lower than the codec limitation (2^m).

- repair-window: The time that spans the source packets and the corresponding repair packets. The size of the repair window is specified in microseconds.

- The repair-window impacts the maximum number of source packets in a FEC block at the sender side, and defines the time which the receiver should wait for the repair packets. The repair-window value may be negotiated between the sender and receiver. the details of such negotiation are out-of-scope for this document.

- element-size: a non-negative integer indicating the length of each encoding elements in bits. This value equals to the "m" parameter in the GF (represented by 2^m).

Optional parameters: None.

Encoding considerations: This media type is framed and binary, see section 4.8 in [RFC4288]

Security considerations: Please see security consideration in [I-D.ietf-fecframe-framework]

Interoperability considerations: None.

Published specification: TBD

Applications that use this media type: Multimedia applications that want to improve resiliency against packet loss by sending redundant data in addition to the source media.

Additional information: None.

Magic number(s): none defined

File extension(s): none defined

Macintosh file type code(s): none defined

Person & email address to contact for further information: Sarit Galanos, sarit@radvision.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP [RFC3550]. Transport within other framing protocols is not defined at this time.

7.1.4. Registration of application/reed-solomon-fec

Type name: application

Subtype name: reed-solomon-fec

Required parameters:

- max_n: The upper limit for the sum of source and repair packets that belong to the same FEC block. max_n is a positive integer. The application can change both k and n-k. max_n is the upper

limit for n. The value of max_n must be equal to or lower than the codec limitation (2^m).

- repair-window: The time that spans the source packets and the corresponding repair packets. The size of the repair window is specified in microseconds.

- The repair-window impacts the maximum number of source packets in a FEC block at the sender side, and defines the time which the receiver should wait for the repair packets. The repair-window value may be negotiated between the sender and receiver. the details of such negotiation are out-of-scope for this document.

- element-size: a non-negative integer indicating the length of each encoding elements in bits. This value equals to the "m" parameter in the GF (represented by 2^m).

Optional parameters: None.

Encoding considerations: This media type is framed and binary, see section 4.8 in [RFC4288]

Security considerations: Please see security consideration in [I-D.ietf-fecframe-framework]

Interoperability considerations: None.

Published specification: TBD

Applications that use this media type: Multimedia applications that want to improve resiliency against packet loss by sending redundant data in addition to the source media.

Additional information: None.

Magic number(s): none defined

File extension(s): none defined

Macintosh file type code(s): none defined

Person & email address to contact for further information: Sarit Galanos, sarit@radvision.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP [RFC3550]. Transport

within other framing protocols is not defined at this time.

7.2. Mapping of SDP Parameters

For a proper operation details of the FEC scheme have to be communicated between the sender and the receiver. Specifically, the receiver has to know the relationship between the source and the repair flows, how the sender applied protection to the source flow and how the repair flows can be used to recover the lost data. One way to provide this information is to use the Session Description Protocol (SDP) [RFC4566].

The mapping of the media type specification for "reed-solomon-fec" and their parameters in SDP is as follows:

- The media type (e.g., "application") goes into the "m=" line as the media name.
- The media subtype ("reed-solomon-fec") goes into the "a=rtpmap" line as the encoding name.
- The remaining required payload-format-specific parameters ("max_n", "repair-window") go into the "a=fmtp" line by copying them directly from the media type string as a semicolon-separated list of parameter=value pairs.

See section 9 for SDP examples.

8. Protection and Recovery Procedures

This section provides a complete specification of the protection and recovery procedures.

8.1. Overview

The FEC repair packets allow end-systems to recover from a loss of media packets. The following sections specify the steps involved in generating the FEC repair packets and reconstructing the missing source packets from the FEC repair packets.

8.2. FEC Repair Packet Construction

The RTP header of a FEC repair packet is formed based on the guidelines given in Section 6.2.1. The FEC header is formed based on the guidelines given in Figure 3. Before Reed-Solomon encoding, two bytes are prepended to each ADU (RTP source packet in our case) that contain this ADU length in bytes, stored in network-order. FEC

encoding can then take place and the n_r = n - k repair symbols are created. Each repair symbol is then appended to its FEC header.

8.3. Source Packet Reconstruction

Recovery requires two distinct operations. The first operation determines which packets (source and repair) must be considered in order to recover the missing packets of a given block. Once this is done, the second step is the reconstruction of the missing data.

8.3.1. Associating the Source and Repair Packets

Association of the FEC source packets and FEC repair packets is done using a combination of the source packet sequence number and the information found in the RTP header and the FEC header of the FEC repair packets. The first step is to accumulate some of the n_r = n - k repair packets that were generated in the protection operation. For that the application has to follow the steps listed below:

- For each received packet, retrieve the payload type parameter from the RTP header to identify the packet as a repair packet of the reed-solomon scheme. In case multiple repair flows are used, different payload types will be used to distinguish between the different repair flows.

- If a FEC repair packet is received, retrieve the sequence number (SN) from the RTP header and the n_r and i parameters from the FEC header. With these parameters, identify the collection of FEC repair packets generated for the source block. For example, if n_r = 4, i = 2 and SN = 1003, the receiver deduces that 4 FEC repair packets with sequence numbers 1001, 1002, 1003 and 1004 have been generated for this source block.

- Still in case of a FEC repair packet, retrieve the BML, pkt_span and optional bit-mask fields. If BML equals 0, then k = pkt_span and the source packets have sequence numbers SN_base up to SN_base + k. If BML is greater than 0, the first pkt_span bits of the bit-mask must be analyzed. k is then equal to the number of bits equal to 1 in this bit-mask. The sequence numbers of the source packets that are acutally part of the source block are equal to SN_base plus the offset of the bits equal to 1 in this bit-mask.

8.3.2. Recovering the source packet

In order to recover the lost source packets, the application has to rebuild the source block according to the guidelines given in Section 5 and append the repair data to it in the correct order. Zero padding will replace the lost packets in the constructed source

block. The size of each source block data packet in bytes will be equal to the size of the repair data found in the repair packets. The repair data size is the size of the RTP payload in the repair packet without the FEC header information (see figure 2). The application will then append the repair data taken from each repair packet. This new block is provided to the Reed-Solomon code.

Reconstruction of lost packets (source or repair packets) is possible only if at least any k packets were received (source or repair).

The Reed-Solomon code will reconstruct the lost data into the provided source block overriding the zero padded blocks. The application can then recover the lost packets as follows:

- The first two bytes specify the RTP packet size.
- According to the RTP packet size the application can retrieve the RTP packet (RTP header and payload).
- Any extra padding bytes if exist are ignored.

9. SDP Examples

The following example demonstrates source flow with flow ID of 0 that is protected by a single repair flow R1.

```
v=0
o=sarit 1122334455 1122334466 IN IP4 fec.example.com
s= Reed Solomon FEC Example
t=0 0
a=group:FEC S1 R1
m=video 30000 RTP/AVP 100
c=IN IP4 233.252.0.1/127
a=rtpmap:100 MP2T/90000
a=fec-source-flow: id=0
a=mid:S1
m=application 30000 RTP/AVP 110
c=IN IP4 233.252.0.2/127
a=rtpmap:110 reed-solomon-fec /90000
a=fmtp:110 max_n:16; repair-window:200000; symbol-size:8
a=mid:R1
```

Figure 4

10. Implementation Considerations

Using Reed-Solomon FEC protection over RTP may be useful for efficiently overcoming newtork packet losses in interactive communications where latency constraints apply. Protection may be applied for small encoding blocks, and therefore latency caused by waiting for the FEC repair packets is minimized.

This document allows the application to set the FEC strength dynamically according to the experienced and measured loss rate, for optimizing bandwidth utilization while recovering from network errors.

When FEC protection is used due to network congestion conditions, it is important that the application will reduce the bandwidth used for FEC protection from the bandwidth used by the source flow, in order not to overload the already congested network with the additional FEC repair packets.

In order to minimize bandwidth overhead for repair packets, algorithm for applying FEC on source packets should be designed carefully. Using source packets with similar lengths (when possible) can minimize the bandwidth overhead of the FEC repair packets.

In order to maximize the FEC strength, when a ratio of k/n is chosen, the larger the source blocks size (n) is, the stronger the FEC protection is. Of course, on the other hand the larger the source block size is, the larger the latency is (caused by waiting for the FEC repair packet). The application should choose carefuly the FEC block size in order to maximize the FEC strength while keeping an acceptable latency at the receiver waiting for the FEC repair packets.

11. Offer/Answer considerations

None.

12. Security Considerations

12.1. Problem Statement

A content delivery system is potentially subject to many attacks. Some of them target the network (e.g., to compromise the routing infrastructure, by compromising the congestion control component), others target the Content Delivery Protocol (CDP) (e.g., to compromise its normal behavior), and finally some attacks target the

content itself. Since this document focuses on various FEC schemes, this section only discusses the additional threats that their use within the FECFRAME framework can create to an arbitrary CDP.

More specifically, these attacks may have several goals:

- o those that are meant to give access to a confidential content (e.g., in case of a non-free content),
- o those that try to corrupt the ADU Flows being transmitted (e.g., to prevent a receiver from using it),
- o and those that try to compromise the receiver's behavior (e.g., by making the decoding of an object computationally expensive).

These attacks can be launched either against the data flow itself (e.g., by sending forged FEC Source/Repair Packets) or against the FEC parameters that are sent either in-band (e.g., in the Repair FEC Payload ID) or out-of-band (e.g., in a session description).

12.2. Attacks Against the Data Flow

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

12.2.1. Access to Confidential Contents

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

12.2.2. Content Corruption

Protection against corruptions (e.g., after sending forged FEC Source/Repair Packets) is achieved by means of a content integrity verification/sender authentication scheme. This service is usually provided at the packet level. In this case, after removing all forged packets, the ADU Flow may be sometimes recovered. Several techniques can provide this source authentication/content integrity service:

- o at the application level, the Secure Real-time Transport Protocol (SRTP) [RFC3711] provides several solutions to authenticate the source and check the integrity of RTP and RTCP messages, among

other services. For instance, associated to the Timed Efficient Stream Loss-Tolerant Authentication (TESLA) [RFC4383], SRTP is an attractive solution that is robust to losses, provides a true authentication/integrity service, and does not create any prohibitive processing load or transmission overhead. Yet, checking a packet requires a small delay (a second or more) after its reception with TESLA. Other building blocks can be used within SRTP to provide authentication/content integrity services.

- o at the Network Layer, IPSec/ESP offers (among other services) an integrity verification mechanism that can be used to provide authentication/content integrity services.

Techniques relying on public key cryptography (e.g., digital signatures) require that public keys be securely associated to the entities. This can be achieved by a Public Key Infrastructure (PKI), or by a PGP Web of Trust, or by pre-distributing the public keys of each group member.

Techniques relying on symmetric key cryptography require that a secret key be shared by all group members. This can be achieved by means of a group key management protocol, or simply by pre-distributing the secret key (but this manual solution has many limitations).

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

12.3. Attacks Against the FEC Parameters

Let us now consider attacks against the FEC parameters included in the FFCI that are usually sent out-of-band (e.g., in a session description). Attacks on these FEC parameters can prevent the decoding of the associated object. For instance modifying the m field (when applicable) will lead a receiver to consider a different code. Modifying the E parameter will lead a receiver to consider bad Repair Symbols for a received FEC Repair Packet.

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

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

a wrong block.

It is therefore RECOMMENDED that security measures be taken to guarantee the Explicit Repair FEC Payload ID integrity. To that purpose, one of the packet-level source authentication/content integrity techniques of Section 12.2.2 can be used.

13. IANA Considerations

New media subtypes are subject to IANA registration. For the registration of the payload formats and their parameters introduced in this document, refer to Section 7.

14. Acknowledgments

Some parts of this document are borrowed from the following documents: [RFC5109], [I-D.ietf-fecframe-1d2d-parity-scheme], [I-D.roca-fecframe-rs], [I-D.ietf-avt-reedsolomon]. The author would like to thank the editors of these documents. The authors would also like to thank the members of the FP7 3DPresence project consortium for their contribution to this draft writing.

15. References

15.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC4288] Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", BCP 13, RFC 4288, December 2005.

[RFC3555] Casner, S. and P. Hoschka, "MIME Type Registration of RTP Payload Formats", RFC 3555, July 2003.

[RFC4756] Li, A., "Forward Error Correction Grouping Semantics in Session Description Protocol", RFC 4756, November 2006.

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

[I-D.ietf-fecframe-framework] Watson, M., "Forward Error Correction (FEC) Framework", draft-ietf-fecframe-framework-09 (work in progress), July 2010.

15.2. Informative References

[I-D.ietf-fecframe-1d2d-parity-scheme] Begen, A., "RTP Payload Format for Non-Interleaved and Interleaved Parity FEC", draft-ietf-fecframe-1d2d-parity-scheme-01 (work in progress), May 2009.

[RFC5109] Li, A., "RTP Payload Format for Generic Forward Error Correction", RFC 5109, December 2007.

[I-D.ietf-avt-reedsolomon] Rosenberg, J. and H. Schulzrinne, "An RTP Payload Format for Reed Solomon Codes", May 1999.

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

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

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

[RFC4383] Baugher, M. and E. Carrara, "The Use of Timed Efficient Stream Loss-Tolerant Authentication (TESLA) in the Secure Real- time Transport Protocol (SRTP)", RFC 4383, February 2006.

[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.

[I-D.roca-fecframe-rs] Roca, V., Cunche, M., Lacan, J., Bouabdallah, A., and K. Matsuzono, "Reed-Solomon Forward Error Correction (FEC)

Schemes for FECFRAME", draft-roca-fecframe-rs-03 (work in progress), July 2010.

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

Authors' Addresses

Sarit Galanos
RADVISION
24 Raul Wallenberg St.
Tel Aviv 69719
Israel

Email: sarit@radvision.com

Orly Peck
RADVISION
24 Raul Wallenberg St.
Tel Aviv 69719
Israel

Email: orlyp@radvision.com

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

Email: vincent.roca@inria.fr