**3GPP TSG-SA WG4 Meeting #127-bis-eS4-240600**

**Online, 8th – 12th Apr 2024**

**Source: Huawei, HiSilicon**

**Title: [5G\_RTP\_Ph2] FEC usage in RTP according to IETF**

**Document for: Discussion**

**Agenda Item: 10.8**

*Abstract: This contribution provides a solution for key issue 3 with summary of different AL-FEC schemes available from IETF.*

# Introduction

The new SA4 Rel-19 study item on “5G Real-time Transport Protocol Configurations, Phase 2” (5G\_RTP\_PH2) has been approved in [SP-240065](https://www.3gpp.org/ftp/TSG_SA/TSG_SA/TSGS_103_Maastricht_2024-03/Docs/SP-240065.zip). The work item lists twelve distinct key issues to improve 5G RTP as defined in TS 26.522, of which key issues number 3 and 4 relate to Application layer forward error correction (FEC):

***Enhancements for application-layer FEC support.*** *According to clause 5.7.4 of TR 26.926 [6], commercial XR split rendering and cloud gaming services use Application Layer Forward Error Correction (FEC). This clause also introduces several RTP based FEC schemes defined by IETF primarily to be used in WebRTC. It is worthwhile to study if any of these FEC schemes can be added to 3GPP specifications, for example to support split rendering.*

***Application-layer FEC awareness for PDU Set handling.*** *Application-layer FEC is also in the scope of SA2 XRM SI phase 2 [3]. In the context of cross-layer design, it is important to understand how to expose the application-layer FEC information to the communication network (UPF, RAN) to enable intelligent resource allocation. Also, there are intricate interactions between the application and the network. In particular, network dropping extra PDUs in a PDU Set encoded with application-layer FEC, if any, may send a false signal to the application on the packet loss rate and the congestion level in the network, and lead to undesired adaptation from the application such as increased redundancy ratio and reduced sending rate. SA4 needs to understand the interactions between the application and the network in the case of application-layer FEC and intentional packet dropping by the network and the impact on the media performance.*

This discussion paper provides as a solution for key issue number 3 as it introduces and summarizes the different AL-FEC schemes available from IETF. =The solution can be used as a basis for working on key issue number 4.

# Discussion

## General

Forward Error Correction, FEC, provides robust transmission by introducing redundant packets into the traffic. At the receiver side, a certain degree of packet loss is still acceptable due to the presence of repair packets. In RTP/UDP based transmission packet loss may happen, and AL-FEC is a useful technology to mitigate the negative effects of packet loss.

## FEC solutions in IETF

IETF defines FEC schemes for RTP in [1-4]. In addition, a generic framework for FEC is defined in RFC6363 [7], this framework enables different FEC solutions to follow similar approach for embedding FEC specific information in both source and repair packets.

## ULP FEC (RFC 5109)

RFC 5109 [1] describes ULP unequal level protection using XOR based on packets (parity codes). For example when packets a, b, c and d are protected:

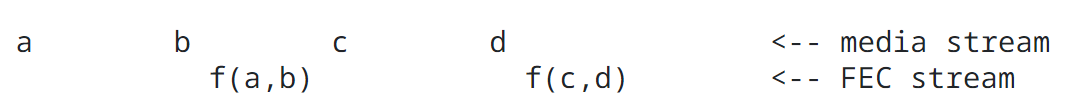


Figure 1 XoR Based FEC

In this example the FEC stream packets are XoR combinations of two media stream packets. At the receiver if b is lost it can be retained by XoRing the repair packet f(a,b) with a, e.g:

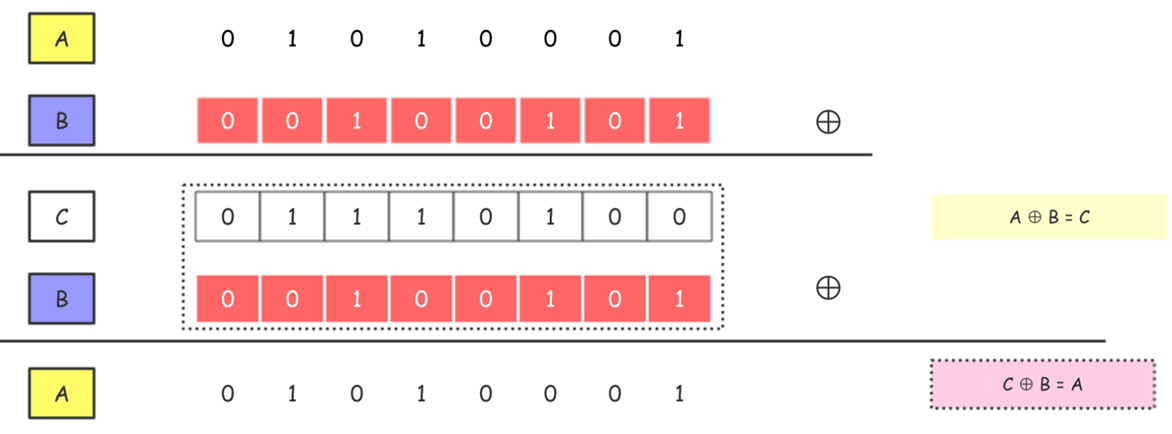


Figure 2 ULP FEC scheme using XoR

More layers of protection can be added, where higher layers can protect more packets, and earlier parts of a packet may be protected stronger than the later parts.

NOTE: In case of congestion, it is likely that multiple subsequent packets get lost and how ULP FEC would overcome this is unclear.

NOTE: RFC 5109 recommends monitoring packet loss and adapting the transmission rate accordingly.

NOTE: In RFC 5109 FEC packets can be added in a separate RTP stream or in the same RTP stream.

NOTE: Guidelines are provided for unequal error protection, for example to apply stronger protection to the beginning of the packet as the end of the packet.

NOTE: The solution can work with encryption (e.g. SRTP), either applying FEC before or after AES encryption, applying FEC after the encryption looks more logical.

NOTE: The RTP payload is constructed by adding FEC header in the payload

## Flex FEC (RFC 8627)

In addition to ULP FEC, Flex FEC introduces two dimensional FEC by interleaving packets and having both parity packets on rows/and/or columns in RFC 8627 [2]. This provides stronger protection to mixed losses, but in some cases of packet loss it may still be impossible to recover the original frame.

For example, the repair packets R\_1 … R\_L or C\_1 … may be generated as follows:

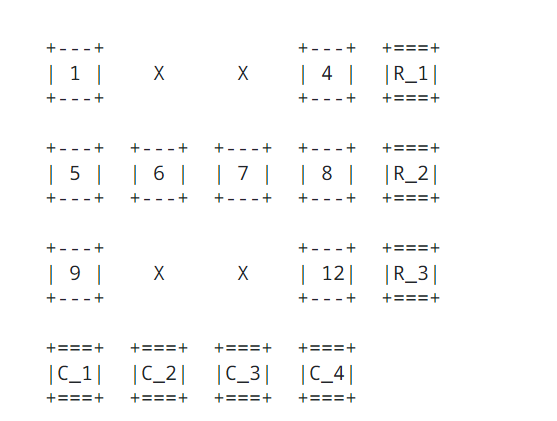


Figure 3 2D Parity check still failing

Flex FEC enables one dimensional or two dimensional parity check based codes and also a flexible mask to enable selection of packets to be protected. Similar to ULP FEC, FEC data may be carried in same RTP stream or a different RTP stream. Also, the RTP Header is used to signal FEC on a RTP packet level, similarly as for ULP FEC. Currently 3 versions of the header are available, one for fixed row column 2D parity, one using a mask and a last option only using retransmissions of source packets. Similar as for ULP FEC receivers that do not understand the FEC can ignore the packet.

Similar points as mentioned in the previous section apply.

Flex FEC is adopted in WebRTC according to [8].

## Raptor and RaptorQ FEC (RFC 6681)

Raptor(Q) FEC is an improved version of the original fountain code. An encoder can generate any number of coded symbols from the source symbols of a data source block on demand. The decoder can recover the source block from any set of encoded symbols that are only slightly larger than the number of source symbols. RaptorQ is an improved Raptor algorithm.

RFC 6681 describes detailed forward error correction policies for Raptor and RaptorQ [3].

This document describes only the FEC schemes (6 in total) of Raptor/RaptorQ and does not describe the mathematical principles of the algorithm. For the mathematical principles, refer to the following IETF reference [5,6].

Similarities and Differences between Raptor and RaptorQ Algorithms are not that big, the coding process of RaptorQ code is almost the same as that of Raptor code.

The traditional Raptor code runs in Galois GF(2) domain, while the new RaptorQ algorithm runs in GF(256) domain instead of GF(2) domain. Operating on a larger finite field RaptorQ can encode 56403 data packets into one source block, while Raptor code can operate only on 8192 data packets, and RaptorQ can generate up to 16777216 encoding symbols, 256 times that of Raptor code;

When the bandwidth and data redundancy are the same, the percentage of users protected by RaptorQ is higher. That is, when the number of redundant data packets received by RaptorQ is the same, the probability of successful decoding is higher.

FEC protection operation mechanism (equivalent to the execution process) can be outlines as follows:

The sender determines, according to the FEC Framework Configuration, a source data packet (a source block) that needs to be protected. The sender arranges the source data packets into a group of source symbols, and each source symbol has a same size. The sender applies the Raptor and RaptorQ protection operations to the source symbols to generate the required number of repair symbols;

The sender packs the repair symbol and sends the repair packet and the source packet to the receiver. According to the requirements of the FEC framework, the sender must transmit source and repair packets in different source and repair streams, or in different RTP streams if the real-time transport protocol (RTP) is used to transmit packets for repair;

At the receiving end, if all source packets are successfully received, FEC recovery is not required and repair packets are dropped. However, if there are lost source packets, repair packets can be used to recover the lost information;

The schemes in RFC 6681 enable generation of source and repair packets. For arbitrary flows, the source packets need to be generated according a specific procedure defined in clause 5 of RFC 6681 which amongst other operations includes the source FEC Payload ID at the end of the packet. This ID includes the source block number and encoding symbol id (ESI).

For a single sequenced flow, the source packets may not need not be modified as it is already possible to derive the application of repair packets from the original stream structure (it is unclear when this would apply, or if the arbitrary flow is the default).

Raptor/RaptorQ FEC is applicable to real-time audio and video conferences, cloud game streaming media, video on demand, and streaming media.

## Reed Solomon FEC for FEC FRAME (RFC 6865)

Reed-Solomon: RS code FEC over Galois finite field GF (2^^m, 2<m<16) may be used to protect Application Data Units (ADUs) (i.e. packets) according RFC 6865 [4]. Reed-Solomon FEC codes are maximum distance separable (MDS) codes. Assuming *k* source symbols, *n* is the number of encoded symbols after encoding, and *n-k* is the number of repair symbols. Theoretically, if *n-k* encoded symbols are lost, the receiver can recover the source symbols.

The RS algorithm limits both the source block size and the number of encoding symbols that can be used. RS FEC is more complex than Raptor FEC.

Similar as for the arbitrary stream in previous section, the source FEC payload id is appended at the end of a source block, and the FEC repair payload id is inserted at the beginning of a repair block. The Reed Solomon FEC enables generating source and repair packets in a single stream.

The scheme from RFC 6865 has some limitations due to the use of this FEC scheme.

- Each ADUI must have a source symbol. Therefore, each ADU must have a source symbol.

- Each FEC repair packet must have a repair symbol.

- Each ADU block must define at least one source block.

ADU block structure limitations are the following:

- Limitation: m in G (2^m) affects a size of a source block and a quantity of encoded symbols. The finite field size q - 1 = 2^m - 1, and this q-1 value is also the theoretical maximum number of encoding symbols that can be generated for the source block.

- The maximum number of ADUs in an ADU block cannot exceed a certain threshold, as it directly affects the decoding latency. The larger the ADU block size, the longer the decoder may have to wait until it receives a sufficient number of encoded symbols to successfully decode, and hence the greater the decoding delay. When the target use case is known, these real-time constraints result in an upper bound on the ADU block size.

## Categorization

Table 1 categorizes available standardized FEC schemes from IETF based on different criteria. In addition, for RFC 6681 and 6865, the source data is modified which may affect backward compatibility and the application of encryption (i.e. if it happens before or after FEC). Performance is considered good if there is general repair capability for any loss without introducing too much latency. Performance is considered medium if there is general repair capability for any loss but introducing some latency and complexity. Performance is poor when reliability is still not guaranteed.

Table 1 categorization of AL-FEC schemes for RTP in IETF

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Name | RFC | Type | format of source packets unchanged/  Backward compatible | Resilience to Arbitrary packet loss | Flexible redundancy | Overhead  (bytes) | Performance (repair capability) |
| ULP FEC | 5109 | Parity/XoR | Yes | NO | YES | High | Low |
| FlexFec | 8627 | Parity/XoR | Yes | NO | YES | High | Low |
| Raptor/  RaptorQ | 6681 | Fountain/LT | Yes/No | YES | YES | Medium | Good |
| Reed  Solomon | 6865 | polynomial | NO | YES | Limited | Medium | Good |

As TS 26.522 only uses RTP Headers that are not in scope of the FEC scheme, it is envisioned that the use of 5G RTP headers from TS 26.522 and FEC from Table 1 are interoperable especially when the 2 byte header is used, as packets would be able to include both the 5G RTP and FEC Headers.

# Proposal

It is recommended to review and discuss this solution and then document this solution to key issue number 3 based on the text provided in clause 2 in the technical report for FS\_5G\_RTP\_ph2.

# References

[1] IETF RFC 5109: “RTP Payload Format for Generic Forward Error Correction (ULP FEC)：Uneven Level Protection, different redundancies for different packets with different importance.”

[2] IETF RFC 8627: “RTP Payload Format for Flexible Forward Error Correction (Flex FEC): flexible FEC.”

[3] IETF RFC 6681: “Raptor Forward Error Correction (FEC) Schemes for FECFRAME：FEC scheme based on the Raptor.”

[4] IETF RFC 6865: “Simple Reed-Solomon Forward Error Correction (FEC) Scheme for FECFRAME：FEC scheme based on Reed-Solomon. “

[5] IETF RFC 5053: “Raptor Forward Error Correction Scheme for Object Delivery”

[6] IETF RFC 6330: “RaptorQ Forward Error Correction Scheme for Object Delivery”

[7] IETF RFC 6363: “Forward Error Correction (FEC) Framework”

[8] IETF RFC 8854: “WebRTC Forward Error Correction Requirements”