FECFRAME version 2

Adding convolutional FEC codes support to the FEC Framework

draft-roca-tsvwg-fecframev2-02

Vincent Roca, Inria, France

Ali Begen, Networked Media, Turkey

November 2016, IETF97, Seoul

Note well

- **we, authors of -02 version, didn’t try to patent** any of the material included in this presentation/I-D
- **we, authors of -02 version, are not reasonably aware** of patents on the subject that may be applied for by our employer
- if you believe some aspects may infringe IPR you are aware of, then fill in an IPR disclosure and please, let us know

FECFRAME / FECFRAMEv2 reminder

- a follow-up of [RFC 6363] describing FECFRAME
  - **RFC 6363, M. Watson, A. Begen, V. Roca, October 2011**
- a shim layer for **robust** and **scalable** distribution of **real-time flows**
  - **already part of 3GPP (e)MBMS standards**
  - **we start to have deployment experience**
- FECFRAME relies on block FEC codes…
- …block codes **add latency to everybody, always**
- this issue is **solved** with convolutional FEC codes
  - **good reception conditions: near zero latency ☺**
  - **bad reception conditions latency: still significantly inferior**
- v2 **adds** convolutional code support
  - in a fully backward compatible way

Differences WRT last July's I-D (01 version)

- added an Implementation Status Section
  - as recommended in RFC 7942
    - **leverages on a FECFRAME implementation (Vincent) being commercialized (Expway), for which interop. tests have been conducted**
    - **FECFRAMEv2 implementation under progress (Vincent)**

- added Appendix B that explains differences WRT RFC 6363

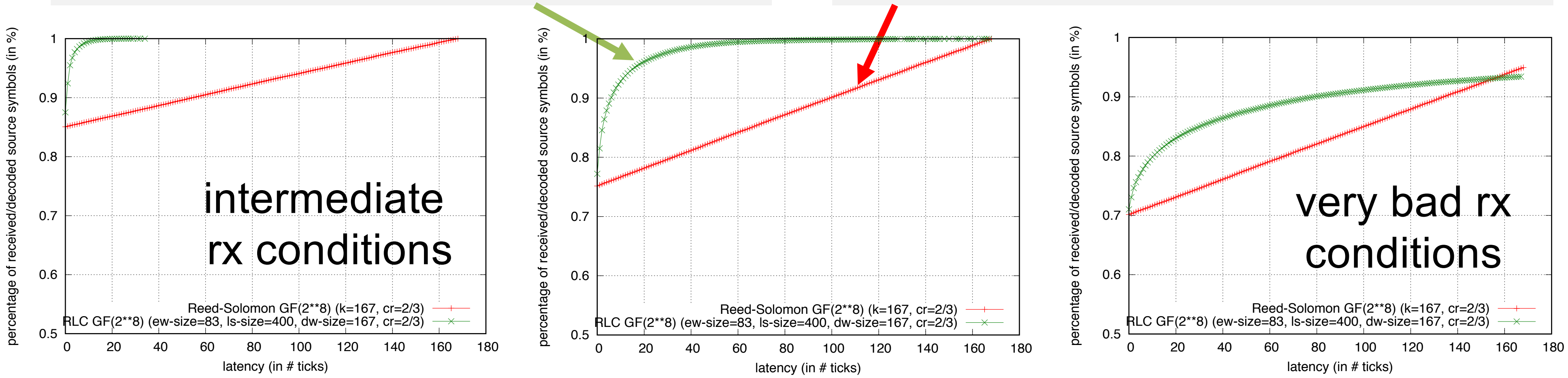
- fixed a few minor things

Differences WRT last July's I-D… (2)

- we made progress in terms of block vs convolutional codes evaluation
  - block FEC codes are totally sub-optimal for real-time flows
  - true with small or larger block/encoding window sizes
  - motivates the need for FECFRAME v2

latency CDF with conv. codes

latency CDF with block R-S codes



(a) R-S vs. RLC CDF when $loss = 15\%$

(b) R-S vs. RLC CDF when $loss = 25\%$

(c) R-S vs. RLC CDF when $loss = 30\%$

V. Roca, B. Teibi, C. Burdinat, T. Tran, C. Thienot , *"Block or Convolutional AL-FEC Codes? A Performance Comparison for Robust Low-Latency Communications"*, *https://hal.inria.fr/hal-01395937*, November 2016.

Q: version 2 or just an update of RFC 6363?

- background
  - version 2 does not remove any capability to FECFRAME
    - **only adds the support of convolutional FEC Schemes**
  - a receiver decides to join or not after processing the SDP
    - **FEC Encoding ID enables the receiver to determine whether it supports the convolutional FEC Scheme**
    - **same mechanism for any unsupported FEC Scheme**
  - no notion of version in FECFRAME anyway
    - **there's no header, only FEC Scheme signaling header/trailer**
  - however, from an implementation viewpoint, there are clear differences
    - **version 2 immediately indicates the capabilities**

Next steps

- we do not expect major changes in future revisions

- TODO 1: finish FECFRAME v2 implementation
    - to be sure we didn't miss anything
    - sender already done, receiver will be okay for IETF98

- TODO 2: propose RLC convolutional FEC Scheme
  - all the convolutional FEC code complexity is here!
    - specify all code details
    - specify all signaling aspects
    - identified by a IANA registered FEC Encoding ID
  - default convolutional code we use in our implementation