About FEC Scheme, signaling and protocol in NWCRG

Vincent Roca, Marie-José Montpetit
IETF102, Montreal, July 19th, 2018
Why this discussion?

- A number of I-Ds of interest:
  - Tetrys (draft-detchart-nwcrg-tetrys-04)
    - a full solution: protocol, signaling/headers, sliding window FEC
    - detailed but un-finished
  - RLNC (draft-heide-nwcrg-rlnce-00)
    - a set of coding solutions, including protocol considerations (but not specification), and signaling/header
    - very detailed in parts, evasive on other aspects
    - more work needed
Why this discussion? (2)

- RLC for FECFRAME (TSVWG, passed WGLC)
  - two FEC Schemes for a well defined protocol
  - full specification, with all required details

- RLC for QUIC (work in progress)
  - leverages on the other one for code specification
  - specifies a different signaling (for QUIC headers) and mechanisms (e.g., application data to symbol mapping)

- and potentially a BATS I-D in near future?
Moving forward?

• Divide and conquer
  - keep code specification, signaling, and protocol aspects separate

  reusable across several contexts
  mostly specific to a protocol
  nwcrg protocol remains TBD
Moving forward? (2)

• Is the FEC Scheme approach appropriate?
  i.e., specify code internals + signaling in order to have a working solution for a **specific target protocol**
  … but we could also limit ourselves to the **code internals + signaling requirements** (without considering a specific format)

what has been done so far

```
+------------------+
| Stream ID (1)    |
| Offset of First Source Symbol in EW (1) |
| Length (1)       |
| Repair Key       |
| DT | NSS (# src symb in ew) |
|------------------|
| Stream Data      |
```

a better approach for nwcrg?

RLC needs:
• coding window description, e.g., (if no gap) 1st symbol id + number symbols
• repair key (necessarily a 16-bit value)
• density threshold (necessarily in {0 ; 15})
Moving forward? (3)

• Investigate key questions
  ▪ is a **universal signaling and header format** feasible? Probably not but if we just focus on code requirements, perhaps 😊
  ▪ what does **inter-flow coding** imply? What type of synchro does it require? Should it be an option for more complex use-cases or something that’s worth to support by default?
  ▪ investigate **code parameter derivation**. E.g., RLC for FECFRAME tries to elaborate on this question, but there’s probably more to say
Moving forward? (4)

- design a protocol for sliding window codes, e.g., “à la Tetrys” (with a different name). Should be (mostly) code agnostic.

- other? <add your own topic here>