Coding for QUIC (update)

draft-swett-nwcrg-coding-for-quic-03

Vincent Roca, François Michel, Marie-José Montpetit, Ian Swett

IETF105, Montreal, July 26th, 2019

"Summary of last episode @ ietf 104"

Two approaches:

- an **"Intra STREAM FEC"** (draft-swett-nwcrg-coding-for-quic-02)
  - Idea: FEC protection within a single STREAM of a QUIC session
    - ☺ protect only the flows that need it within a QUIC session
    - ☺ absolutely no change to source QUIC packets
    - ☹ no notion of boundaries (it's a stream abstraction) making it incompatible with DATAGRAM FRAMES
    - ☹ a source symbol may straddle several QUIC packets, hence inefficiencies

"Summary of last episode @ ietf 104" (2)

- a **"Cross packet FRAMES FEC"** proposal (François M.)
  - Idea: FEC protection of all/a subset of the FRAMES of a QUIC packet
    - ☺ compatible with any FRAME type, including future DATAGRAM FRAMES
    - ☺ preserves packet boundaries
    - ☺ source symbols never straddle QUIC packets
    - ☹ a slight change of source packets (adding FPI signaling)
    - ☹ more difficult to protect a subset of a QUIC packet (signaling cost)

What we decided: "cross packet FRAMES"

- being FRAME agnostic and compatible with DATAGRAM FRAMES is a must
- we totally changed the approach
  - draft **version -03 implements the "Cross packet FRAMES FEC" paradigm** plus additional features

What we decided: "cross packet FRAMES" (2)

- many things didn't change
  - (non exhaustive list)
  - FEC encoding is applied **before** any QUIC encryption
    - middleboxes cannot interfere (no re-encoding)
  - define a **generic** framework to use FEC protection in QUIC -- this I-D
    - leave code specificities (e.g., block or sliding window code) in additional FEC Scheme I-Ds
  - use dedicated **RECOVERED** FRAMES to signal **"loss but recovery"** of a QUIC packet
    - do not interfere with congestion control but avoid retransmission

QUIC packet to source symbols mapping (1)

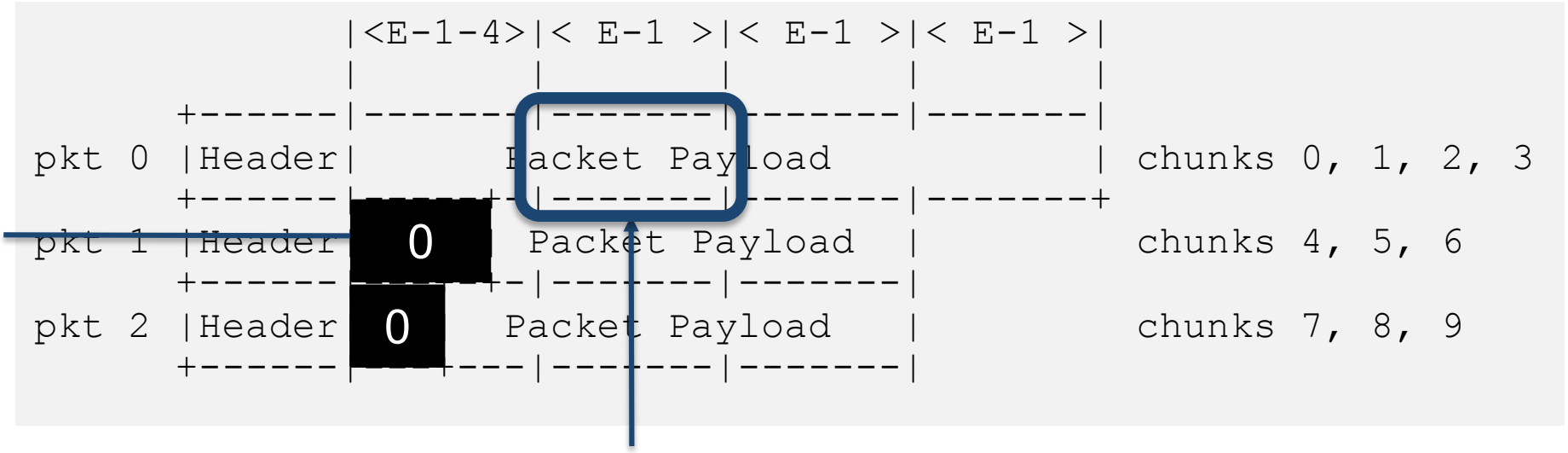
- requirements:
  1. packets are of variable size, symbols are of fixed size (E), so **we need a mapping**
  2. the **symbol size, E**, needs to be initialized wisely:
     - "small E" is fine when the QUIC packet sizes is largely variable (very small + a few very large packets), but has a cost
     - "large E" makes it simple (everything fits in a single symbol) but is suboptimal with a majority of small packets, and it may require to fragment the QUIC/UDP at IP level
     - E could be adjusted depending on the target use-case (if known)
  3. anticipate the potential need to **avoid exceeding the PMTU** (we add extra FEC related signaling)
     - choose E small enough

QUIC packet to source symbols mapping (2)

- **Step 1:** from QUIC packet payload to **chunks**
  - packet payload data is of **variable** size but source symbols are **fixed** size (E bytes)
  - use QUIC **zero padding FRAMES** **before*** the payload to align payload size

* to avoid problems with STREAM frames that do not encode their length

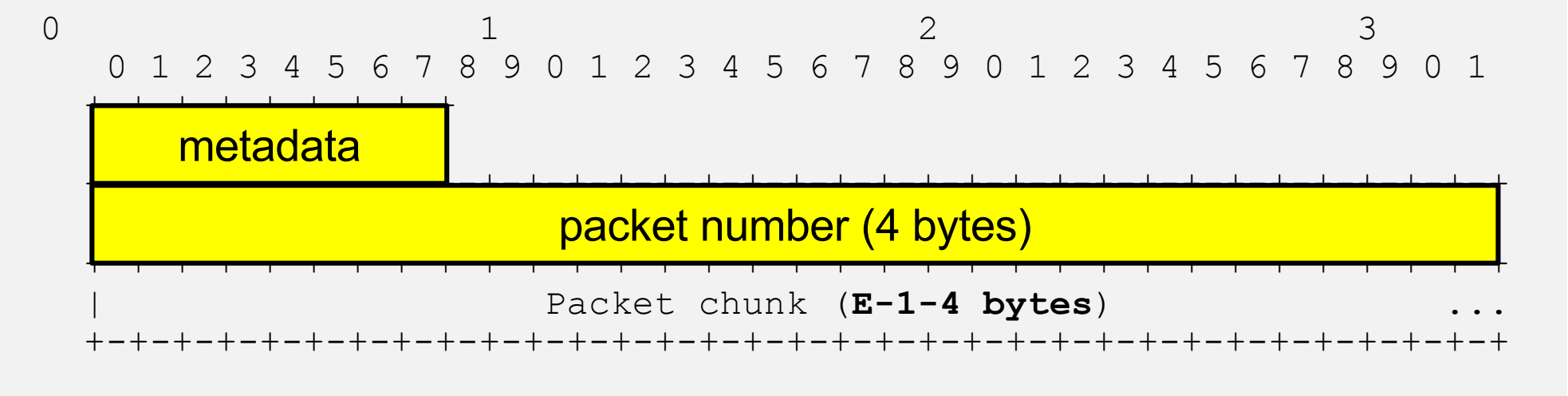
(1) pad with initial padding FRAMES to align packet size



(2) segment padded payloads into E-1 byte long chunks
(E-5 for the 1st one to leave room for additional info)

QUIC packet to source symbols mapping (3)

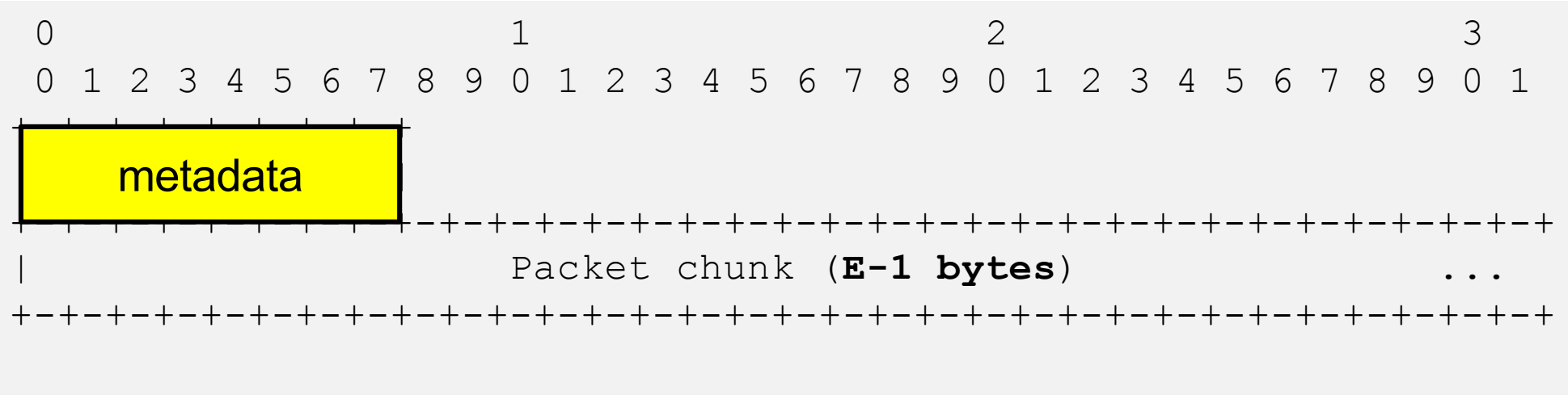
- **Step 2:** from a packet chunks to **source symbols**
  - **1st chunk:** prepend a long signaling (5 bytes)



<meta-data + packet number + 1st chunk> constitute the source symbol

QUIC packet to source symbols mapping (4)

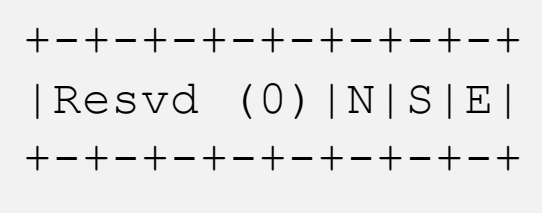
- **Step 2:** from a packet chunks to **source symbols**
  - 1st chunk: prepend a long signaling (5 bytes)
  - **following chunks:** prepend a short header (1 byte)



<meta-data + chunk> constitute the source symbol

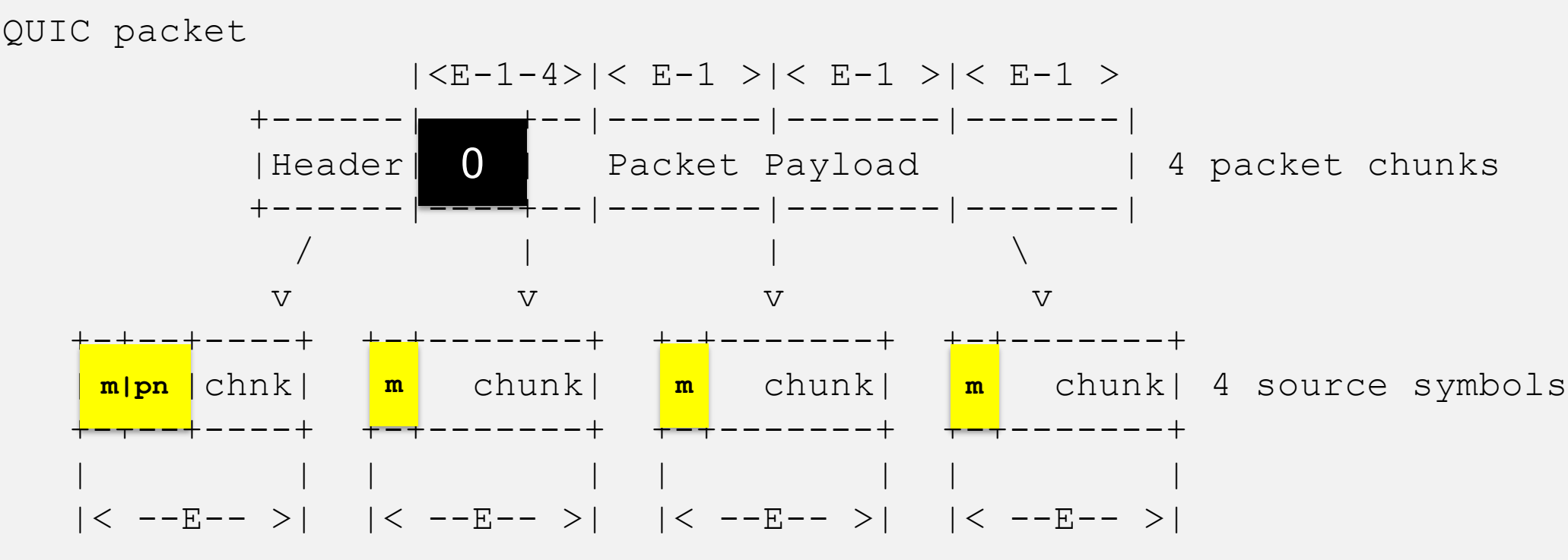
QUIC packet to source symbols mapping (5)

- **Step 2:** from a packet chunks to **source symbols**
  - meta-data (all chunks):
    - N: packet Number is there
    - S (start): first chunk of a QUIC packet
    - E (end): last chunk of a QUIC packet



  - Packet Number (4 bytes) (first chunk only, optional):
    - when "N" (Packet Number) field is 1 in the meta-data
    - required at a receiver to determine the QUIC packet number associated after decoding all the symbols of the lost packet

The big picture: from QUIC packet to source symbols (1)



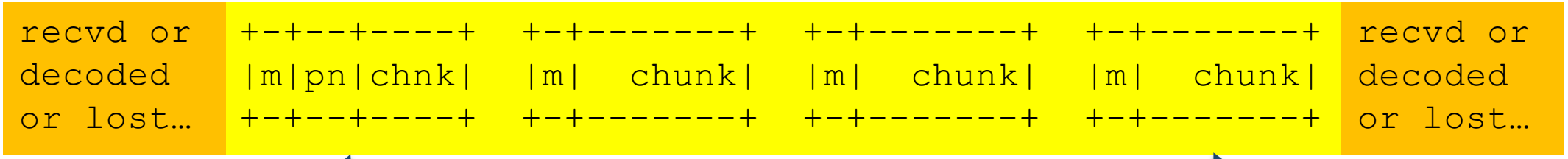
The big picture: from QUIC packet to source symbols (2)

- add a **FEC_SRC_FPI frame** to the original QUIC packet, then send it
  - no other change to the original QUIC packet (no padding, no meta-data, we do not transmit source symbols per se)
  - the FEC_SRC_FPI frame is **ignored** by a QUIC receiver that does not support FEC

- transmit repair symbols in dedicated **REPAIR frames**
  - it includes repair FPI signaling information
  - one or more REPAIR frames can be packed in the same QUIC packet (e.g., if E is small WRT the PMTU), for reduced overhead
  - a REPAIR frame is **ignored** by a QUIC receiver that does not support FEC

Yes, we need this extra meta-data / packet number

decoding successful for those 4 source symbols



S=1: we know the chunk contains one or more padding FRAMES plus original FRAMES

N=1: we can also recover the corresponding QUIC Packet Number

E=1: we know this chunk is the last one of packet

we have all chunks (S=1 | middle | middle | E=1) of the QUIC packet. Done ☺

Open points

- considering a subset of the QUIC packet frames
  - because protecting these FRAMES is not crucial
    - e.g., ACK FRAMES
    - e.g., FRAMES of non delay sensitive streams
    - …
  - how to inform both ends?

- choice of E
  - we define general rules, but details is left to the developer/use-case/data flow features

To Be Done

- update our companion RLC for QUIC I-D…
  - https://datatracker.ietf.org/doc/draft-roca-nwcrg-rlc-fec-scheme-for-quic/
- finish the SWIF codec…
- … and continue tests