Localized Optimizations over Path Segments

IETF 106, NWCRG meeting 2019-11-21

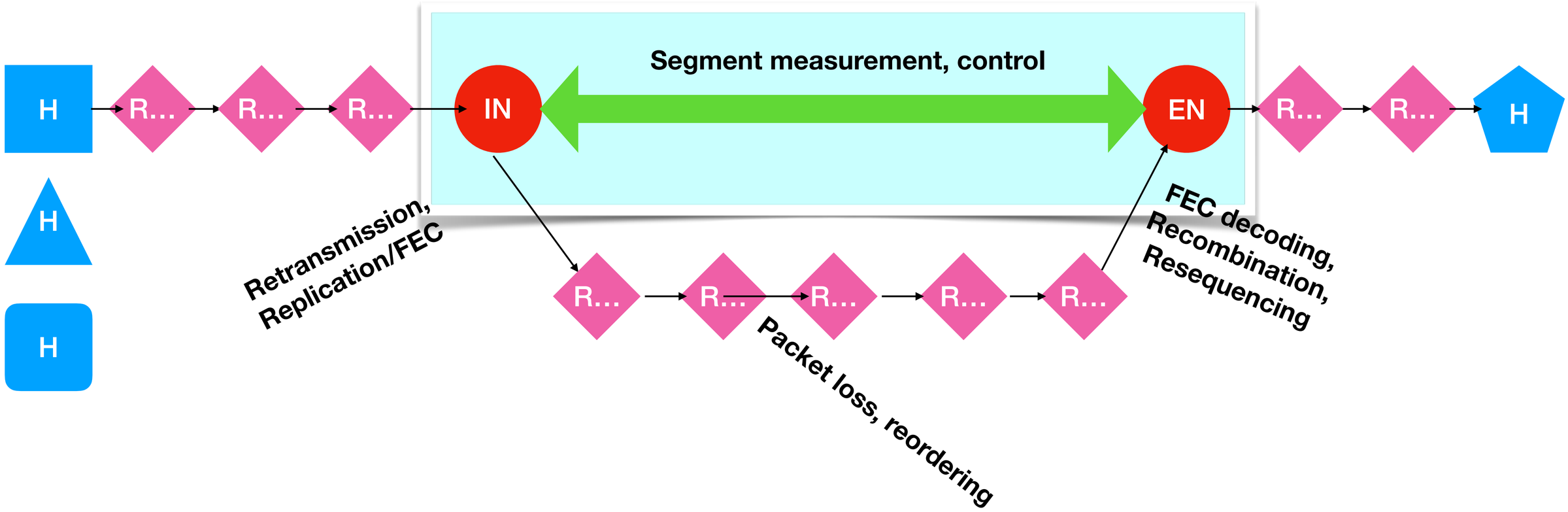
Packet loss

Packet loss

Packet loss

Tail loss!

LOOPS Opportunity



Recover Packets Locally

Reduce end-to-end packet loss

Recover locally, **where needed**, with low latency

In the network

Host participation not required

Don’t look

Don’t touch

Works with any kind of IP packets

How to recover?

- **Retransmission**
  - Reverse information needed: ACK/NACK ← Piggyback (Tunnel), separate Packets
  - Forward information: sequence numbering (if needed) ← Tunnel
- **Forward Error Correction** (redundancy)
  - Can use dynamic selection of block size/rate: measurement input
  - “Retransmission” also possible by adding FEC

- Aim for low setup overhead
- Keep most setup out of protocol (“controller model”)

How not to blow up the Internet

- Concealing losses removes important congestion signal
  - End-hosts would ramp up to higher rates, increase congestion

- Need **congestion feedback**
  - Preferred: ECN
  - Fallback: Selective dropping (selective recovery, actually)
- Host transport protocol improvements will help improve LOOPS performance, but are not prerequisite to obtaining benefit

LOOPS vs. transport protocols

- LOOPS is separate from the end-to-end transport protocol
  - Hands-off approach: don’t meddle
  - Do not assume the end-to-end protocol is out to help us, either
  - No direct control over sending rate (cc feedback only)
- LOOPS should not just be a classical transport protocol
  - Residual loss is OK
  - More choices: Tight interaction with the path segment being optimized

Where “transport protocol” intuition may not even work

- Relatively controlled/managed environment;
  setup mechanism assumed (can supply parameters so not everything needs to be high dynamic range)
- No full reliability intended; remaining gaps are OK (and at some point must leave the focus of attention)
  - Setup might set upper bound for overhead volume (e.g., 10 %), can well be “risky” in the way that this is used
- Tunnels usually have packets in flight (possibly a large number); tail processing rarely invoked (but may still be desired); don’t need overly conservative RTO

Documents out there

- Use cases and problem statement: “LOOPS (Localized Optimizations on Path Segments) Problem Statement and Opportunities for Network-Assisted Performance Enhancement” <draft-li-tsvwg-loops-problem-opportunities>
- Protocol: “LOOPS Generic Information Set” <draft-welzl-loops-gen-info>
- One of the Encapsulations: “Embedding LOOPS in Geneve” <draft-bormann-loops-geneve-binding-00.txt>

- Charter proposal for a LOOPS WG <https://github.com/loops-wg/charter>
- LOOPS mailing list loops@ietf.org

Related work (see IETF105 BOF)

- Encapsulations: Many (e.g., NVO3 for Geneve; GUE; GRE?)
- **FEC: NWCRG for e.g., sliding window FEC, encapsulation techniques**
- Tunnel congestion Feedback (TSVWG)
- Also: measurement work, IOAM;
  knowledge about behavior of transport protocols (TCP, QUIC)
  adaptation layer retransmission work (6Lo Fragment Recovery)

Sliding Window FEC

- Sliding windows fit quite well to LOOPS application
  (Can also use traditional block formats)

- Various drafts for FEC scheme and specific embeddings in NWCRG and TSVWG, e.g.,

  - "Sliding Window Random Linear Code (RLC) Forward Erasure Correction (FEC) Schemes for FECFRAME" <draft-ietf-tsvwg-rlc-fec-scheme-16.txt>

  - "Forward Error Correction (FEC) Framework Extension to Sliding Window Codes" <draft-ietf-tsvwg-fecframe-ext-08.txt>

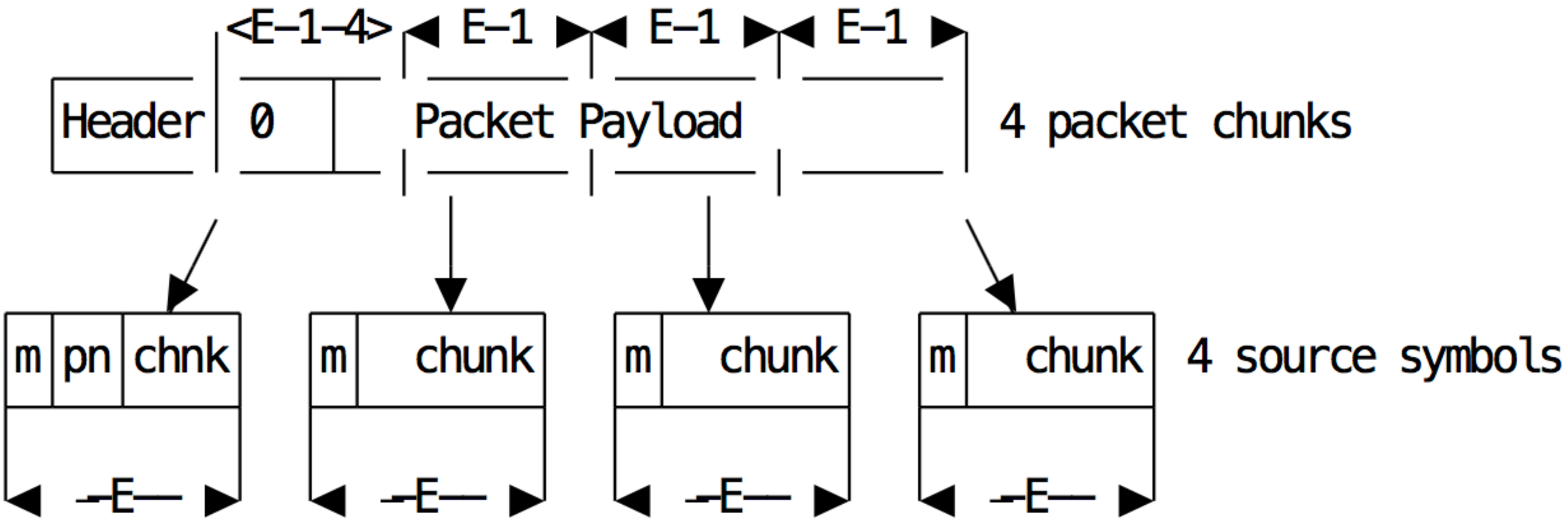
LOOPS FEC approach

- Support multiple classes of FEC schemes, e.g.:
  - Very simple parity (as in SMPTE 2022)
  - Fountain Codes (e.g., RaptorQ)
  - Sliding Window schemes (e.g., RLC)
- Assume all codes are systematic (needed for transparent mode)
  - Except for transparent mode, augment payload packets by FEC indices
- Possibly add special handling for larger-than-tunnel-MTU packets
- Add repair packets with repair information

LOOPS FEC approach

- LOOPS can provide:
  - forward: place for FEC indices, separate format for repair packets
  - reverse: Block 2 acknowledgements, or aggregate loss rate feedback
- Assumption: large size variance of payload packets (avg 400..700 B)
  - Payload packets are divided up before being funneled into FEC
  - Not necessarily related to the way they are sent forward
  - Any piggybacking for repair segments?
    Recombining/splitting of payload packets (also for MTU reasons)?

From draft-roca-nwcrg-rlc-fec-scheme-for-quic-02:



FEC: Design choices

- Classes of FEC schemes (that can be handled equivalently by LOOPS)
  - What are the FEC indices to be added to payload packets?
    (Tunnel: right there; Transparent: separately)
- Do we put in some MTU mitigation (breaking up payload packets)?
  Piggy-backing runts/short packets/repair symbols?
- Feedback:
  - For controlling FEC rate — what is the time scale?
  - For filling in repair packets?
- Details of the construction of FEC input and repair packets;
  how are reconstructed packets put together again?

LOOPS: Next Steps

While we are not a WG…

- Continue on, *working* like a WG
  - Explore design space, maybe holding back on tough decisions for now
- Continue improving the set of documents, possibly adding FEC document
  - Identify authors and reviewers
- Employ github.com/loops-wg and loops@ietf.org for coordination
- Review charter proposal at github; react to AD input on this
- Aim for being a WG at IETF 107 (Vancouver, March 2020)