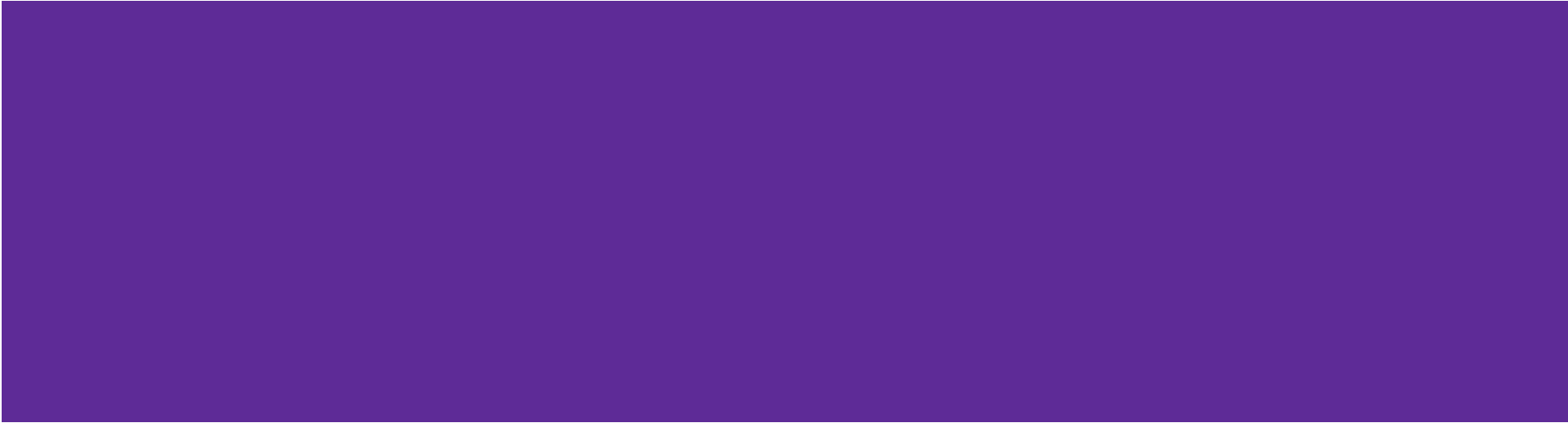


# Proportional Rate Reduction for TCP

## draft-ietf-tcpm-prr-rfc6937bis-04

ietf 117 tcpm

Yuchung Cheng ([ycheng@google.com](mailto:ycheng@google.com)), Neal Cardwell ([ncardwell@google.com](mailto:ncardwell@google.com))



# RFC6937

- Published in 2013 as an experimental RFC
  - Only implemented by Linux; implemented without RFC8985 (RACK-TLP)
- The **mini** congestion control during fast recovery
  - Send packets at the ratio of CC's cwnd reduction
    - Reno: 0.5, Cubic: 0.7
  - If inflight drops below ssthresh
    - Mode SSRB: slow-start
    - Mode CRB: packet conservation
  - Implementation has to pick SSRB or CRB option
- Interesting fact: a flow that frequently operates in fast recovery, its cwnd is mostly controlled by RFC6937 instead of Cubic/CC
  - e.g. policed video streaming flows using Cubic

# RFC 6937-bis

10 years later

- PRR is default-enabled in Linux, FreeBSD, Netflix-BSD/RACK, MS Windows TCP
- tcpm voted to revise and publish as a standard RFC

Main revisions

- Algorithm refinements
  - Merge SSRB and CRB modes: automatically choose the mode based on if the last ACK indicate further losses // bis-01
  - Force a fast retransmit on entering recovery to maintain ACK clock // bis-02
  - Non-SACK support // bis-03
  - Streamline the sending process if experienced higher network reordering previously // **bis-04**
- Editorial clarifications
  - Do not slow start on ACKs that trigger RFC 6675 “last resort” retransmission as it may indicate further losses
  - Relationships with RFC6675 (pipe), RFC8985 (RACK-TLP)
  - Removed experiments section

# RFC 6937-bis-04: RecoverFS refinement

Upon entering fast recovery:

```
RecoverFS = FlightSize = snd.nxt - snd.una  
RecoverFS = pipe
```

For each ACK:

```
if (pipe > ssthresh) /* Proportional Rate Reduction */  
    sndcnt = CEIL(prr_delivered * ssthresh / RecoverFS) - prr_out
```

## Rationale

- FlightSize overestimates the actual data in flight
  - Particularly if the sender starts recovery after >3 SACKs due to previous reordering events
- The overestimation causes an uneven sending rate over the PRR round (below the intended rate, then above the intended rate)