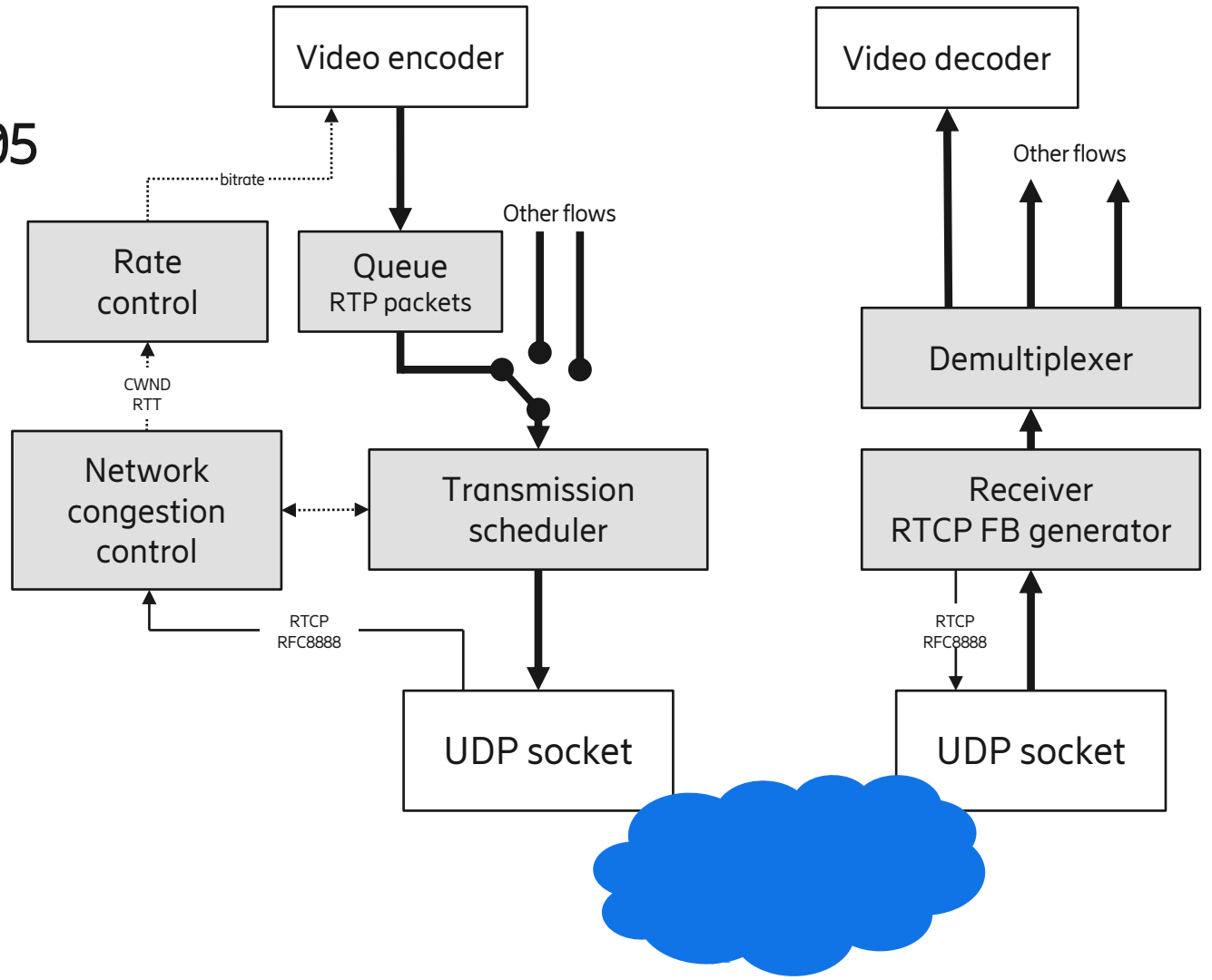
SCReAM v2

draft-johansson-ccwg-rfc8298bis-screamv2-05

IETF-124 Montreal, Canada



Ingemar Johansson
Magnus Westerlund
Mirja Kühlewind
Ericsson AB

ingemar.s.johansson@ericsson.com

Basic function

Network congestion- and transmission control

- ~~Congestion~~ reference window (ref_wnd)
- Bytes in flight can be higher than ref_wnd
  - Avoid unnecessary sender side queuing of large video frames
- ref_wnd increase is continuous
  - < 1MSS/RTT if close to last known max
  - > 1MSS/RTT if long since congested (multiplicative increase)
  - Limited by bytes in flight
- ref_wnd decrease on congestion
  - Packet loss detected
  - ECN/L4S
  - Estimated queue delay increases
- Packet pacing ~150% of target_bitrate

Media rate control

- Objective: maintain just about enough data to fill available capacity
- target_bitrate = 8*ref_wnd/s_rtt
- Challenging part
  - Media coder can have a slow rate control loop
  - Large I-frames complicate congestion control
    - Gradual decoder refresh recommended
  - Sometimes media rate <> target_bitrate
- Sender side queue can be discarded if too large

See also https://datatracker.ietf.org/meeting/119/materials/slides-119-ccwg-screamv2

Progress since IETF-119
draft versions -01 -- -05

- Draft is mainly transport protocol agnostic
- Major clean-up
- Discussion on large reference window overhead and “sluggish” video encoders
- Discussion on issues with clock drift and clock skipping

- Delay based CC is always active
- Modified delay based virtual L4S marking backoff
- Slow down ref_wnd growth when close to last known max value
- ref_wnd decrease/increase limited when ref_wnd/mss is small
- Fast attack slow decay filter for l4s_alpha
- ref_wnd increase limited if L4S is likely non-active and queue delay increases

Missing + additional improvements

To be added in later draft version

- Additional section that describe remedies to clock drift and clock skipping (already in code on github)
- Improve stability for case with e.g. sluggish media coders
  - Adaptive REF_WND_OVERHEAD
  - Window overhead is reduced if RTT variance increases → reduces overshoot and queue delay spikes

WebRTC implementation by Google WebRTC team

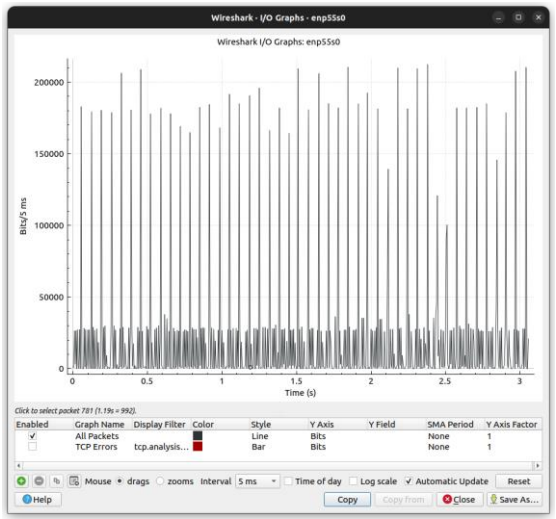
- Available in chrome canary build
- Enabled with : /opt/google/chrome-canary/chrome --enable-logging=stderr --disable-webrtc-encryption --vmodule=scream_v2=1 --force-fieldtrials="WebRTC-RFC8888CongestionControlFeedback/Enabled, offer:true/WebRTC-Bwe-ScreamV2/Enabled/" https://l4s-testbed-4smqe.ondigitalocean.app/
- No window limitation
- Quite relaxed packet pacing

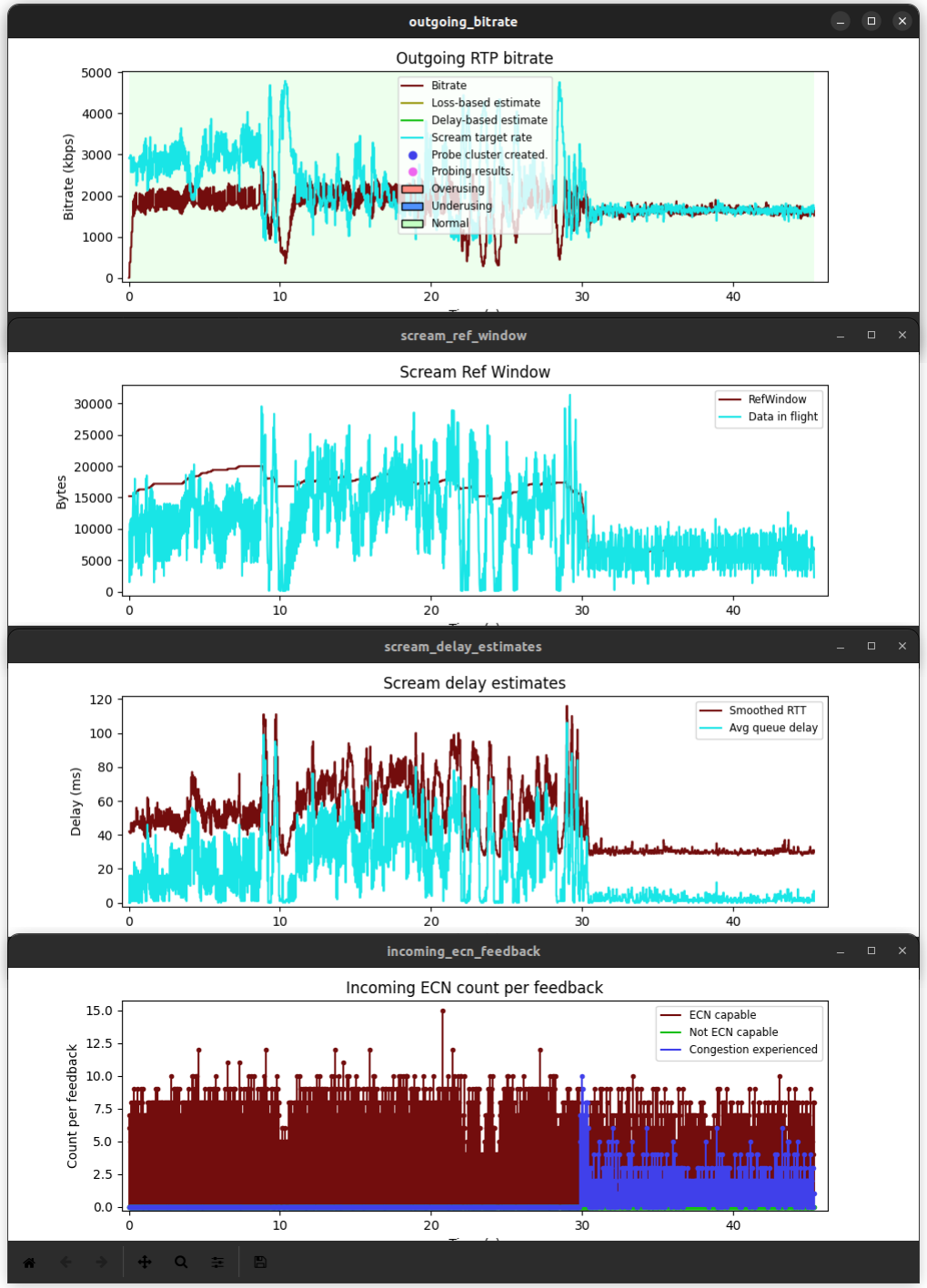
WebRTC SCReAM logs

Chrome canary 143.0.7498.2

- Example: 2Mbps emulated bottleneck with 25ms min RTT
  - fq_codel enabled after 30s
  - L4S mark threshold = 5ms

- L4S improves stability
- Frame rate is more stable
- Bitrate goes down to 1.5Mbps because of bursty transmission





WG item ?

- Algorithm is well tried out
- Evaluation in 5G L4S product
- Remote control car demos
- Trial test implementation in WebRTC
- Minor changes in algorithm recently
  - But additional improvements are expected based on additional experiments

65.584626698771166, 22.145146891851766

Photo : Maine Jäderberg