draft-hoeiland-joergensen-aqm-fq-codel

Toke Høiland-Jørgensen

Karlstad University

toke.hoiland-jorgensen@kau.se

IETF AQM WG, November 10, 2014

Outline

- The FQ-CoDel algorithm
- Changes to version -01 of the draft
- Moving forward

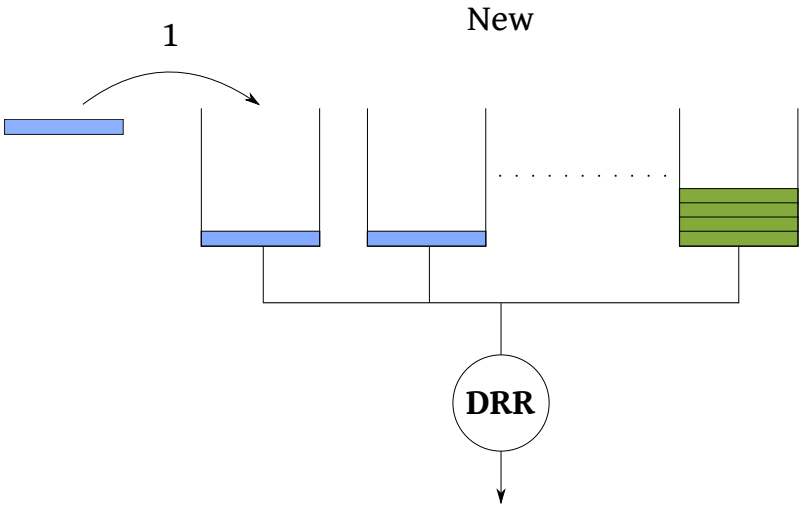
The basis for FQ-CoDel

- SFQ: Stochastic Fairness Queueing (McKenney, 1990)
  - Hash flows into queues, round-robin per-packet dequeue
- DRR: Deficit Round-Robin (Shreedhar and Varghese, 1996)
  - Track deficit at dequeue to approximate byte fairness
- DRR++ (MacGregor and Shi, 2000)
  - Add a separate tier for priority traffic. De-prioritise flows that send too much data.
- ...and of course the CoDel AQM.

The Flow Queueing scheduling algorithm

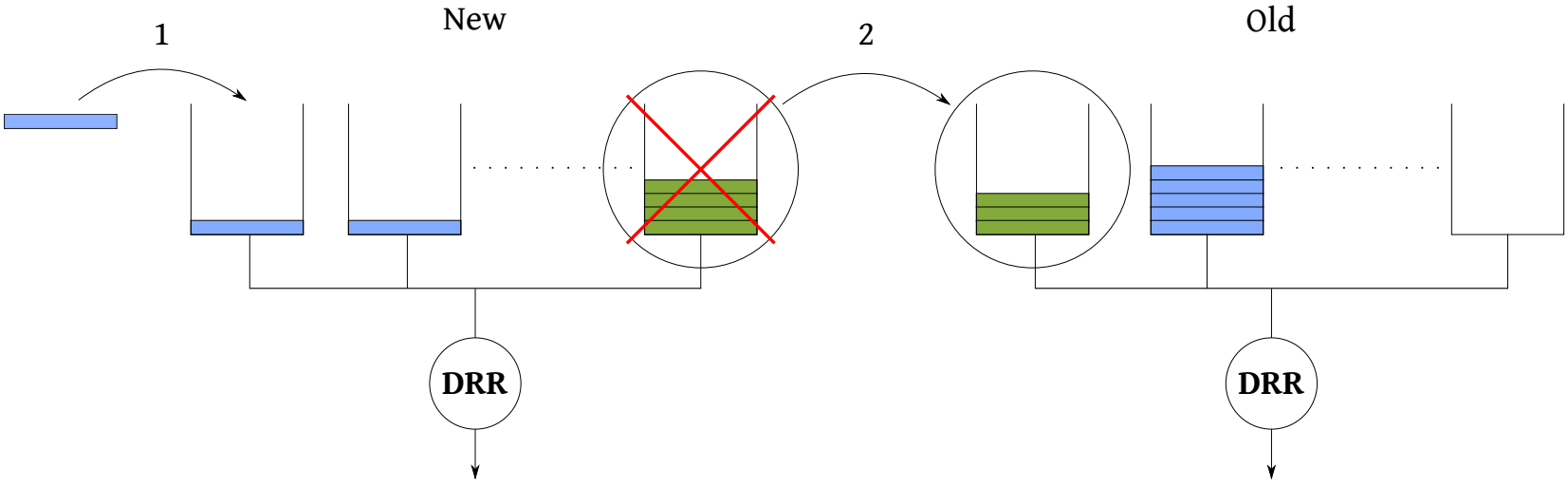
Flows are identified. The first packet(s) from new flows are scheduled before packets from flows that have built a queue. If a flow's queue empties, it is eligible to again be considered a new flow. CoDel is applied to control the queue length in any flow. Result: timely delivery, and mixing of flows.

The Flow Queueing scheduling algorithm



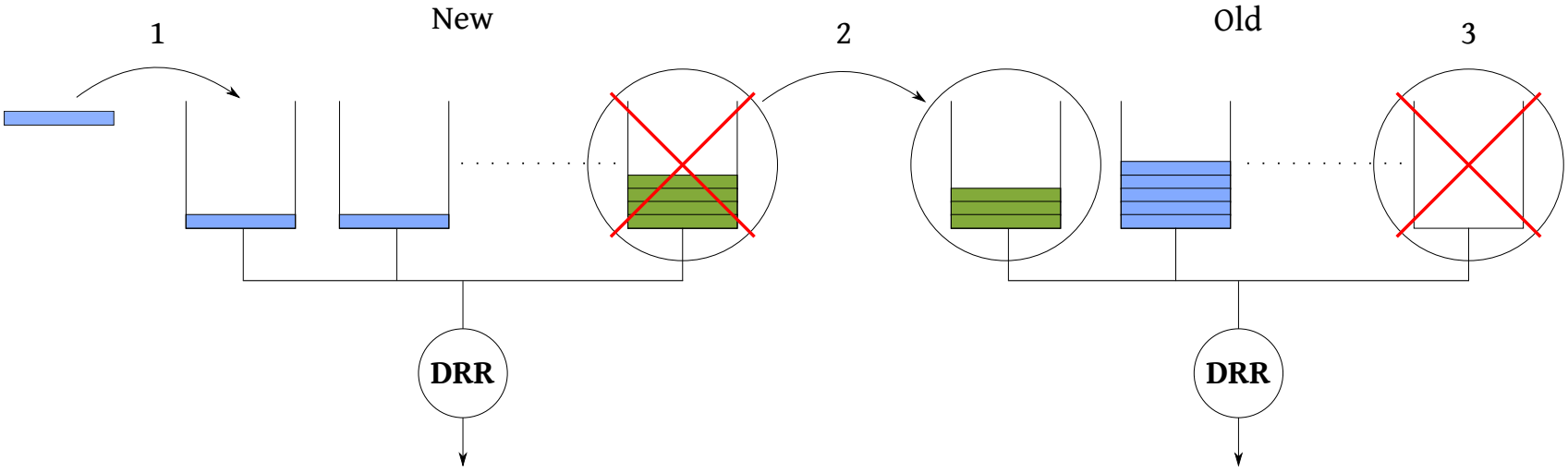
1. Packet from a new flow comes in, is put into a new queue.

The Flow Queueing scheduling algorithm



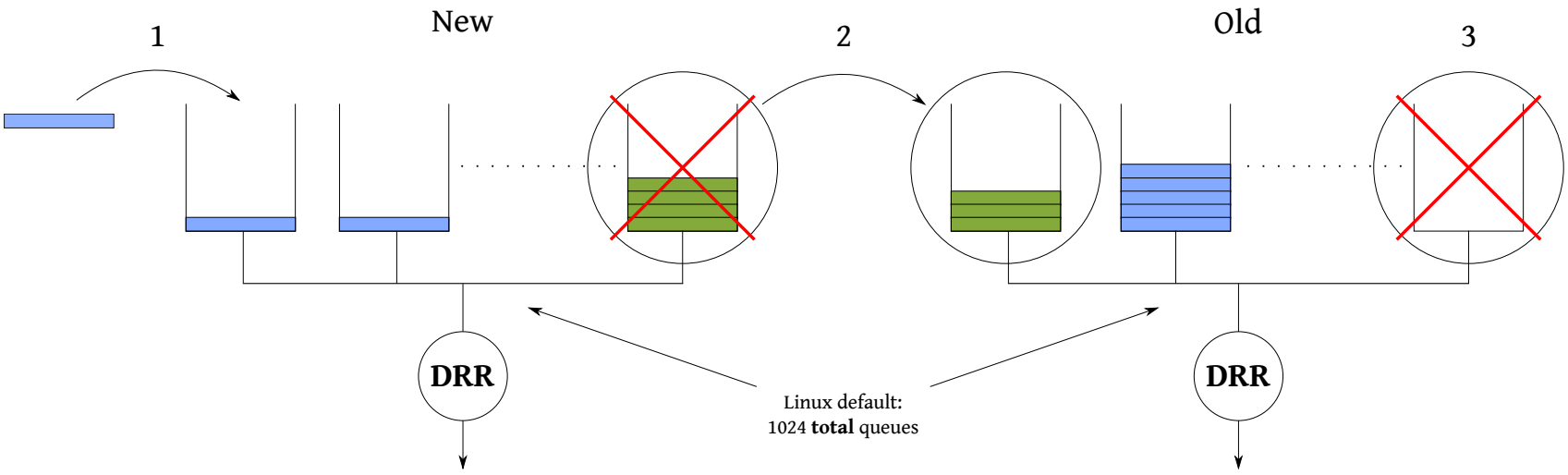
1. Packet from a new flow comes in, is put into a new queue.
2. An existing 'new' queue dequeues a quantum of bytes, and is moved to the old queues.

The Flow Queueing scheduling algorithm



1. Packet from a new flow comes in, is put into a new queue.
2. An existing 'new' queue dequeues a quantum of bytes, and is moved to the old queues.
3. An empty queue gets its turn to dequeue and is removed.

The Flow Queueing scheduling algorithm



1. Packet from a new flow comes in, is put into a new queue.
2. An existing 'new' queue dequeues a quantum of bytes, and is moved to the old queues.
3. An empty queue gets its turn to dequeue and is removed.

Benefits of FQ-CoDel

- Flow isolation
- Prioritisation of many latency-sensitive applications
- Mixing of flows
- Almost completely solves bufferbloat at the edge!

Evaluation results presented in ICCRG tomorrow.

Status of FQ-CoDel deployment

- FQ-CoDel has been in Linux since v3.4 (May 2012).
- One line of configuration to enable (v3.11+):
  - `echo 'net.core.default_qdisc=fq_codel' > /etc/sysctl.d/qdisc.conf`
- On by default in several places. E.g:
  - OpenWrt Barrier Breaker.
  - Systemd 217+.
- Major component of QoS systems:
  - OpenWrt, DD-WRT, CeroWrt, IPFire, Gentoo Linux, Ubnt Edgerouter, Qualcomm Streamboost, Netgear "Dynamic QoS", others.

Changes to version -01 of the draft

- Added more references to prior work – esp. DRR++
- Added a section on limitations of the algorithm
  - Fairness between things other than flows
  - Flow bunching by opaque encapsulation
  - Low-priority congestion control algorithms
- Minor changes to language to clear things up

Moving forward

We have the running code – how do we get the rough consensus?

- Adopting the draft in its current form?
- Restructuring to decouple AQM and scheduling?
- Other ideas?

References

- P.E. McKenney (1990). *Stochastic fairness queueing*.
- M. Shreedhar and G. Varghese (1996). *Efficient fair queuing using deficit round-robin*.
- M.H. MacGregor and W. Shi (2000). *Deficits for Bursty Latency-critical Flows: DRR++*.
- The draft itself: `https://datatracker.ietf.org/doc/draft-hoeiland-joergensen-aqm-fq-codel/`.
- The CoDel algorithm: `https://datatracker.ietf.org/doc/draft-ietf-aqm-codel/`.