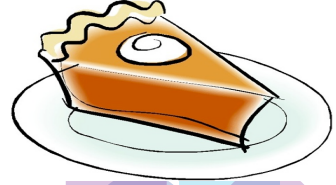
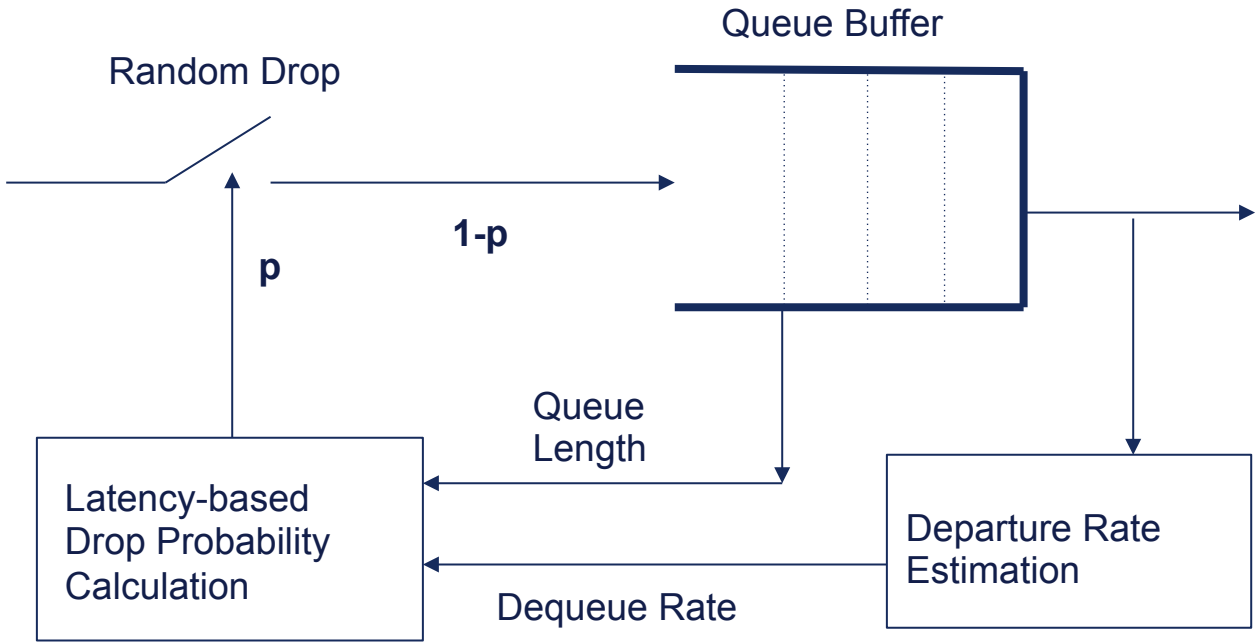
PIE: A lightweight latency control to address the bufferbloat problem

Rong Pan, Preethi Natarajan, Fred Baker, Bill Ver Steeg, Mythili S. Prabhu, Chiara Piglione, Vijay Subramanian and Greg White

November 10, 2014, IETF 91

The block diagram of PIE





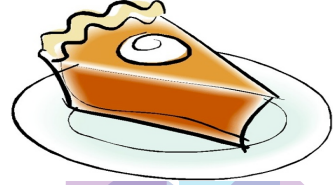
The design of PIE

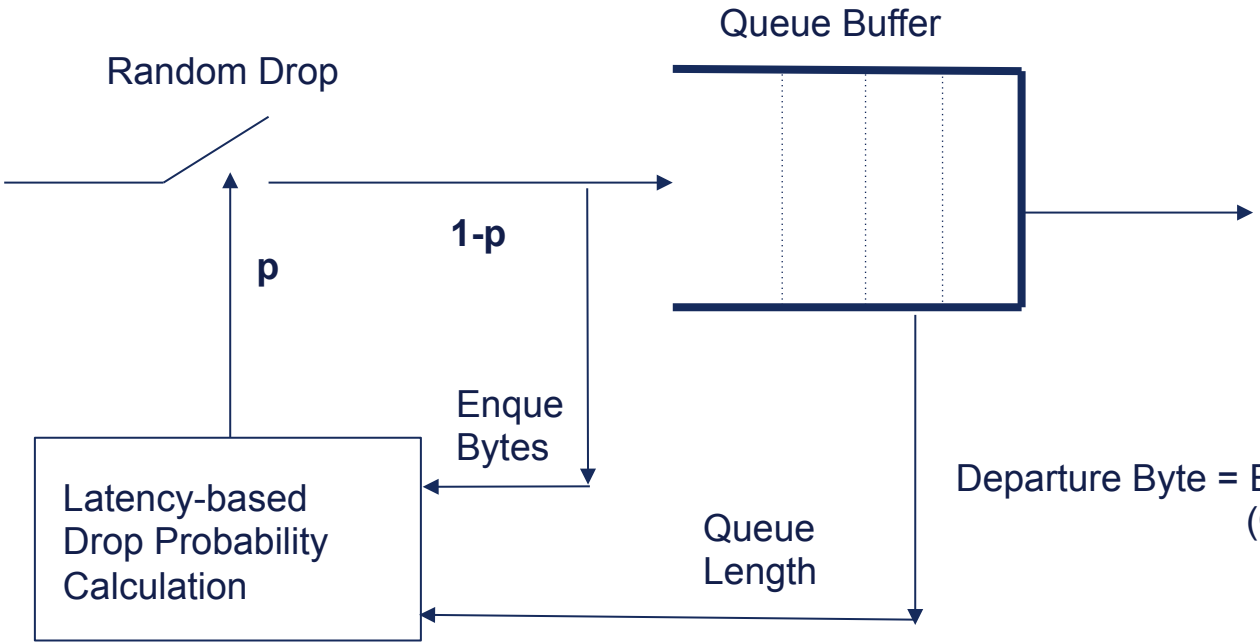
- Upon every packet arrival
  - randomly drop a packet based on drop_prob calculated below
- Every T_{update} interval
  - estimated_delay, est_del = queue_length/depart_rate
  - drop_prob += a*(est_del – target_delay) + b* (est_del – est_del_old)
  - est_del_old = est_del;
  - depart_count = 0;
- In a measurement cycle
  - Upon a packet's departure: depart_count += deque_packet_size;
  - if dq_count > deq_threshold then
  - depart_rate = deqart_count/(now-start);
  - dq_count = 0; start = now;

PIE Work Update

- Turning PIE on/off automatically
  - Spurious uptick in queueing delay would cause packet drops
- Extending auto-tuning range of PIE
  - Extend the auto-tuning region all the way up to 0.001% drop probability
- Enhanced Burst Tolerance
  - Related to the first bullet, burst tolerance is only triggered when PIE is active
  - Spurious spike will not be counted towards burst tolerance
- De-randomization
  - Random tosses could cause drops too close to each other or too far from each other
  - Add a mechanism to mitigate the outliers
- FQ_PIE
  - First pass of Linux implementation, the test results are promising

The block diagram of enque-based PIE





Departure Byte = Enque Bytes –
(Qlen – Qlen_old)

extra bytes stored in the queue

Thank you.