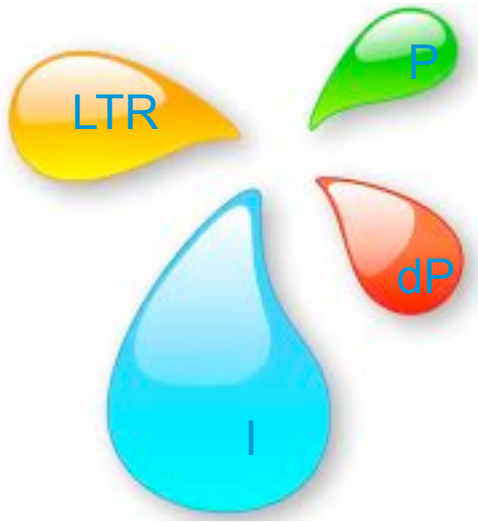
IETF88 Vancouver

Congestion control for video and priority drops

Background for

draft-lai-tsvwg-normalizer-02.txt

Toerless Eckert, eckert@cisco.com



Summary

- draft-lai-tsvwg-normalizer-02.txt discusses key problem of larger solution
  - Provide overview of larger solution here: Priority dropping
- Interest for priority dropping due to p2p video resilience work
- Overlap/beneft also for p2mp switched video
- What can the network do ?
- Consider how priority drops can be beneficial for CC and video quality
- What is missing ?

Use case 1: RT P2P video resilience
A historical perspective

- Loss of video packets during congestion happen.. and is unavoidable
  - Today Internet traffic far from ideal congestion control
  - Even with ideal congestion control: bad competing traffic, burst collisions,…
- Mitigation:
  - Retransmission (incurs delay ~RTT)
  - Concealment (in video layer of application, interpolation == delay)
  - Redundancy/Protection/FEC
- Optimize Protection by taking video packet priority into account
  - Loss of higher priority video packet has bigger impact on quality
  - Streaming: I (high), P (medium), B (low), Conferencing: LTRF (high), P (medium), discardable P (low)
  - Use unequal protection: more FEC for I/LTRF, less for P, none for B/dP
  - BW-cost of FEC still high, efficiency limited by acceptable delay,
  - Effectiveness limited by loss profile (bursty loss = hard to protect with low delay)

Dear network, please drop only low priority video packets

- Avoids FEC downsides: overhead, limited effectiveness, delay

Use case 2: RT P2MP video rate-adaptation.

- Switched MCU video conference:

  Sender -> switching MCU -> multiple receivers

- Congestion from MCU to receiver requires rate-adaptation at MCU

  switching MCU == no codec layer == no transrating/transcoding.

  Rate-adaptation via:

  1. Shaping (== delay == bad)
  2. Select next-best spatial encoded video from sender (eg: QCIF, CIF, 4CIF, 16CIF,…)
  3. Drop frames from that encoded video to match available rate.

     Hierarchical temporal encoding with discardable P-frames.

     Dropping dP frames minimizes visual impact.

- Priority dropping in network can improve this:

  Unavoidable network drops are like the P2P use case (bad or FEC,…)

  Faster: low-prio dropping in network will drop low priority immediately.

  MCU dropped packets (“holes”) make flow more bursty == difficult for CC.

  Loss rate on these flows may get higher or achievable rate lower

Proactive relying on low-priority drops: Better than no drops ?!

- Minimizing drops via rate-control impacts throughput
  - Especially with bursty traffic.
  - RT video traffic has great justification/need for burstyness
- Example 1: Compare delay-variation/ECN with eg: PCN
  - PCN can achieve lower loss than delay-variation/ECN rate-control alone.
  - PCN “Headroom” is unused bandwidth available to bursts.
- Example 2: Similar effects for conservative rate-control
  - The less rate-control “probes” the limit to loss, the less loss there will be.
  - And the less throughput.
- Claim 1 (intra-flow): Visual impact of loosing low-priority video packet may be lower than a reduction in overall bitrate of video flow.
- Claim 2 (inter flow): Aggregate quality result is better when high-prioirty packets can burst more without loss – at the expense of low-priority packets sometimes getting dropped.

So, what could a network do ?

- Assume we have video packets marked with priority
- Queues in network devices can quite effectively drop low priority packets over high priority packets.
  - Leveraging existing HW queue options: Droptail profiles / WRED
  - Example with three priorities:
    - high (20% bitrate of flows), normal (60% bitrate of flows), low (20% bitrate of flows)
      - Rates are longer term average – eg: over ½ second
    - Queue:
      - high priority packets dropped on 100% queue length
      - Normal priority packets dropped on 90% queue length
      - Low priority packets dropped on 40% queue length
    - On any loss under 10% can achieve >> 99.9% loss in only low priority packets
    - On loss at 40%, can achieve “ideal loss” – all low priority, 20% medium priority, no high priority.
- Guess: the more priorities to distinguish, the less crisp the results.

So, what is missing then ?

- Useful priority markings

  DSCP difficult/overloaded – if priority dropping would be useful for N existing types of traffic (realtime video, streaming video, market-data/telemetry,…) we would need at least 3 * N DSCP.

  For video, RTP header extension with “drop priority” would be ?ideal?

  RTP header extension would require onpath signaling to let routers know (eg: MALICE).

- Application support

  Mark the priority of packets

  Encode video to best utilize packet priorities.

  Optimize rate-adaptation & congestion control if priority dropping is supported.

  If loss in network is only in low-priority packets, application know that congestion happens at a point in network that supports it.

  - Better/faster upspeeding
  - less need to shape/avod bursts
  - No protection overhead, … adjusted CC parameters (longer rate averaging)

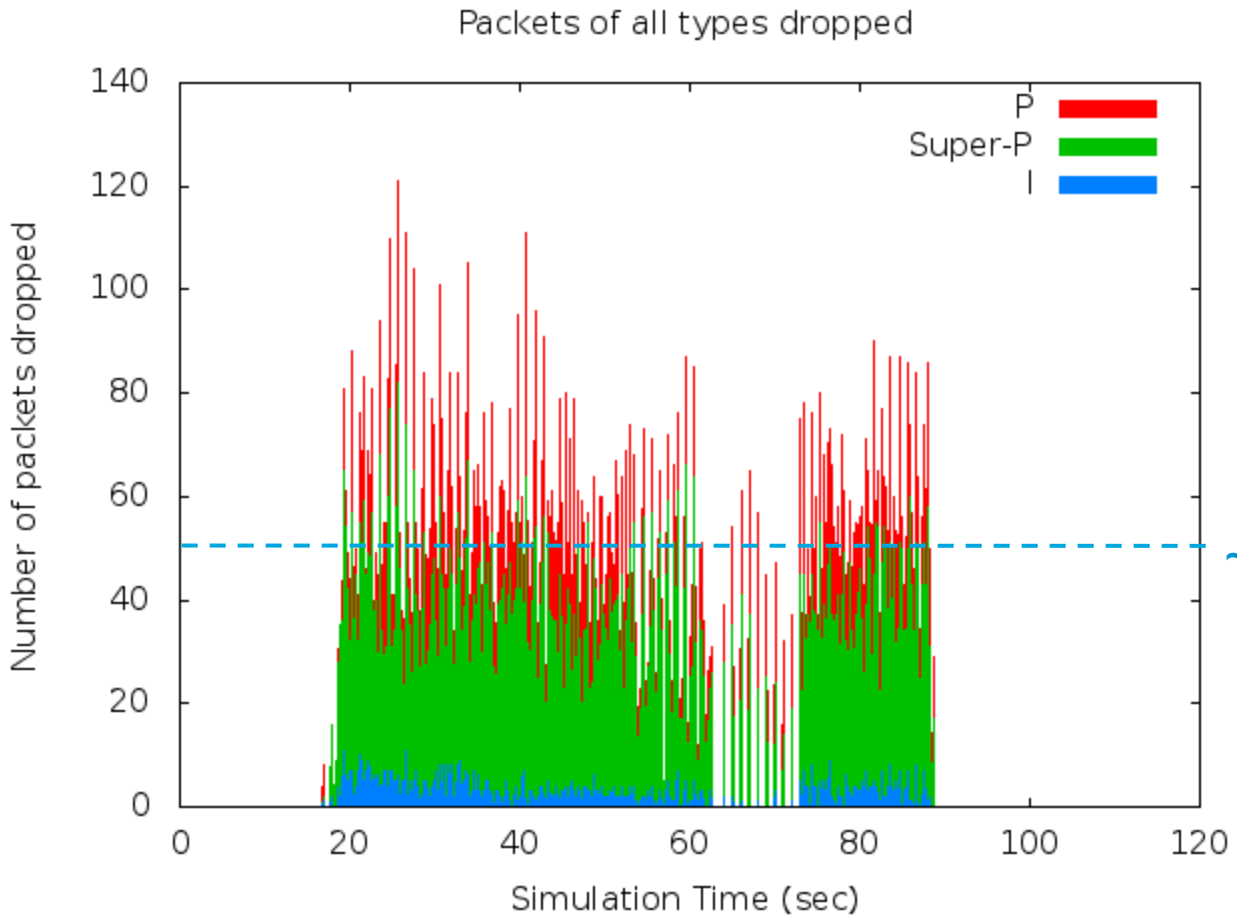
- Fairness, Normalization, standard profiles ?

Fairness ?

- Unfairness with existing queuing setups:
  - The lower the average priority in a flow, the more loss it will see vs. other flows
  - No motivation for applications to honestly mark priority of packets
  - Great incentive to mark packets with only high-priority
- draft-lai-tsvwg-normalizer-02.txt solves this problem
  - Router analyzes distribution of priorities in flow.
  - Remaps priorities (internally, not visible in packet) so that distirbution of priorities match a normalized profile (eg: 20% high, 60% medium, 20% low).
  - All flows now compete fairly in the queue and see same amount of drops.
  - Running code. But unclear if this scales to higher end routers
- More generic approach ?
  - Agree on a simple standard profile 20/60/20 ?
  - Perform normalization or filtering only on “trust” edge
    - Exactly like for any other QoS function.

Background: some simulation results

Droptail
all packet priorities can fill whole queue

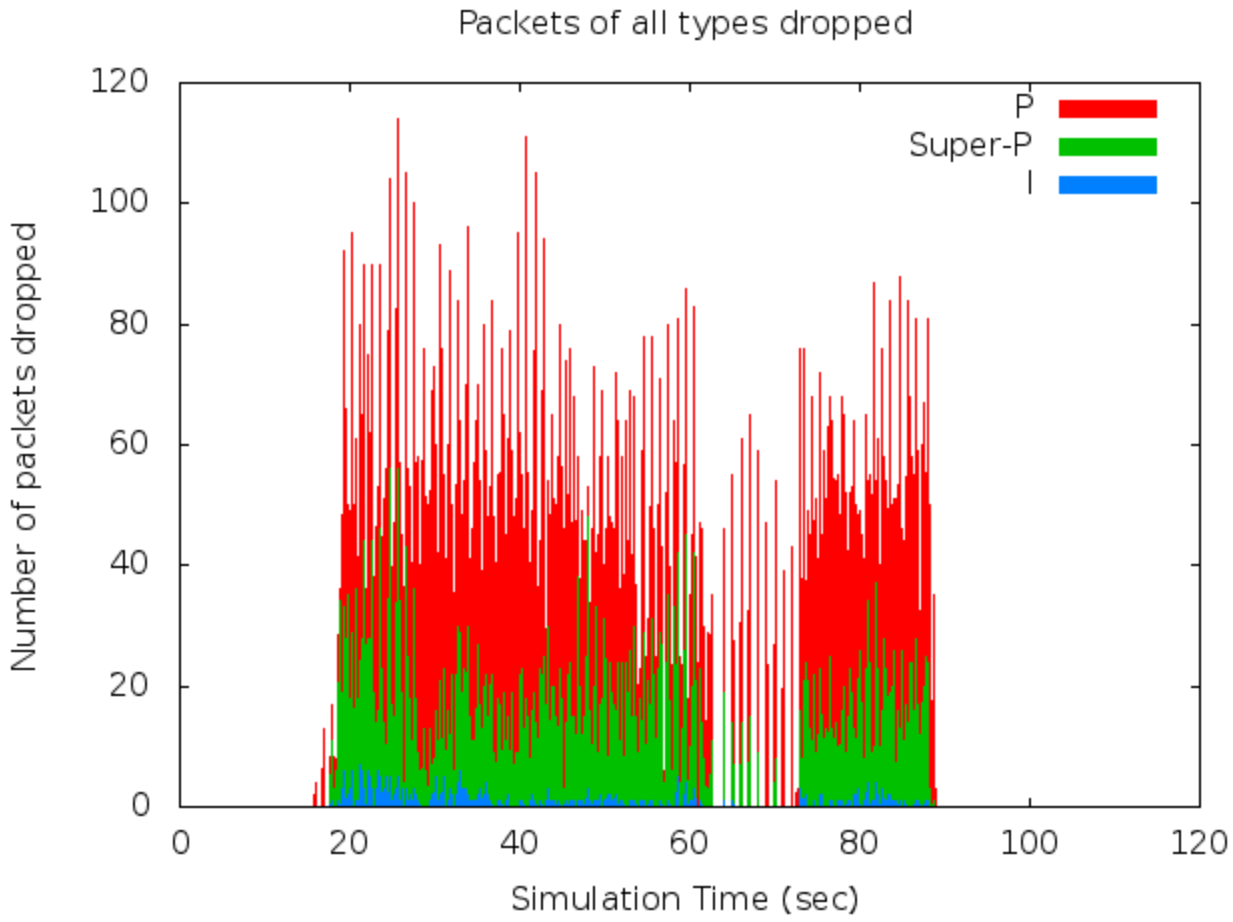


Wrong labels:
red = dP
Green = P
blue = LTRF

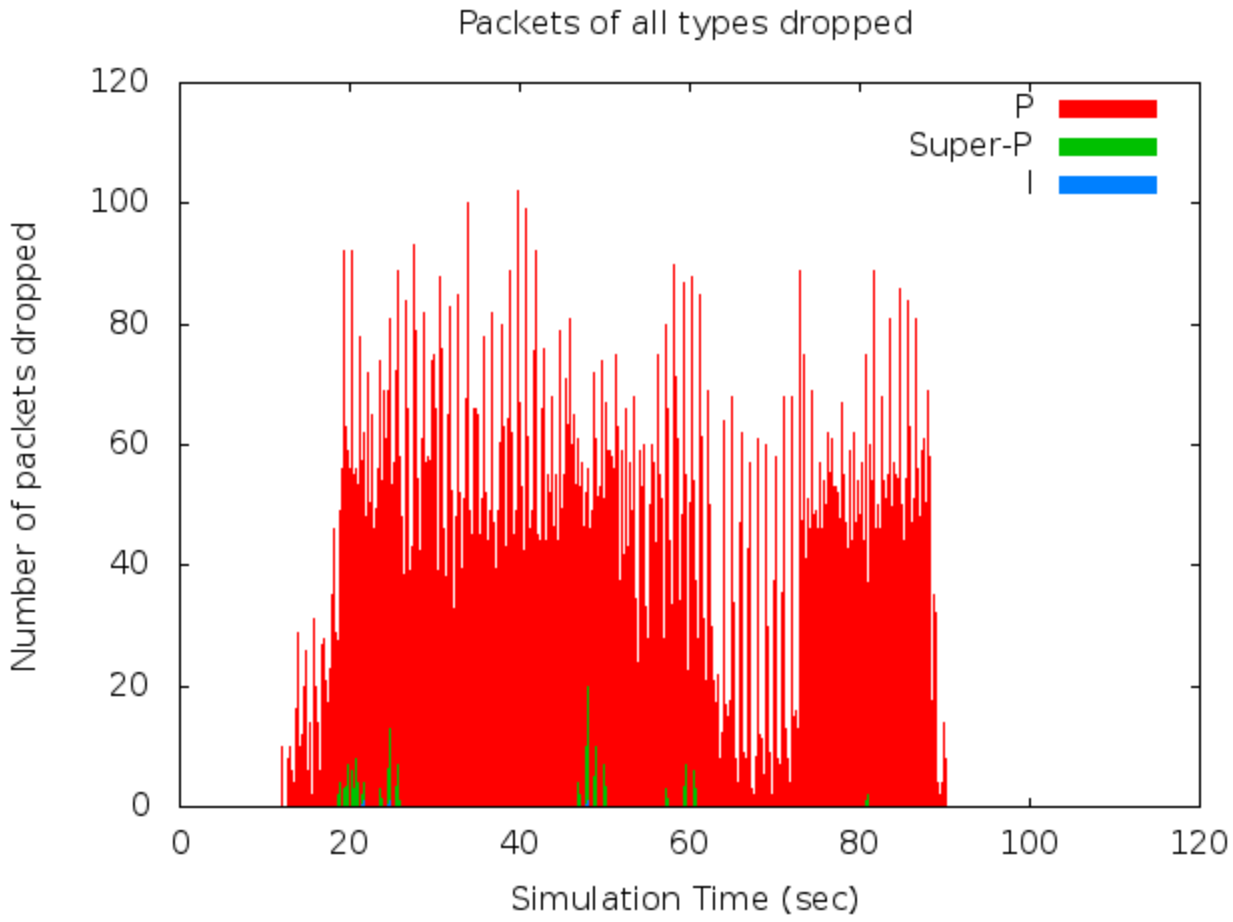
~10% dropped

- Measured: # of packets dropped every 0.1 second

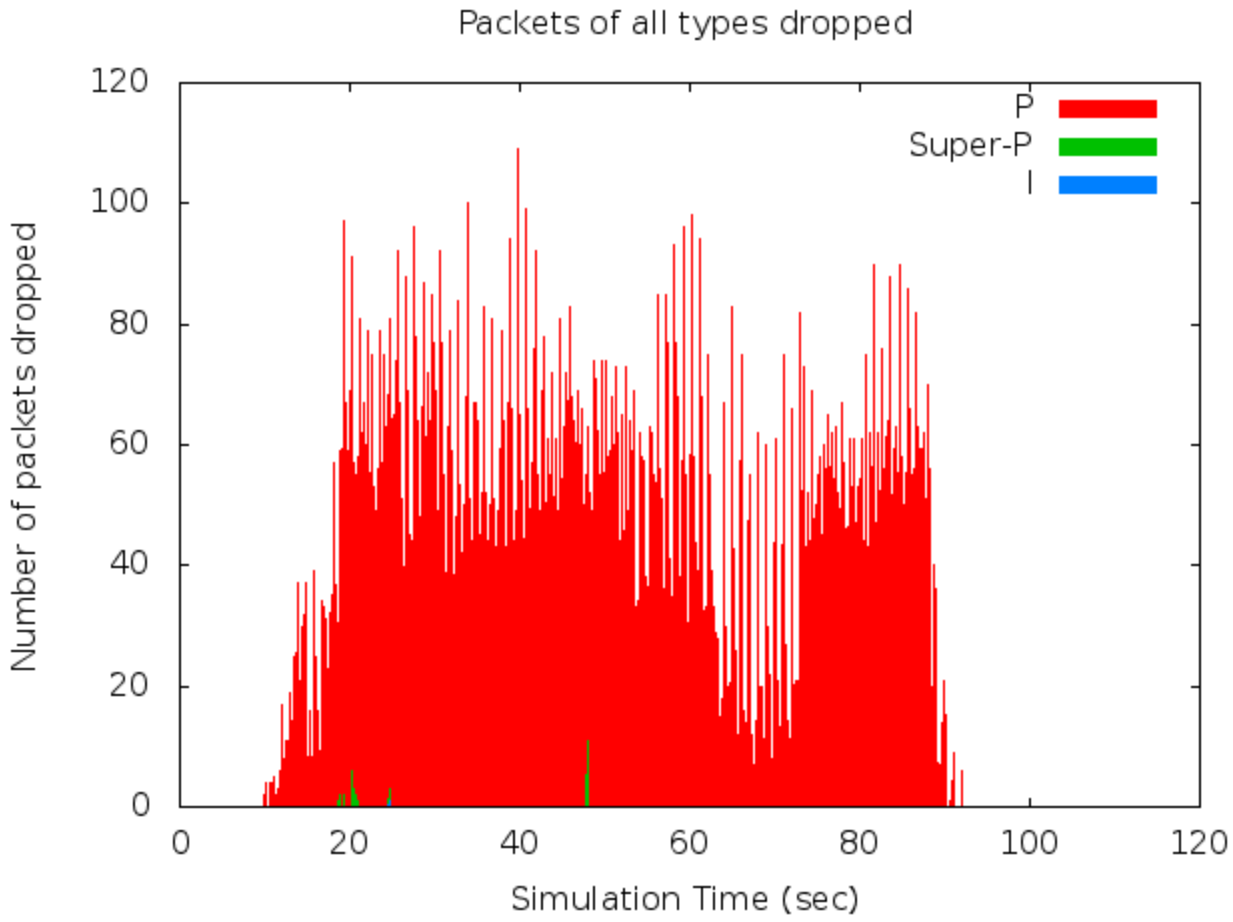
Threshold_dP=90%



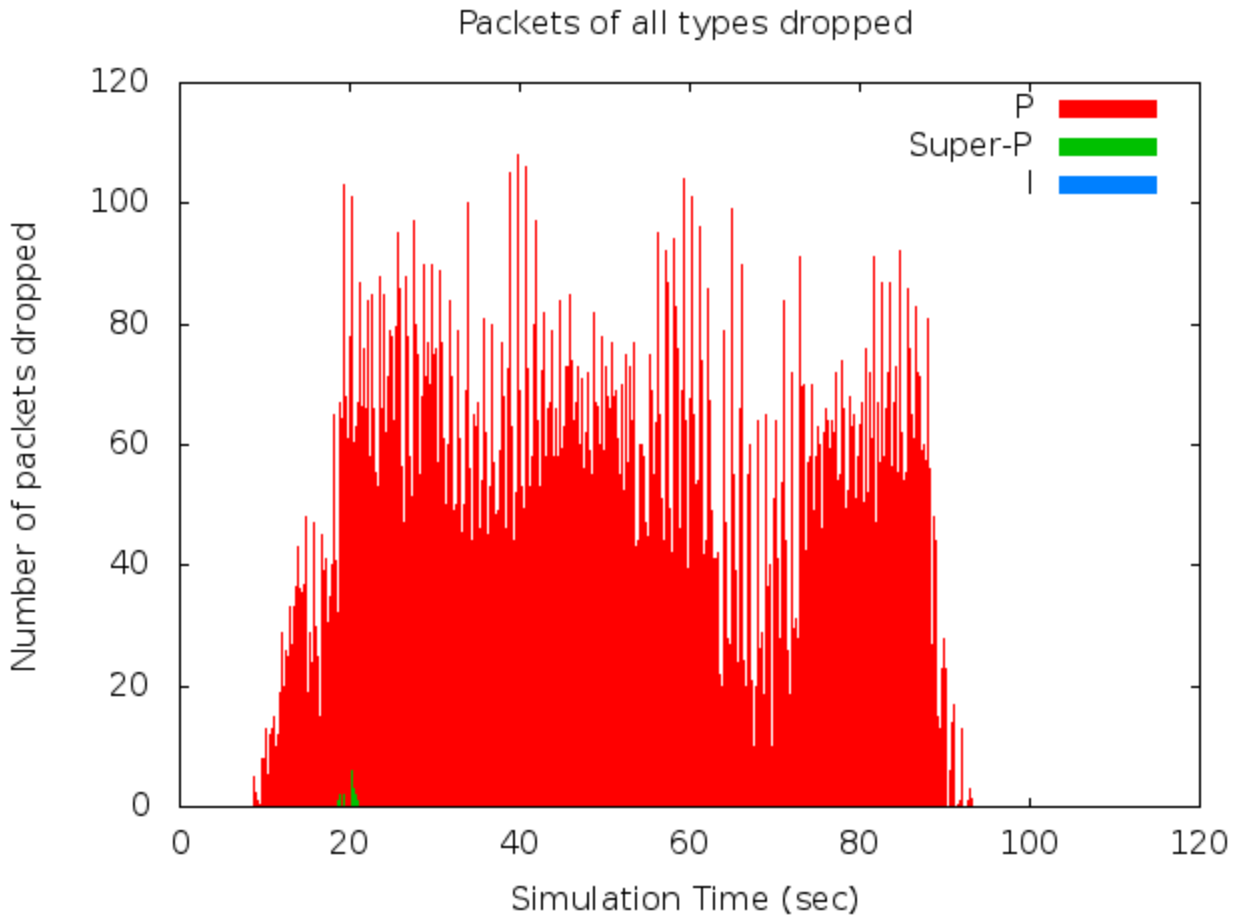
Threshold_dP=50%



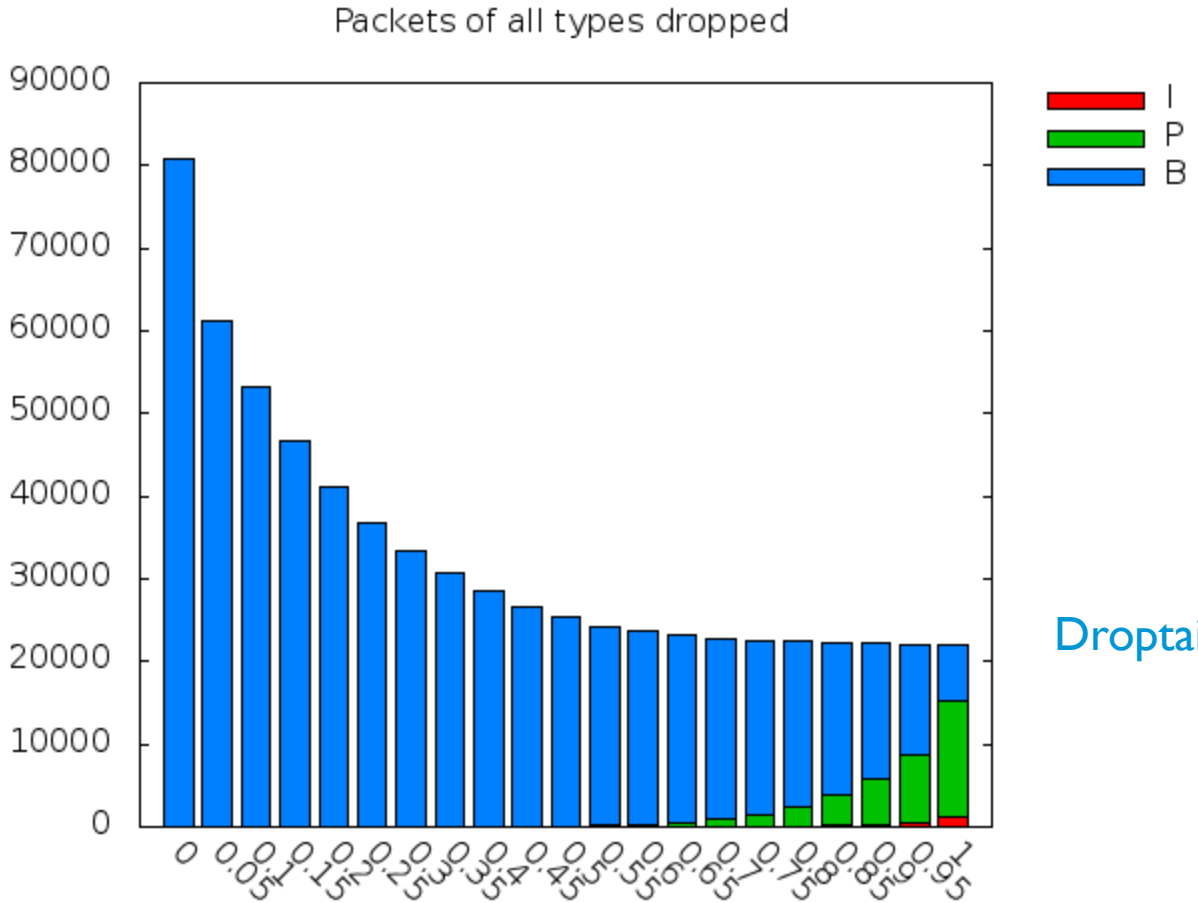
Threshold_dP=40%



Threshold_dP=30%



Varying drop thresholds for dP-Frames percent of queue-length



Droptail