FastLane: Making Short Flows Shorter with Agile Drop Notification
David Zats, Anand Iyer, Ganesh Ananthanarayanan, Rachit Agarwal, Randy Katz, Ion Stoica, Amin Vahdat

Motivation
- Many real-world workflows consist of large numbers of short flows
- Application-layer performance depends on the last flow that finishes
- Network is a limiting factor → can cause workflows to miss deadlines

Drops are Costly!
- Transport guesses when they occur!
- Relies on duplicate acks / timeouts
- Takes long time to detect & react
- Prior approaches to reducing drops:
  - DCTCP learns across flows
  - D3 / PDQ has switches set tx rates

Motivation
- Avoid timeouts
  - Particularly helpful for short flows
- Don’t need in-order delivery to recover
  - Enables per-packet load balancing
- Avoid delays at hotspots by notifying sources directly (unlike CP)

Challenges
- Switch overheads from generating notifications:
  ✓ Transform, don’t generate notification

Advantages
- Network overheads from notification transmission:
  ✓ Cap rate of notification generation
  ✓ If cap exceeded, best to time out
- Ping-pong where packet retransmitted too early and dropped again:
  ✓ Measure ping-pong behavior (maintain counter for every retransmitted packet)
  ✓ Exponentially throttle retransmissions as ping-pong behavior increases

Evaluation
- Simulation (NS-3):
  - 128 Server FatTree
  - 10Gbps links, 4x Oversubscription
- Baselines:
  - TCP-Codel w/ 1ms timeouts
  - pFabric w/ 250us timeouts
- Workload:
  - 2, 4, 8, 16, 32 KB many-to-one flows
  - 1 MB all-to-all flows
- Metric: reduction in 99.9th percentile completion time

FastLane
1. Switches notify sources of drops
   ✓ Direct notification -> informs sources as quickly as possible
2. Sources respond agilely, retransmitting and reducing rates
3. Network overheads from notification transmission:
   ✓ Cap rate of notification generation
   ✓ If cap exceeded, best to time out
   - Ping-pong where packet retransmitted too early and dropped again:
     ✓ Measure ping-pong behavior (maintain counter for every retransmitted packet)
     ✓ Exponentially throttle retransmissions as ping-pong behavior increases

Can we address the problem directly by reducing the cost of drops?