Low Extra Delay Background Transport

Presented by Joel Bricker
October 25th, 2012

Inspired by:
Rajan, University of Delaware
Andrew Sprouse, Northeastern University
Dario Rossi, TELECOM ParisTech

CISC 856 TCP/IP and Upper Layer Protocols

References

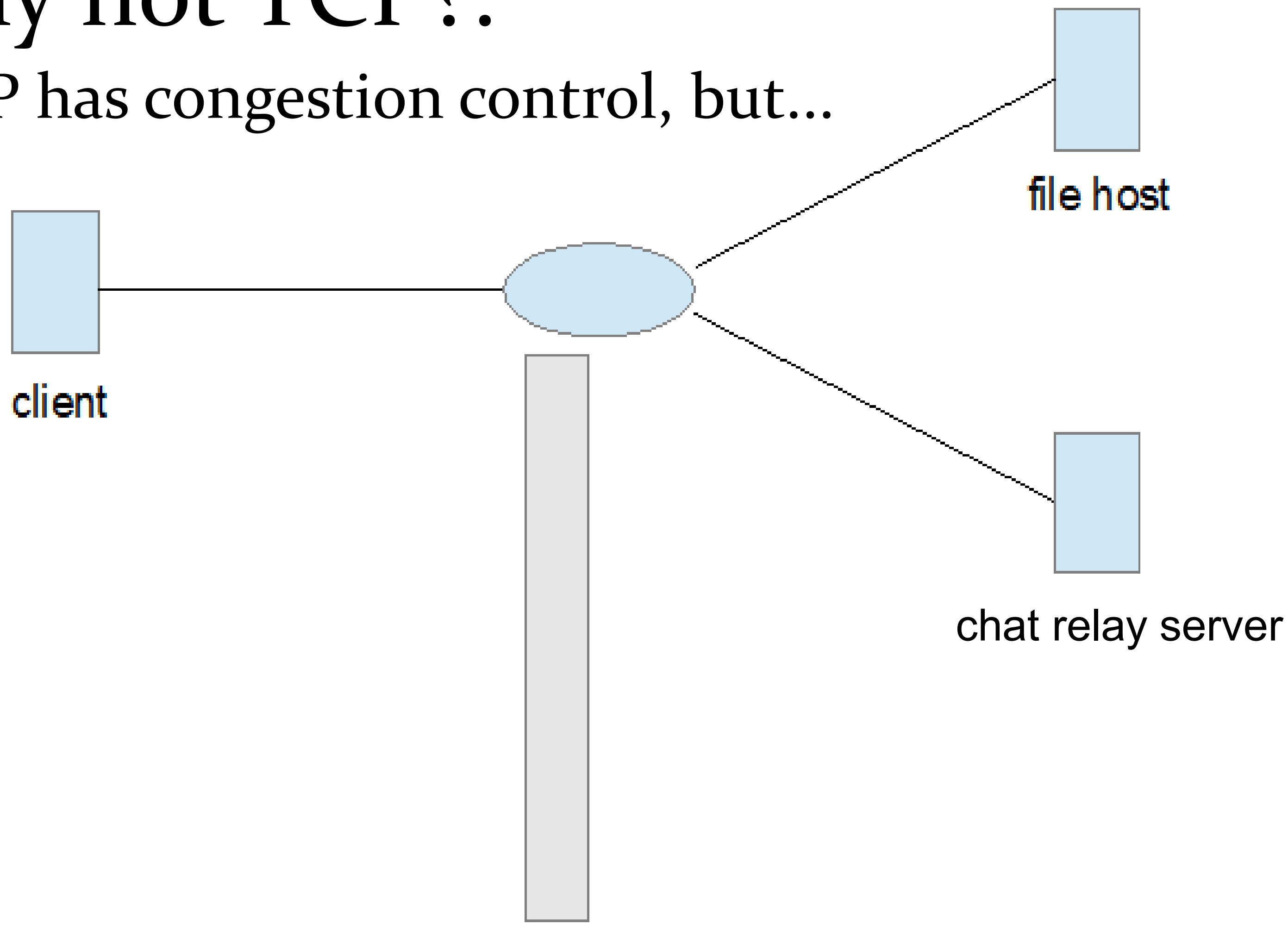
- Low Extra Delay Background Transport (LEDBAT) draft-ietf-ledbat-congestion.txt
- LEDBAT: the new BitTorrent congestion control protocol - Dario Rossi, Luca Muscariello, Computer Communications and Networks (ICCCN), 2010 Proceedings of 19th International Conference
- LEDBAT specification: http://www.bittorrent.org/beps/bep_0029.html

Goals of LEDBAT:

- Yield to higher priority internet traffic
  - Minimize congestion by minimizing queuing delay
- Utilize unused bandwidth fully

Why not TCP?:

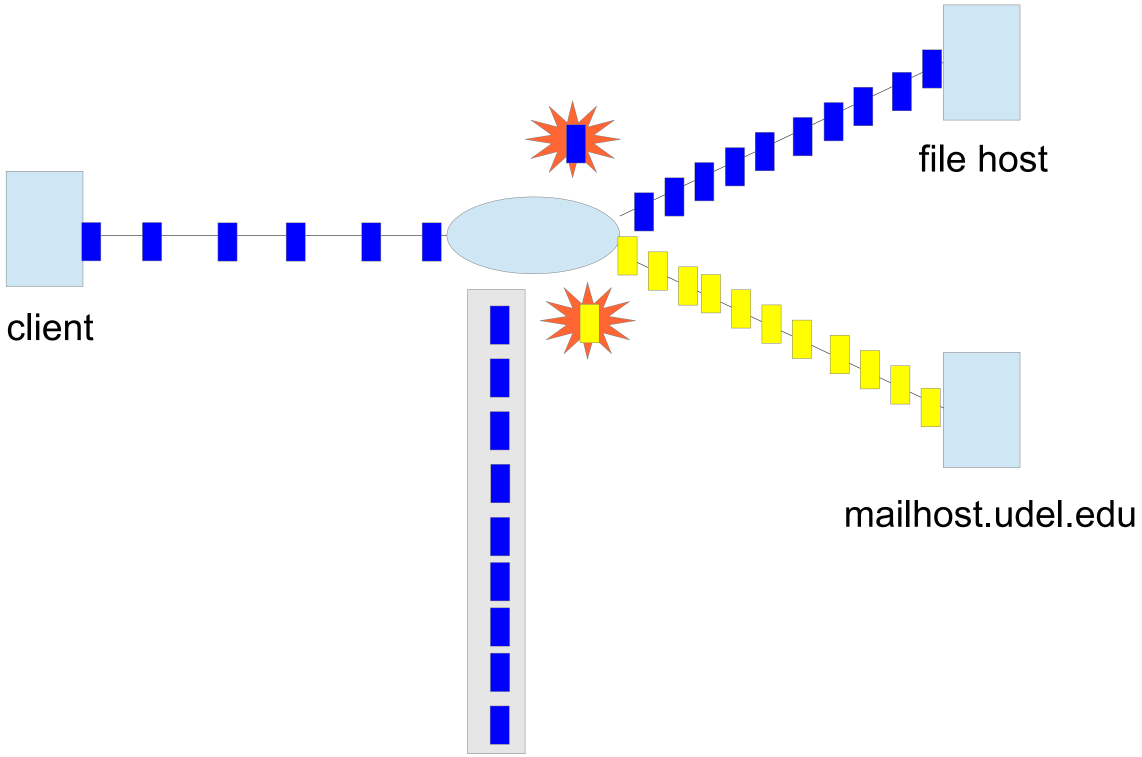
- TCP has congestion control, but…



- Some modems have queues on the order of seconds, so even if the queue is not full delay will be excessive

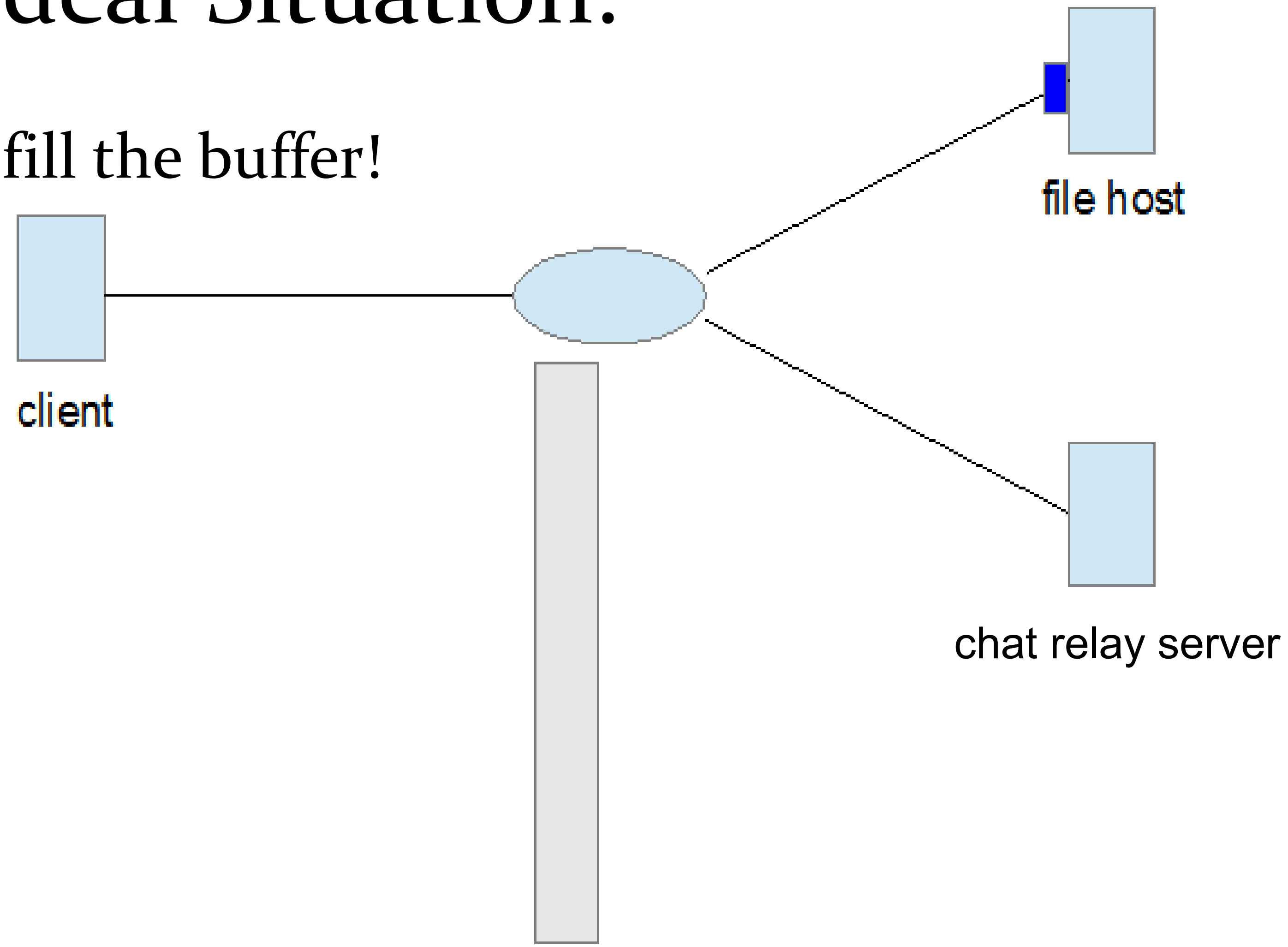
Motivation:

- TCP has congestion control, but…



The Ideal Situation:

- Don't fill the buffer!



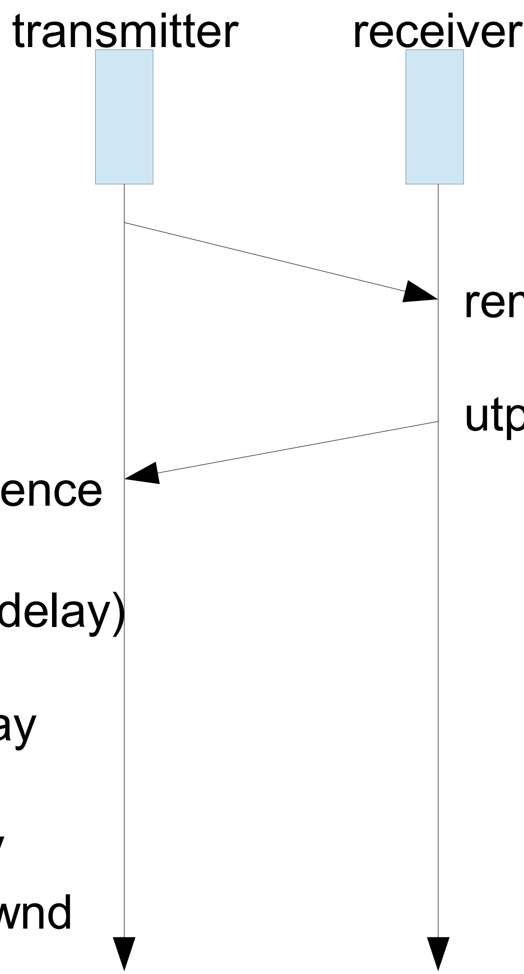
uTP "Transport" Protocol:

- Estimate queuing delay with One-Way Delay
  - One Way Delay is required to avoid reverse-path traffic impacting the calculation
- Operates in the application layer
  - Uses UDP as its transport layer protocol
  - Implements most of the functionality of TCP in A-PCI
  - Allows LEDBAT to implement its own congestion control protocol

uTP A-PCI:

0	4	8	16	32
type	ver	extension	connection_id	
timestamp_microseconds				
timestamp_difference_microseconds				
wnd_size				
seq_nr			ack_nr	

uTP Congestion Window:



remote_timestamp =
utp_pdu.timestamp

utp_pdu.timestamp_difference =
localtime() - remote_timestamp

current_delay =
utp_pdu.timestamp_difference

base_delay =
min(base_delay, current_delay)

queuing_delay =
current_delay - base_delay

off_target =
TARGET - queuing_delay

cwnd += GAIN * off_target / cwnd

TARGET, off_target, and cwnd

- TARGET is the maximum queuing delay that LEDBAT tolerates
- Off_target = TARGET - queuing delay
- For example if TARGET = 20 ms
  - If queuing delay > 20, then off_target is < 0, and sender rate is decreased
  - If queuing delay < 20, then off_target is > 0, and sender rate is increased
- cwnd += GAIN * off_target / cwnd
- Where GAIN = 1 / TARGET, the rate at which the congestion window responds to changes in queuing delay.

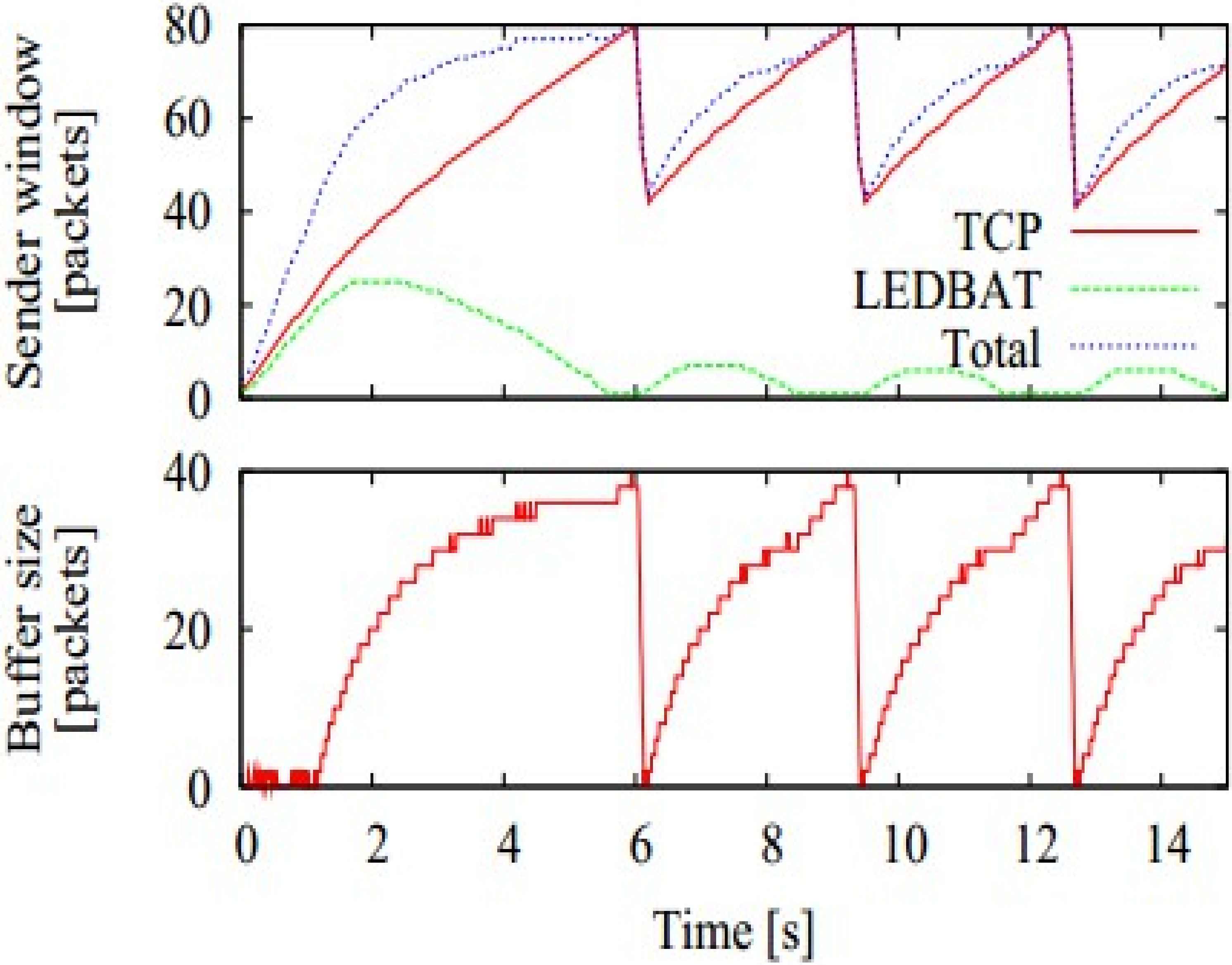
uTP Congestion Window:

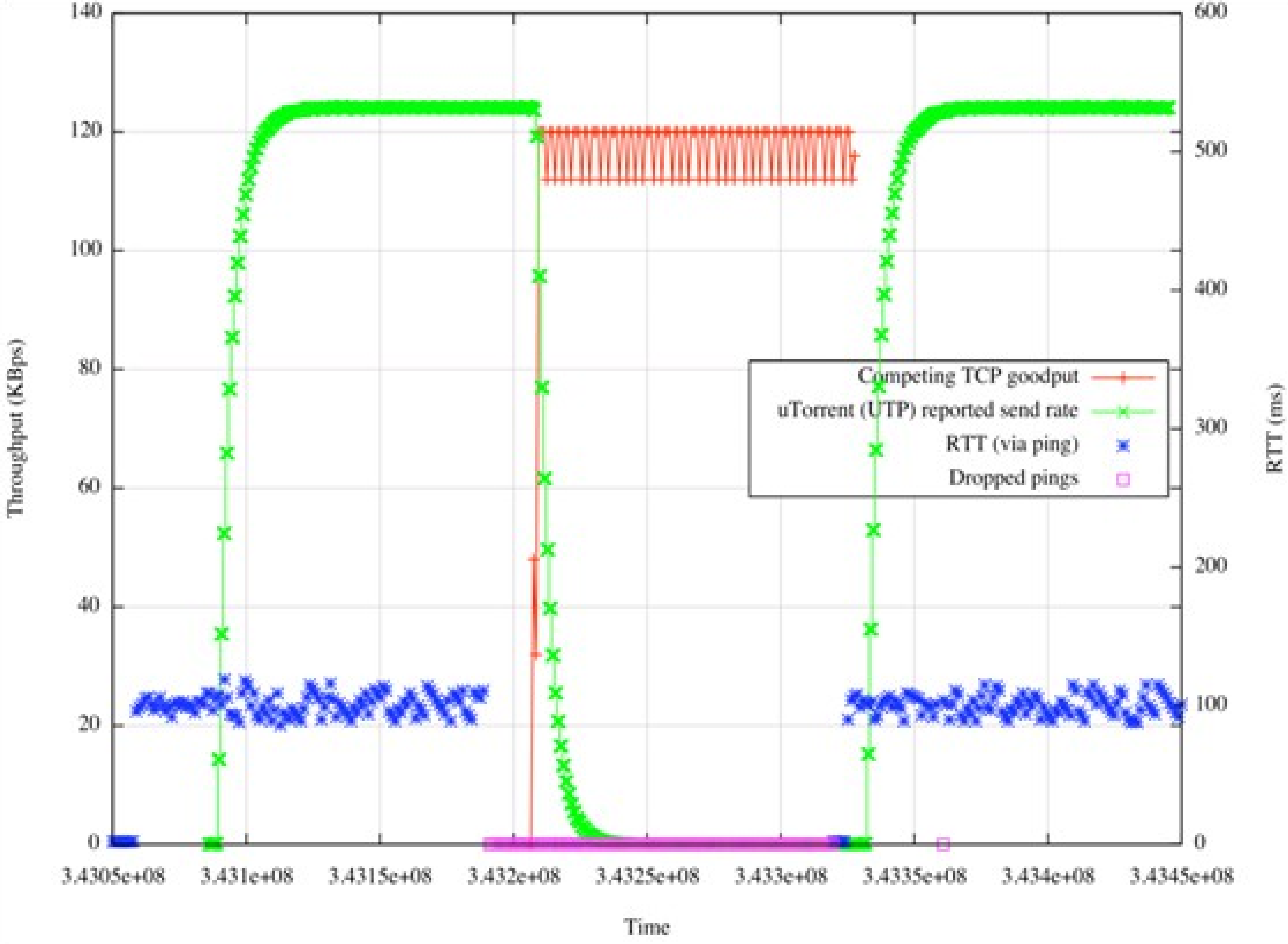
- cwnd += GAIN * off_target / cwnd, increase or decrease is directly proportional to off_target
  - On start-up, allows faster ramp up
  - As queuing_delay approaches TARGET the changes are less significant preventing excessive oscillation
- On loss, behave like TCP
  - Halve congestion window

uTP Congestion Window:

- Worst case: degenerate into TCP
  - Critical to avoid a user of LEDBAT from maliciously taking an unfair share of the bandwidth
  - Worst case is somebody maliciously sets their TARGET to be ∞ or alters timestamps to show a queuing_delay of 0, this means:
    - cwnd += GAIN * off_target / cwnd
    - GAIN = (1 / ∞), off_target = (∞ - 0)
    - cwnd += (1 / ∞) * (∞ - 0) / cwnd, therefore:
      - cwnd += 1 / cwnd per ACK arrival, which is exactly the behavior of TCP when operating in congestion avoidance

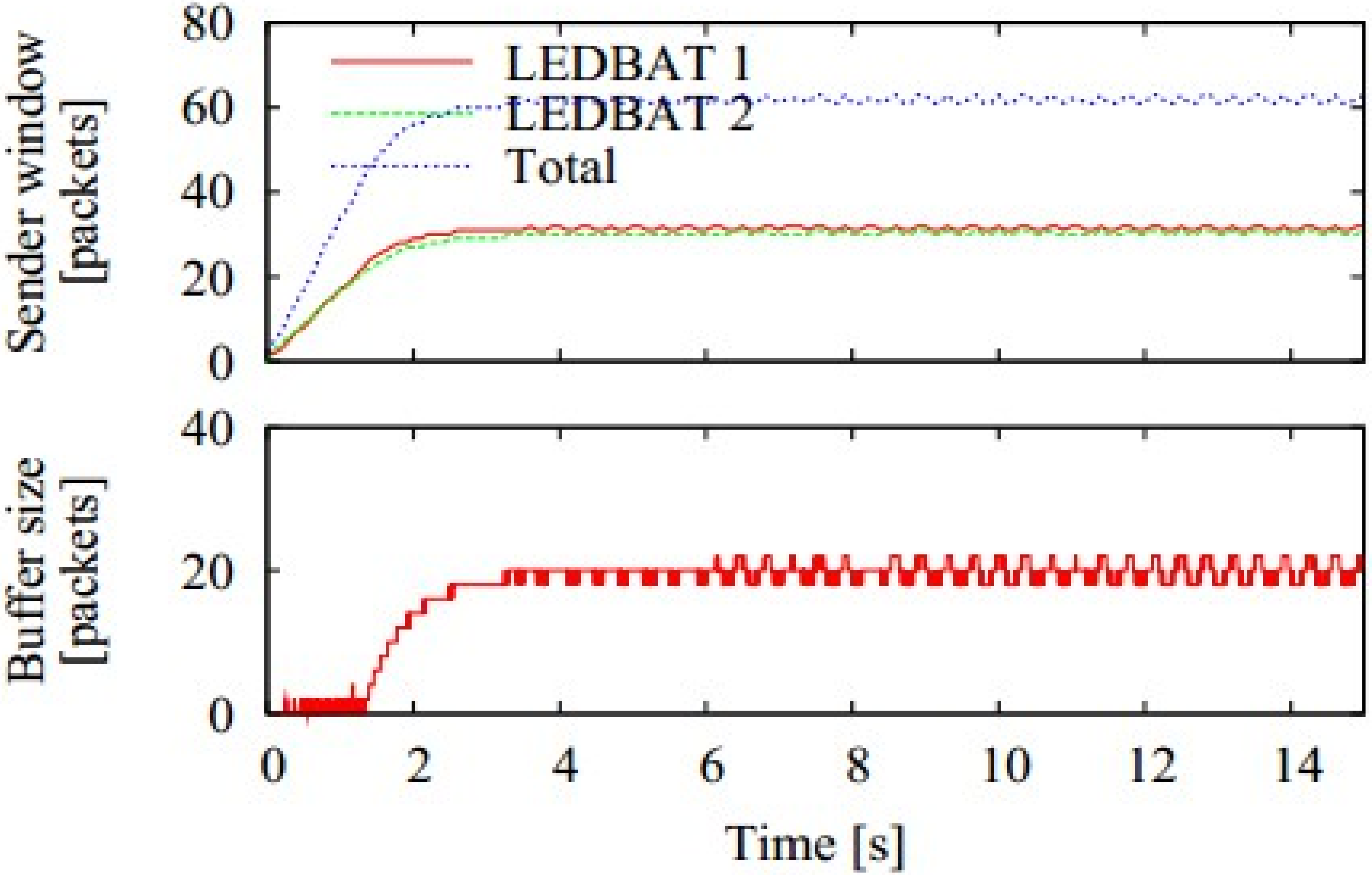
LEDBAT vs. TCP:



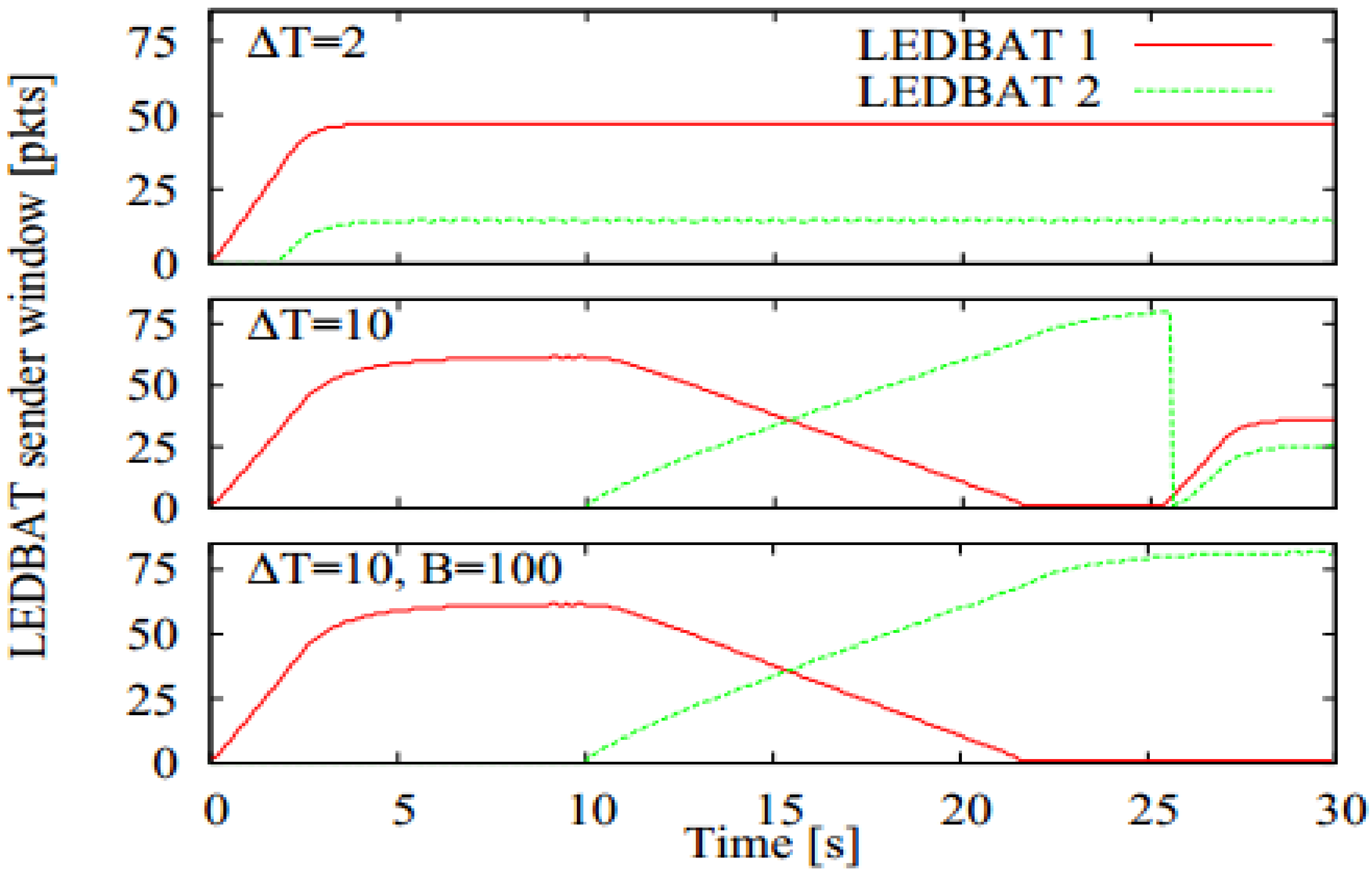


Competing TCP goodput
uTorrent (UTP) reported send rate
RTT (via ping)
Dropped pings
Throughput (KBps)
RTT (ms)
Time
0
20
40
60
80
100
120
140
0
100
200
300
400
500
600
3.4305e+08
3.431e+08
3.4315e+08
3.432e+08
3.4325e+08
3.433e+08
3.4335e+08
3.434e+08
3.4345e+08

LEDBAT vs. LEDBAT:



Late-comer Phenomenon:



Late-comer Advantage:

- LEDBAT_1 maintains a queue size of TARGET
- LEDBAT_2 incorrectly calculates the base delay
- LEDBAT_1 detects increased delay caused by LEDBAT_2 and LEDBAT_1 decreases its sending rate
- LEDBAT_2 does not detect an increase in queuing_delay because LEDBAT_1 is decreasing its sending rate so LEDBAT_2 increases its sending rate

Late-comer Disadvantage:

- LEDBAT_1 has not yet caused a queue to build in the bottleneck router
- LEDBAT_2 starts transmitting when queuing_delay calculates to 0
- As LEDBAT_1 has started first, its sending rate will be greater than LEDBAT_2 causing LEDBAT_1 to receive a larger percentage of the bandwidth when the TARGET queuing_delay is reached