BurstRadar

Practical Real-time Microburst Monitoring for Datacenter Networks

Raj Joshi[1], **Ting Qu**[2], Mun Choon Chan[1], Ben Leong[1], Boon Thau Loo[3]

NUS National University of Singapore [1]

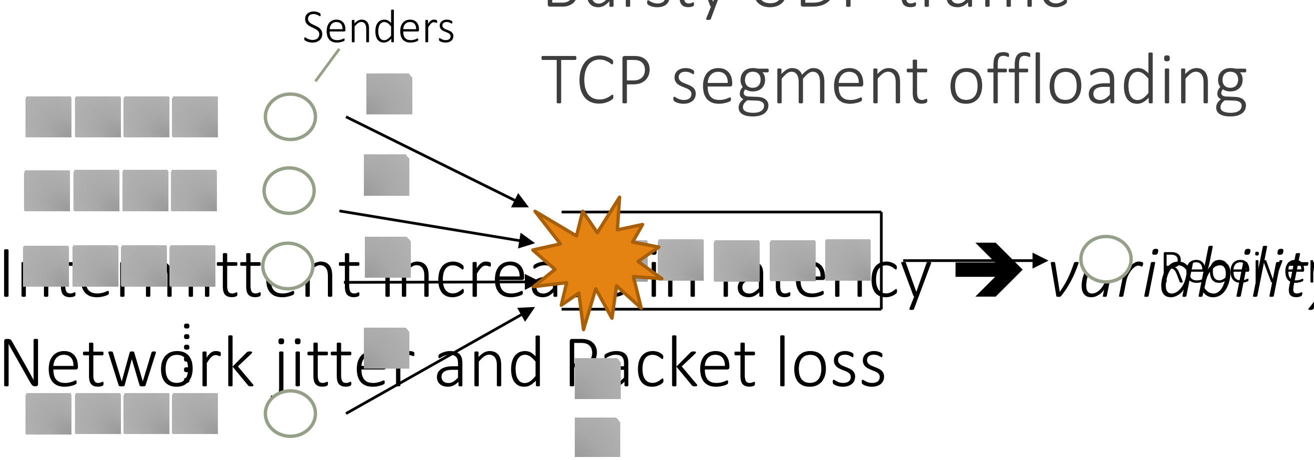
国防科學技術大學 National University of Defense Technology [2]

Penn University of Pennsylvania [3]

Microbursts (µbursts)

Events of intermittent congestion lasting 10’s or 100’s of µs

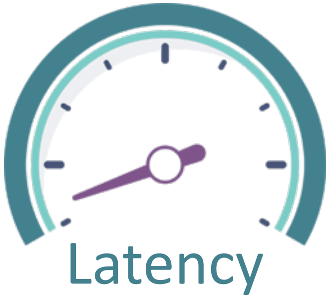
- Common Causes: TCP Incast
  Bursty UDP traffic
  TCP segment offloading



- Intermittent increase in latency ➔ *variability*
- Network jitter and Packet loss

Modern Datacenter Networks

> 10 Gbps



10’s of µs

Small amounts of queueing (microbursts):

Performance

Modern Datacenter Networks

Detect the occurrence of µbursts
&
identify the contributing flows!

Detecting & characterizing µbursts is hard

Measurement study from FB's datacenter

- Last for less than 200 µs
- Occur unpredictably

Traditional sampling-based techniques

NetFlow

sFlow®

- Cannot even detect microbursts

Commercial Solutions

CISCO

ARISTA

- Can detect the occurrence of microbursts
- Provide no information about the cause

New Advancements

Programmable dataplanes and dataplane telemetry

In-band Telemetry (INT)

- Adds queuing telemetry info into packets & exports it to monitoring servers from the last-hop switches

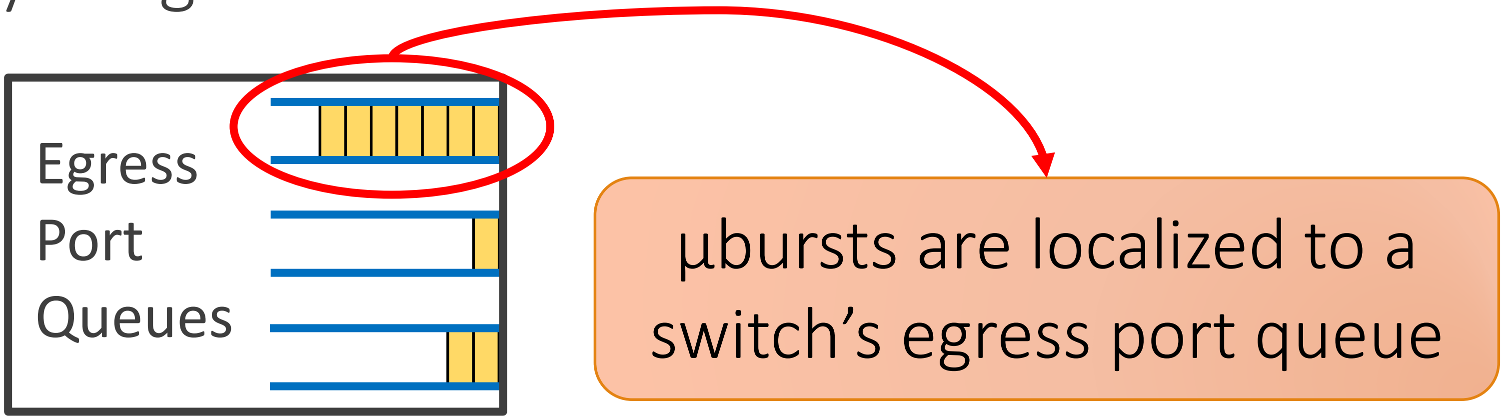
Challenges: Effective & real-time monitoring

Using INT to detect μbursts is wasteful

- Need to capture and export/process telemetry data for all packets
  - Since μbursts are unpredictable
- Expensive computation and delay
  - Correlate monitoring data from different points in the network

Solution: Out-band

Key Insight:



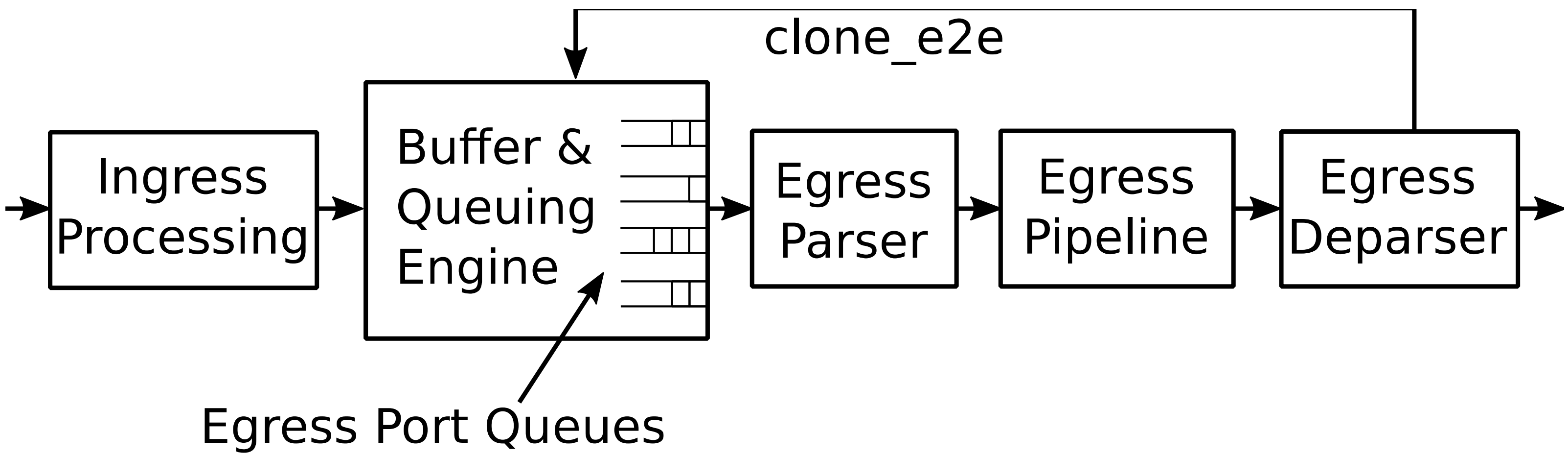
Switch's Queuing Engine

Key Idea:

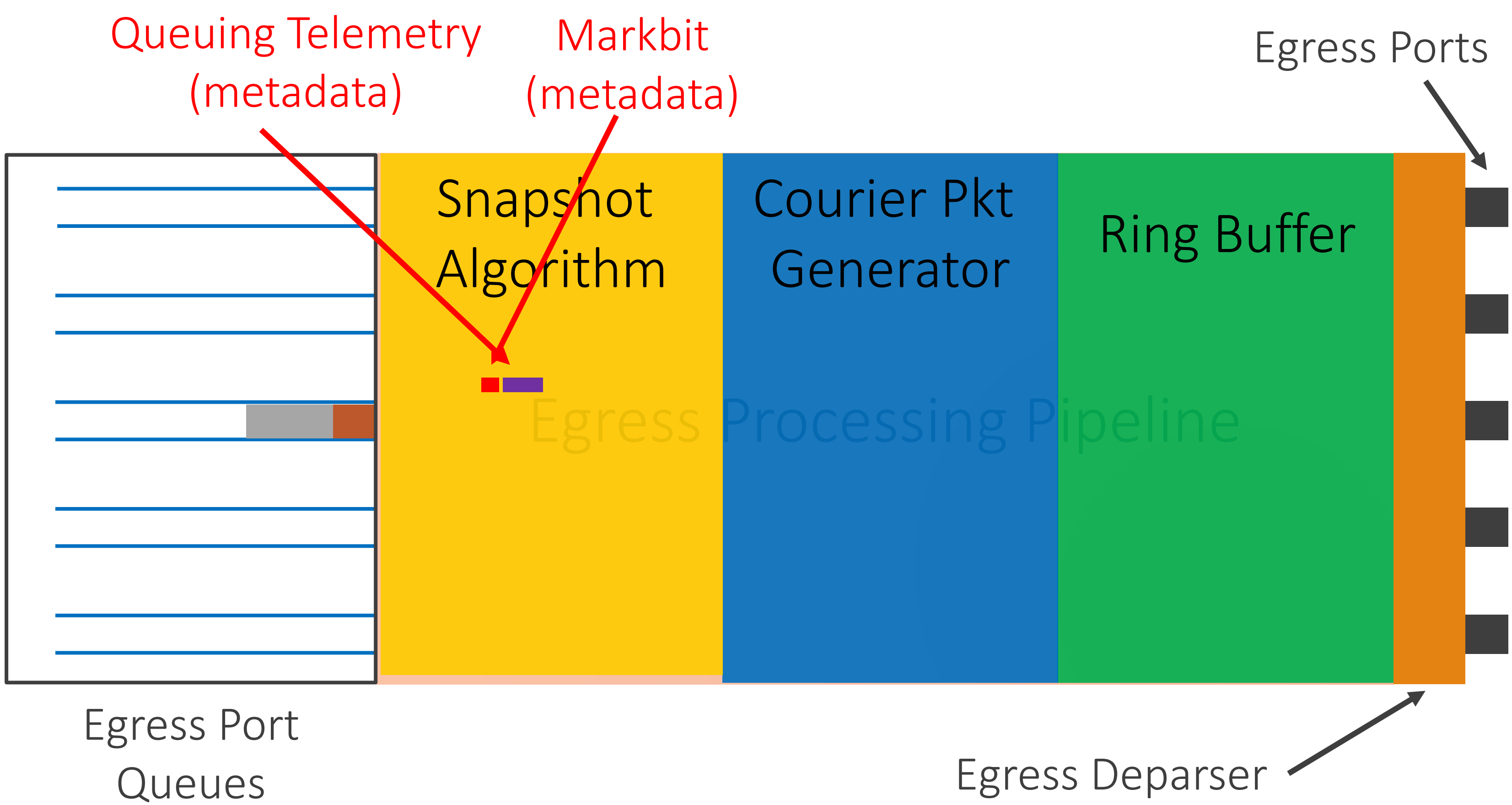
- We can detect the microburst directly on the switch where it happens

Solution: egress pipeline

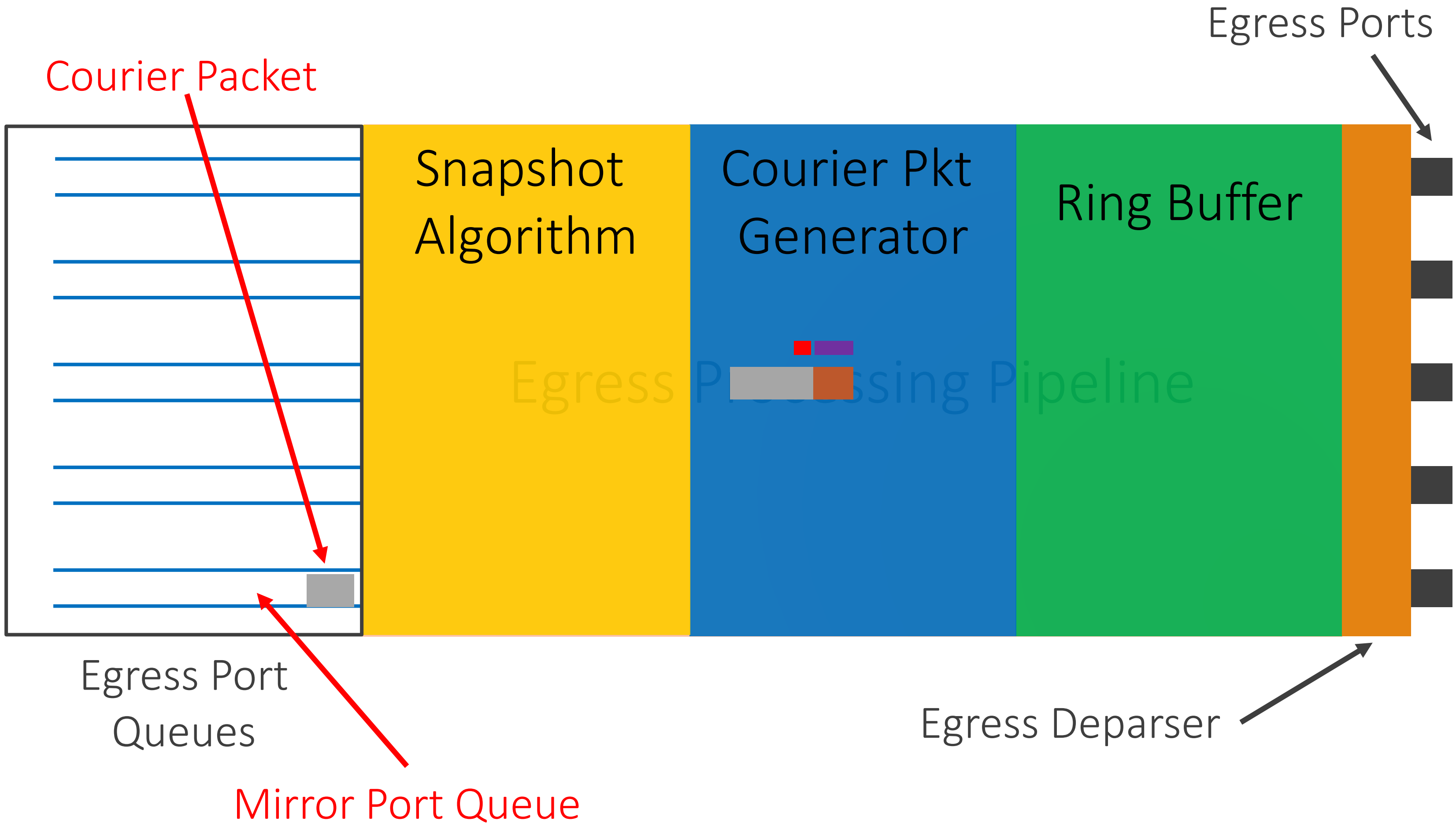
- Switching ASIC's "Buffer and Queuing Engine" (BQE) does not provide any support to peek into the contents of any queue



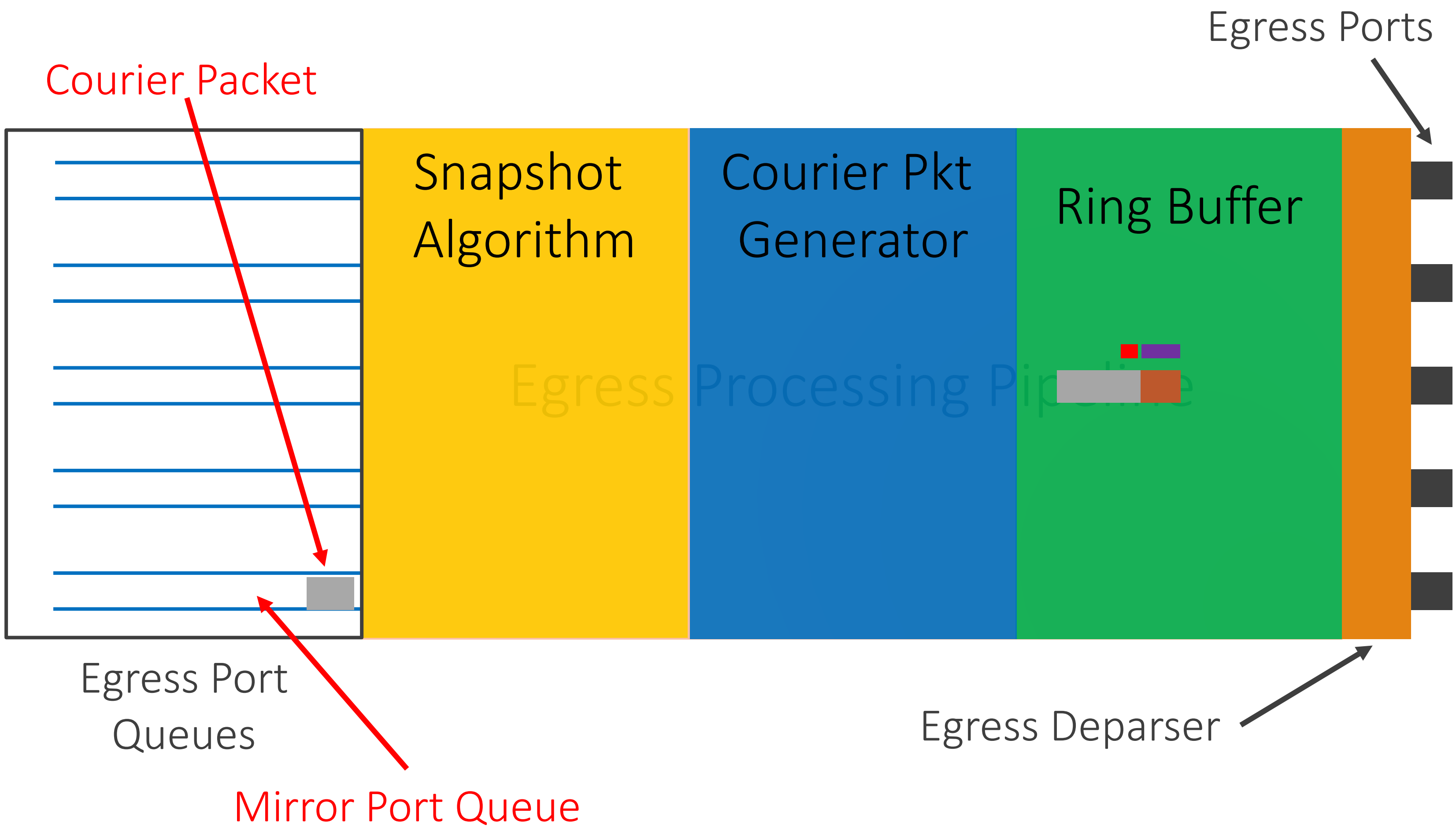
BurstRadar Overview



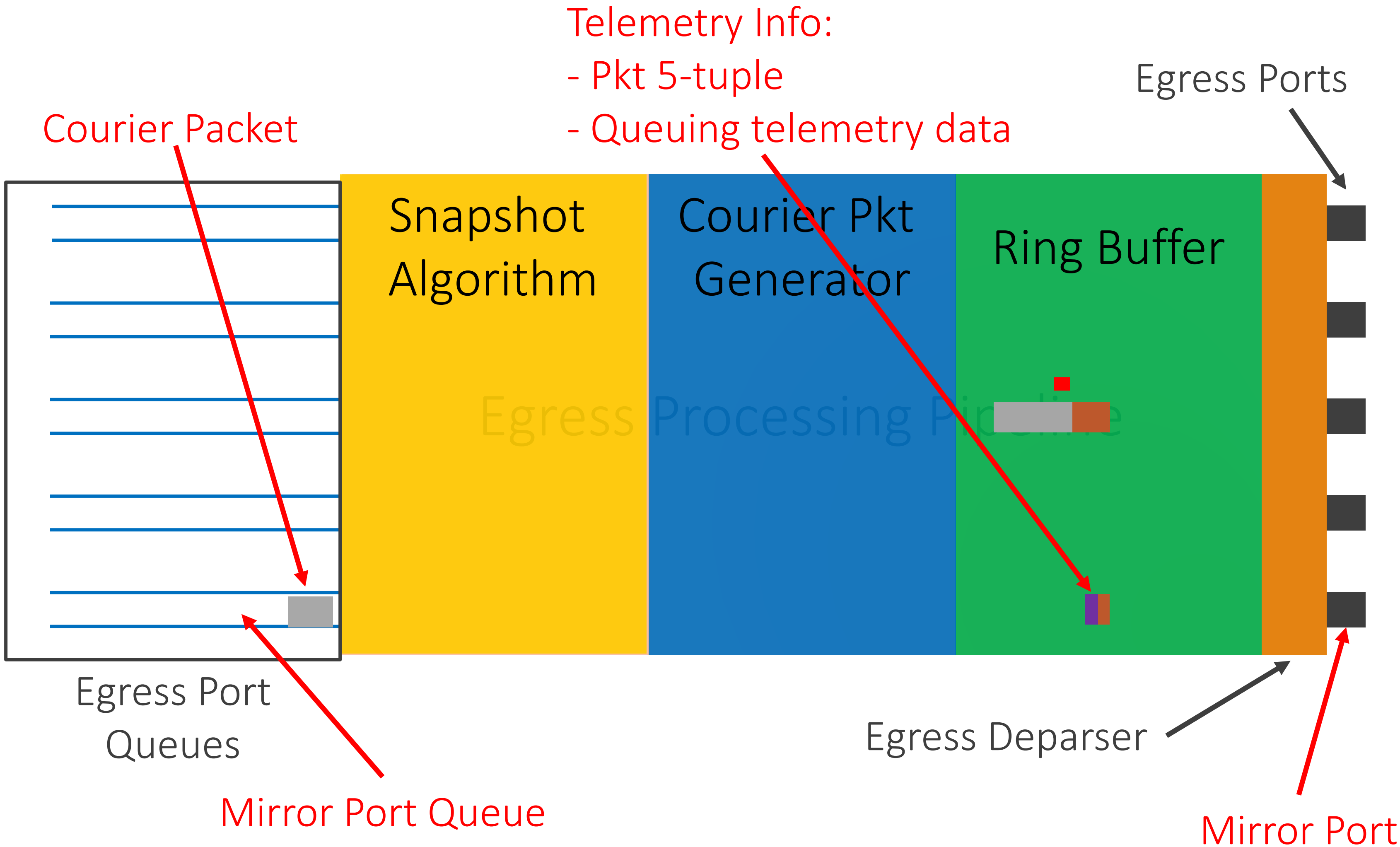
BurstRadar Overview



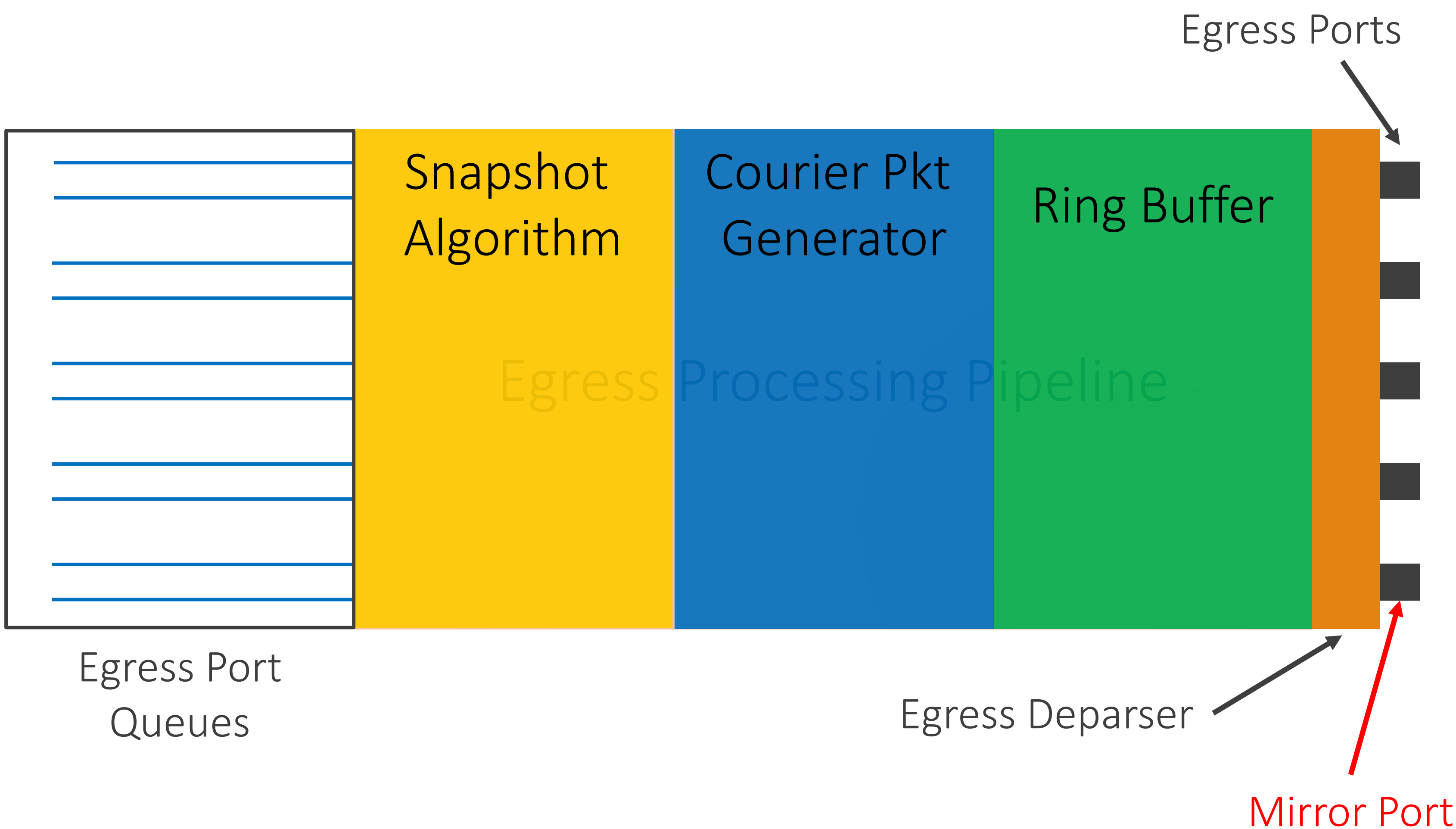
BurstRadar Overview

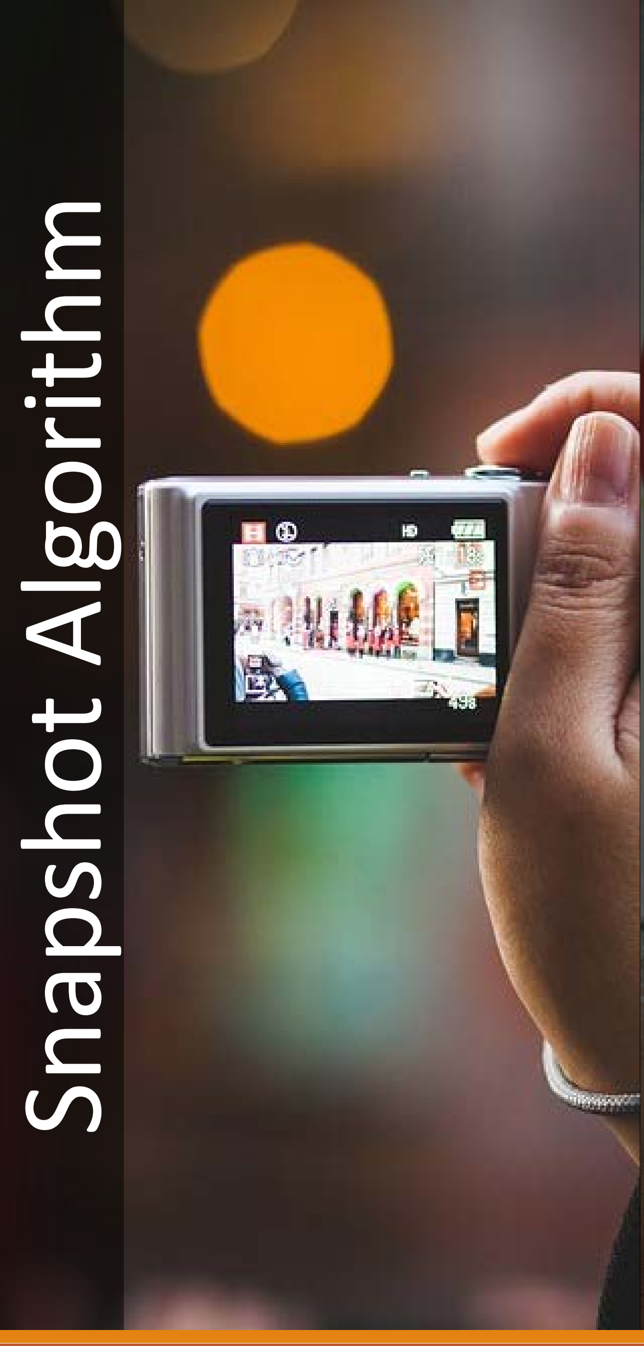


BurstRadar Overview



BurstRadar Overview




Snapshot Algorithm


Courier Pkt Generator
पत्र
letters
680020-8
भारतीय डाक
India Post
1355


Ring Buffer

Snapshot Algorithm

"Snapshot" the telemetry info of *only* the packets involved in µbursts

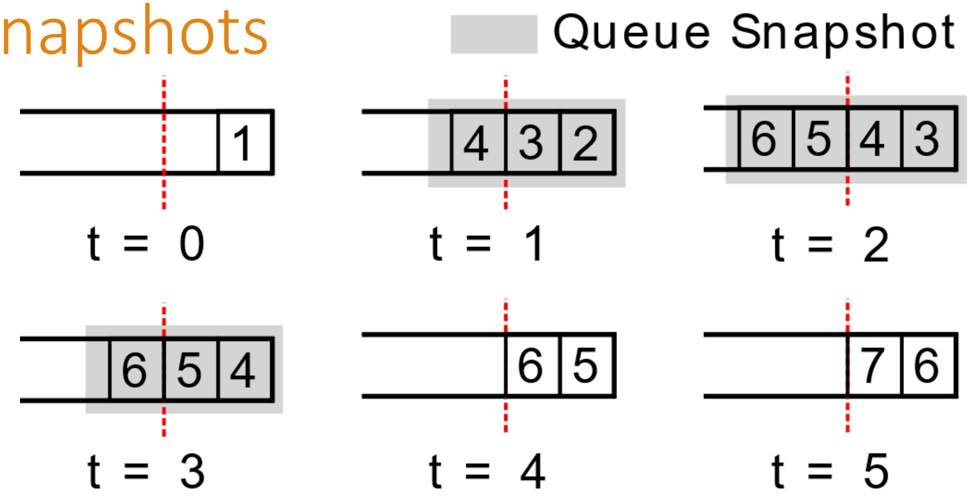
Telemetry info: 5-tuple (packet header)
ingress/egress timestamps
enqueue/dequeue queue depths
(metadata)

Snapshot Algorithm

Latency Increase Threshold

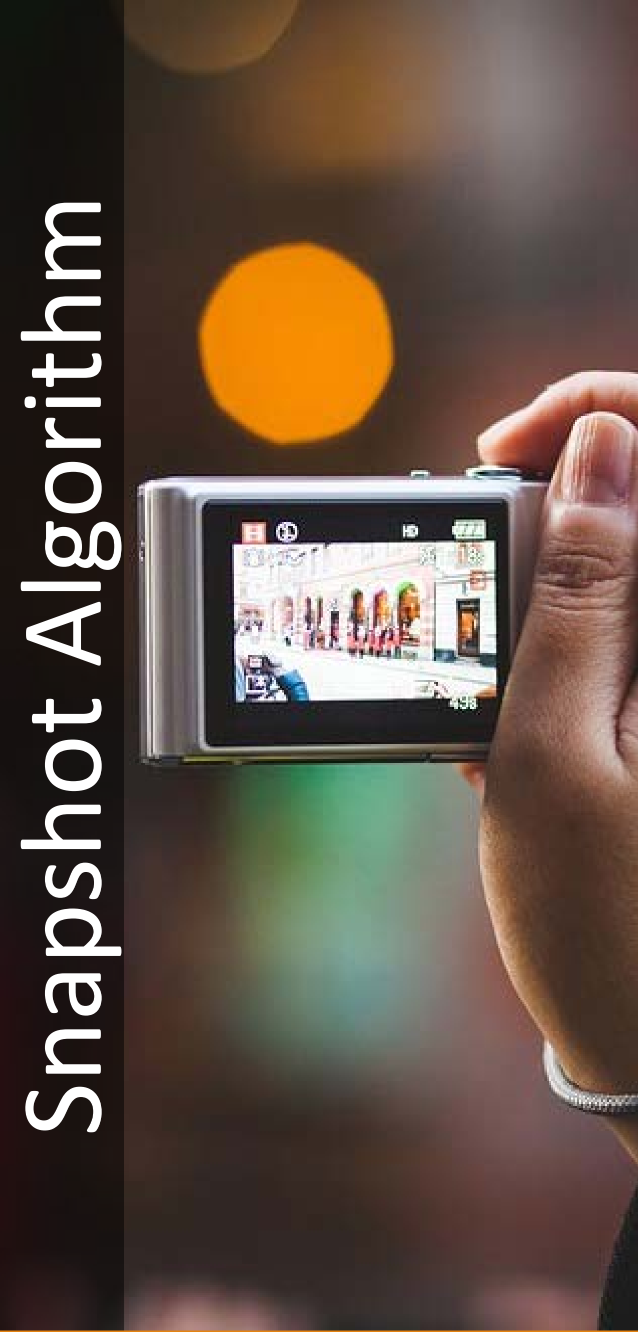
- Eg. RTT = 50µs, threshold = 30%, i.e., maximum delay = 15µs

Queue Snapshots



Snapshot Algorithm

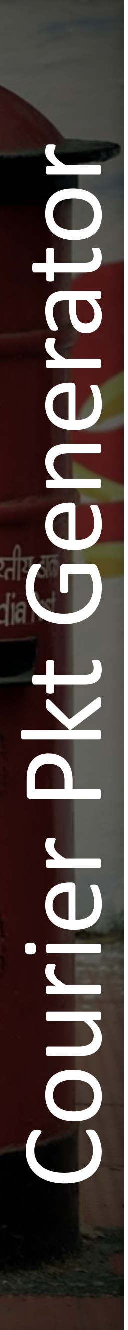
- Each packet reports deqQdepth
- if deqQdepth > threshold, then mark pkt → snapshot
  - Track remaining bytes in the queue
- elif tracked bytes still remaining then mark pkt → snapshot


Snapshot Algorithm


Courier Pkt Generator
पत्र
letters
680020-8
भारतीय डाक
India Post
1355


Ring Buffer

Courier Pkt Generator



"Courier" Packets transport the telemetry info via the switch's mirror port (out-of-band)

All the data stays *together* → Avoids the expensive correlation on the monitoring servers

Courier Pkt Generator

Each marked packet → generate new courier packet

clone egress to egress, clone_e2e

- Copy of the exiting marked packet
- Place it in the egress queue of the mirror port

Snapshot Algorithm

Courier Pkt Generator

Ring Buffer

Ring Buffer

"Ring Buffer" temporarily stores the telemetry info of marked packets until they can be copied into the courier packets.

Evaluation

Evaluation Setup

Hardware Testbed

BurstRadar Prototype

Send/Receive µburst Traffic

- About 550 lines of p4 code

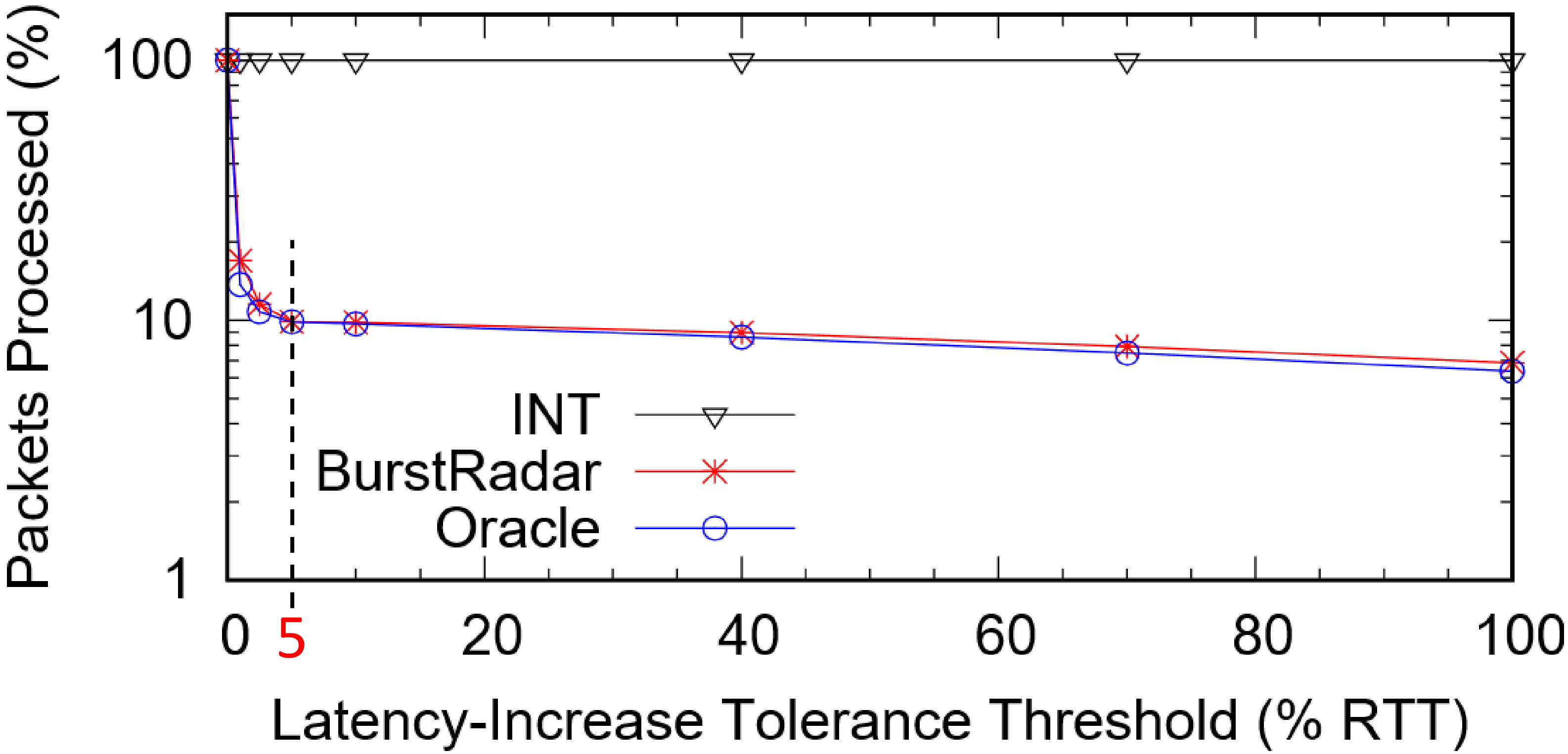
Generated µburst Traffic Traces

- µbursts data for “web” and “cache” traffic [IMC ‘17]

Compare BurstRadar against

- In-band Telemetry (INT) → dataplane-based solution
- “Oracle” Algorithm → ground truth (exact pkts in µbursts)

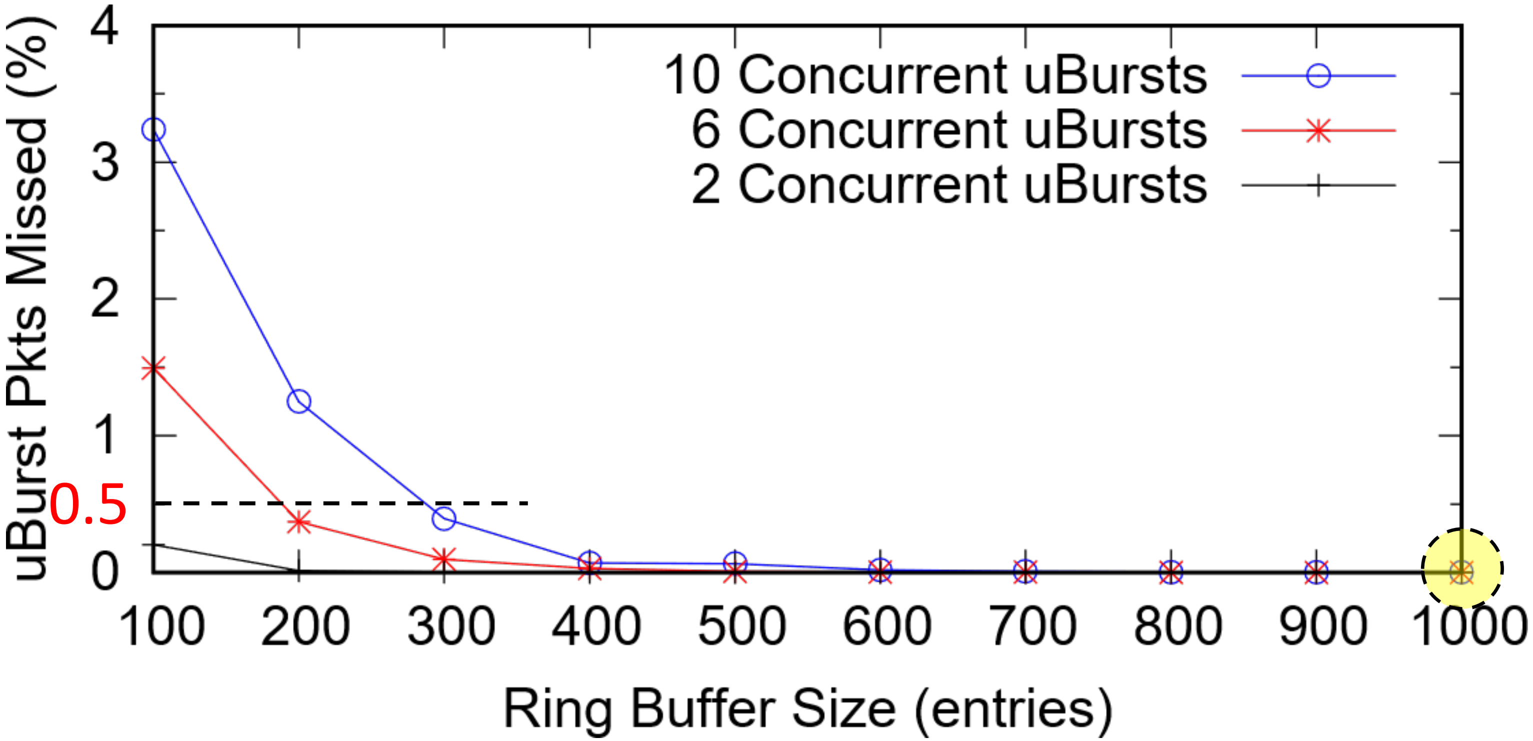
Efficiency



5% RTT → 10 times less packets compared to INT

Note: 5% RTT ≈ 1.25µs of queuing @10Gbps in our testbed

Handling Concurrent µbursts



300 entries (8.7KB SRAM)→ 10 concurrent µbursts (< 0.5%)

Note: 1000 entries (29KB SRAM) fully handle 10 concurrent µbursts

Dataplane Resource Utilization

Tofino Resource Utilization (Ring Buffer = 1000 entries)

Resource	switch.p4*	BurstRadar	Combined
Match Crossbar	50.13%	3.39%	53.52%
Hash Bits	32.35%	4.83%	37.18%
SRAM	29.79%	4.06%	33.85%
TCAM	28.47%	0.69%	29.16%
VLIW Actions	34.64%	4.69%	39.33%
Stateful ALUs	15.63%	12.5%	28.13%

* resource utilization of a fully-featured datacenter ToR switch

Very little resources → combined with switch.p4

Conclusion

- Microburst monitoring is important

  -High impact on performance

- BurstRadar can detect and identify Microbursts effectively and continuously

  -Capture and report the telemetry information of only the packets involved in microbursts

- BurstRadar demonstrates that modern programmable ASICs have made it practical to detect and characterize microbursts at multi-gigabit line rates in high-speed datacenter networks.


Thank
You
Vinaka
köszönöm
Kiitos
Maake
Asante
Shukria
Dhanyavadagalu
감사합니다
Dankscheen
спасибо
Kam Sah Hammida
Manana
Dankon
Diolch i Chi
Terima Kasih
Matondo
Taiku
Dank Je
Mauruuru
Biyan
Tack
Blagodaram
Ngiyabonga
Dziekuje
Chokrane
Arigato
Grazie
Juspaxar
Mochchakkeram
Gracias
Bedankt
Dakujem
धन्यवाद
cảm ơn bạn
Khap
Paldies
Tingki
நன்றி
Grazas
Ua Tsaug Rau Koj
Mèsi
Gratias Tibi
Obrigado
Kia Ora
Kop Khun
Děkuji
Nirringrazzjak
ありがとう
Suksama
Nuwun
Hvala
Di Ou
Rahmat
谢谢
хвала
Welalin
Djiere Dieuf
Misaotra
Matur
Danke
Merci
Go Raibh Maith Agat
Eskerrik Asko
Tuke
Salamat
Najis