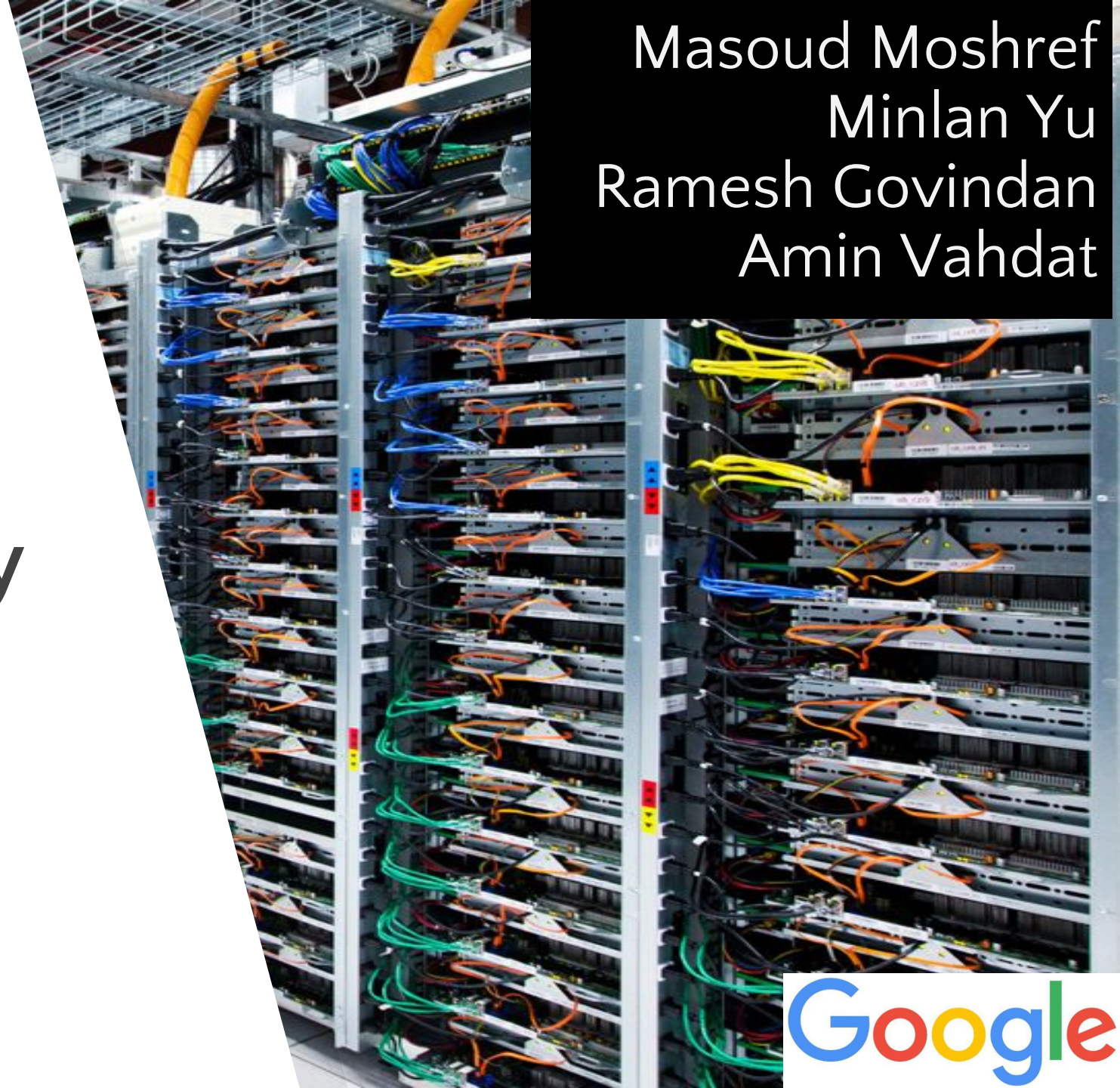
USC University of Southern California



Masoud Moshref
Minlan Yu
Ramesh Govindan
Amin Vahdat

Trumpet: Timely and Precise Triggers in Data Centers

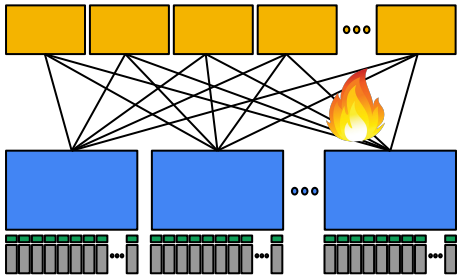
Google

Evolve or Die, SIGCOMM 2016

Long failure repair times in large networks

Human-in-the-loop failure assessment and repair

Humans in the Loop



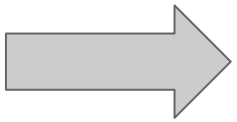
Detect

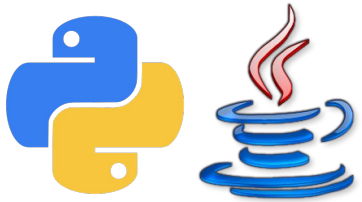
Examine ongoing network **events** to find possible **root-causes**

Operators examine dashboards

Our Focus

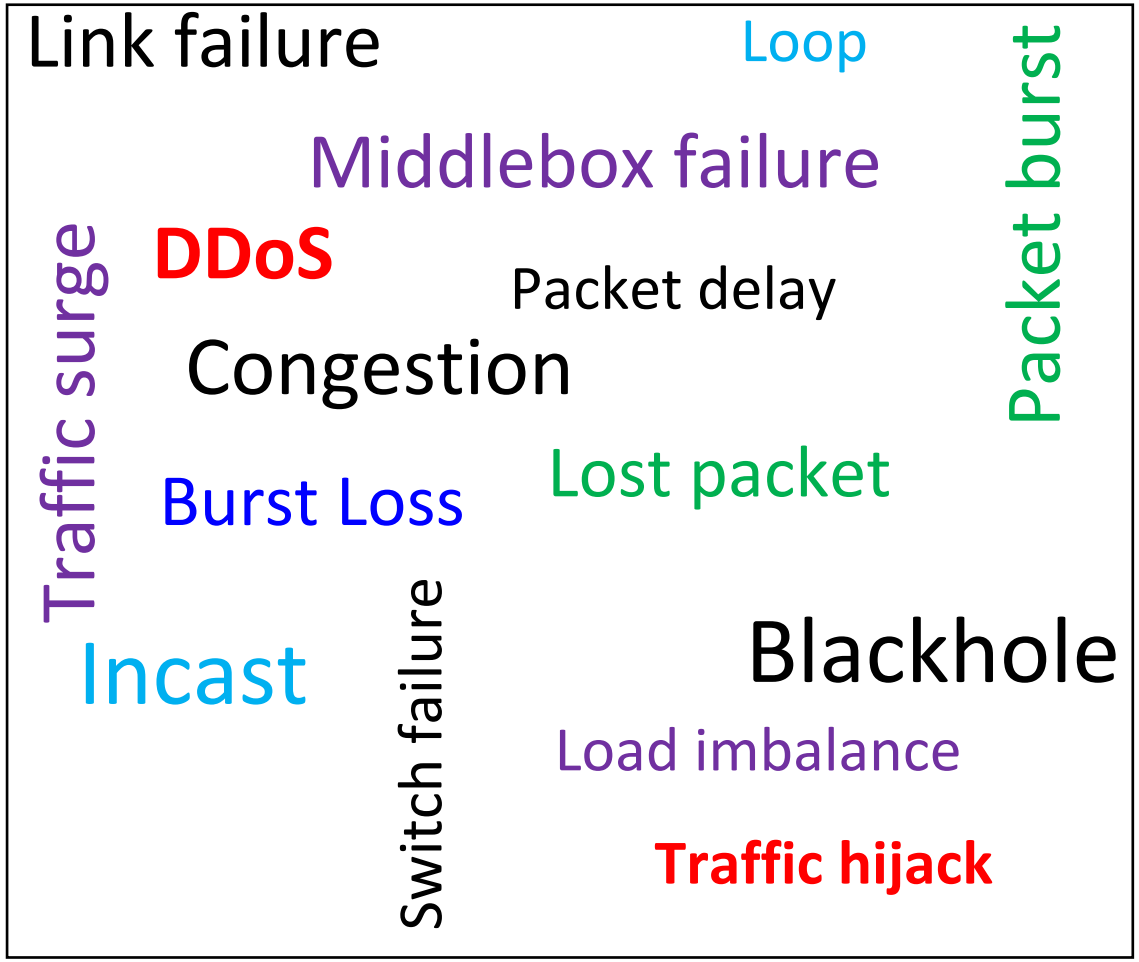
Humans





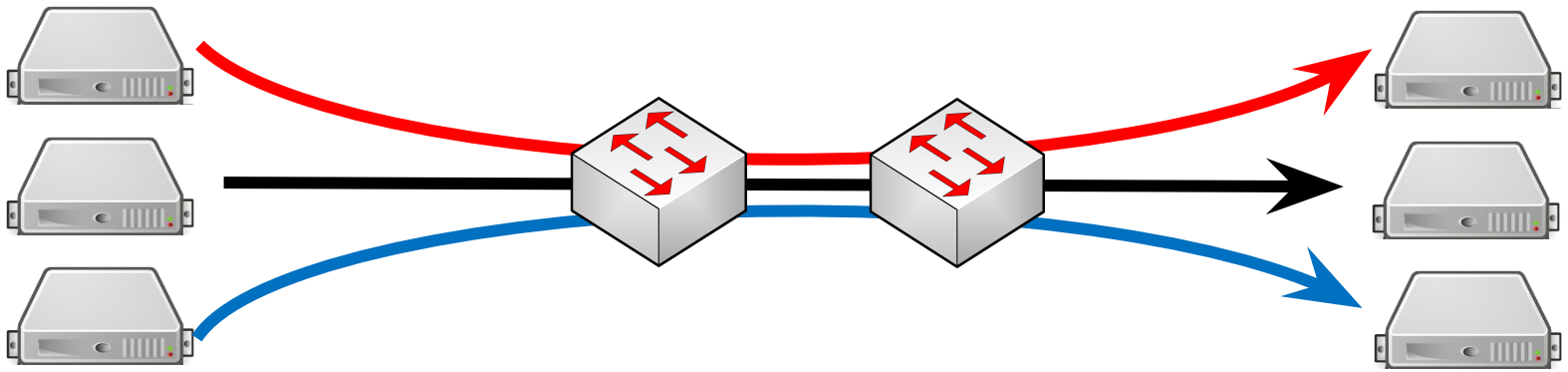
Programs

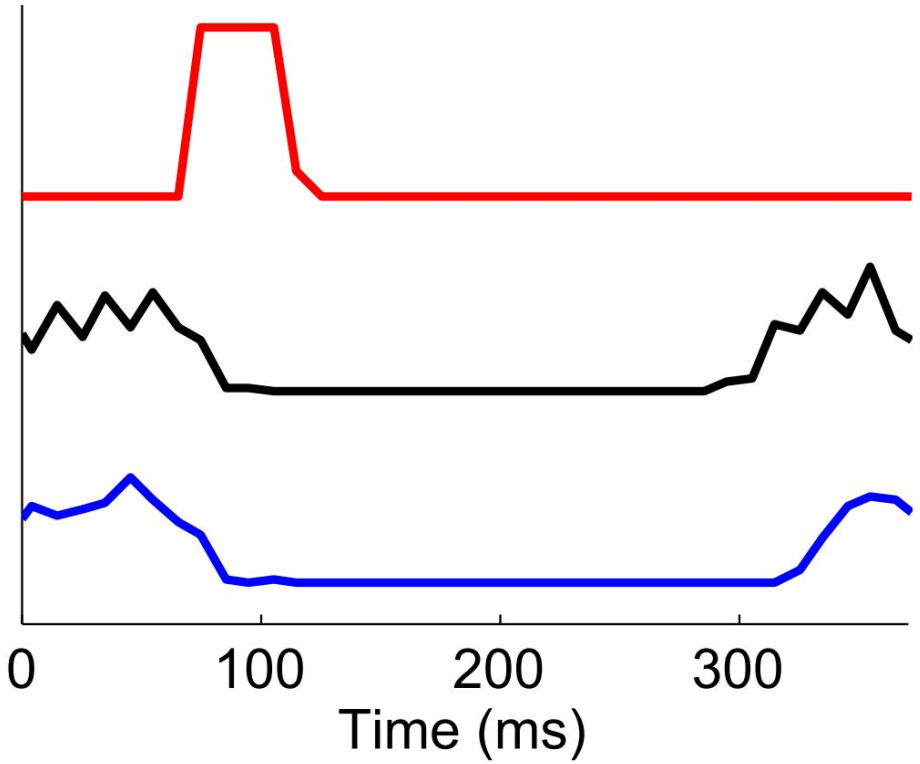
A framework for **programmed** detection of ***events*** in large datacenters



- Availability
- Performance
- Security

Detecting Transient Congestion





40 ms burst

Timeouts lasting several 100 ms

Fine Timescale Events

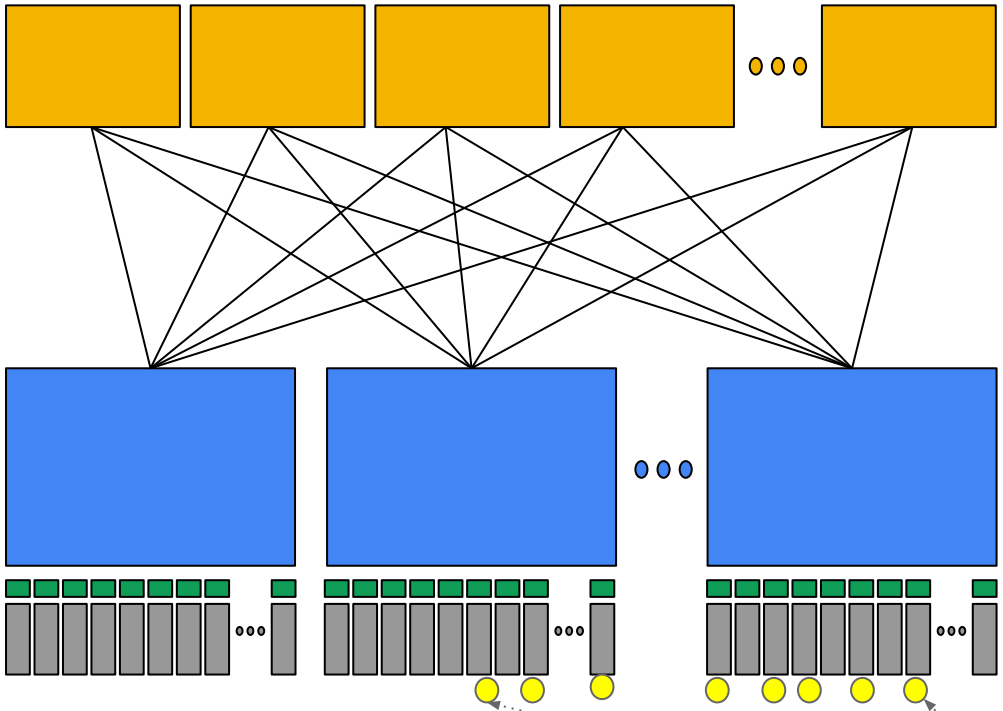
Alternative Approach

Detect

Dashboard data insufficent

Aggregated, often **sampled** measures of network health

Detecting Attack Onset

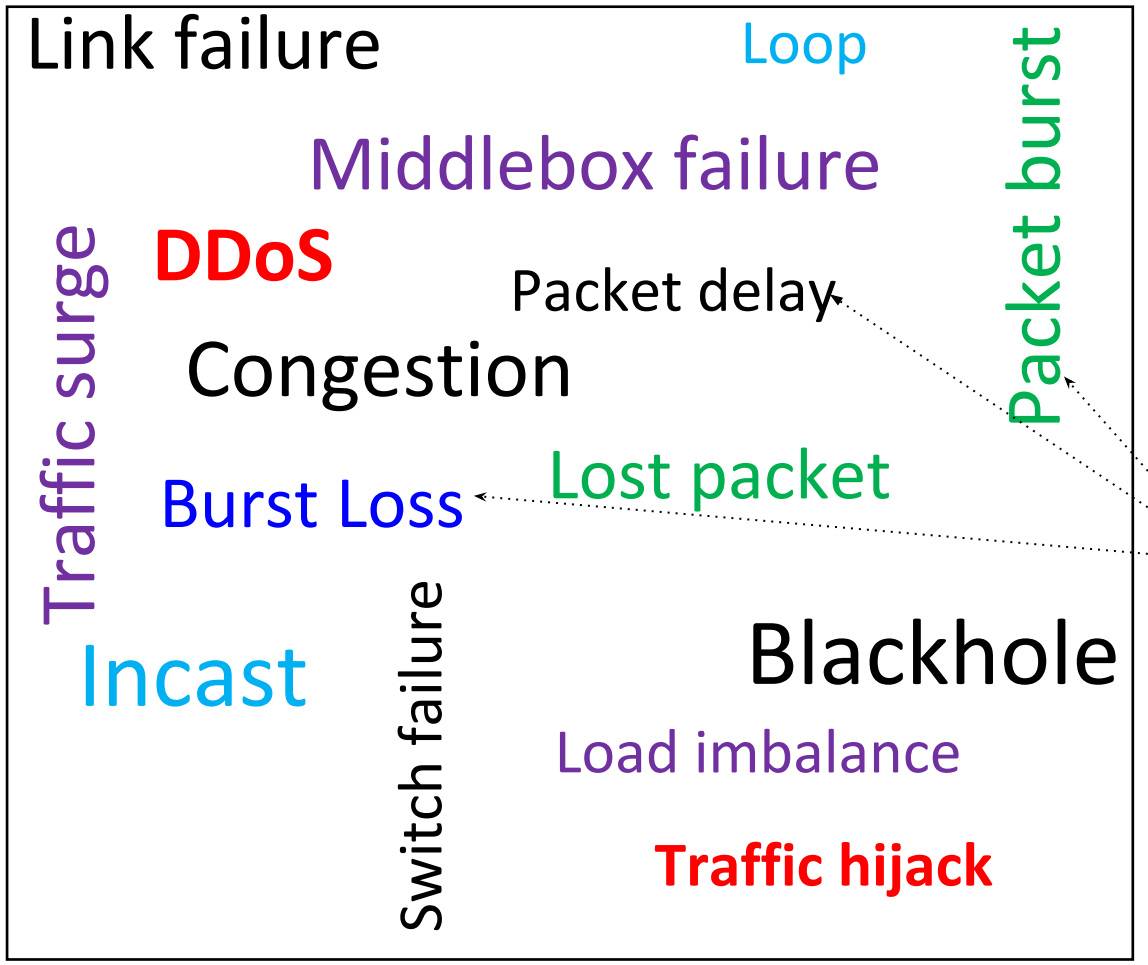


Data center

Did this **tenant** see a sudden increase in traffic over the last few **milliseconds**?

Fine Timescale Events

Inspect Every Packet



Some events may require **inspecting every packet**

Eventing Framework Requirements

Expressivity

★ Set of possible events not known *a priori*

Fine timescale eventing

★ Capture transient and onset events

Per-packet processing

★ Precise event determination

Because data centers will require high availability and high utilization

Where do we place eventing functionality?

A Key Architectural Question

Switches

NICs

Hosts

- Are programmable
- Have processing power for fine-time scale eventing
- Already inspect every packet

Trumpet *is a* ***host-based*** *eventing framework*

Research Questions

What eventing architecture permits programmability **and** visibility?

How can we achieve precise eventing at fine timescales?

What is the *performance envelope* of such an eventing framework?

Research Questions

What eventing architecture permits programmability **and** visibility?

How can we achieve precise eventing at fine timescales?

What is the performance envelope of such an eventing framework?

Trumpet has a logically centralized *event manager* that aggregates *local events* from per-host *packet monitors*

For each packet matching **Filter**

group by **Flow-granularity**

and report every **Time-interval**

each group that satisfies **Predicate**

Flow volumes, loss rate,
loss pattern (bursts), delay

Is there any flow sourced by a service that sees a burst of losses in a small interval?

For each packet matching **Service IP Prefix**

group by **5-tuple**

and report every **10ms**

any flow whose **sum (is_lost & is_burst) > 10%**

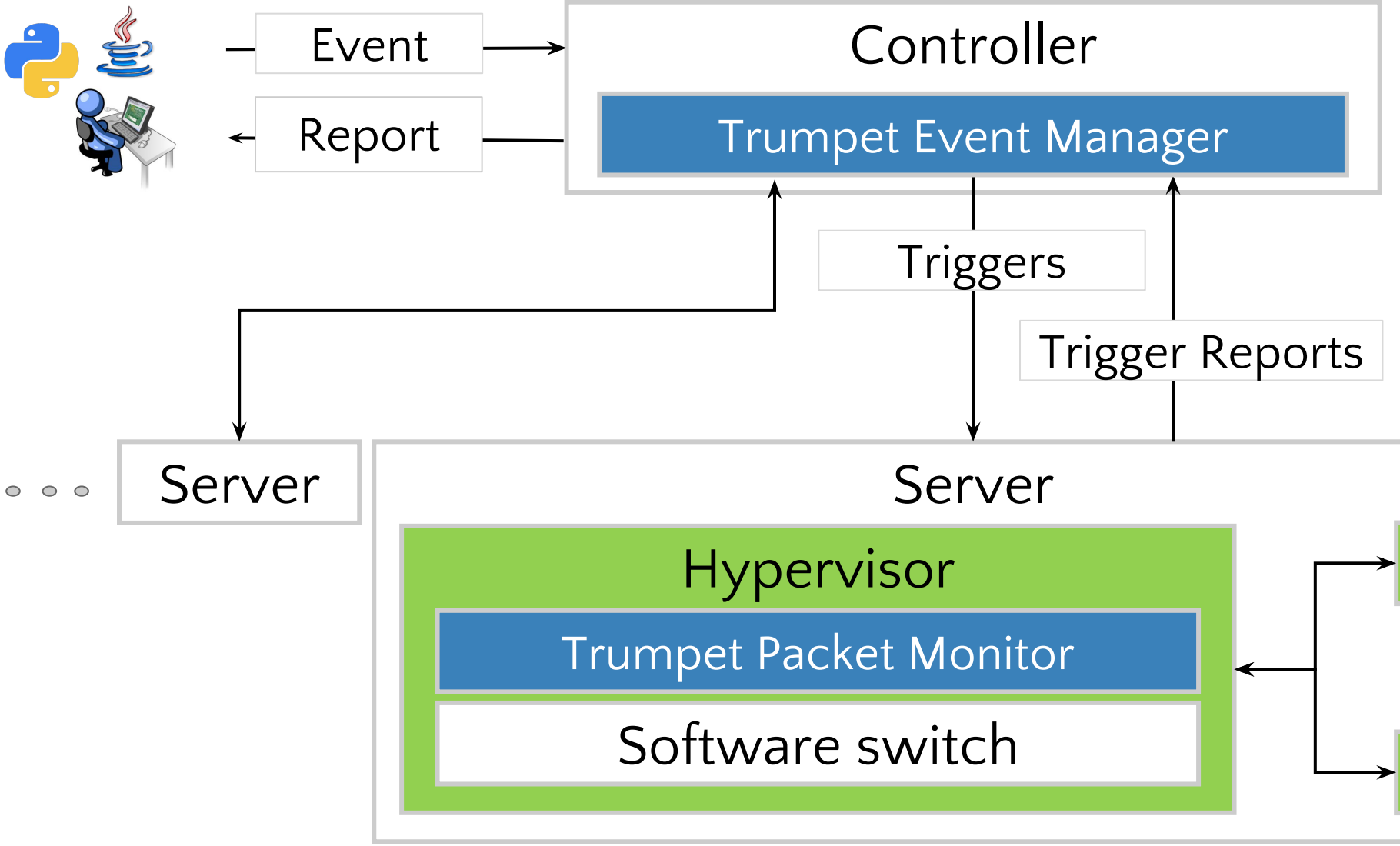
Is there a job in a cluster that sees abnormal traffic volumes in a small interval?

For each packet matching **Cluster IP Prefix and Port**

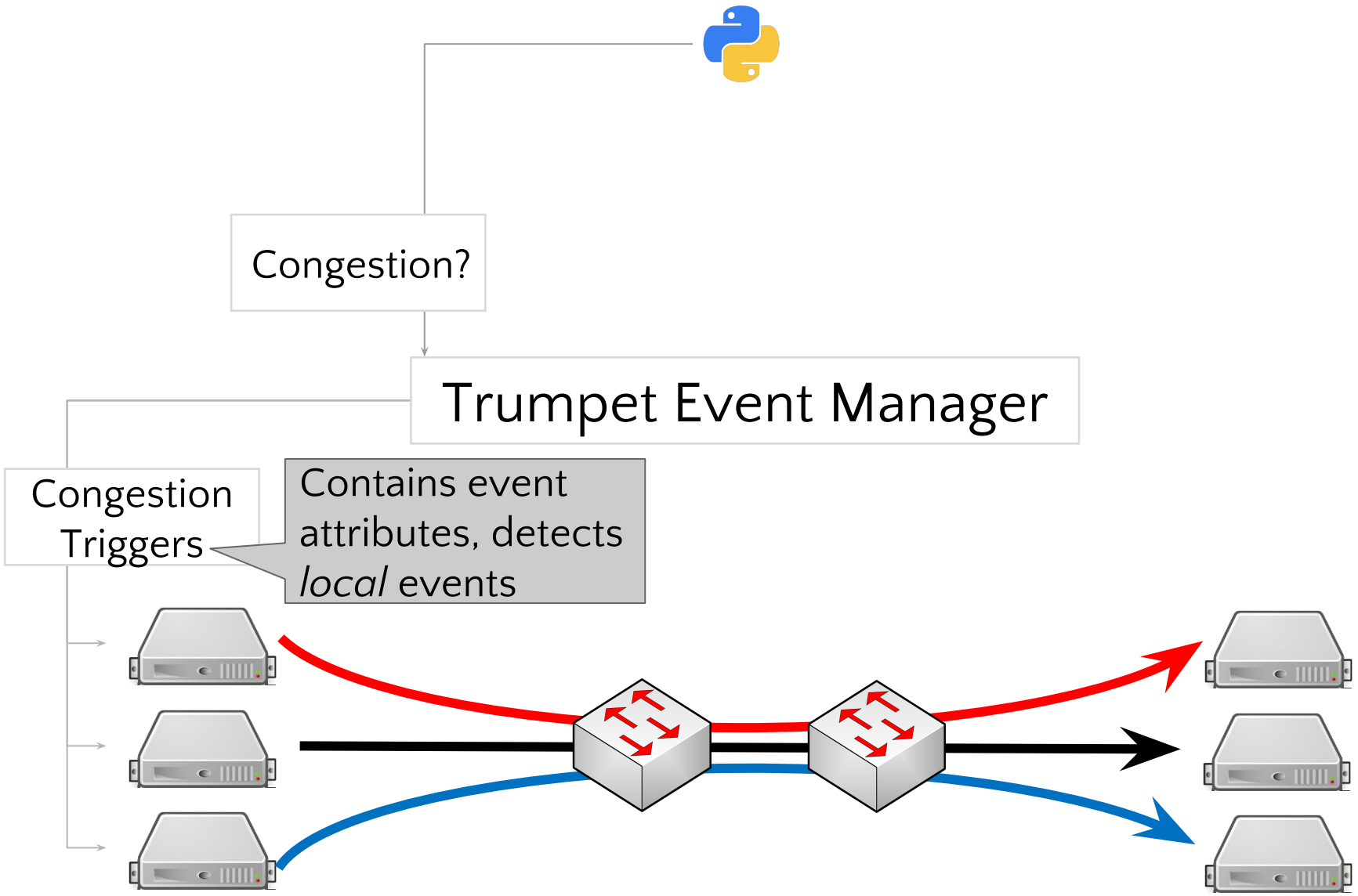
group by **Job IP Prefix**

and report every **10ms**

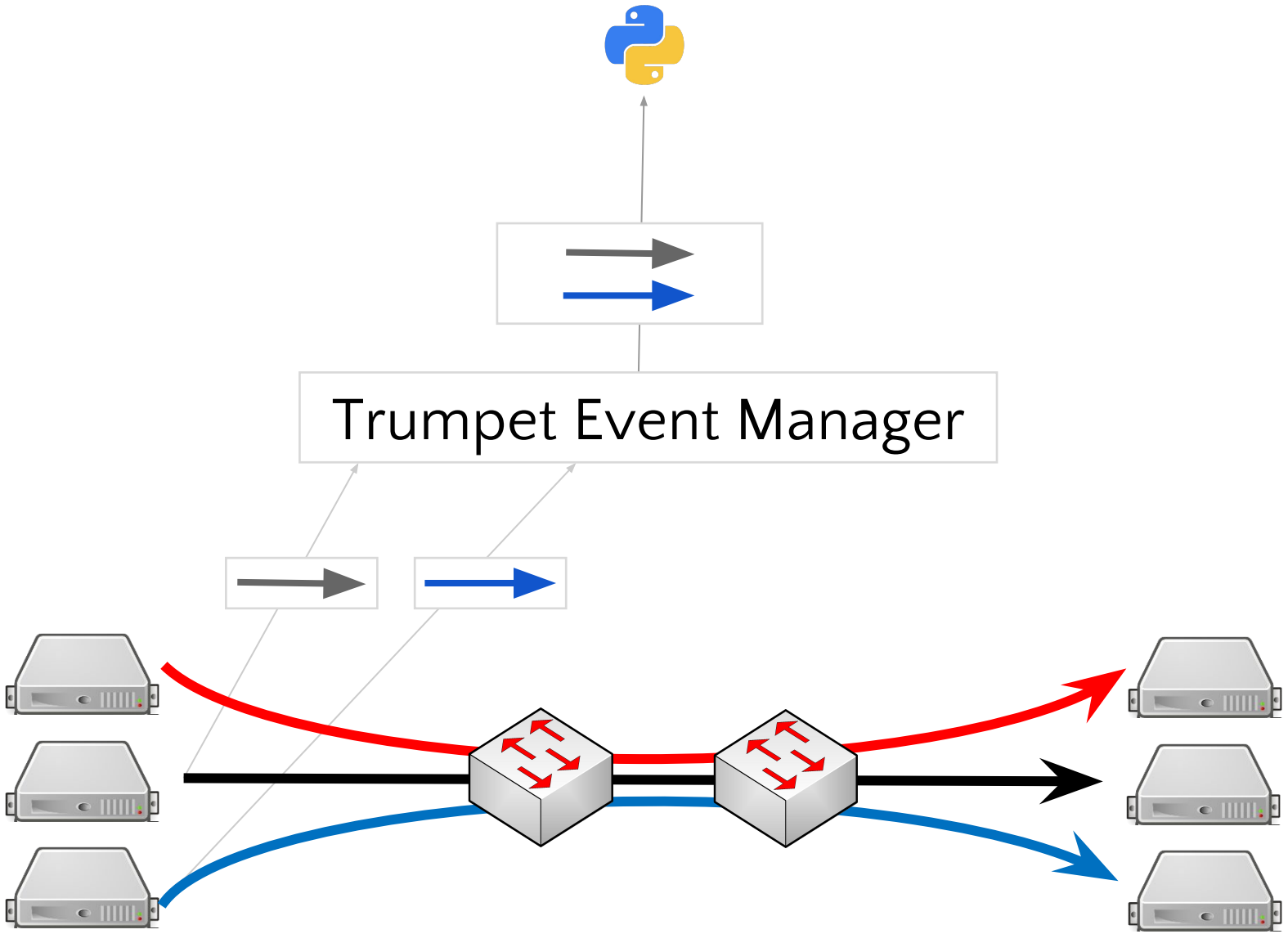
any job whose **sum (volume) > 100MB**


Event
Report
Controller
Trumpet Event Manager
Triggers
Trigger Reports
Server
Server
Hypervisor
Trumpet Packet Monitor
Software switch
VM
VM

Trumpet Event Manager

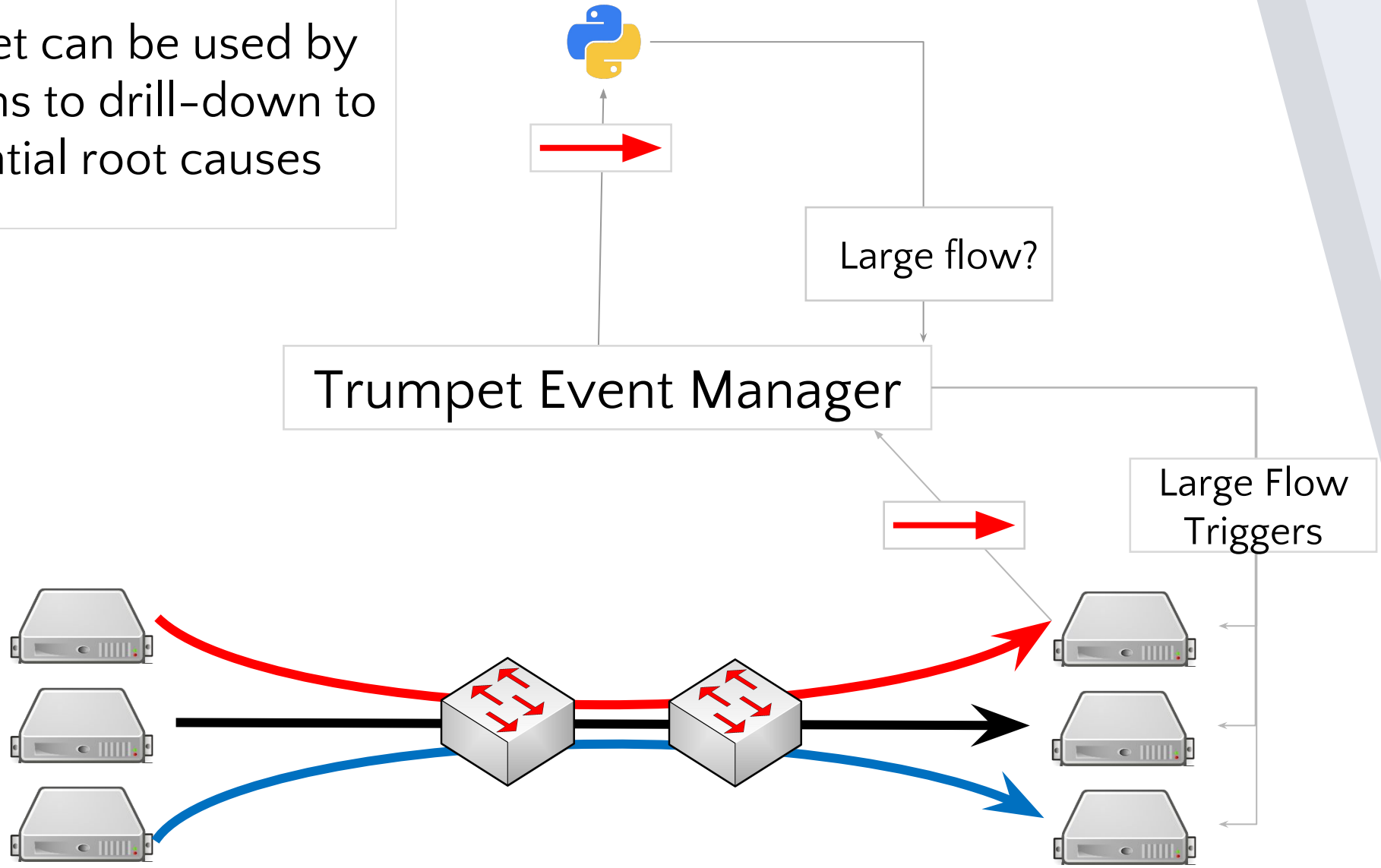


Trumpet Event Manager



Trumpet Event Manager

Trumpet can be used by programs to drill-down to potential root causes



Research Questions

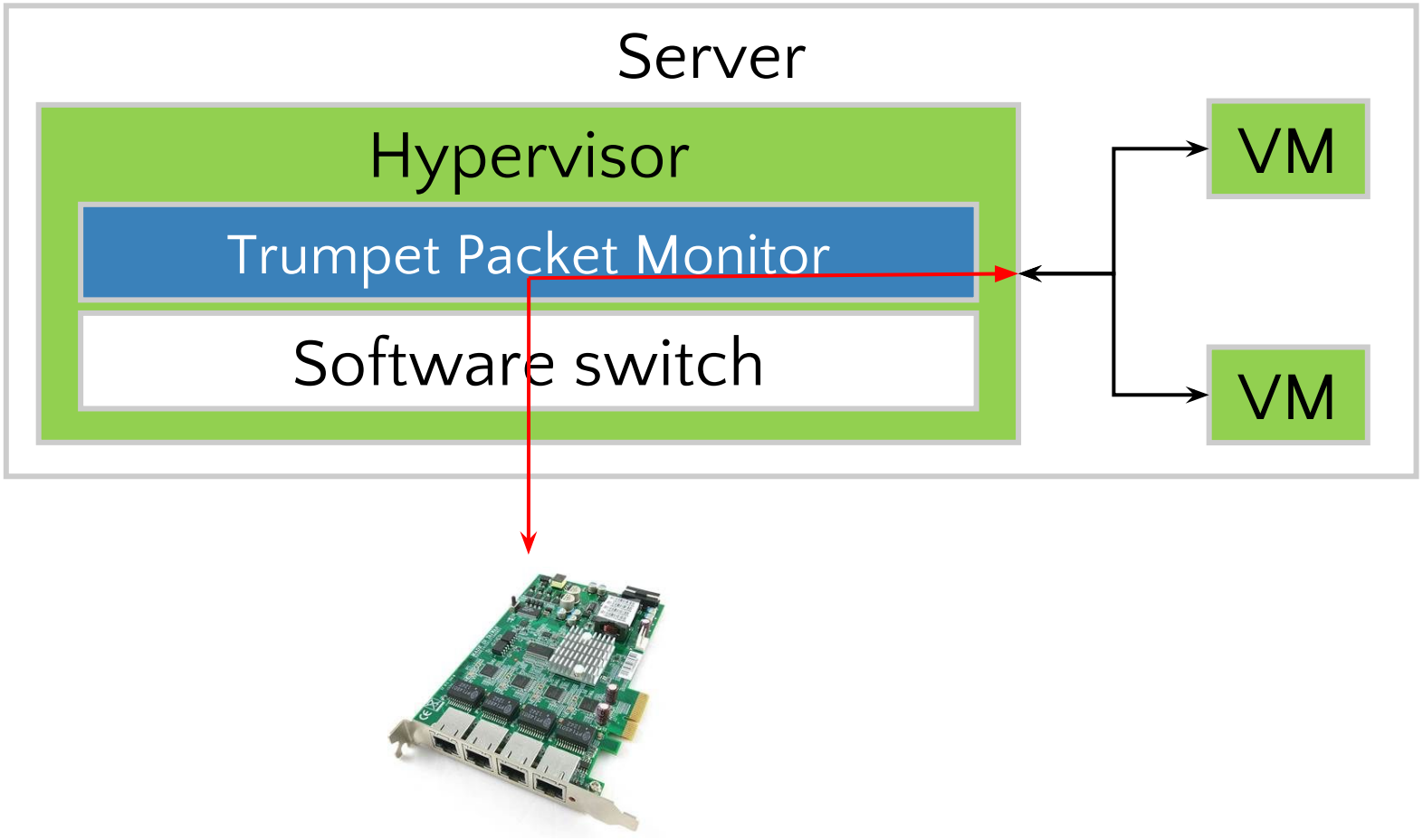
What eventing architecture permits programmability **and** visibility?

How can we achieve precise eventing at fine timescales?

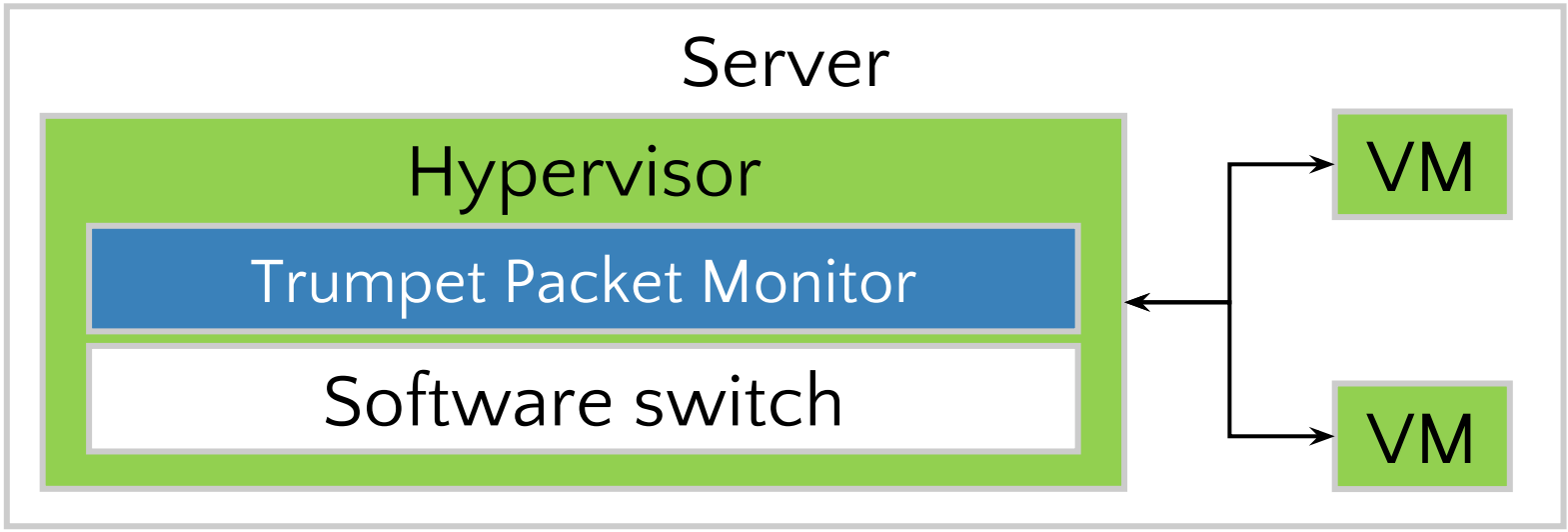
What is the *performance envelope* of such an eventing framework?

The monitor optimizes packet processing to inspect every packet and evaluate predicates at fine timescales

The Packet Monitor



A Key Design Decision



Run monitor on CPU core used by software switch

- Conserves CPU resources
- Avoids inter-core synchronization

Can a single CPU core monitor thousands of triggers at full packet rate (14.8 Mpps) on a 10G NIC?

Two Obvious Tricks

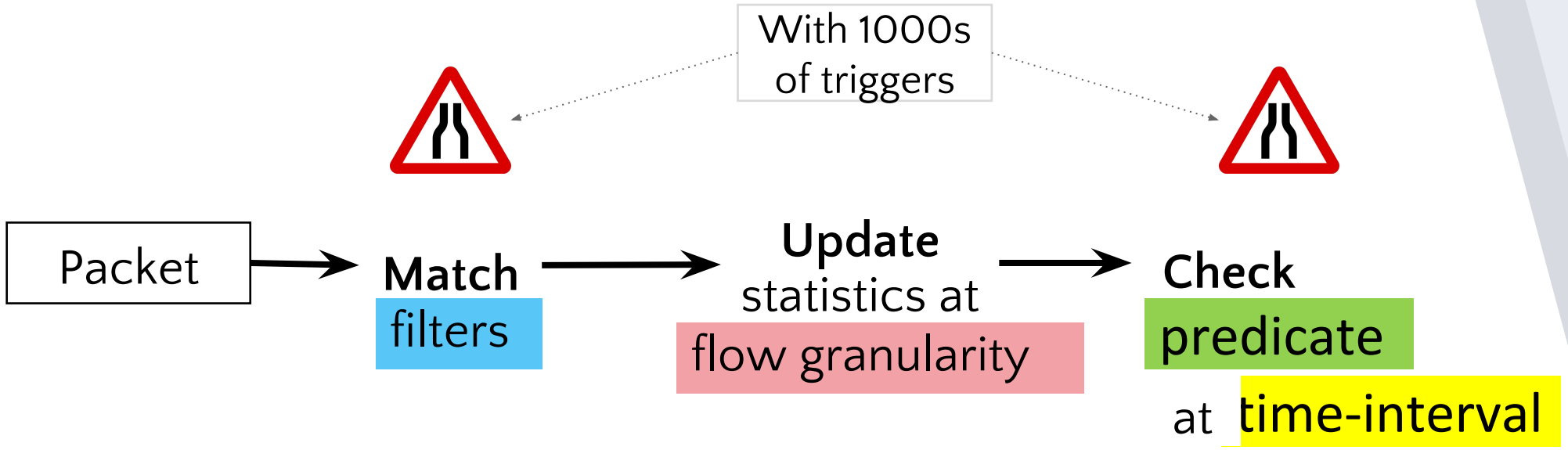
Use kernel bypass

- Avoid kernel stack overhead

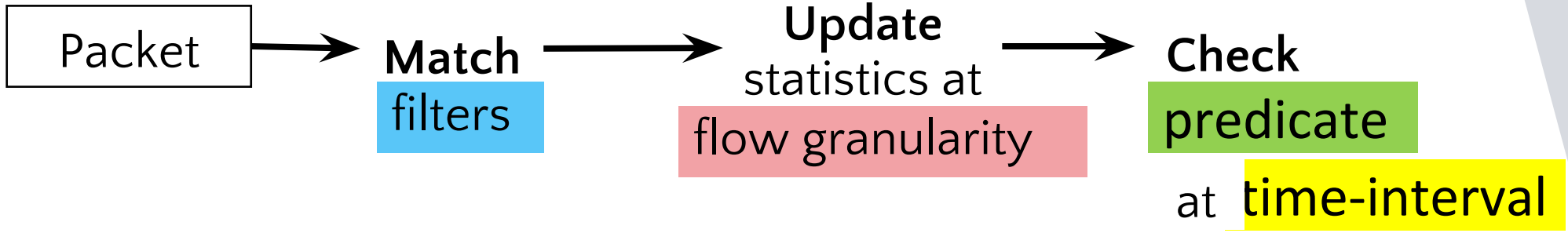
Use polling to have tighter scheduling

- Trigger time intervals at 10ms

Necessary, but far from sufficient….

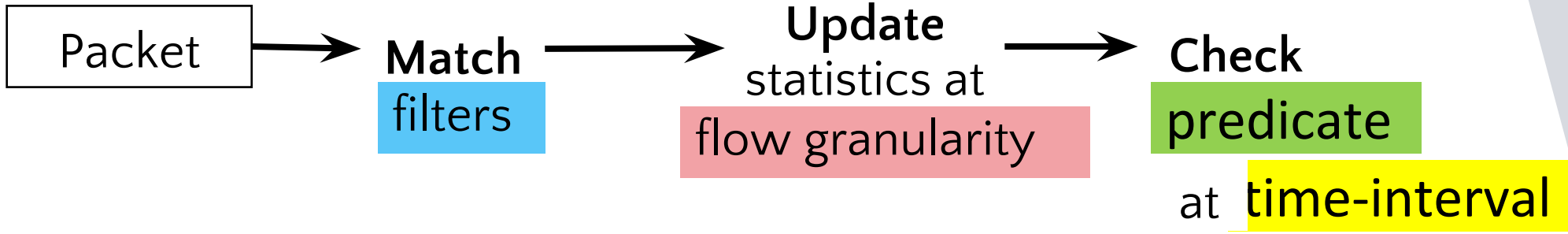


Filter	Flow granularity	Time interval	Predicate
Source IP = 10.1.1.0/24	5-tuple	10ms	Sum(loss) > 10%
Source IP = 20.2.2.0/24	Service IP prefix	100ms	Sum(size) < 10MB



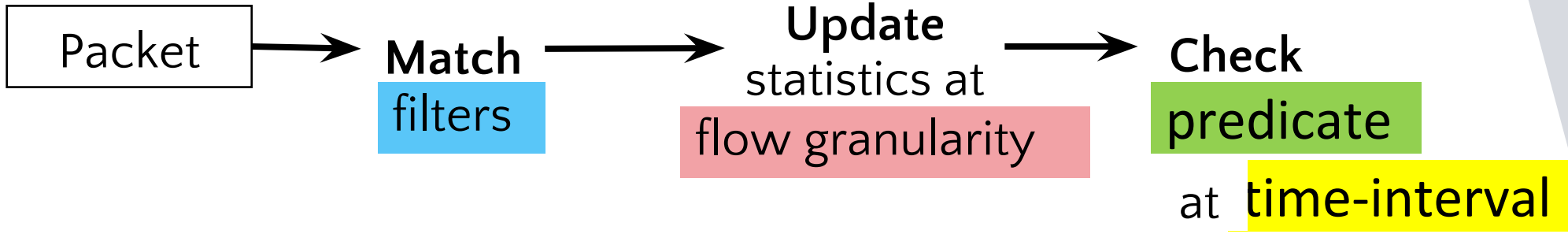
Which of these should be performed

- *On-path*
- *Off-path*



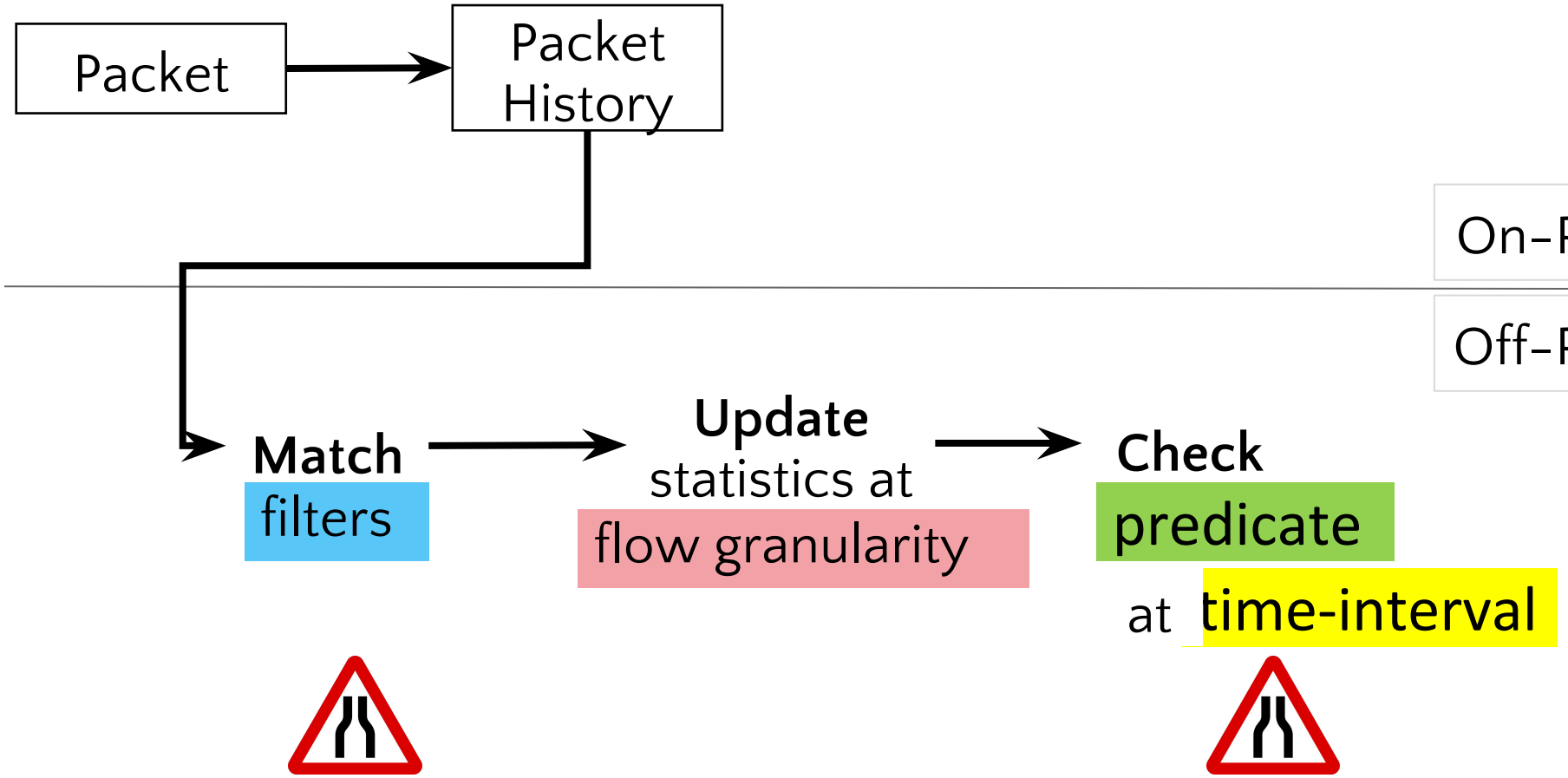
Which operations to do on-path?

- 70ns to forward and inspect packet

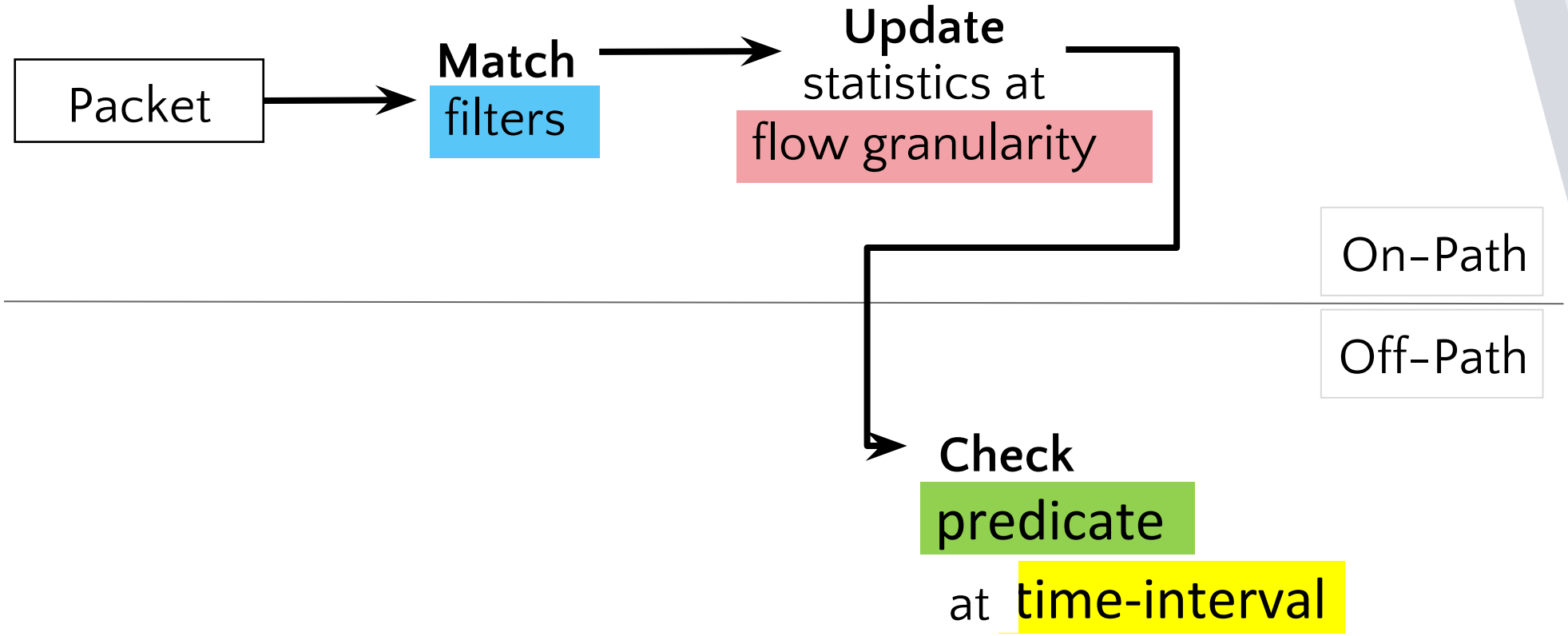


How to schedule off-path operations?

- Off-path on same core, can delay packets
- Bound delay to a few µs

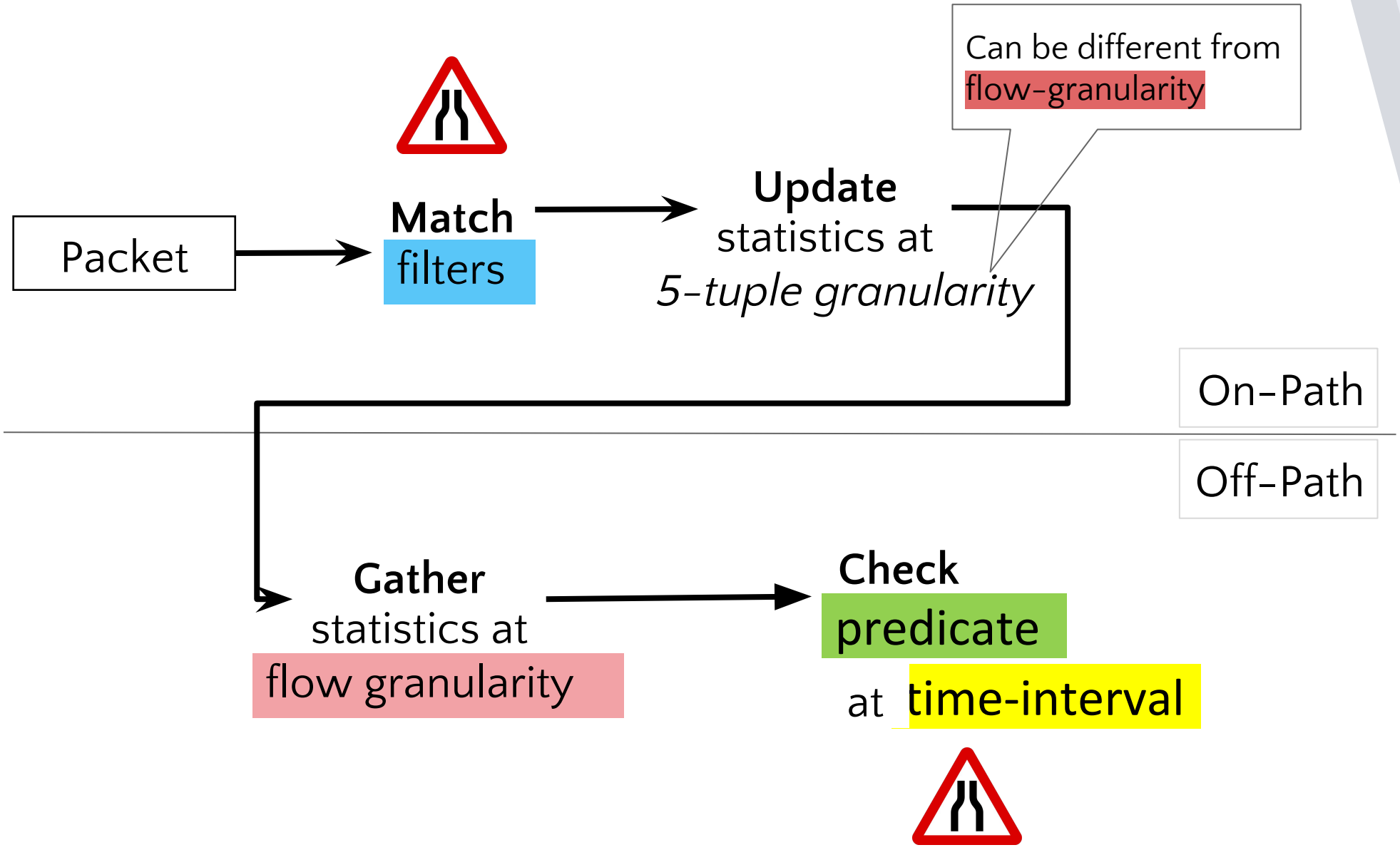


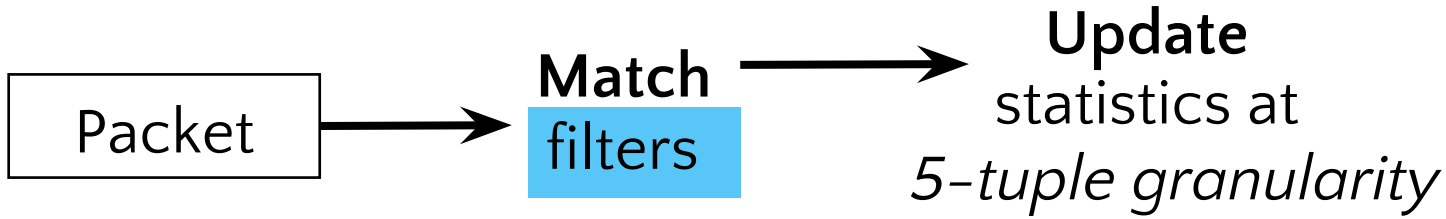
Doesn't scale to large numbers of triggers



Still cannot reach goal

- Memory subsystem becomes a bottleneck


Can be different from
flow-granularity
Packet
Match
filters
Update
statistics at
5-tuple granularity
On-Path
Off-Path
Gather
statistics at
flow granularity
Check
predicate
at time-interval

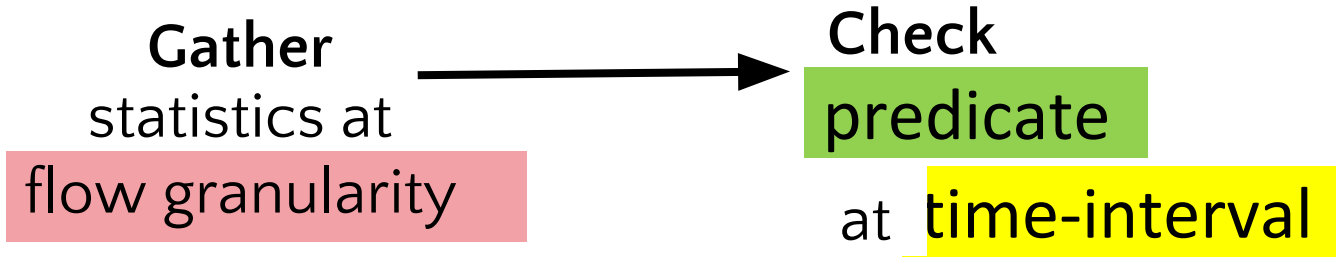


On-Path

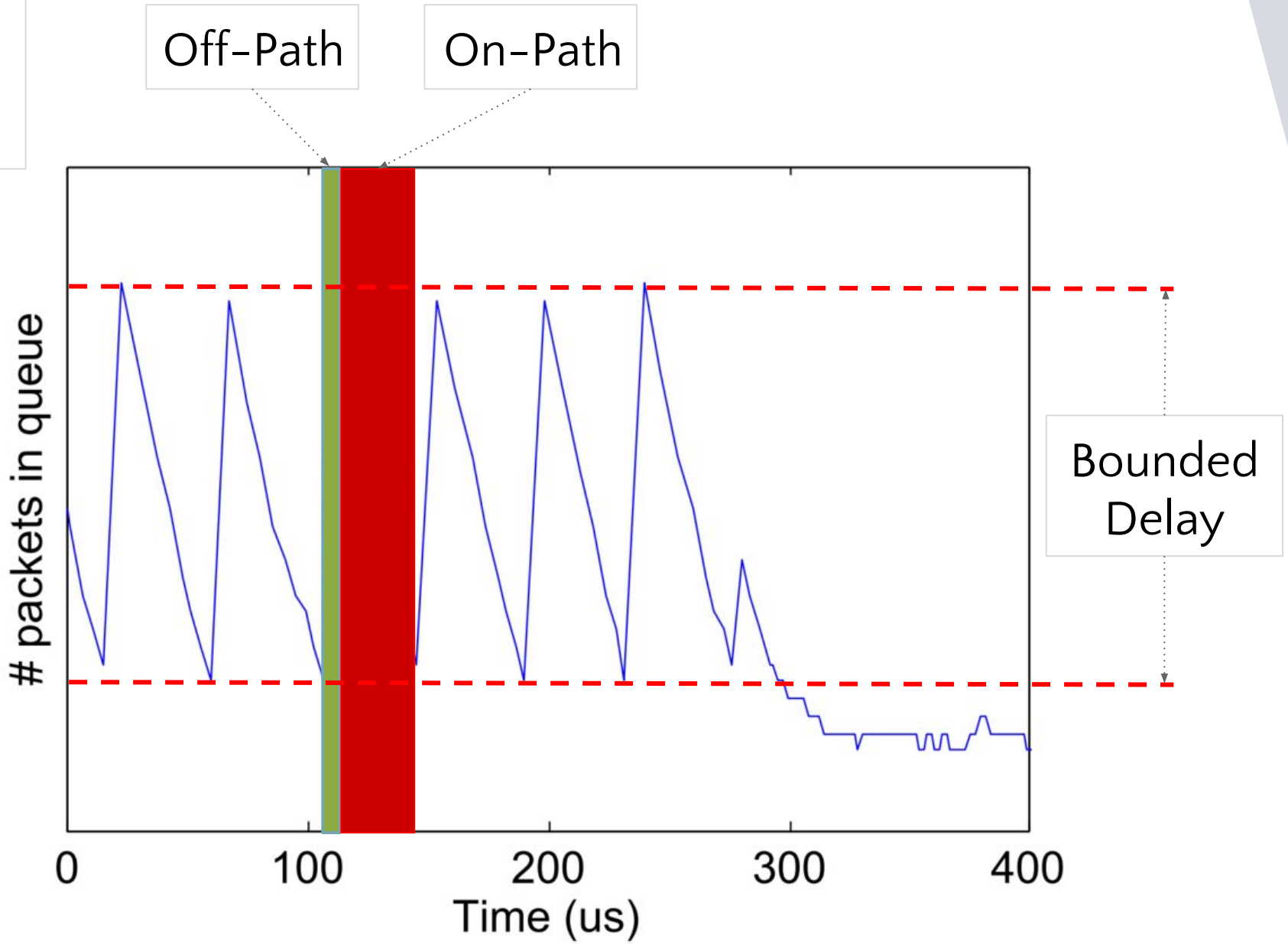
- Use tuple-space search for matching
- Match on first packet, cache match
- Lay out tables to enable cache prefetch
- Use TLB huge pages for tables

- Lazy cleanup of statistics across intervals
- Lay out tables to enable cache prefetch
- *Bounded-delay cooperative scheduling*

Off-Path



Bound delay to a few µs



Bounded Delay Cooperative Scheduling

Research Questions

What eventing architecture permits programmability **and** visibility?

How can we achieve precise eventing at fine timescales?

What is the *performance envelope* of such an eventing framework?

Trumpet can monitor thousands of triggers at full packet rate on a 10G NIC

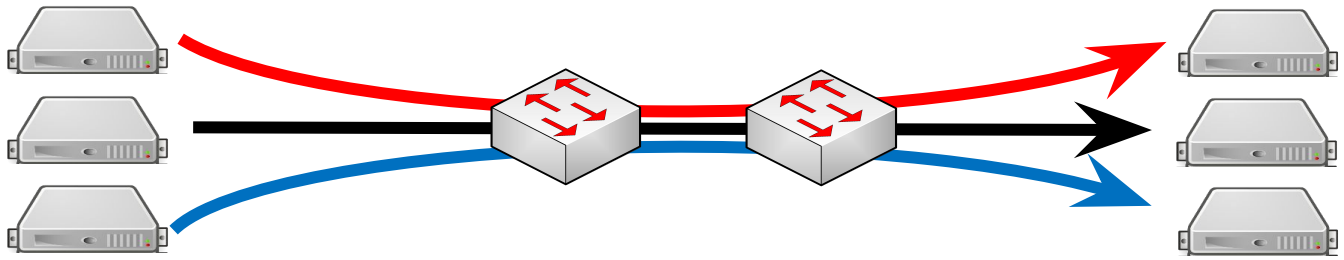
Trumpet is expressive

- Transient congestion
- Burst loss
- Attack onset

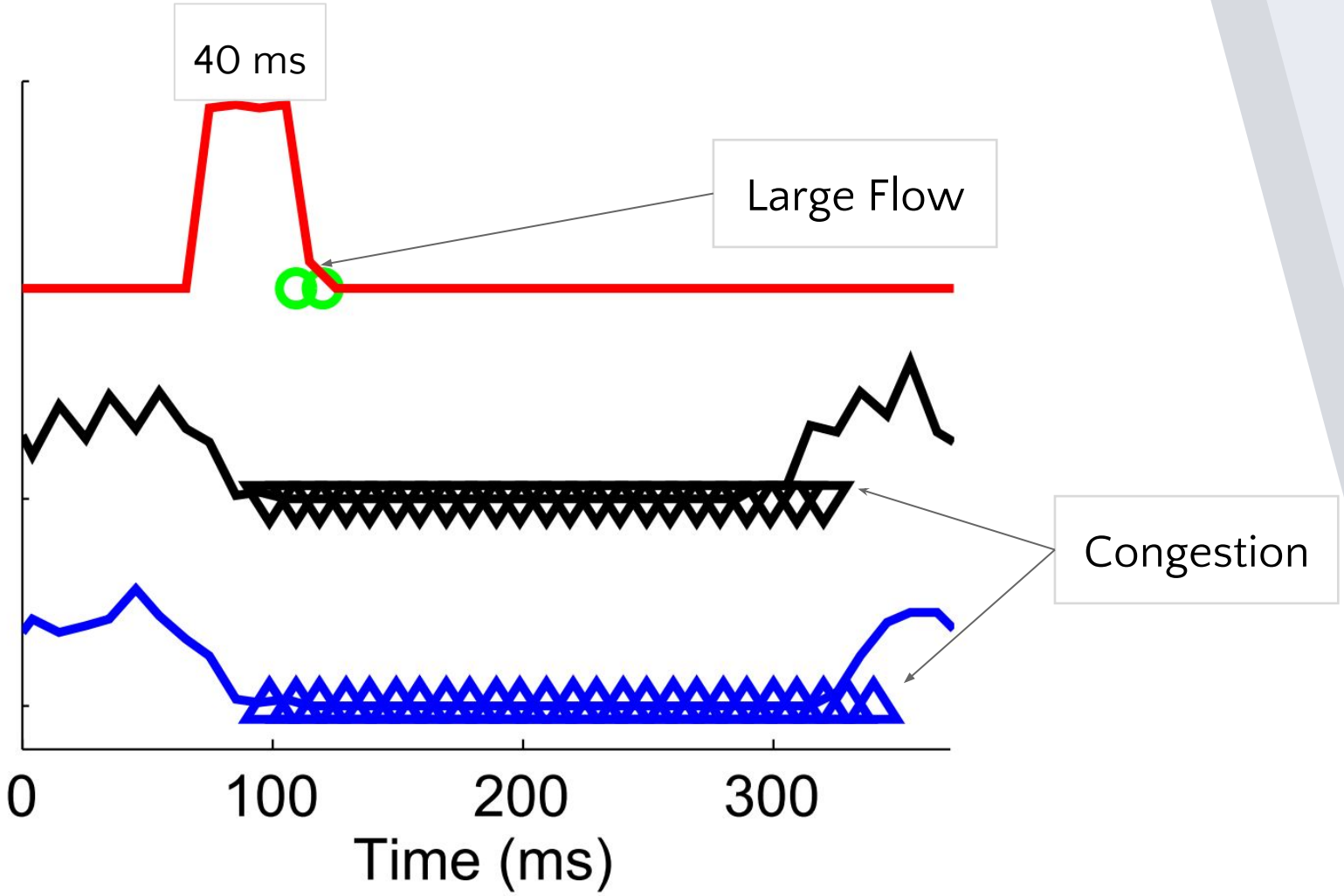
Trumpet scales to thousands of triggers

Trumpet is DoS-Resilient

Detecting Transient Congestion



Trumpet can detect millisecond scale congestion events



Trumpet can process* 14.8 Mpps

- 64 byte packets at 10G
- 650 byte packets at 4x10G

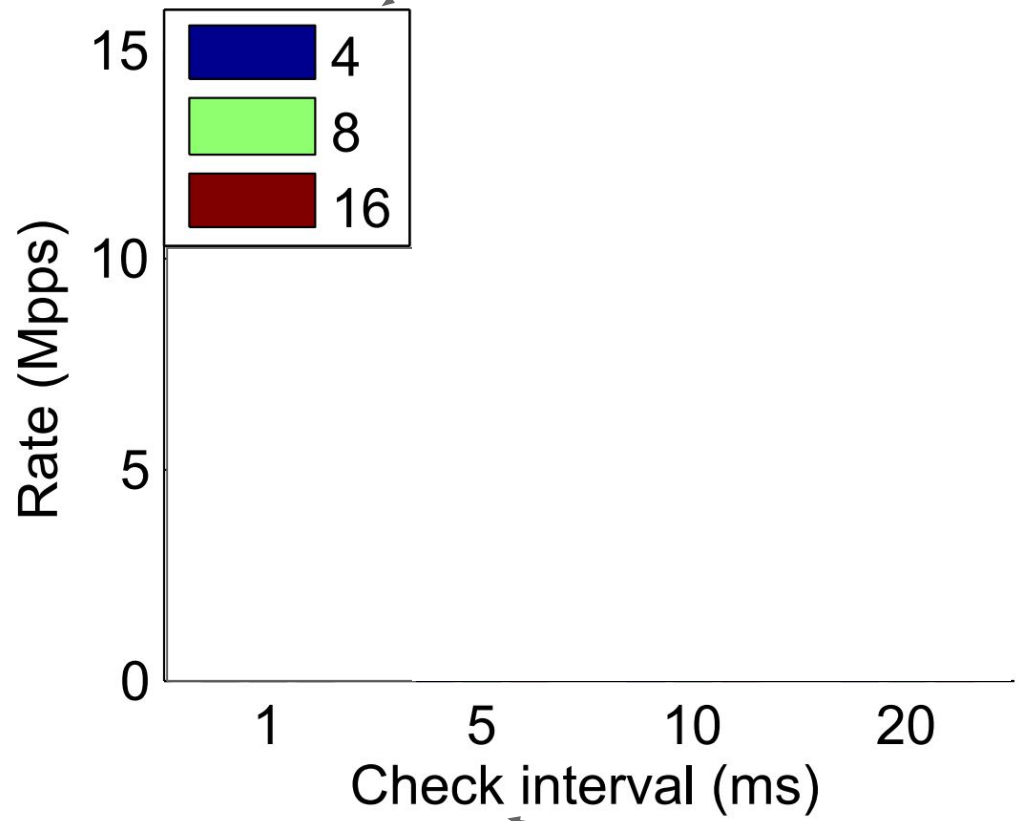
... while evaluating 4K triggers at 10ms granularity

*Xeon ES-2650, 10-core 2.3 Ghz, Intel 82599 10G NIC

Performance Envelope

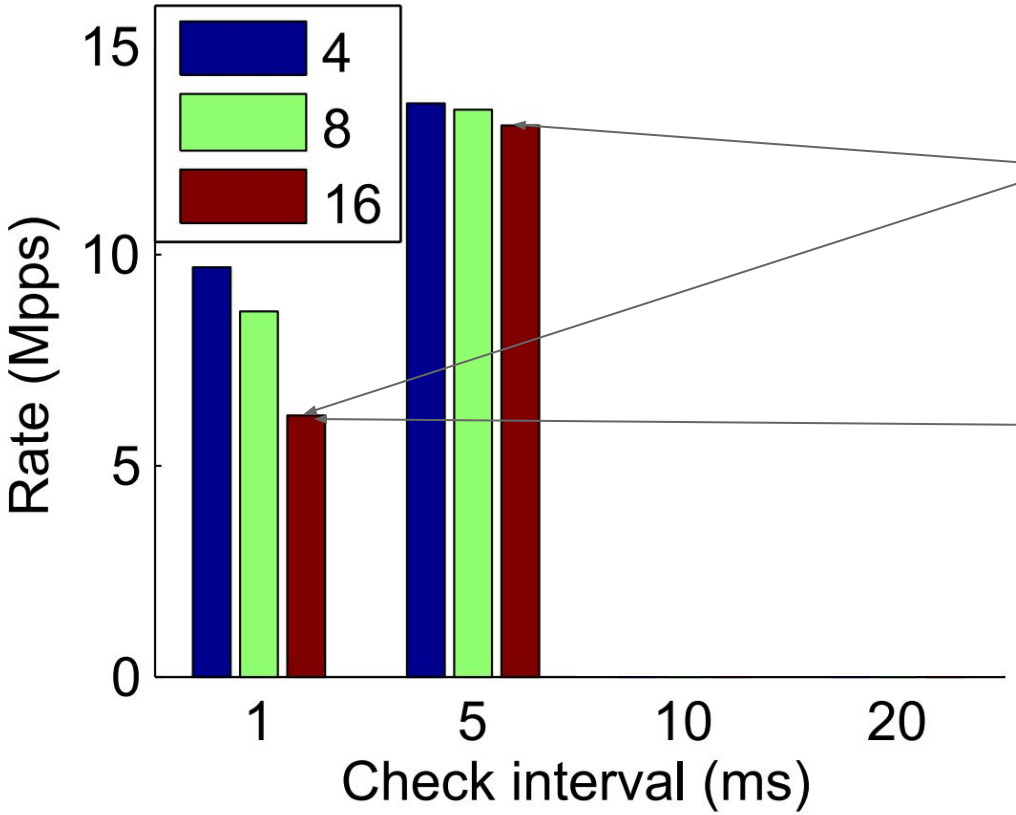
Triggers matched by each flow

Above this rate, Trumpet would **miss events**



How often each predicate is checked

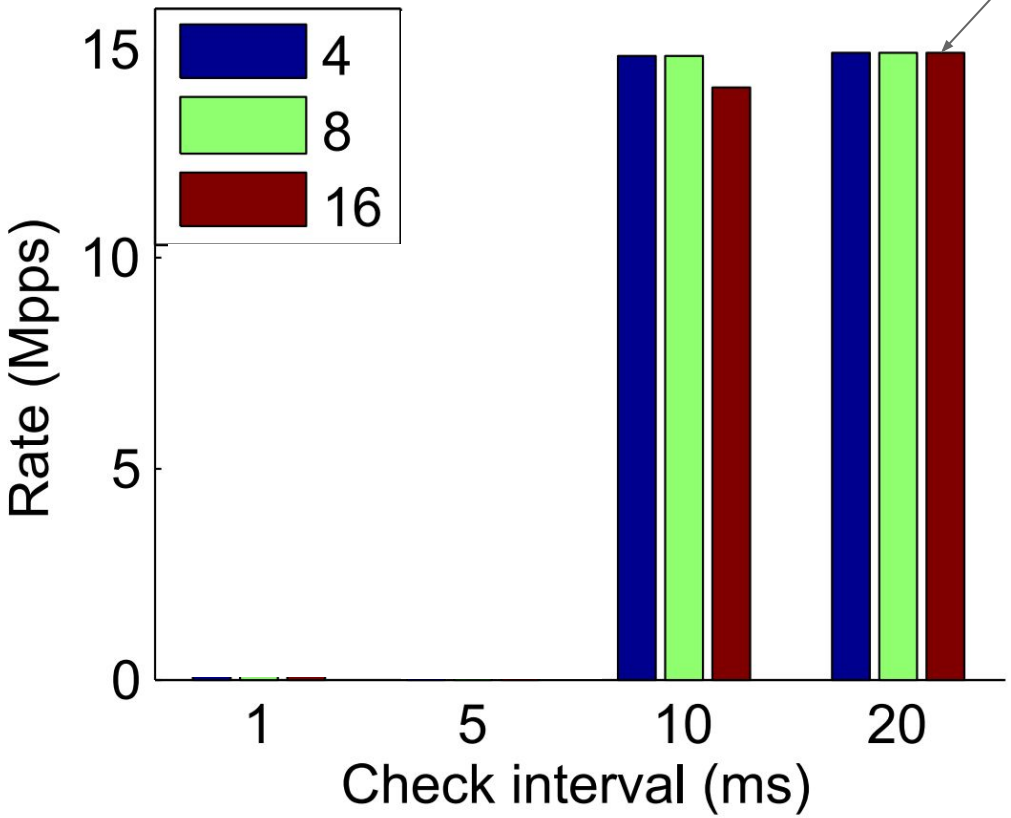
Performance Envelope



Number of <trigger, flow> pairs increases statistics gathering overhead

At moderate packet rates, can detect events at 1ms

Performance Envelope



Above 10ms, CPU can sustain full packet rate

Need to profile and provision Trumpet deployment

Conclusion

Future datacenters will need fast and precise eventing

- Trumpet is an expressive system for host-based eventing

Trumpet can process 16K triggers at full packet rate

- … without delaying packets by more than 10 µs

Future work: scale to 40G NICs

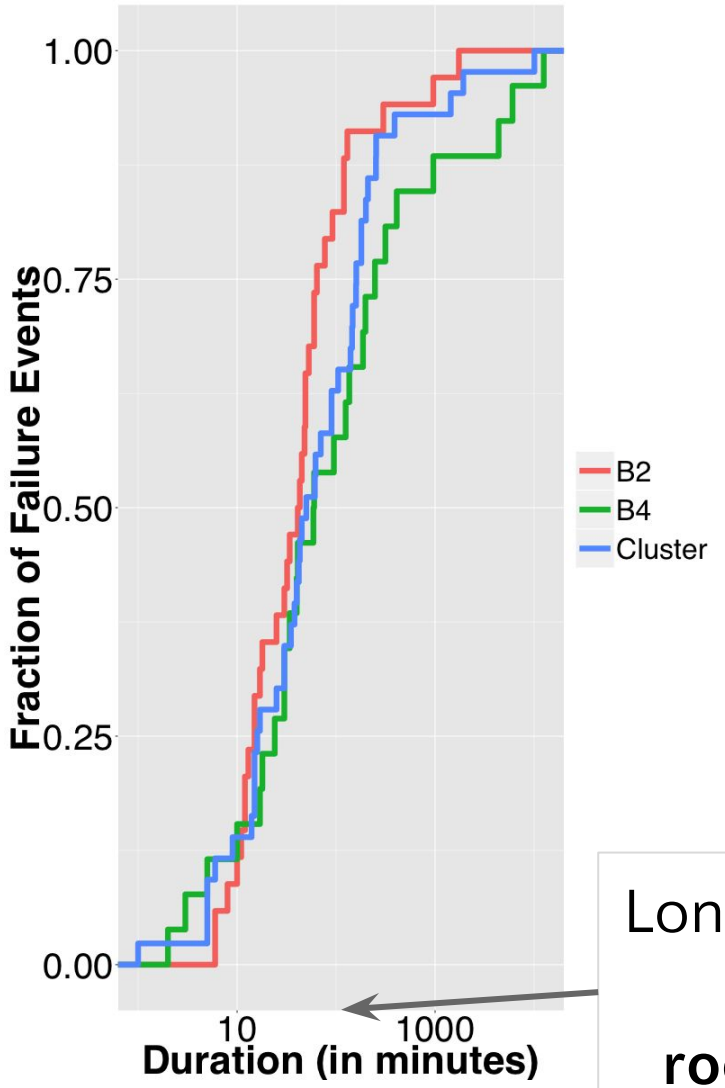
- … perhaps with NIC or switch support

https://github.com/USC-NSL/Trumpet

A Big Discrepancy

Outage budget for **five 9s** availability

24 seconds per month

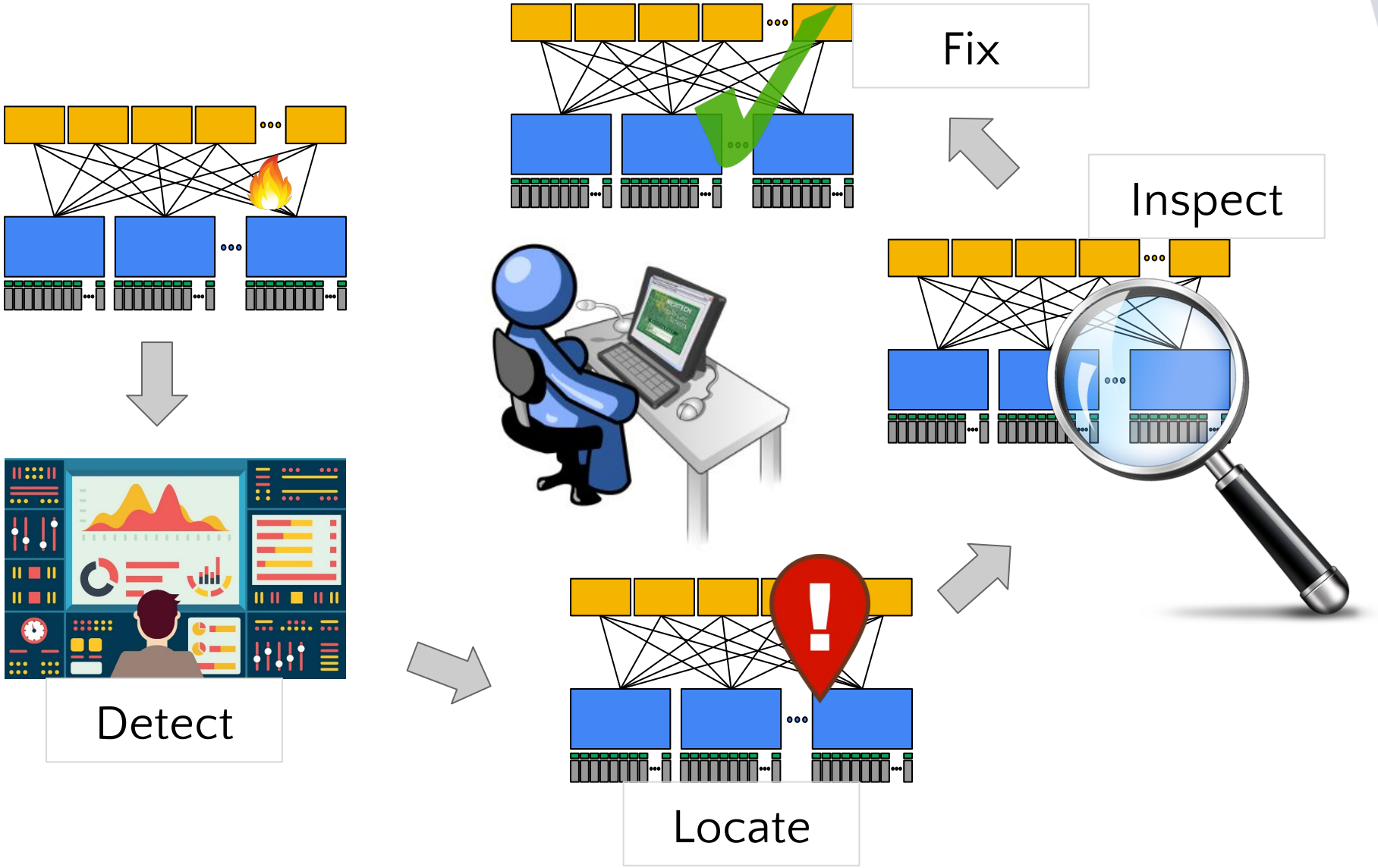


Long failure durations due to time to **root-cause** failures

Every optimization is necessary❇

❇Details in the paper

Humans in the Loop



Programs in the Loop