Self-driving/monitoring networks

in the age of deep network programmability

Laurent Vanbever

nsg.ee.ethz.ch

TMA

June 19 2019

1962

Paul Baran (1926–2011)

American electrical engineer

created the notion of a distributed network

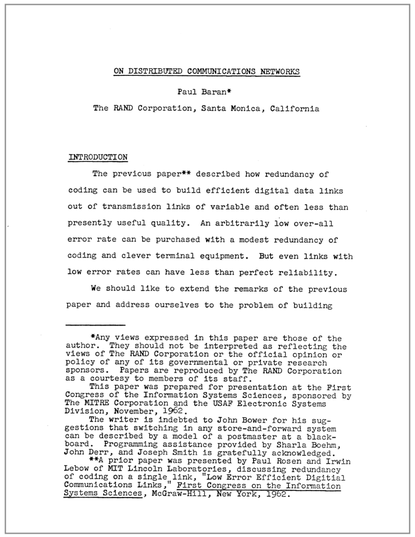
which could maintain communications

in the face of a thermonuclear attack

inventor of packet switching
(with Donald Davies)

On distributed communication networks

Paul Baran—Sept. 1962



ON DISTRIBUTED COMMUNICATIONS NETWORKS

Paul Baran*

The RAND Corporation, Santa Monica, California

INTRODUCTION

The previous paper** described how redundancy of coding can be used to build efficient digital data links out of transmission links of variable and often less than presently useful quality. An arbitrarily low over-all error rate can be purchased with a modest redundancy of coding and clever terminal equipment. But even links with low error rates can have less than perfect reliability.

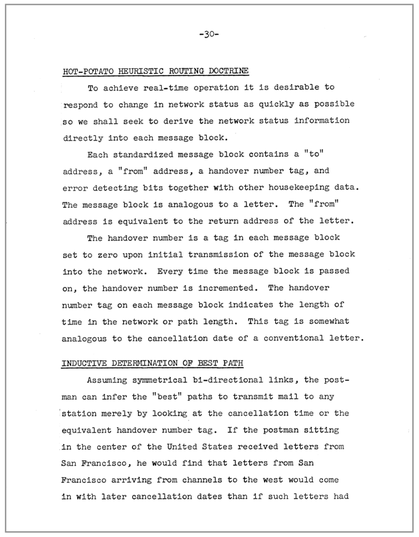
We should like to extend the remarks of the previous paper and address ourselves to the problem of building

*Any views expressed in this paper are those of the author. They should not be interpreted as reflecting the views of The RAND Corporation or the official opinion or policy of any of its governmental or private research sponsors. Papers are reproduced by The RAND Corporation as a courtesy to members of its staff.

This paper was prepared for presentation at the First Congress of the Information Systems Sciences, sponsored by The MITRE Corporation and the USAF Electronic Systems Division, November, 1962.

The writer is indebted to John Bower for his suggestions that switching in any store-and-forward system can be described by a model of a postmaster at a blackboard. Programming assistance provided by Sharla Boehm, John Derr, and Joseph Smith is gratefully acknowledged.

**A prior paper was presented by Paul Rosen and Irwin Lebow of MIT Lincoln Laboratories, discussing redundancy of coding on a single link, "Low Error Efficient Digitial Communications Links," First Congress on the Information Systems Sciences, McGraw-Hill, New York, 1962.



-30-

HOT-POTATO HEURISTIC ROUTING DOCTRINE

To achieve real-time operation it is desirable to respond to change in network status as quickly as possible so we shall seek to derive the network status information directly into each message block.

Each standardized message block contains a "to" address, a "from" address, a handover number tag, and error detecting bits together with other housekeeping data. The message block is analogous to a letter. The "from" address is equivalent to the return address of the letter.

The handover number is a tag in each message block set to zero upon initial transmission of the message block into the network. Every time the message block is passed on, the handover number is incremented. The handover number tag on each message block indicates the length of time in the network or path length. This tag is somewhat analogous to the cancellation date of a conventional letter.

INDUCTIVE DETERMINATION OF BEST PATH

Assuming symmetrical bi-directional links, the postman can infer the "best" paths to transmit mail to any station merely by looking at the cancellation time or the equivalent handover number tag. If the postman sitting in the center of the United States received letters from San Francisco, he would find that letters from San Francisco arriving from channels to the west would come in with later cancellation dates than if such letters had

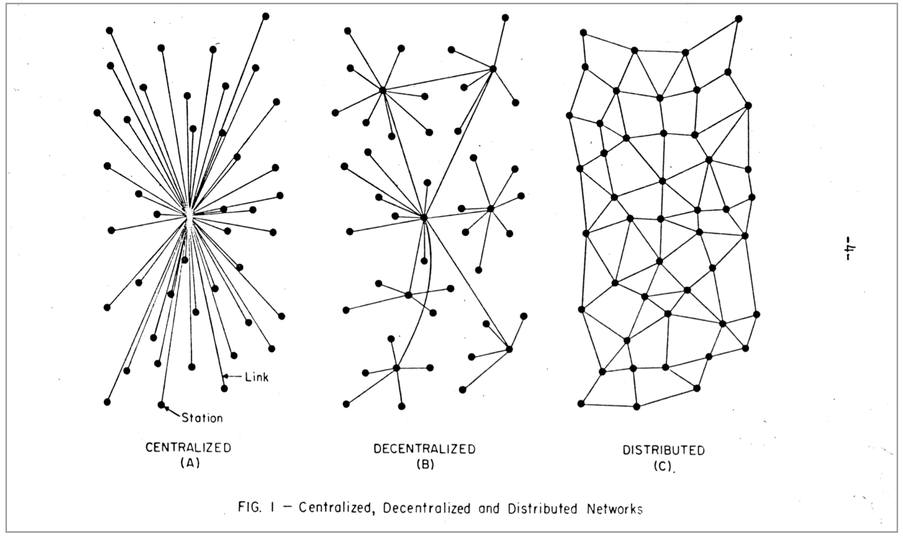


FIG. 1 — Centralized, Decentralized and Distributed Networks

Introduces the concept of

- packet switching
- distributed routing
- reliable transport

among many others…

Thanks to Paul Barlan et al.

the Internet infrastructure is extremely resilient

ability to route packets around failures

(not necessarily fast though)

Thanks to Paul Barlan et al.

the Internet infrastructure is extremely resilient

… at least when it comes to thermonuclear wars…

Thanks to Paul Barlan et al.

the Internet infrastructure is extremely resilient

… at least when it comes to thermonuclear wars…

but how does it fare against, say…

Thanks to Paul Barlan et al.

the Internet infrastructure is extremely resilient

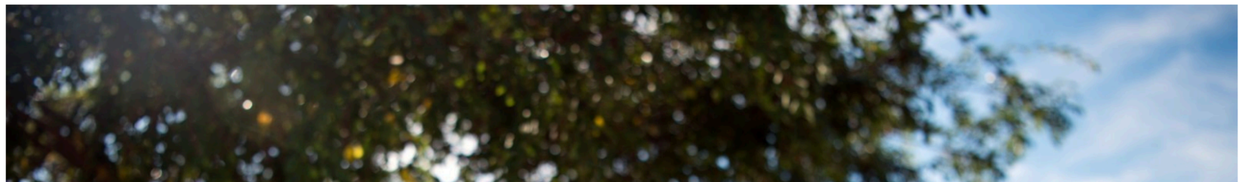
… at least when it comes to thermonuclear wars…

but how does it fare against, say… us humans?

Google accidentally broke the internet throughout Japan

A mistake led to internet outages for about half of the country.

Mallory Locklear, @mallorylocklear
08.28.17 in Internet



27 August 2017

Google made a configuration mistake which caused their Chicago point-of-presence to wrongly advertise 160k IP prefixes to its neighbors.

Google made a configuration mistake which caused their Chicago point-of-presence to wrongly advertise 160k IP prefixes to its neighbors.

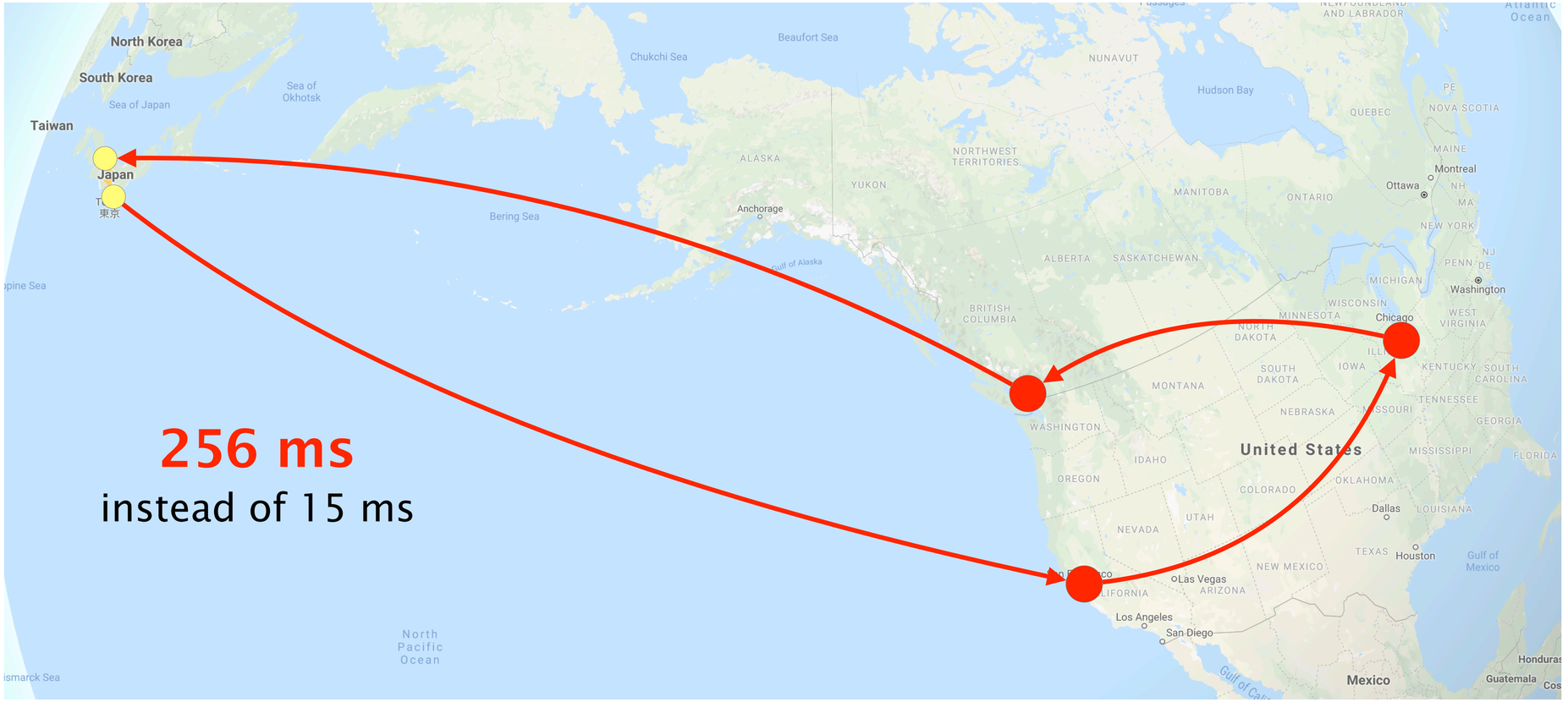
These advertisements propagated in the Internet and got picked up by Japanese giants (IIJ and KDDI) which started to direct local Japanese traffic to Google.

Tokyo to Nagoya

(normally)

Tokyo to Nagoya via… Chicago?!

A **17x** increased latency!



Google made a configuration mistake which caused their Chicago point-of-presence to wrongly advertise 160k IP prefixes to its neighbors.

These advertisements propagated in the Internet and got picked up by Japanese giants (NTT and KDDI) which started to direct local, Japanese traffic to Google.

The outage in Japan *only* lasted a couple of hours but was so severe that the country's ministries wanted carriers to report on what went wrong.

This is far from being an isolated event...

Forbes / Personal Finance

The Little Black Book of Billionaire Secrets

JUL 8, 2015 @ 03:36 PM 11,261 VIEWS

United Airlines Blames Router for Grounded Flights

'Configuration Error' Blamed for AWS Outage

By David Ramel ■ 08/12/2015

The summer of network misconfigurations

Amazon's massive AWS outage was caused by human error

One incorrect command and the whole internet suffers.

By Jason Del Rey | @DelRey | Mar 2, 2017, 2:20pm EST

CenturyLink: 750 calls to 911 missed during Aug. 1 outage caused by human error in Minnesota, North Dakota

By Barry Amundson on Aug 15, 2018 at 4:43 p.m.

Data Centre ▸ Networks

Level3 switch config blunder for US-wide VoIP blackout

affected Comcast, Spectrum, Verizon and AT&T customers

BY: CNN

Data Centre ▸ Networks

CloudFlare apologizes for Telia screwing you over

Unhappy abou

By Kieren McCarthy in

Facebook struggles to deal with epic outage


By Donie O'Sullivan and Heather Kelly, CNN Business

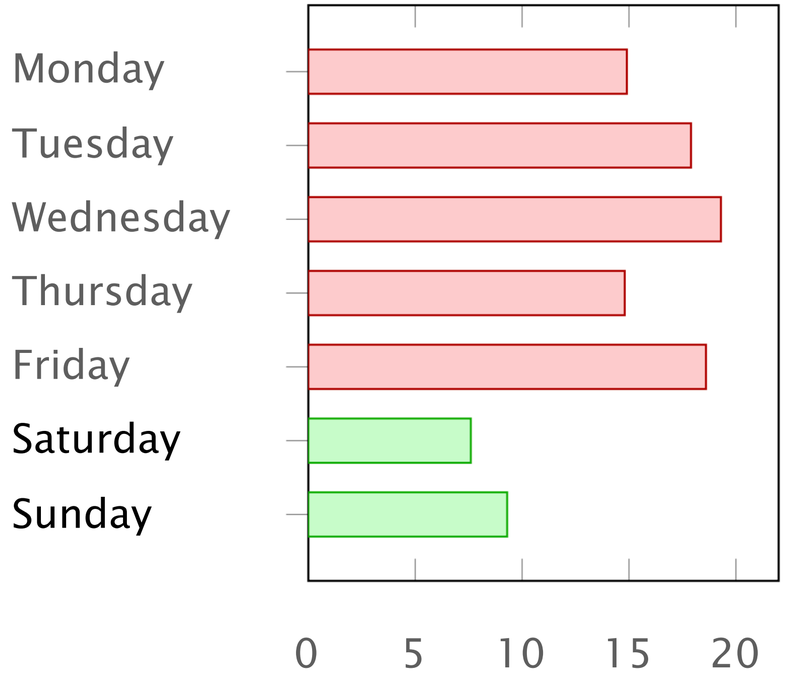
Updated 0654 GMT (1454 HKT) March 14, 2019

"Human factors are responsible

for 50% to 80% of network outages"

Juniper Networks, *What's Behind Network Downtime?*, 2008

A perhaps ironic consequence is that the Internet works better during the week-ends



source: Job Snijders, 2008–2016

"Human factors are responsible

for 50% to 80% of network outages"

Juniper Networks, *What's Behind Network Downtime?*, 2008

“Human factors are responsible

for [redacted] of [redacted]”

“Human factors are responsible

for >90% of car accidents”

NHTSA, National Motor Vehicle Crash Causation Survey, February 2015

Enters…

Self-Driving Car

Enters…

Self-Driving Car

Wikipedia

A vehicle that is capable of monitoring its environment and moving with little or no human input

Enters…

Self-Driving Car

Wikipedia — A vehicle that is capable of monitoring its environment and moving with little or no human input

Promise — A drastic reduction in the number of (fatal) accidents

Enters…

Self-Driving Car

Enters…

Self-Driving

Enters…

Self-Driving Network

Enters…

Self-Driving Network

Definition

A network that is capable of monitoring its environment and adapting its behavior accordingly with little or no human input

Enters…

Self-Driving Network

Definition

A network that is capable of monitoring its environment and adapting its behavior accordingly with little or no human input

Questions

How do we build and deploy such networks?

Enters…

Self-Driving Network

Definition

A network that is capable of monitoring its environment and adapting its behavior accordingly with little or no human input

Questions

How do we build and deploy such networks?

Until recently, innovating in networks was hard because devices were completely locked down

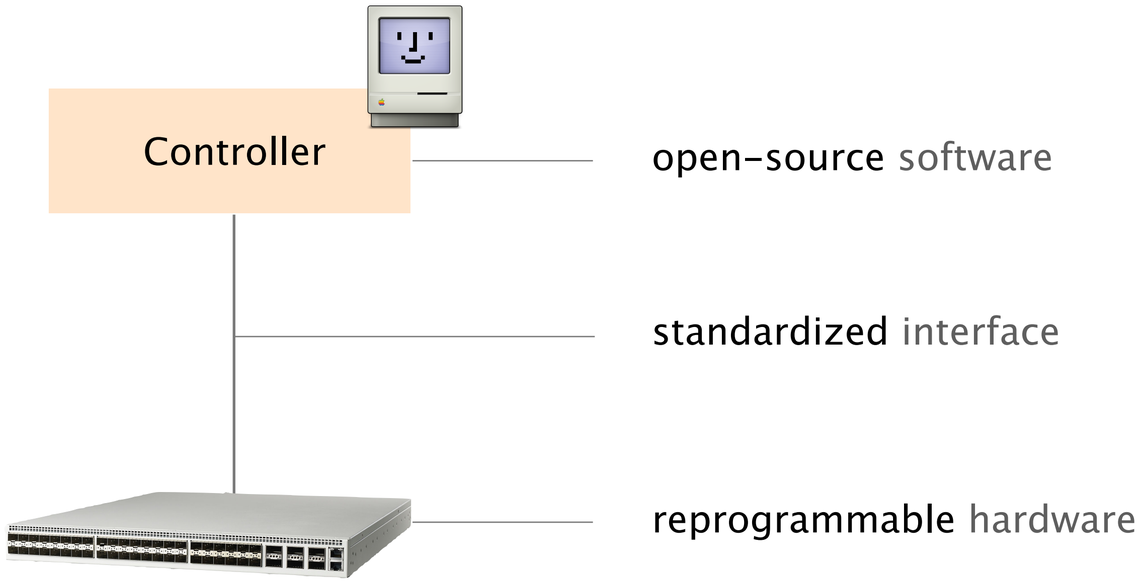
closed software

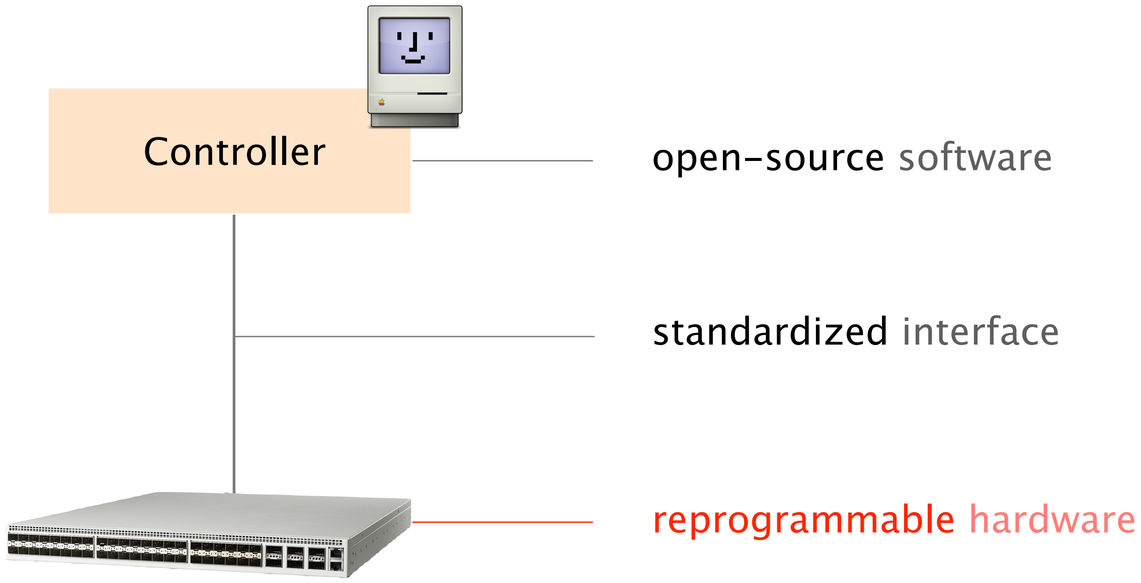
closed hardware

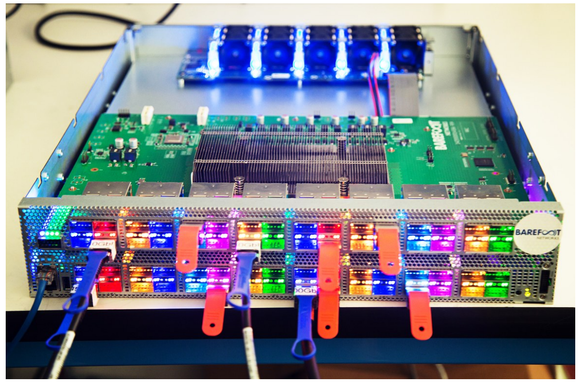
Cisco™ device

Things are changing though!

Networks are on the verge of a paradigm shift towards deep programmability

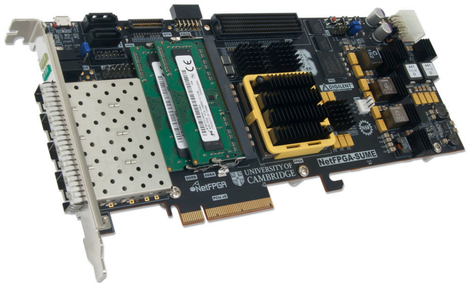



Controller
open-source software
standardized interface
reprogrammable hardware

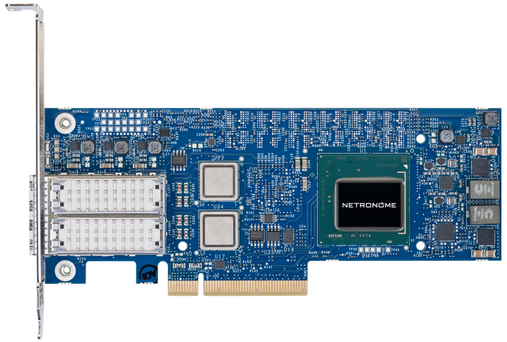


Barefoot Tofino 12.8 Tbps

programmable network switch



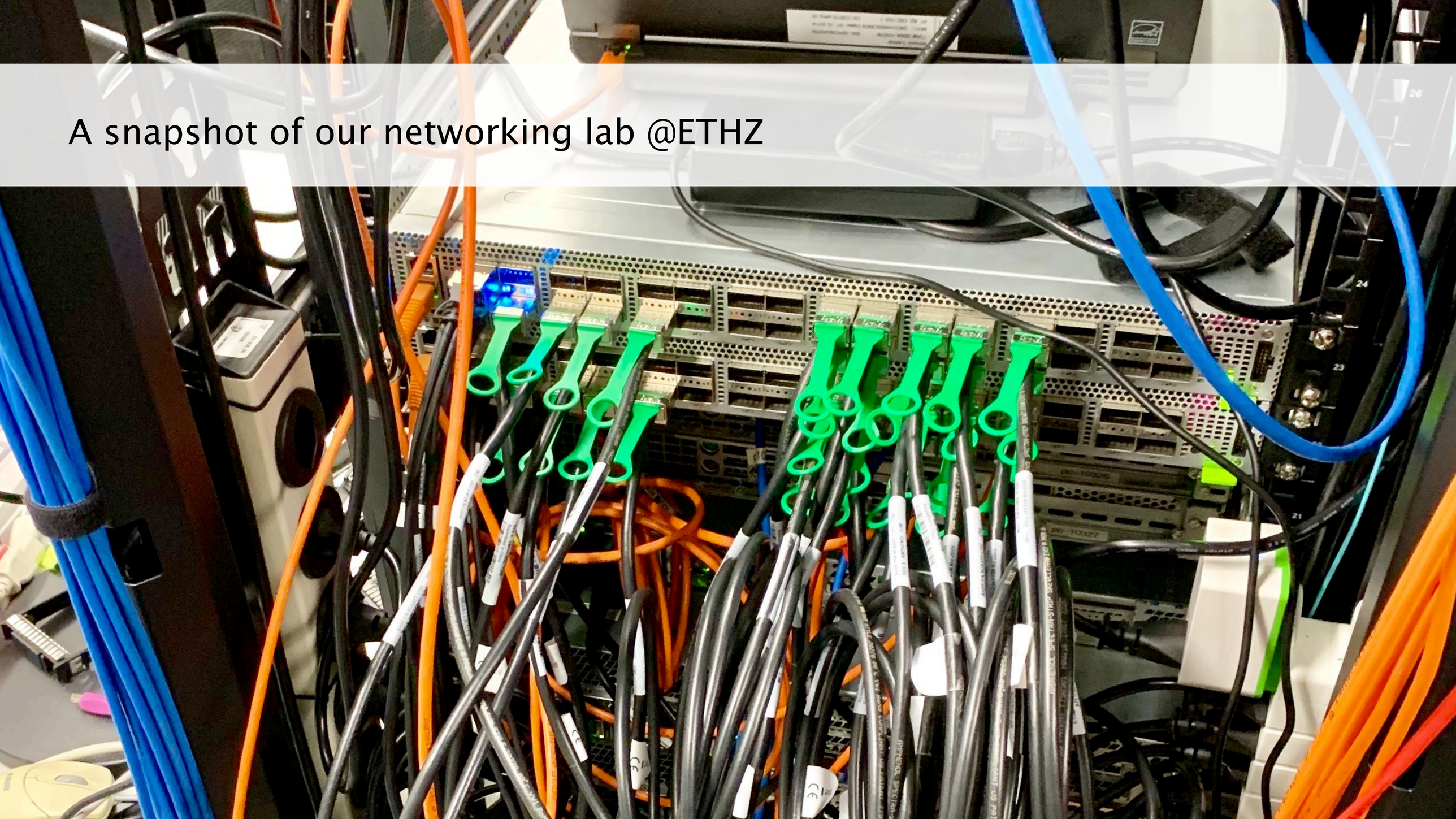
NetFPGA SUME



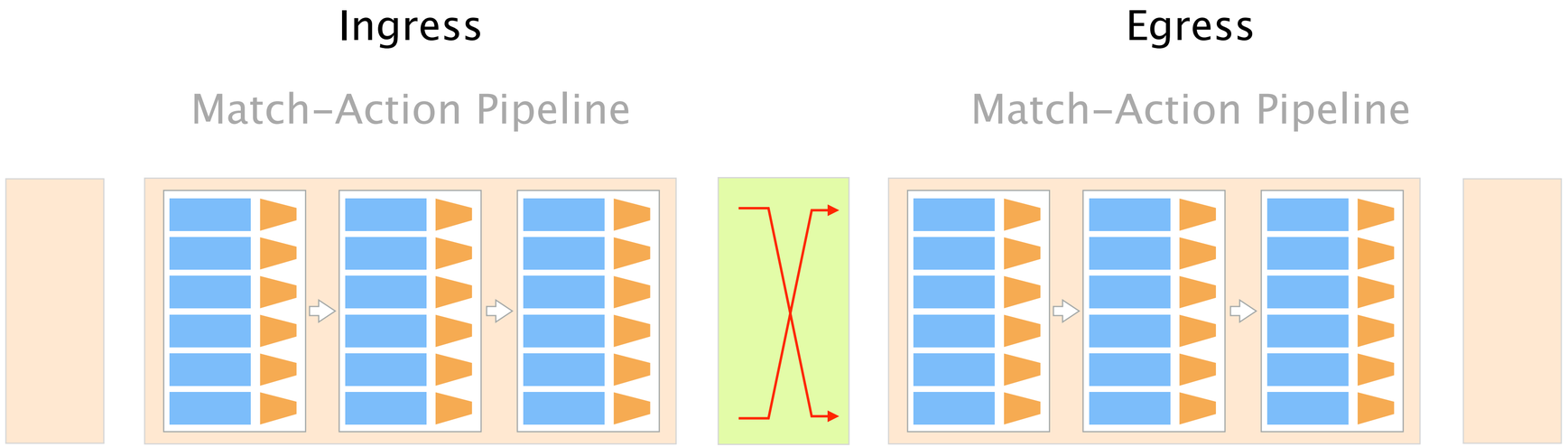
Netronome Agilio CX

100 Gbps programmable network card

A snapshot of our networking lab @ETHZ



Reprogrammable network hardware allows to completely redefine the forwarding logic



Programmable networks can be made "self-driving"

Programmable devices can

- measure
- perform statistical inference
- adapt their forwarding decisions

at line rate, on a per-packet basis

Enters…

Self-Driving Network

Definition

A network that is capable of monitoring its environment and adapting its behavior accordingly with little or no human input

Questions

How do we build and deploy such networks?

Levels of autonomy in self-driving cars

level 0	no automation	
1	driver assistance	human monitors the environment
2	partial automation	
3	conditional automation	system monitors human as fallback
4	high automation	
5	full automation	no more human

Self-driving/monitoring networks

in the age of deep network programmability

1 operator assistance

2 partial automation

Part I
assisting operators

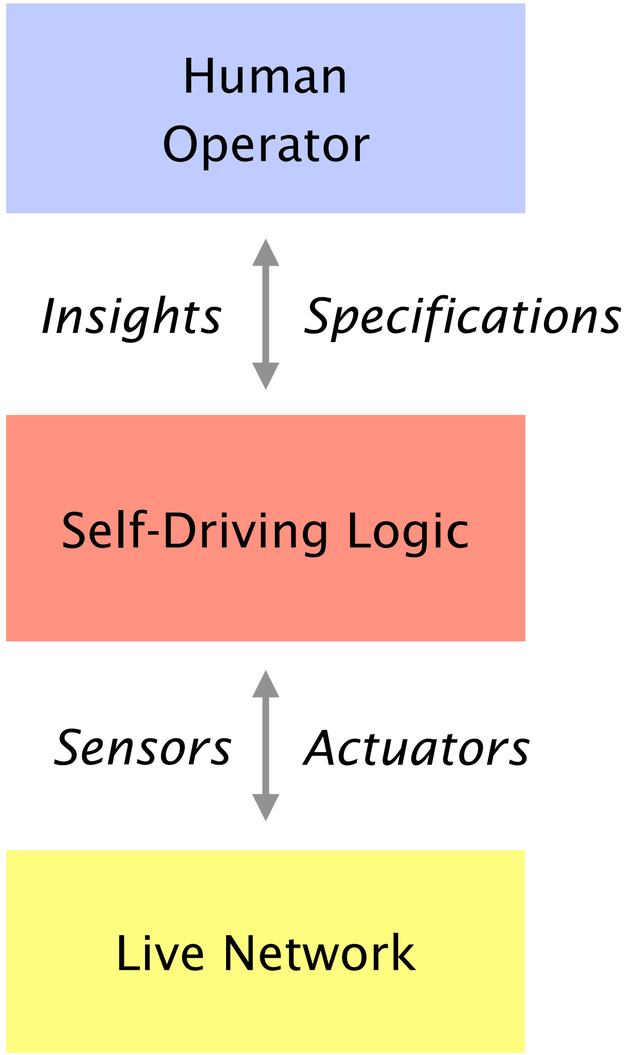
3 conditional automation

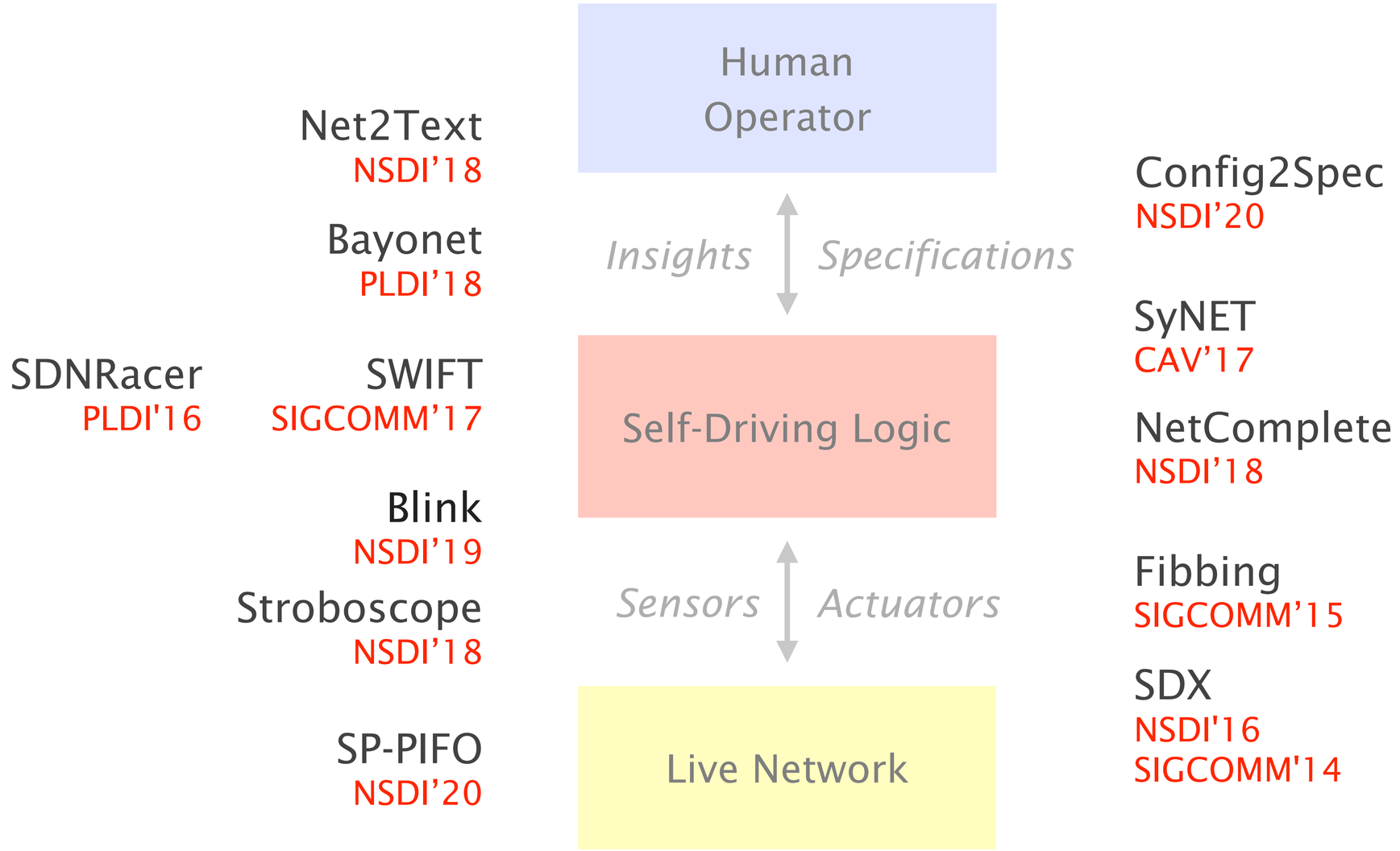
4 high automation

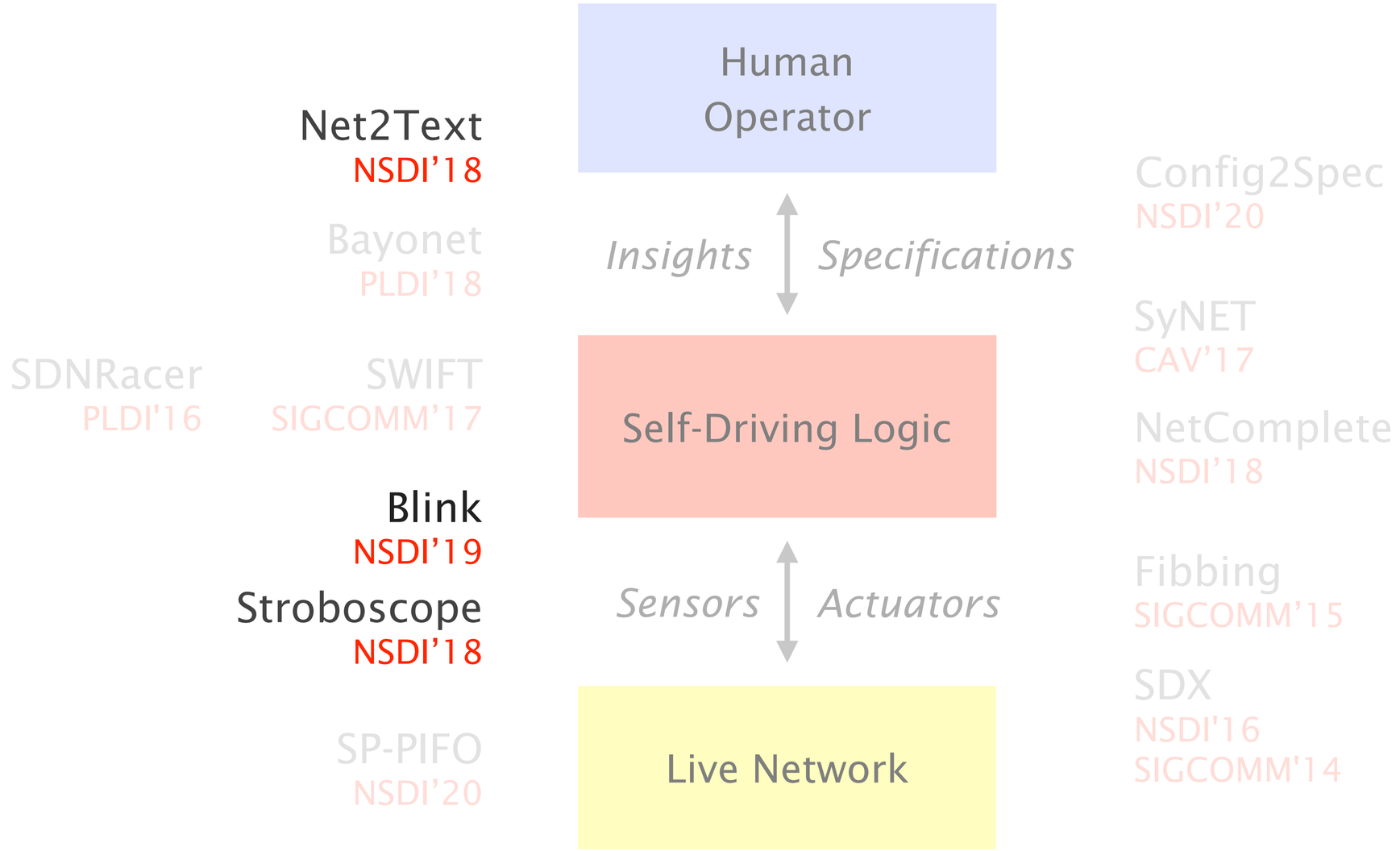
Part II
taking control

5 full automation

too futuristic? 


Human
Operator
Insights
Specifications
Self-Driving Logic
Sensors
Actuators
Live Network


Human Operator
Insights
Specifications
Self-Driving Logic
Sensors
Actuators
Live Network
Net2Text
NSDI'18
Bayonet
PLDI'18
SDNRacer
PLDI'16
SWIFT
SIGCOMM'17
Blink
NSDI'19
Stroboscope
NSDI'18
SP-PIFO
NSDI'20
Config2Spec
NSDI'20
SyNET
CAV'17
NetComplete
NSDI'18
Fibbing
SIGCOMM'15
SDX
NSDI'16
SIGCOMM'14


Human Operator
Insights
Specifications
Self-Driving Logic
Sensors
Actuators
Live Network
Net2Text
NSDI'18
Bayonet
PLDI'18
SDNRacer
PLDI'16
SWIFT
SIGCOMM'17
Blink
NSDI'19
Stroboscope
NSDI'18
SP-PIFO
NSDI'20
Config2Spec
NSDI'20
SyNET
CAV'17
NetComplete
NSDI'18
Fibbing
SIGCOMM'15
SDX
NSDI'16
SIGCOMM'14

Self-driving/monitoring networks

in the age of deep network programmability

1 operator assistance

2 partial automation

Part I
assisting operators

3 conditional automation

4 high automation

Part II
taking control

5 full automation

too futuristic? 

Self-driving/monitoring networks

in the age of deep network programmability

1	operator assistance	Part I assisting operators
2	partial automation	
3	conditional automation	Part II taking control
4	high automation	
5	full automation	too futuristic? ☺

Stroboscope: Declarative Network Monitoring on a Budget

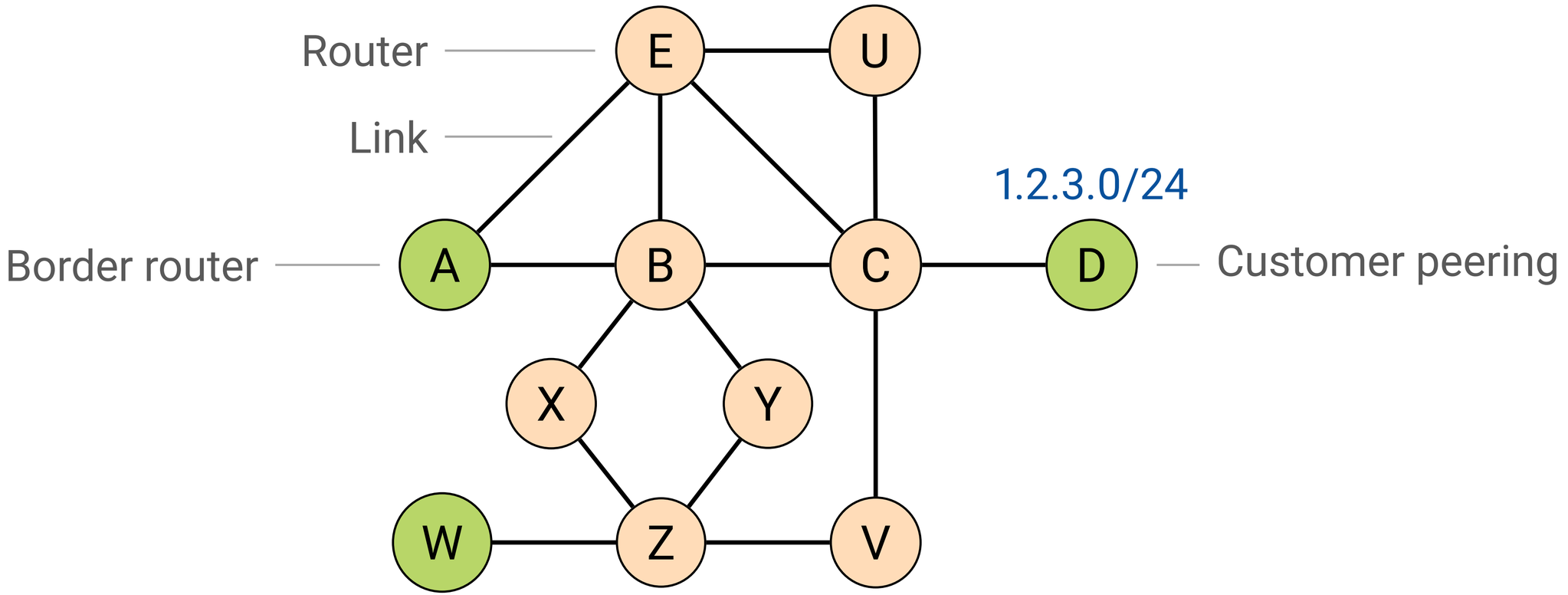




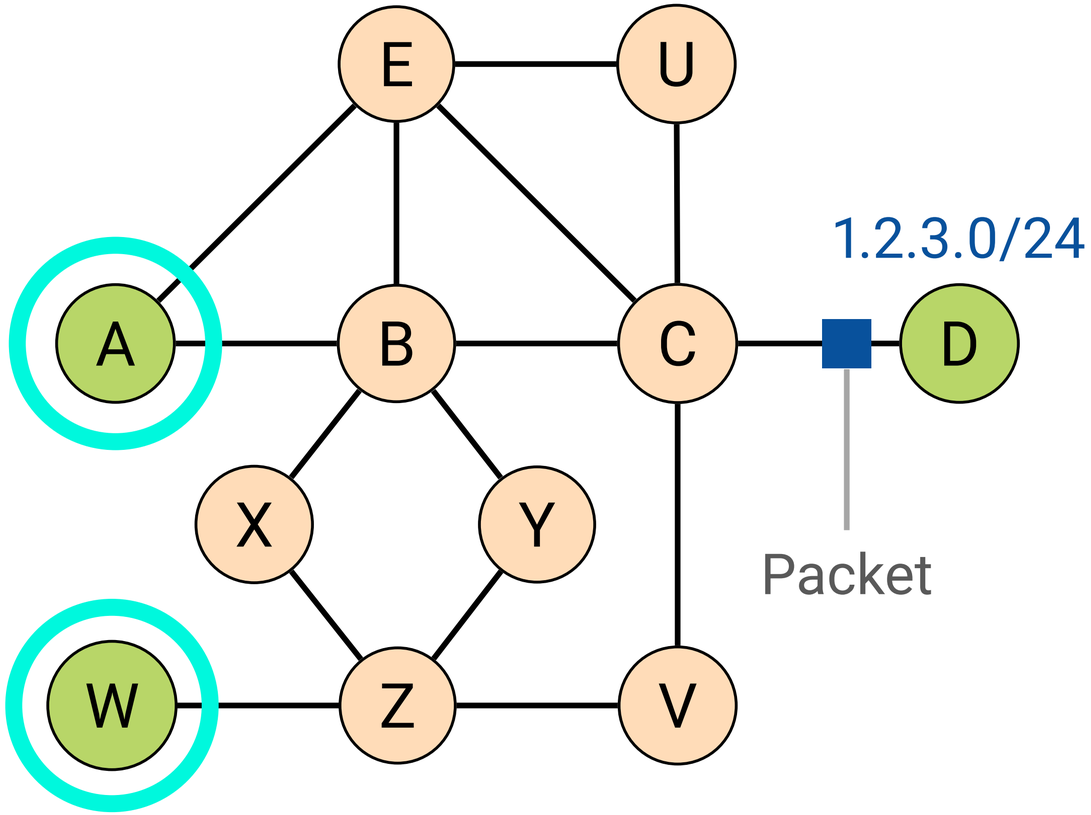
Olivier Tilmans Tobias Bühler Ingmar Poese Stefano Vissicchio Laurent Vanbever

USENIX Symposium on Networked Systems Design and Implementation. April 2018.

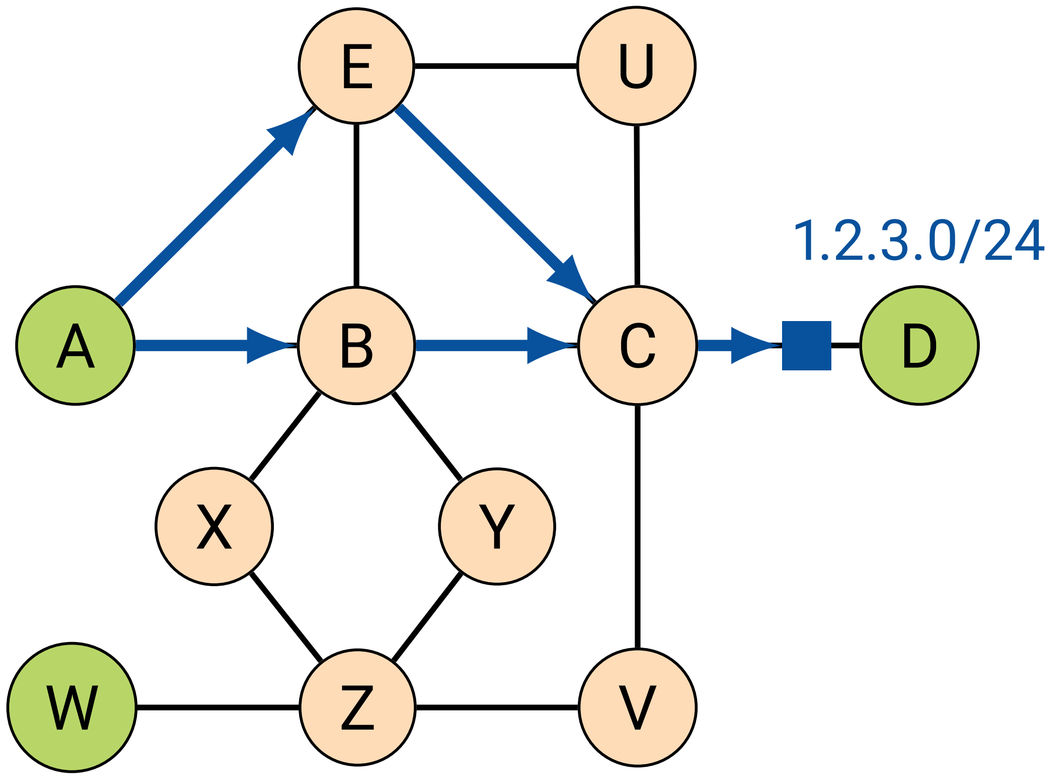
Consider this example ISP network topology



What is the ingress router for this packet arriving at router D?

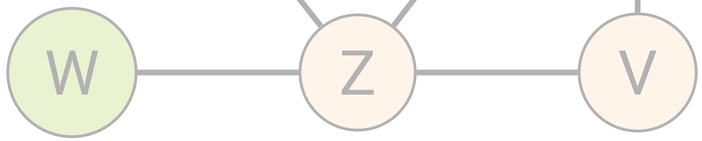


Which paths does the traffic follow?

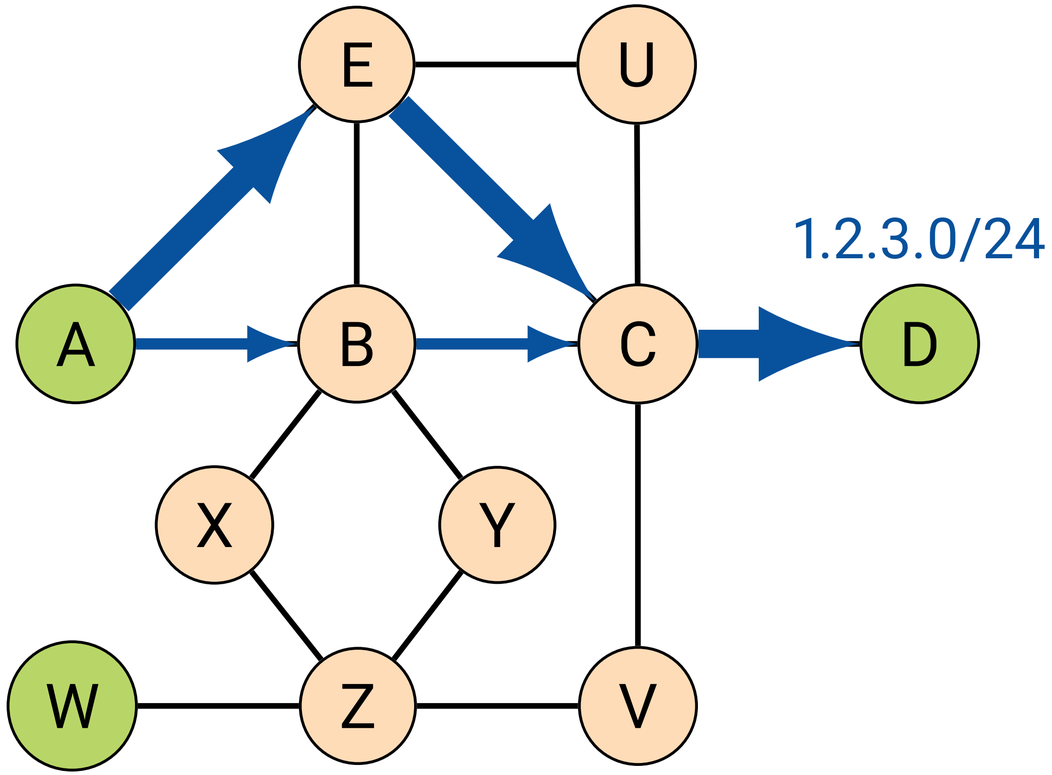


Which paths does the traffic follow?

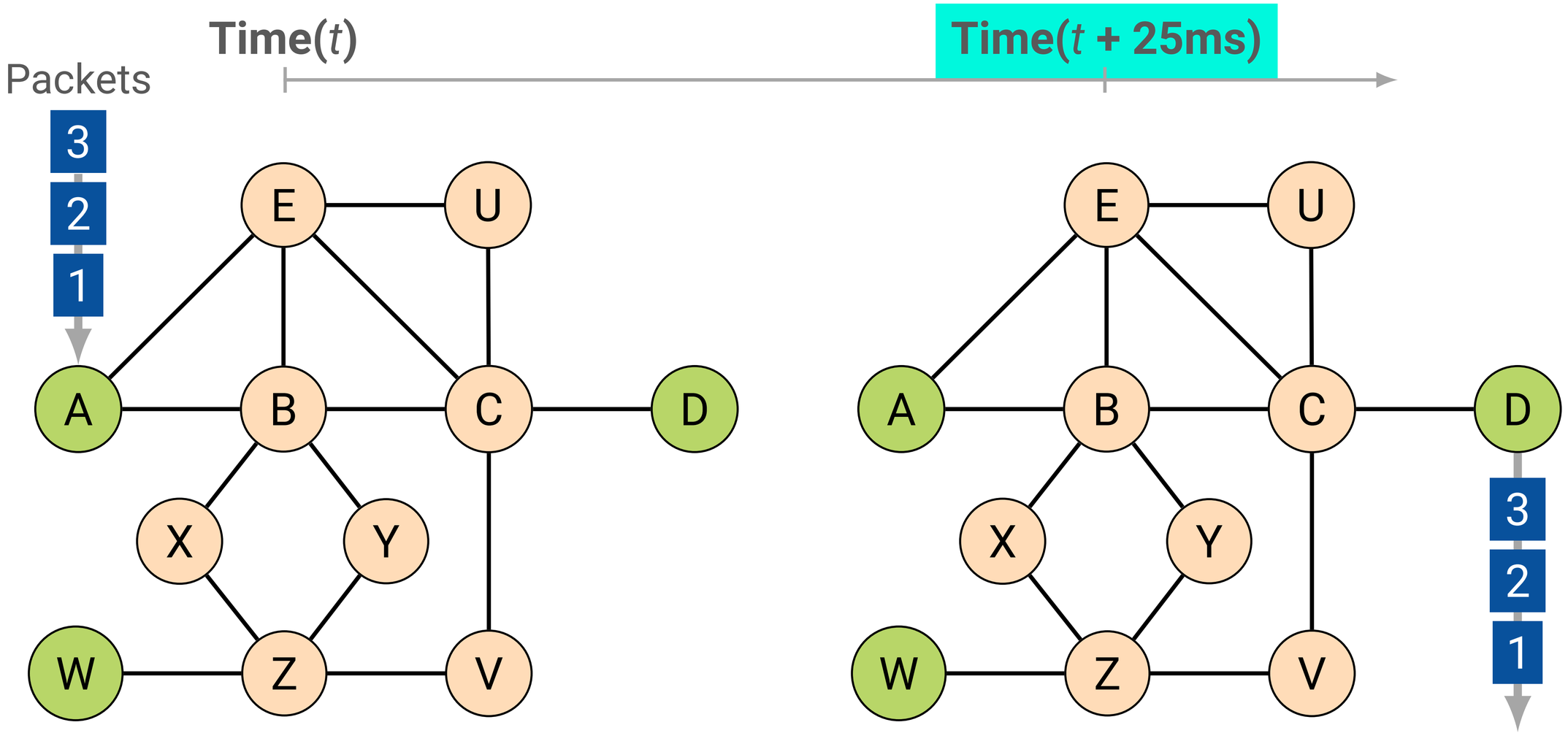
Tracking flows network-wide requires to
match measurements across multiple vantage points



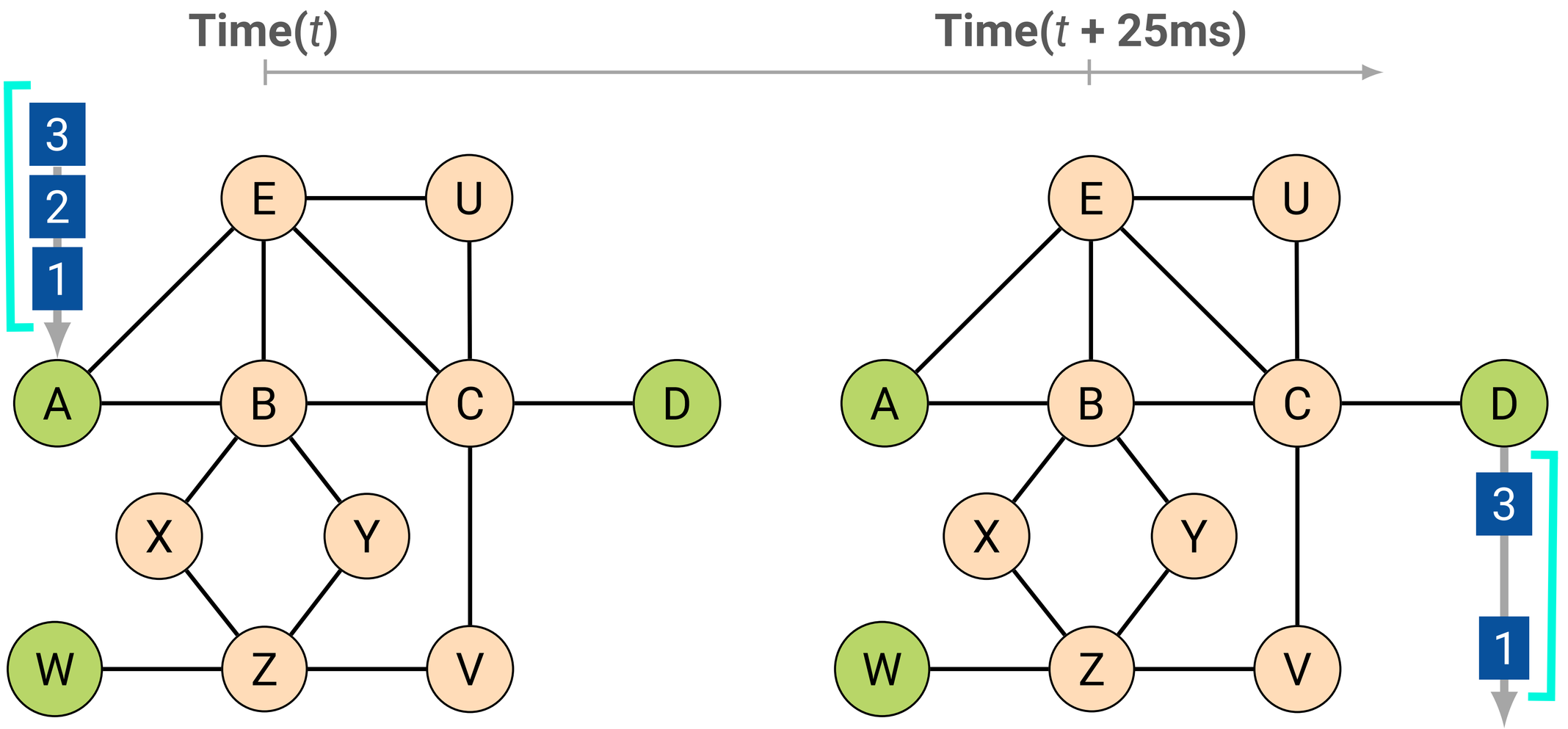
Is traffic load-balanced as expected?



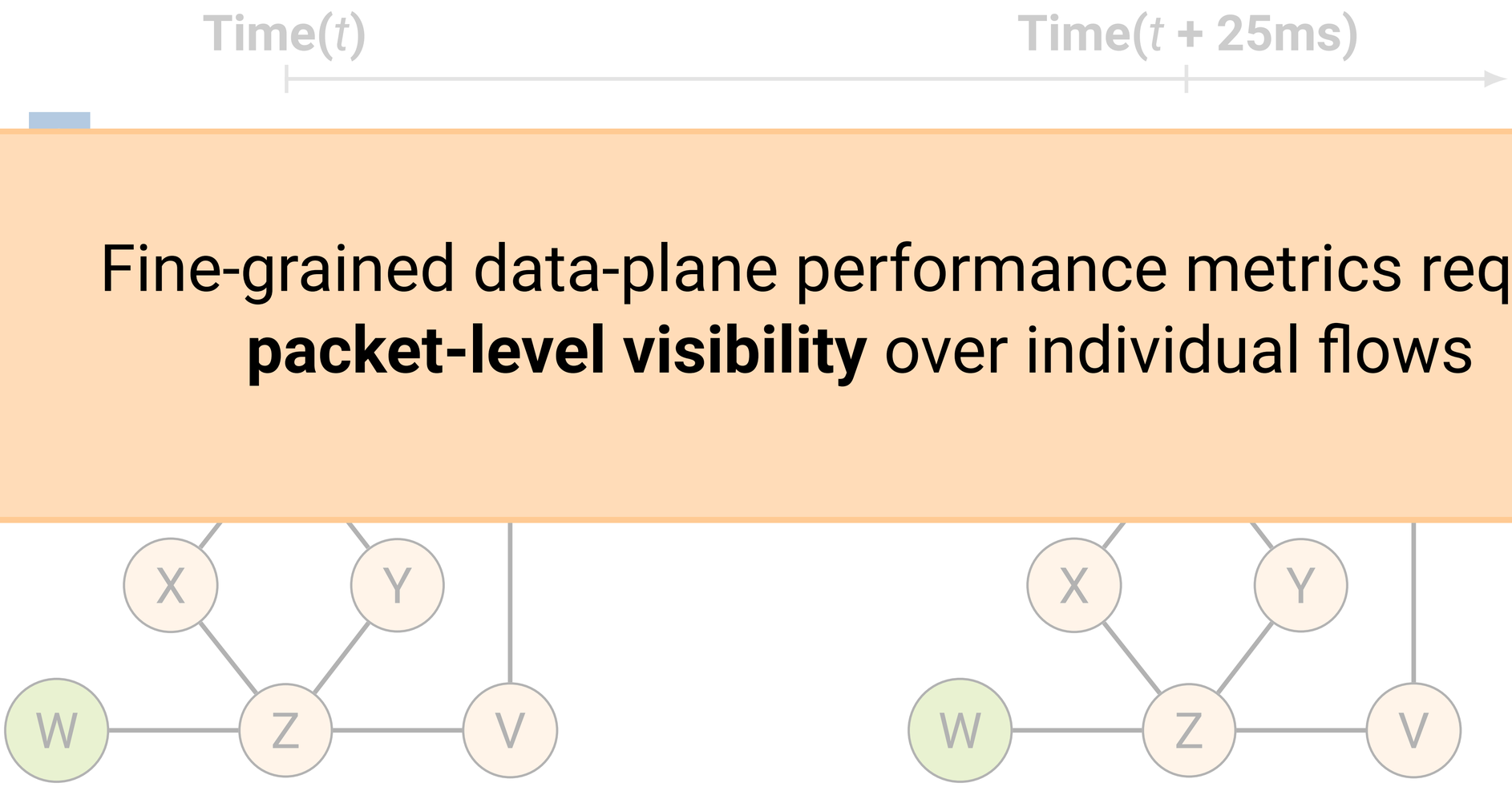
Is the latency acceptable?



Are there losses?



Are there losses?



Fine-grained data-plane performance metrics require **packet-level visibility** over individual flows

Fined-grained network monitoring is widely researched

Gigascope [SIGMOD'03]

Planck [SIGCOMM'14]

Everflow [SIGCOMM'15]

Compiling Path Queries [NSDI'16]

Trumpet [SIGCOMM'16]

Marple [SIGCOMM'17]

Fined-grained ISP network monitoring poses unique and unmet challenges

- No control over end hosts

Gigascope [SIGMOD'03]

Planck [SIGCOMM'14]

Everflow [SIGCOMM'15]

Compiling Path Queries [NSDI'16]

~~Trumpet [SIGCOMM'16]~~

Marple [SIGCOMM'17]

Fined-grained ISP network monitoring poses unique and unmet challenges

- No control over end hosts
- Limited data-plane flexibility

Gigascope [SIGMOD'03]

Planck [SIGCOMM'14]

Everflow [SIGCOMM'15]

~~Compiling Path Queries [NSDI'16]~~

~~Trumpet [SIGCOMM'16]~~

~~Marple [SIGCOMM'17]~~

Fined-grained ISP network monitoring poses unique and unmet challenges

- No control over end hosts
- Limited data-plane flexibility
- Limited monitoring bandwidth

~~Gigascope [SIGMOD'03]~~

~~Planck [SIGCOMM'14]~~

~~Everflow [SIGCOMM'15]~~

~~Compiling Path Queries [NSDI'16]~~

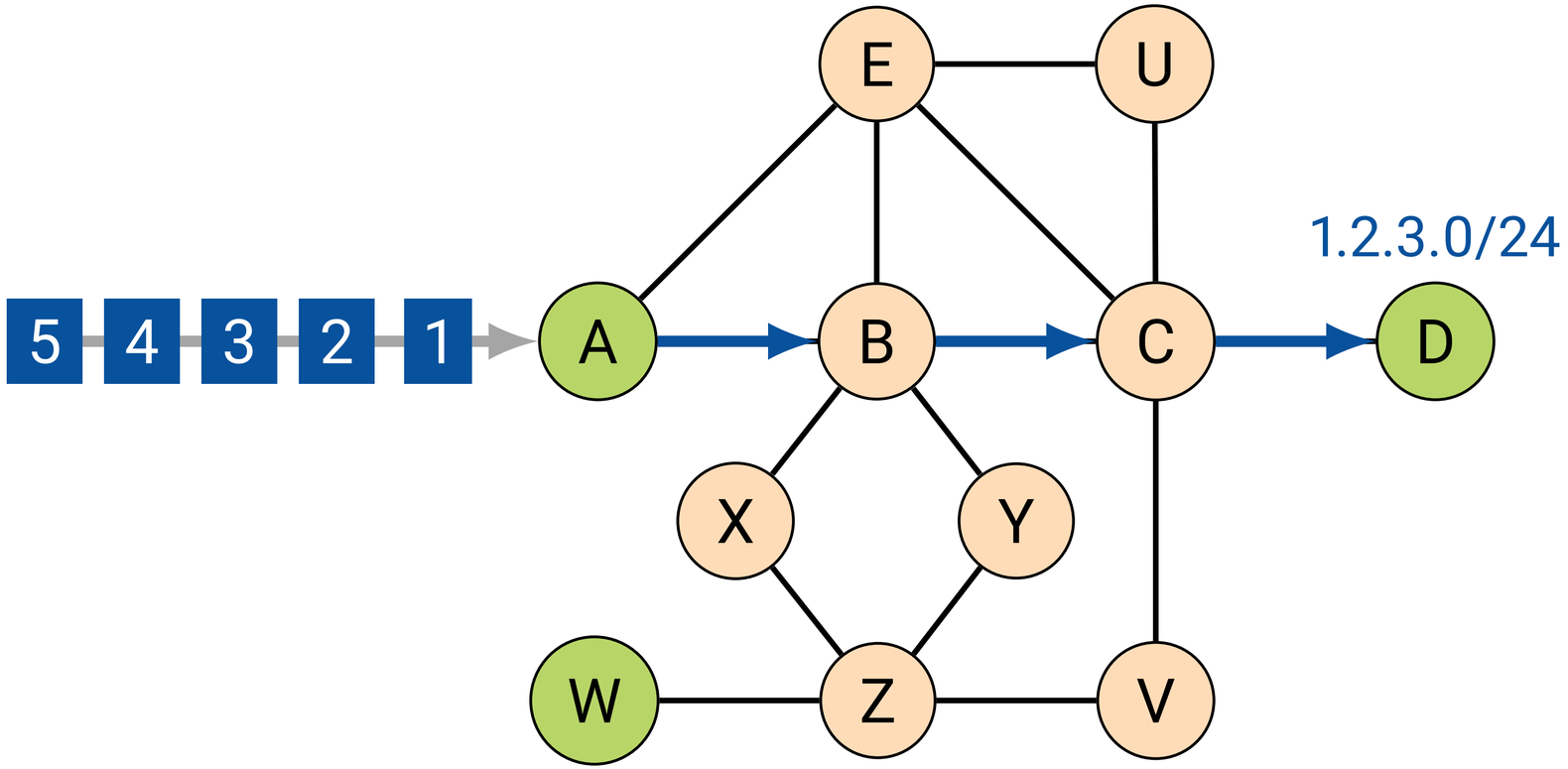
~~Trumpet [SIGCOMM'16]~~

~~Marple [SIGCOMM'17]~~

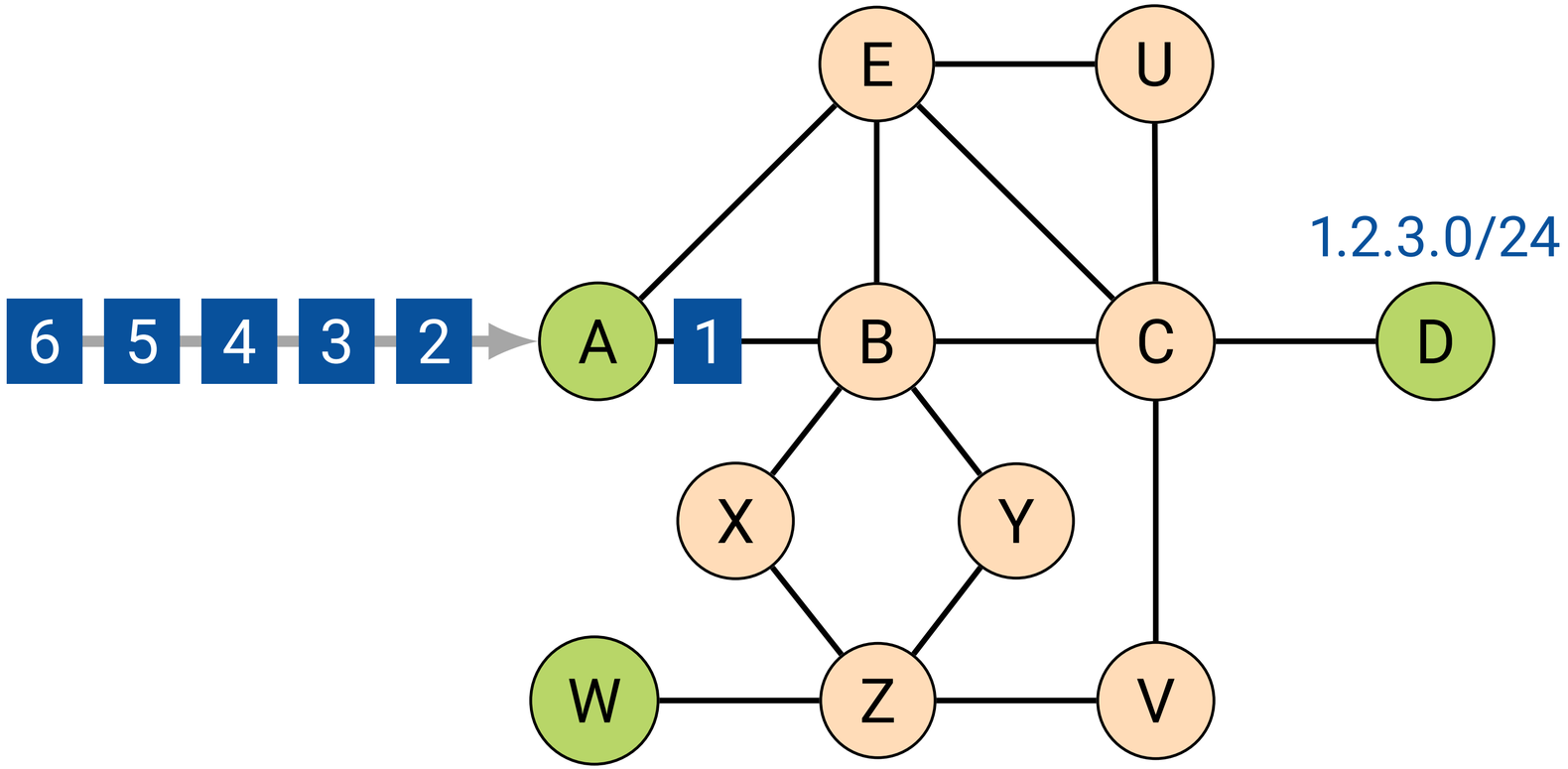
Stroboscope: Declarative Network Monitoring on a Budget

- Collecting traffic slices to monitor networks
- Adhering to a monitoring budget
- Using Stroboscope today

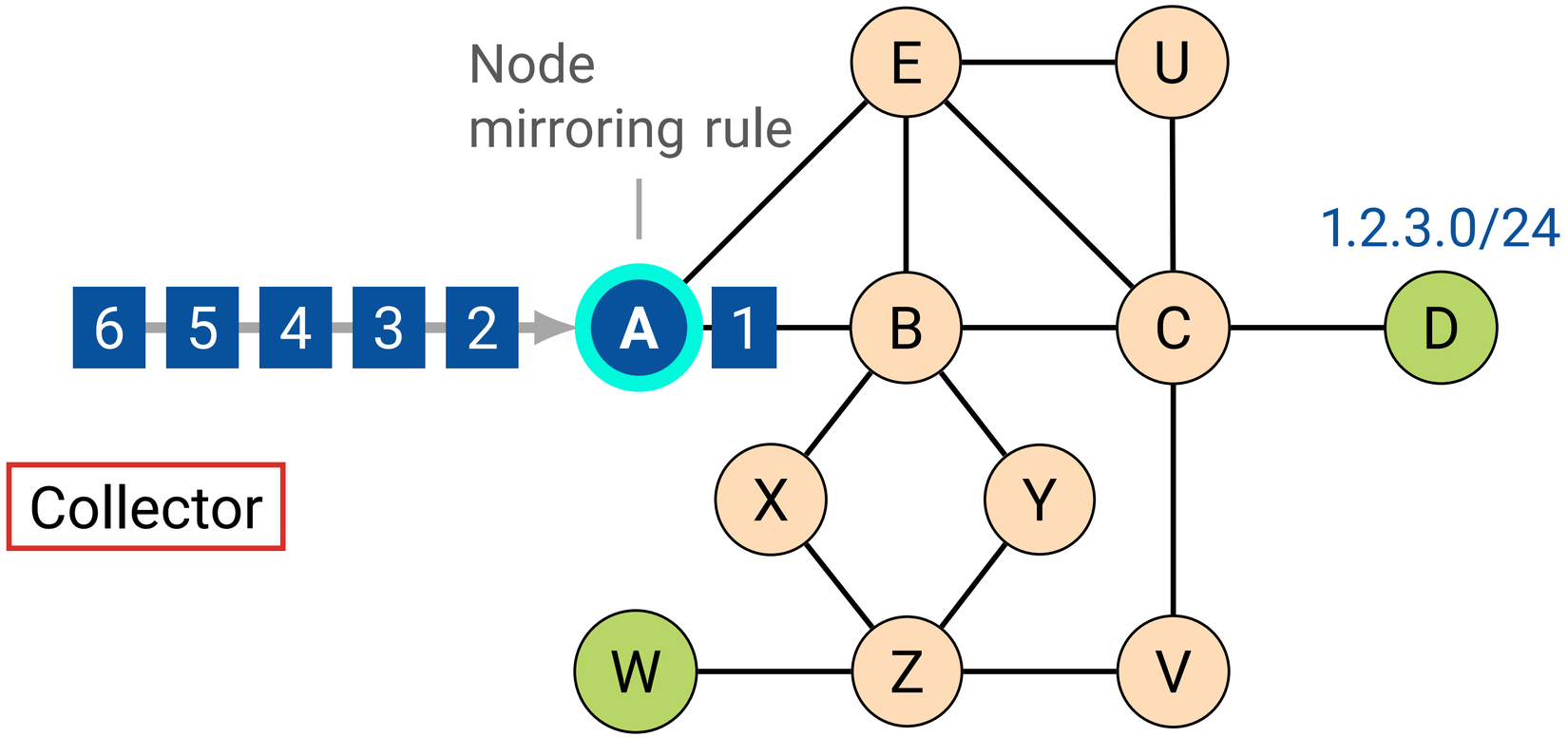
Consider the following flow of packets



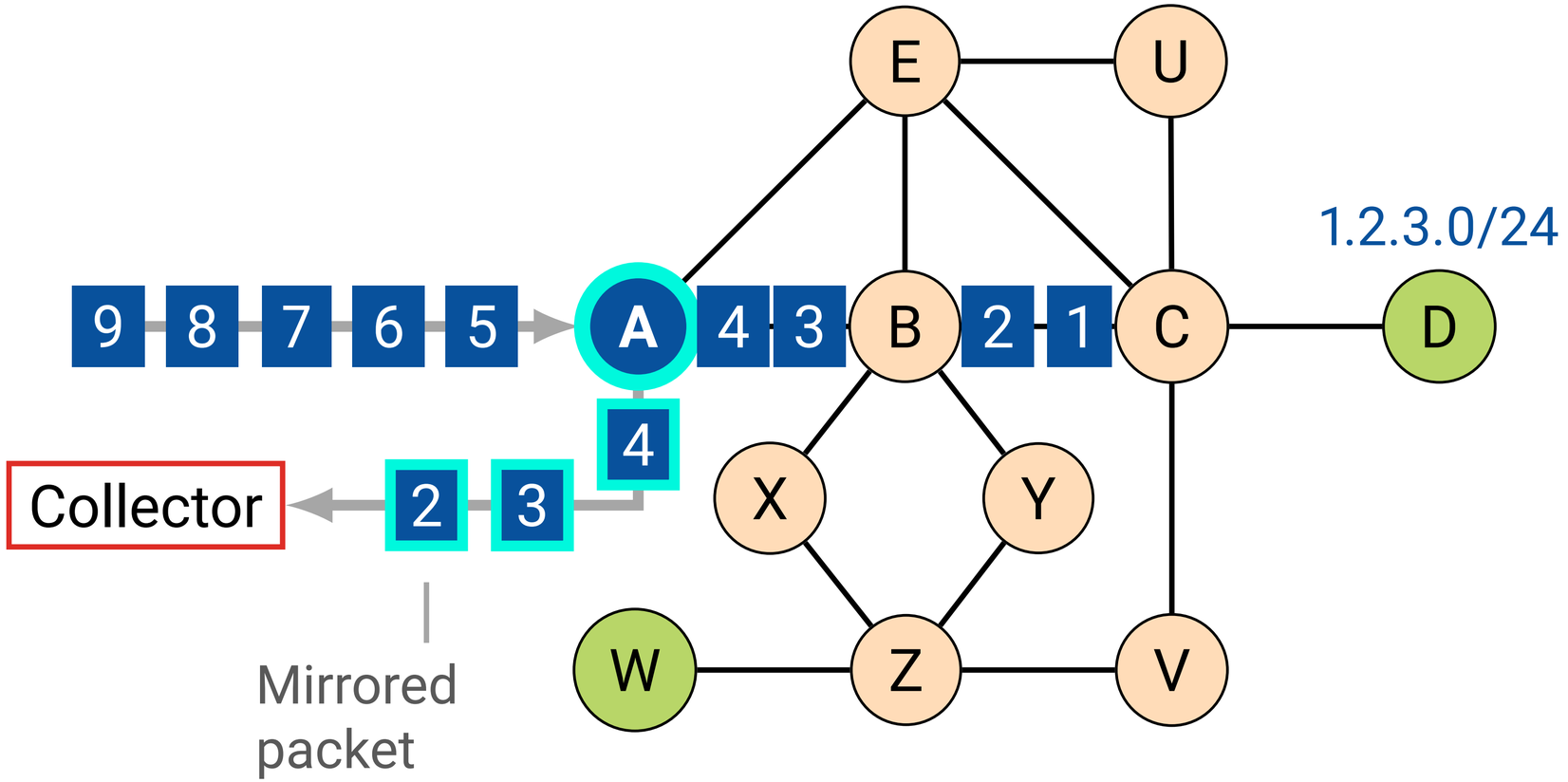
Consider the following flow of packets



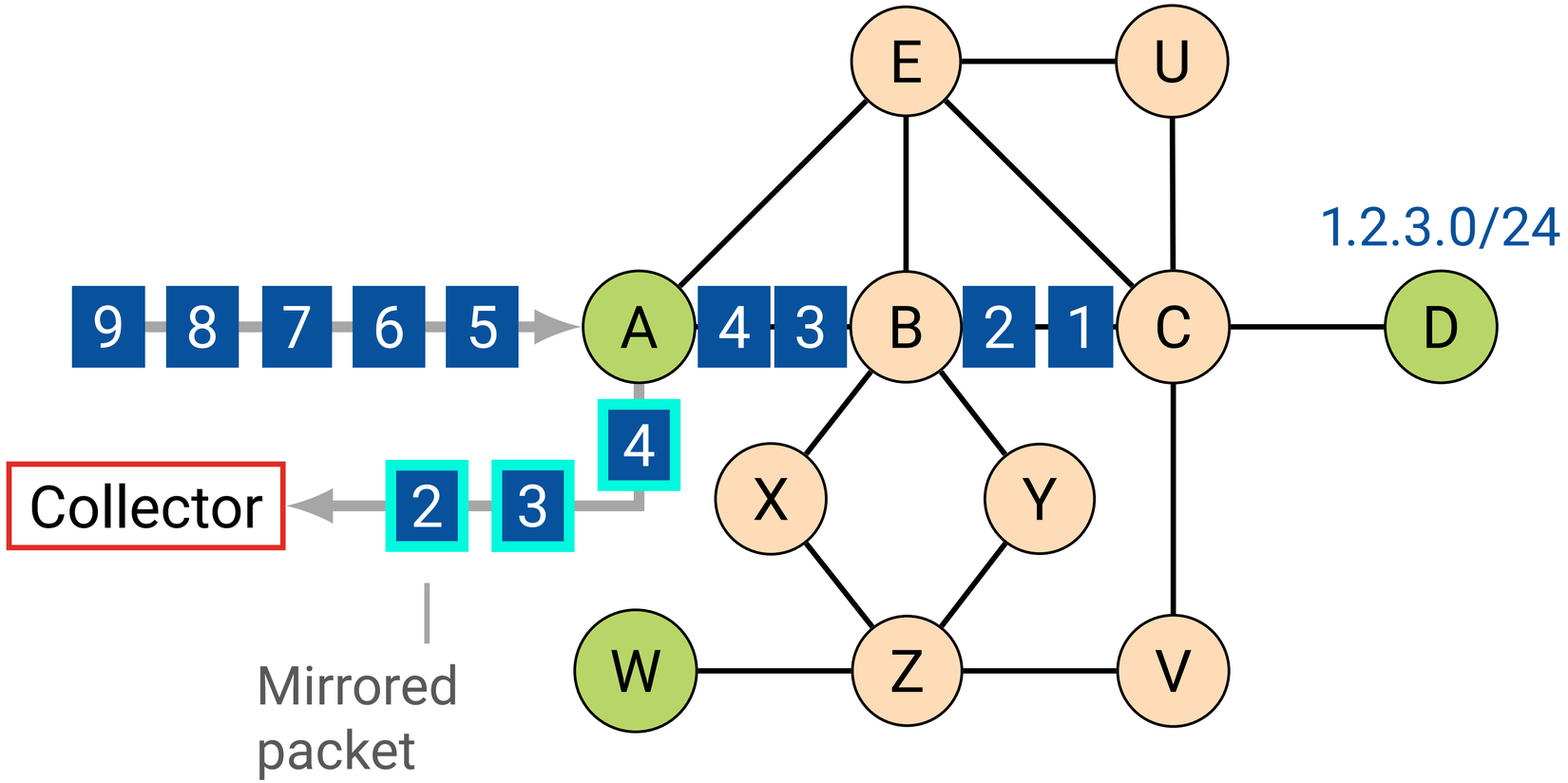
Stroboscope activates mirroring for the flow



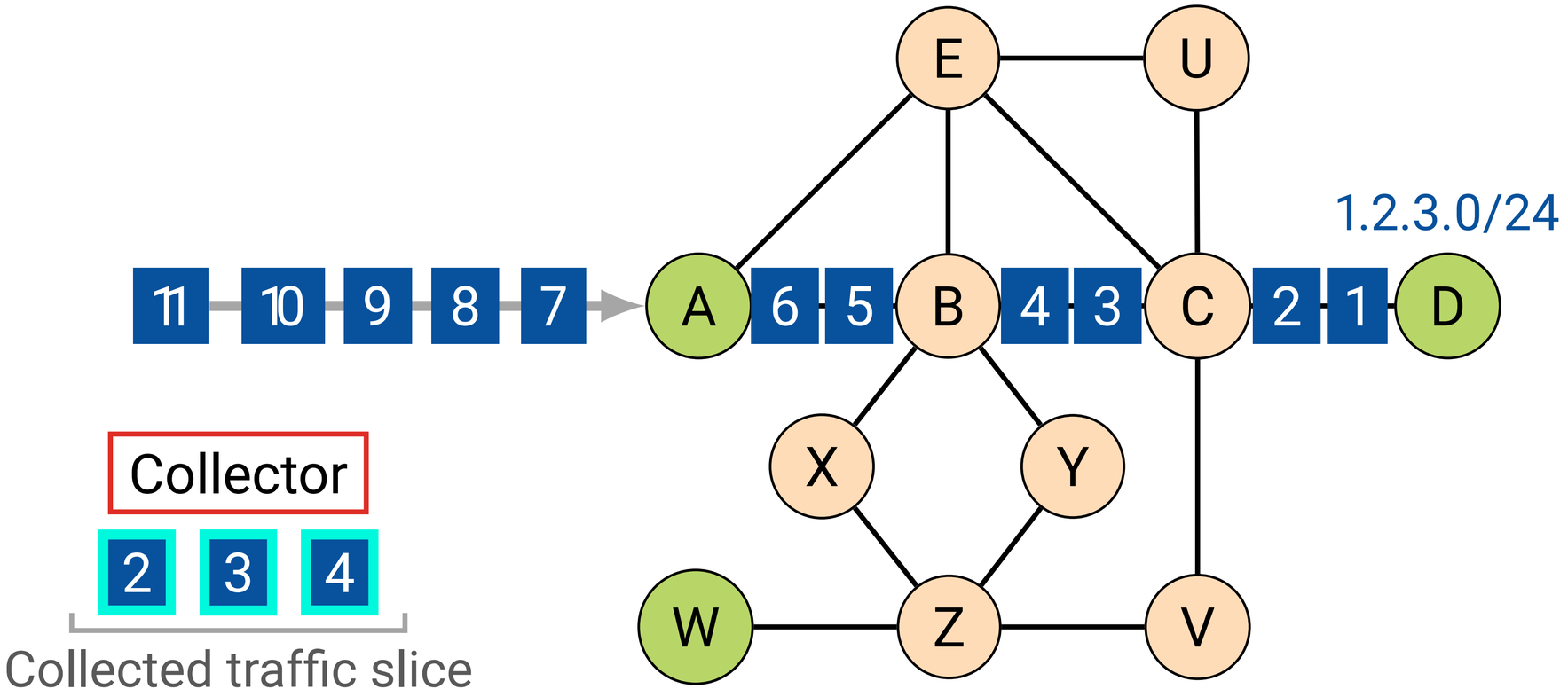
Packets are copied and encapsulated towards the collector



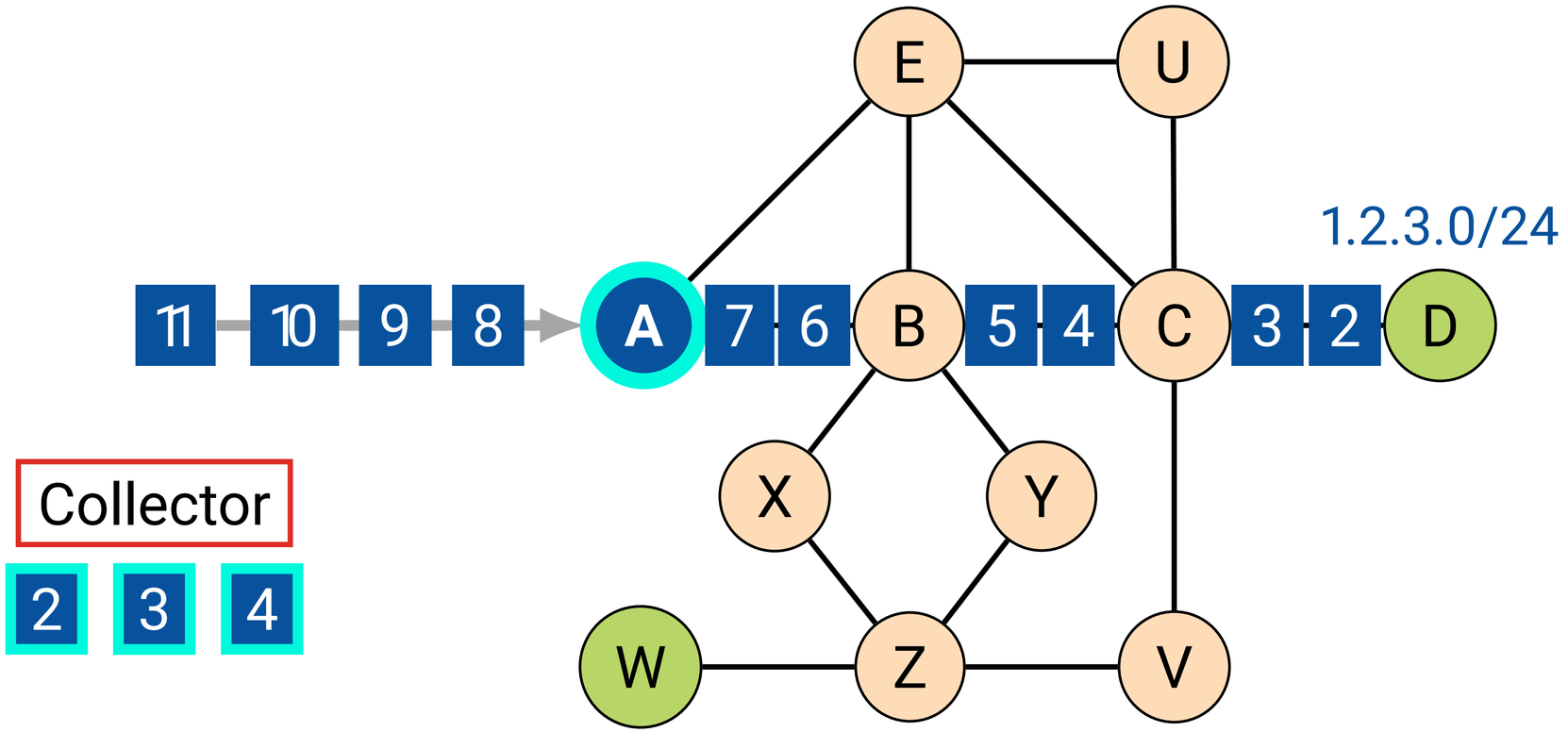
The mirroring rule is deactivated after a preset delay



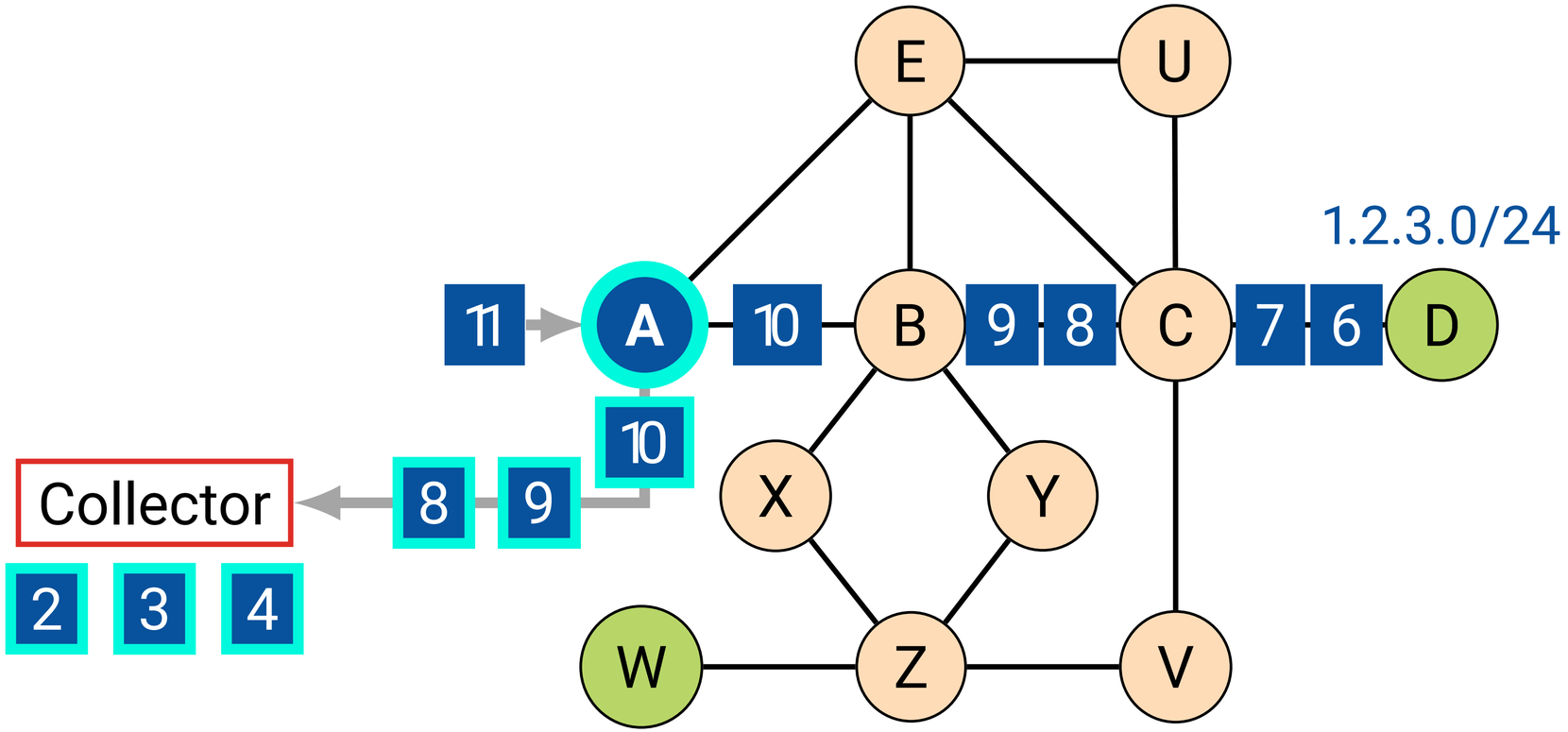
Stroboscope stores the traffic slice for analysis



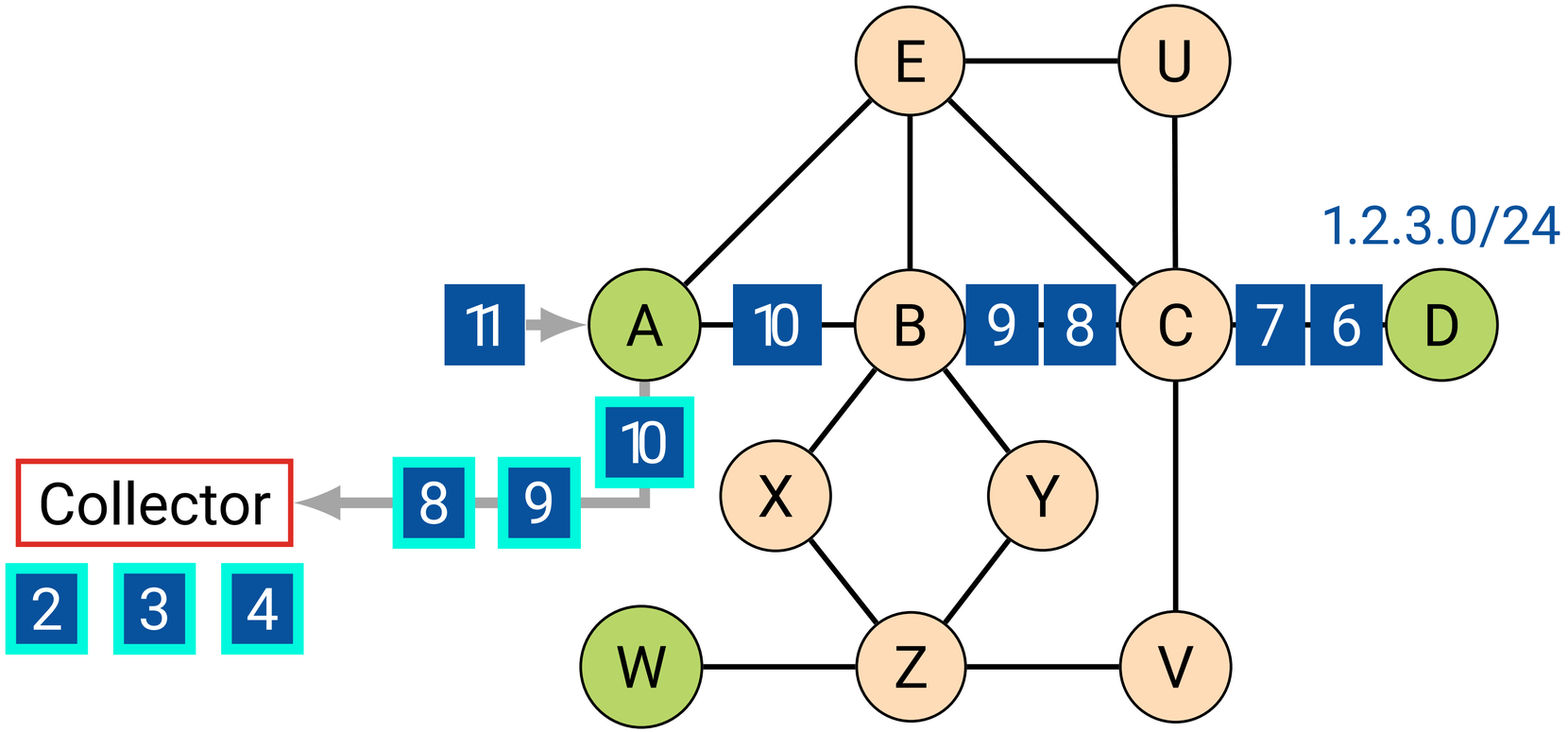
Stroboscope periodically toggles the mirroring rule



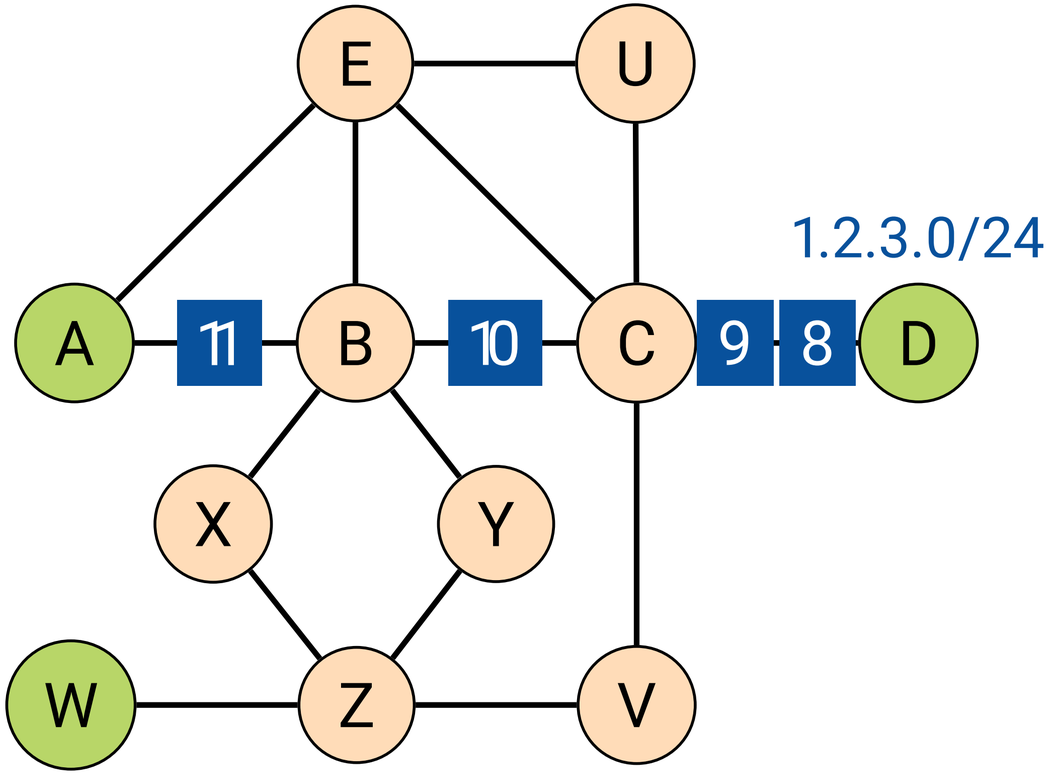
Stroboscope periodically toggles the mirroring rule



Stroboscope periodically toggles the mirroring rule



Stroboscope collects multiples traffic slices over time



Analyzing matching packets across traffic slices enables fine-grained measurements at scale

Analyzing matching packets across traffic slices enables fine-grained measurements at scale

Forwarding paths discovery, timestamp reconstruction, payload inspection, . . .

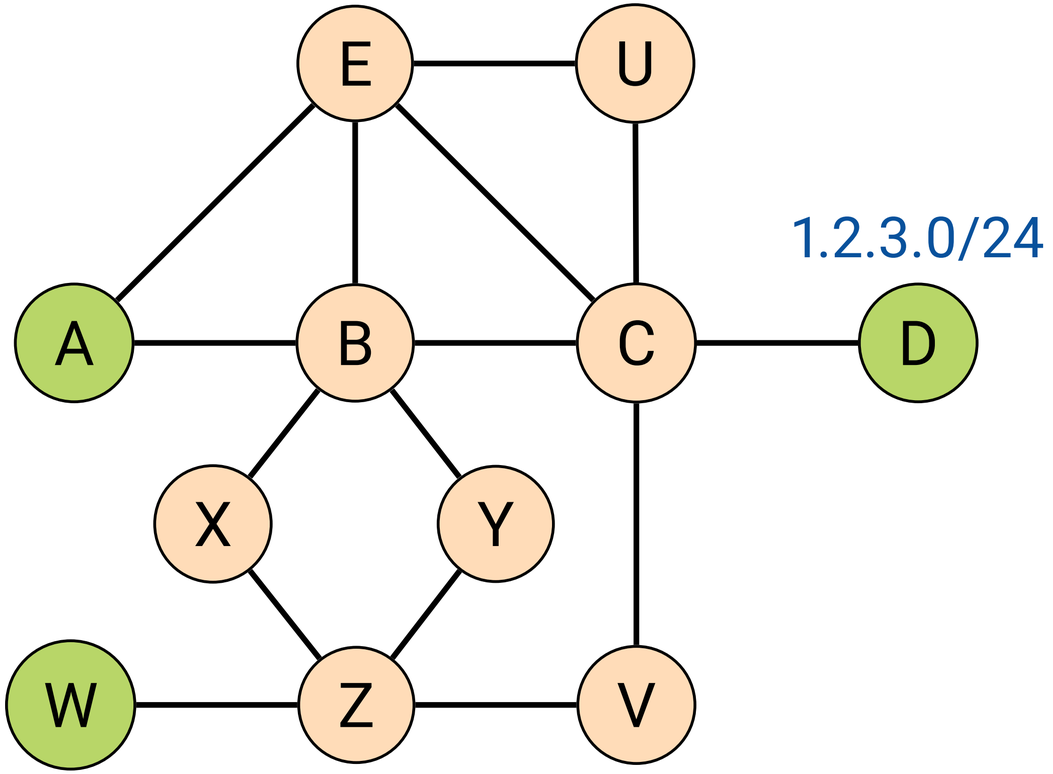
Stroboscope defines a declarative requirement language

```
MIRROR  1.2.3.0/24 ON [A B C D], [A E C D]
MIRROR  1.2.3.0/24 ON [A -> D]

CONFINE 1.2.3.0/24 ON [A B E C D]
CONFINE 1.2.3.0/24 [A -> D]

MIRROR  1.2.3.0/24 ON [ -> D]
CONFINE 1.2.3.0/24 ON [ -> D]

USING 15 Mbps DURING 500 ms EVERY 5 s
```



Stroboscope defines two types of queries

MIRROR

CONFINE

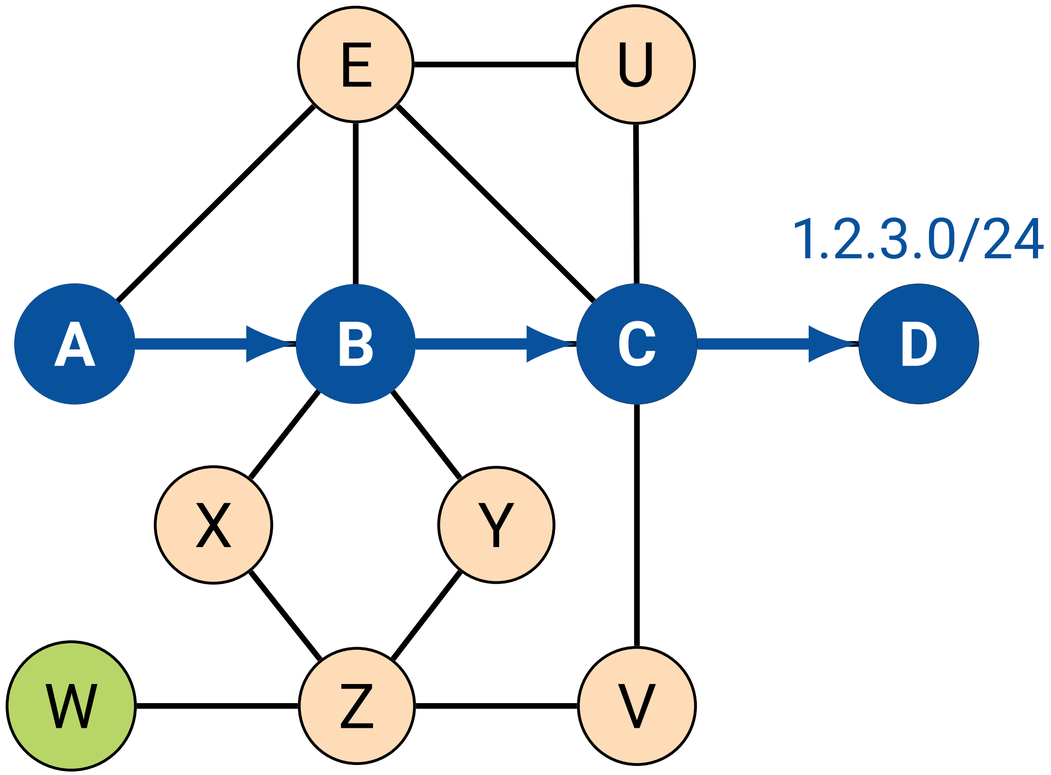
Stroboscope defines two types of queries

MIRROR

CONFINE

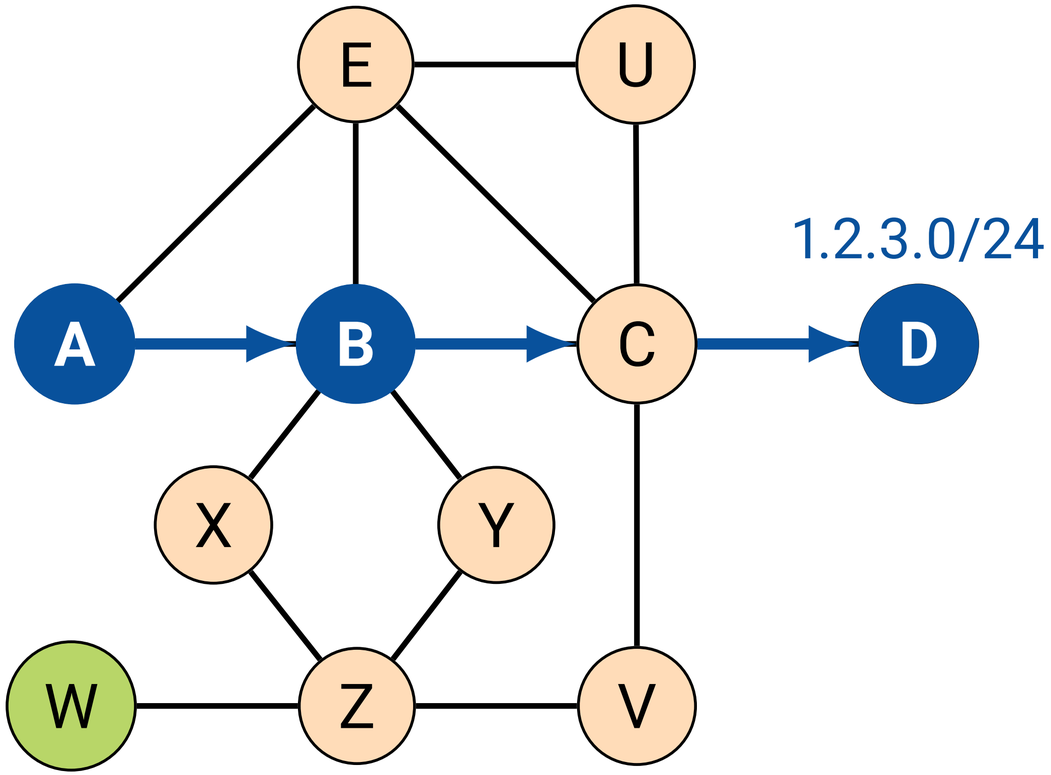
MIRROR queries reconstruct the path taken by packets

`MIRROR` 1.2.3.0/24 `ON` [A B C D]



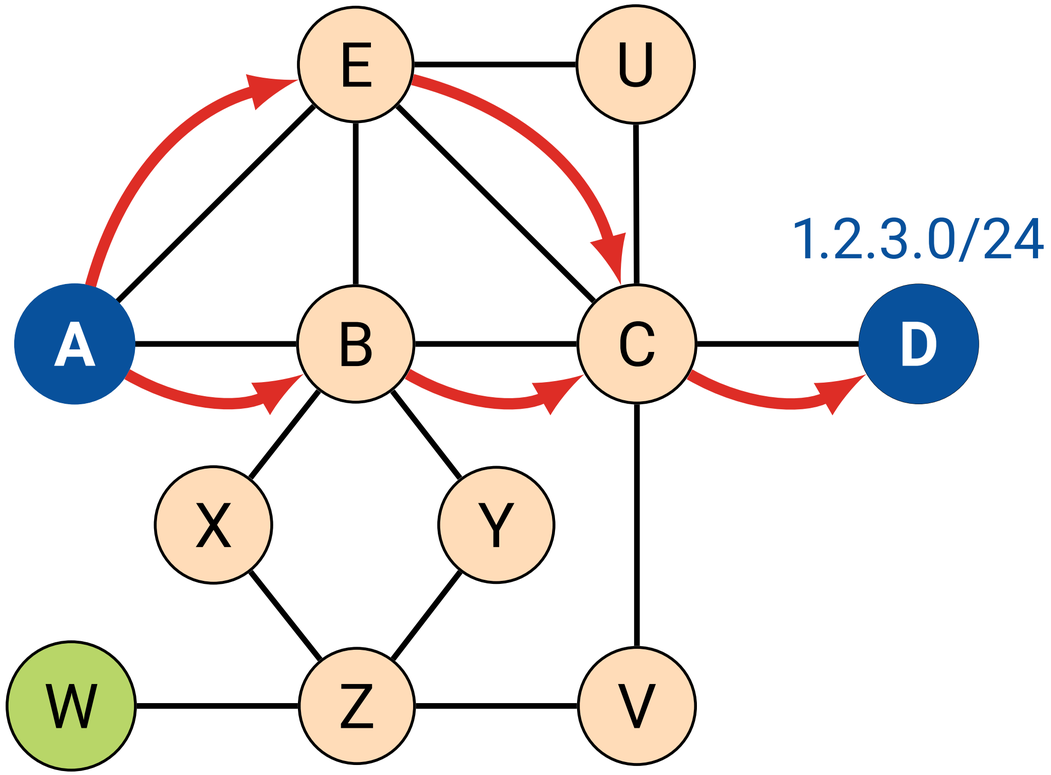
Fewer mirroring rules reduces bandwidth usage

`MIRROR` 1.2.3.0/24 `ON` [A B C D]



Too few mirroring rules creates ambiguity

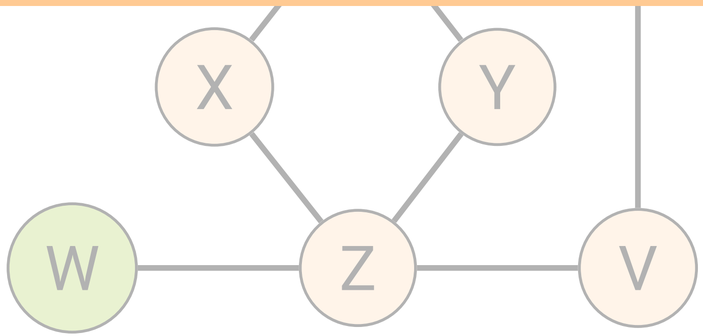
MIRROR 1.2.3.0/24 ON [A B C D]



Too few mirroring rules creates ambiguity

`MIRROR 1.2.3.0/24 ON [A B C D]`

The **Key-Points Sampling** algorithm minimizes mirroring rules and guarantees non-ambiguous reconstructed paths



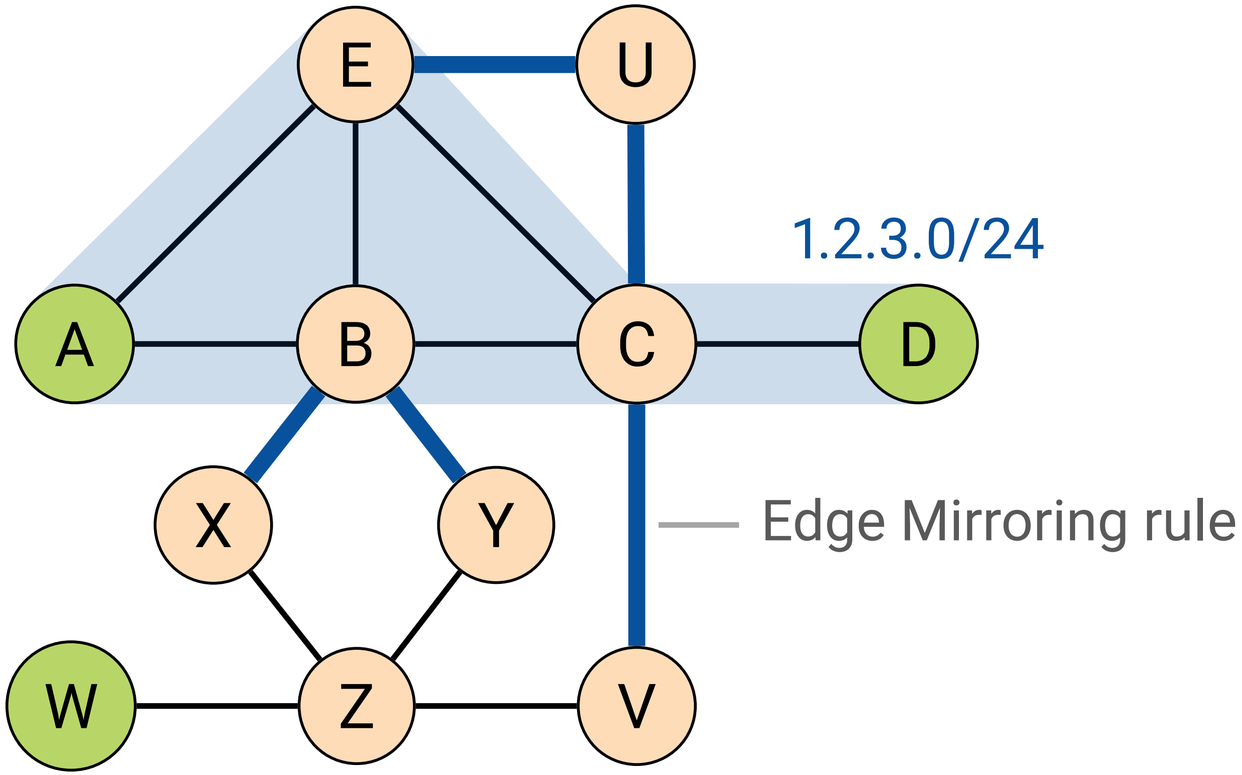
Stroboscope defines two types of queries

MIRROR

CONFINE

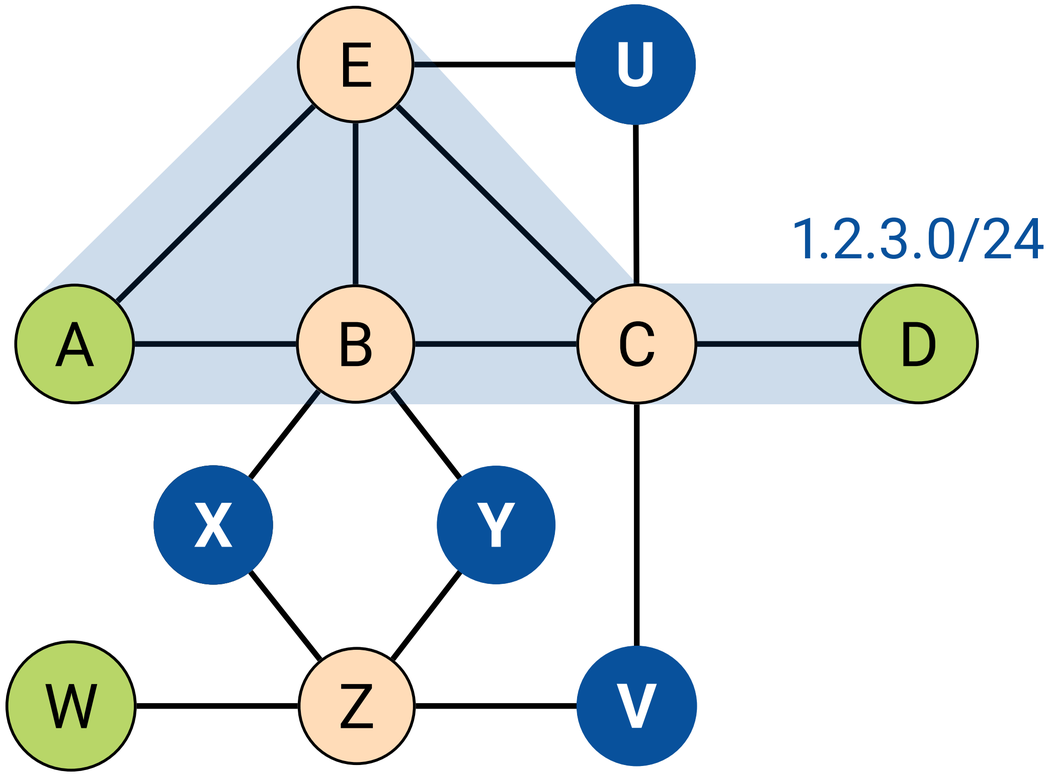
CONFINE queries mirror packets leaving a confinement region

CONFINE 1.2.3.0/24 ON [A B E C D]



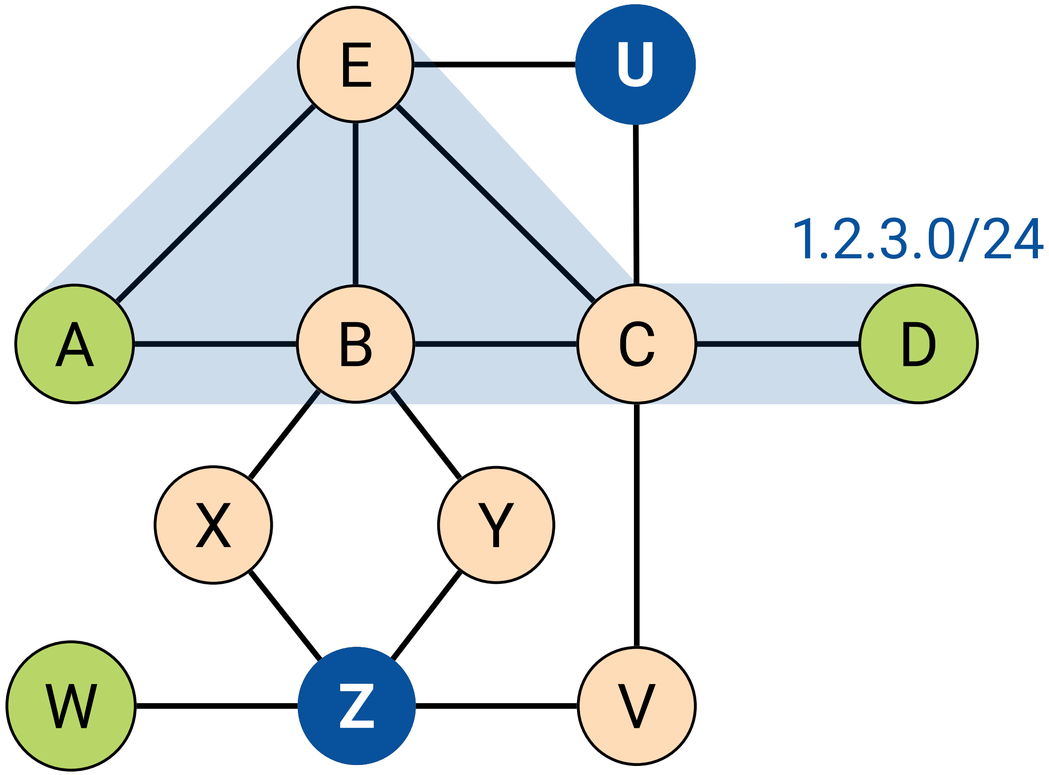
Fewer mirroring rules minimizes control-plane overhead

CONFINE 1.2.3.0/24 ON [A B E C D]



The lower bound is a multi-terminal node cut

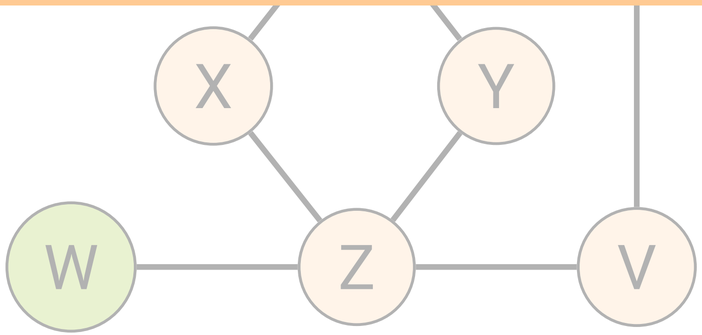
CONFINE 1.2.3.0/24 ON [A B E C D]



The lower bound is a multi-terminal node cut

`CONFINE` 1.2.3.0/24 `ON` [A B E C D]

The **Surrounding** algorithm minimizes mirroring rules and guarantees to mirror any packet leaving the confinement region



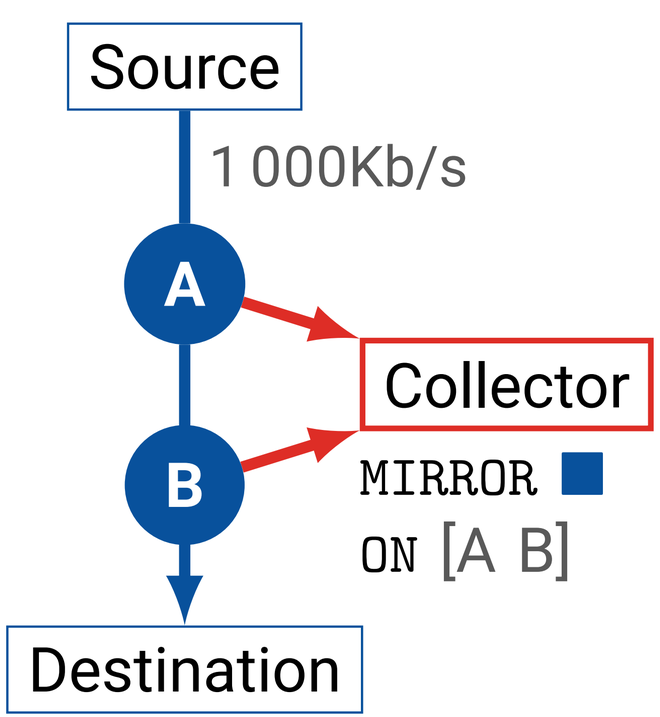
Stroboscope: Declarative Network Monitoring on a Budget

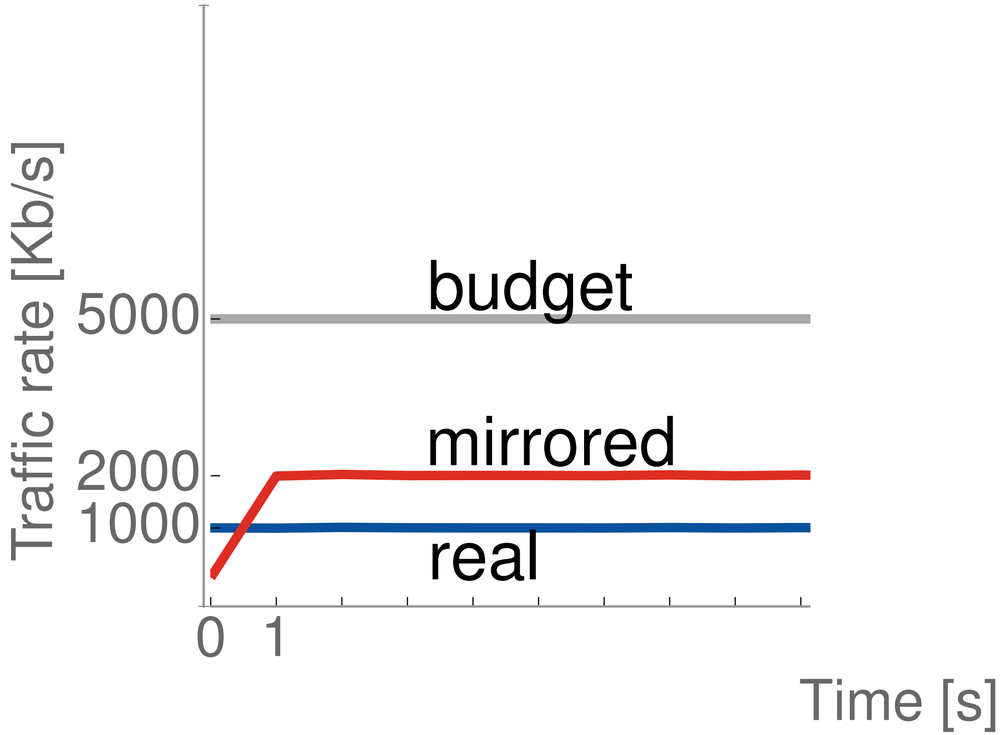
- Collecting traffic slices to monitor networks
- Adhering to a monitoring budget
- Using Stroboscope today

Stroboscope works with currently deployed routers

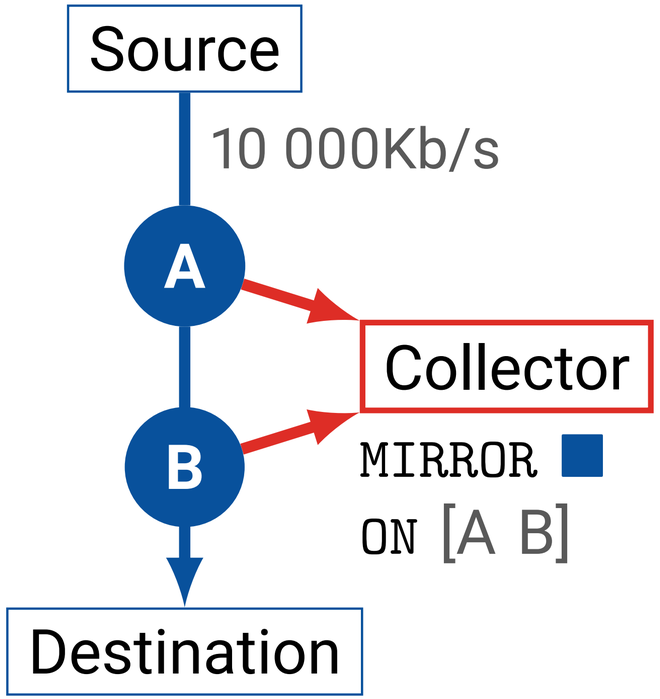
- Most vendors provide traffic mirroring and encapsulation primitives
- The collector activates mirroring for a flow by updating one ACL
- Routers autonomously deactivate mirroring rules using timers
- Traffic slices can be as small as **23 ms** on our routers (Cisco C7018)

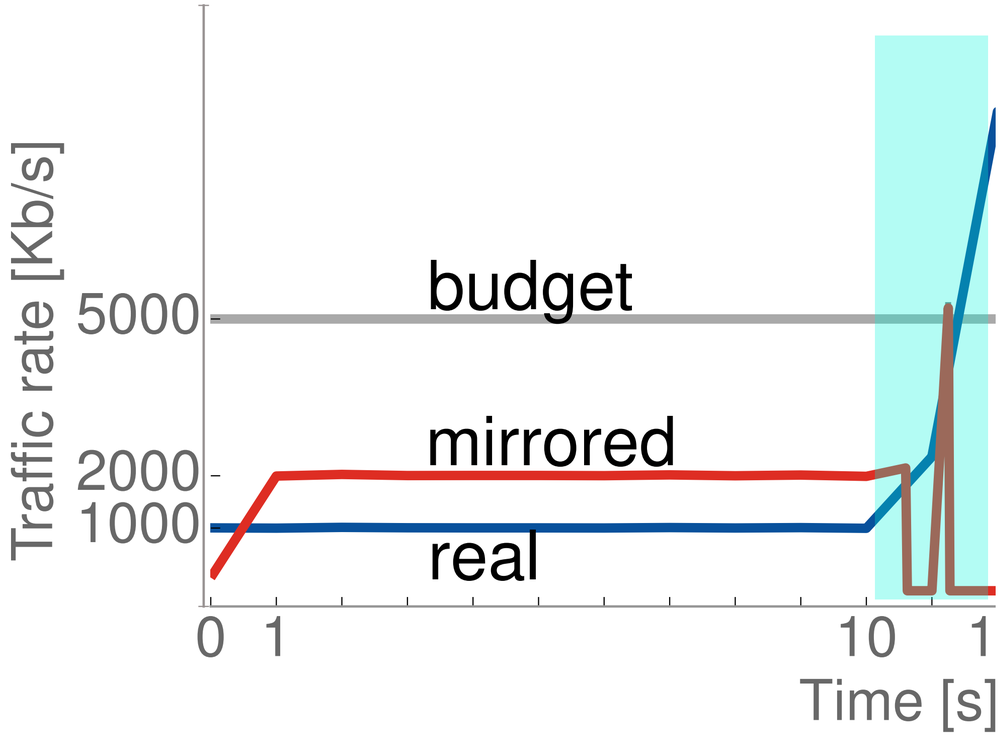
Stroboscope tracks the rate of mirrored traffic in real time



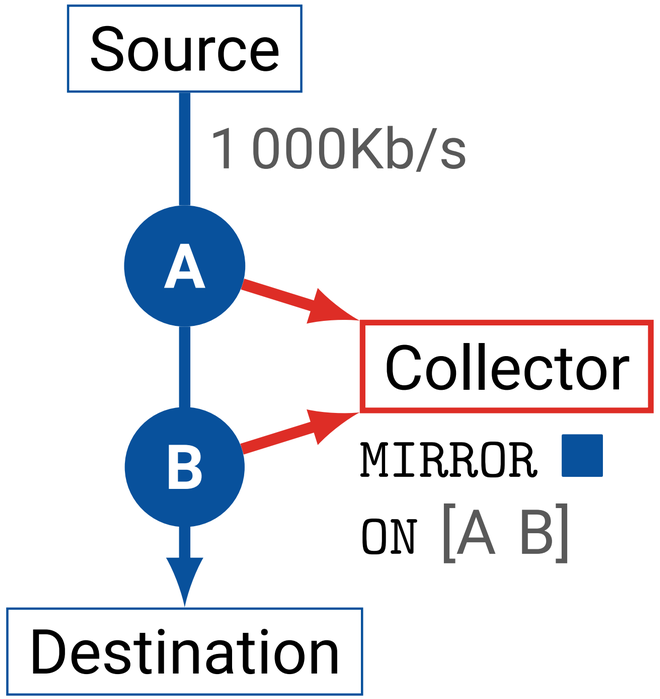


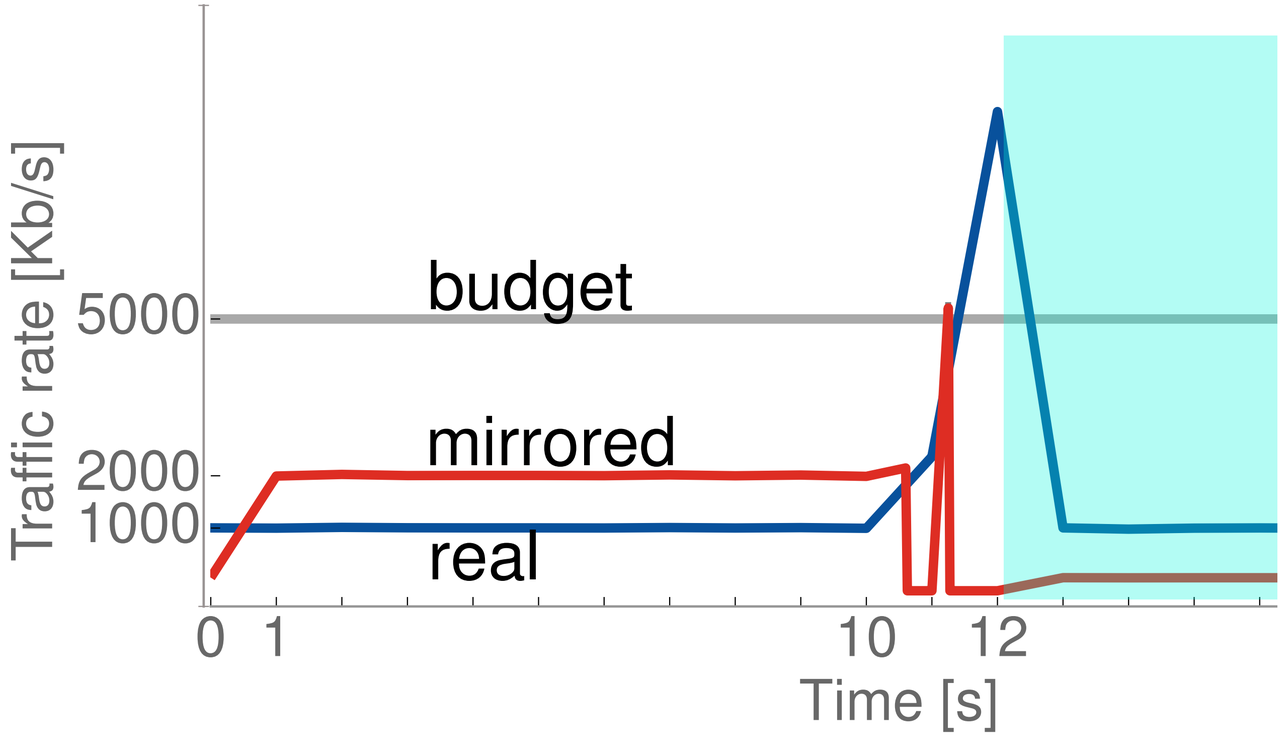
Measurement campaigns are stopped early if the estimated demand are exceeded



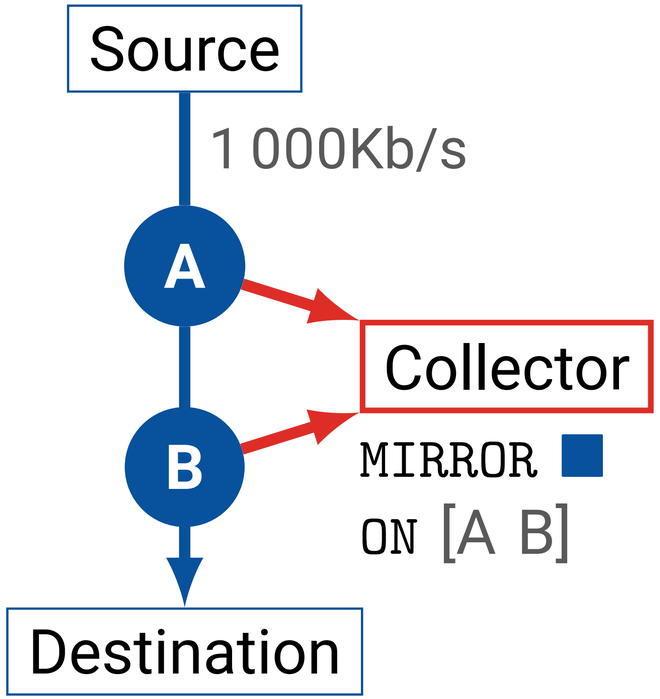


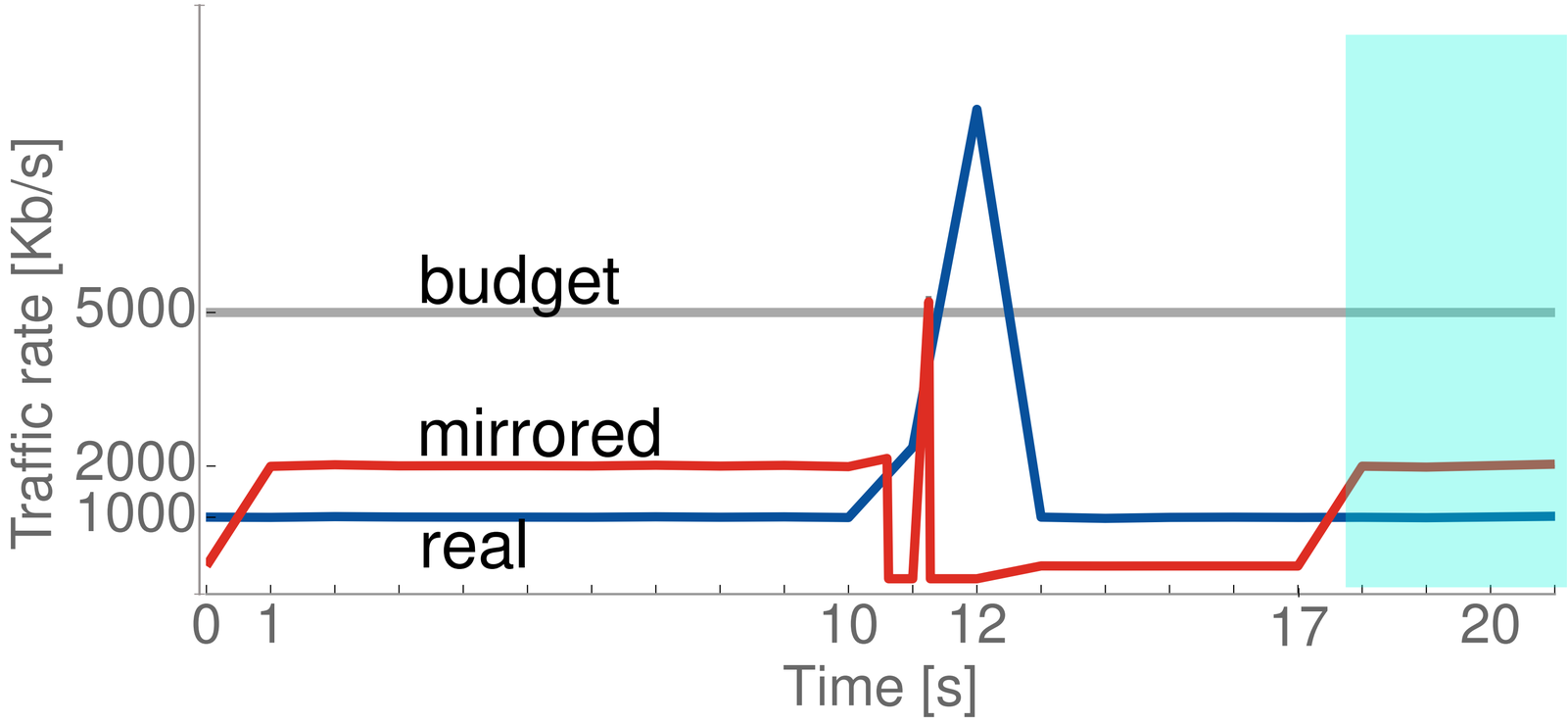
Exceeding the total budget schedules the query once per measurement campaign



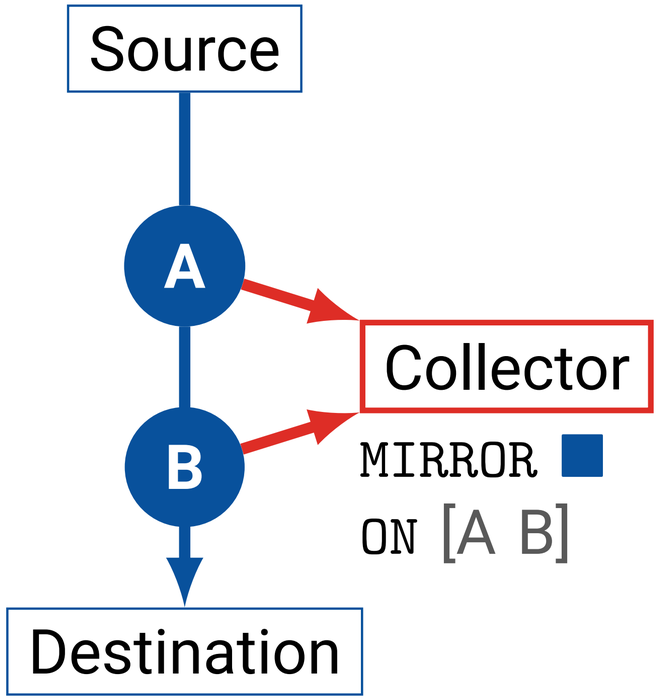


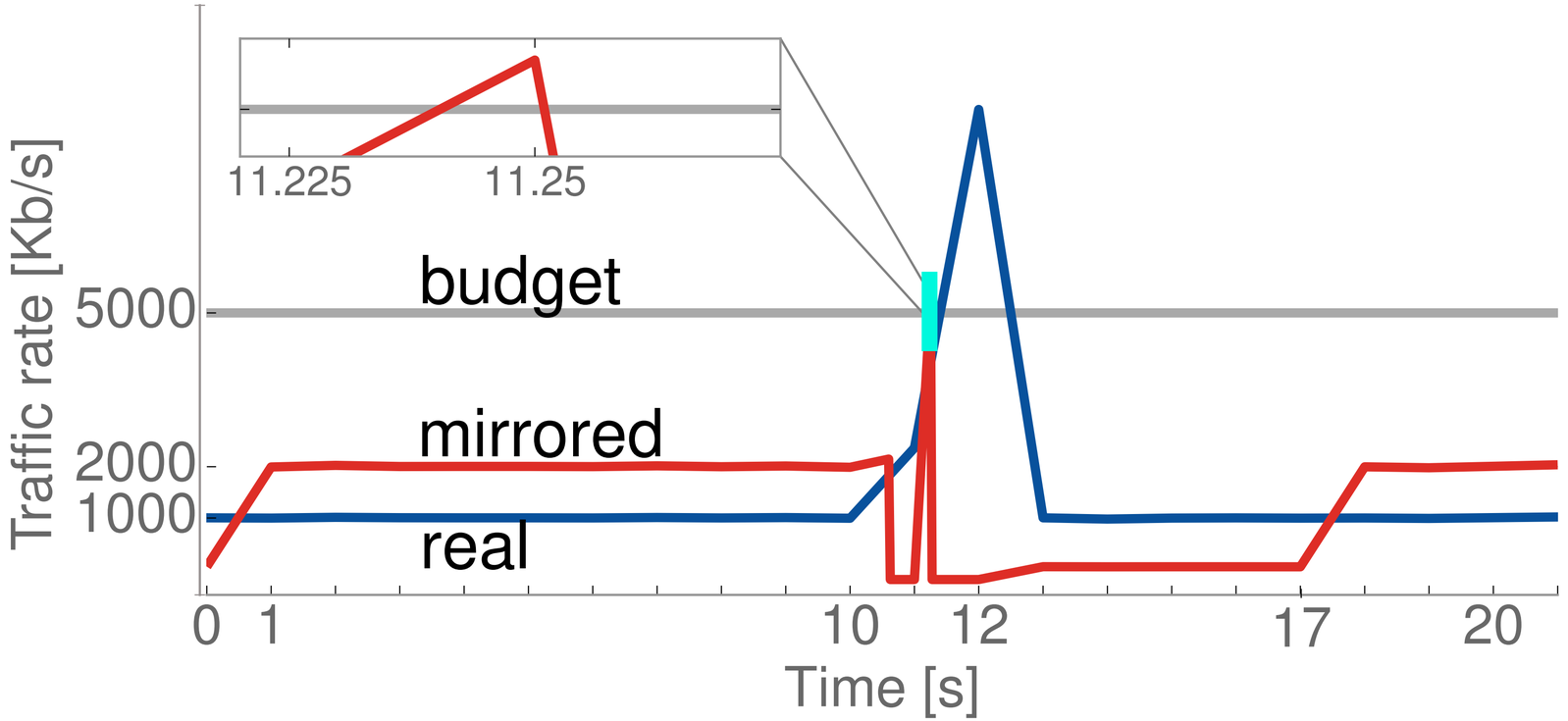
Stable recorded traffic rates are used for future estimations





Stroboscope exceeds the monitoring budget for at most one timeslot





Stroboscope: Declarative Network Monitoring on a Budget

- Traffic slicing as a first-class data-plane primitive
- Strong guarantees on budget compliance and measurement accuracy
- Measurement analysis decoupled from measurement collection

Net2Text: Query-guided Network Captioning




Rüdiger Birkner Dana Drachsler-Cohen Martin Vechev Laurent Vanbever

USENIX Symposium on Networked Systems Design and Implementation. April 2018.


Net2Text

Net2Text

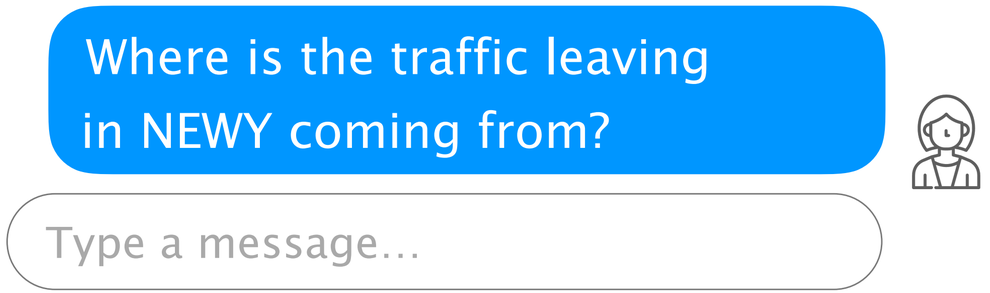
Type a message...

Net2Text

Where is the traffic...

Net2Text

question *in*
natural language



Net2Text

Where is the traffic leaving
in NEWY coming from?

•••

Type a message...

Net2Text

Where is the traffic leaving in NEWY coming from?

The traffic enters mostly in PHIL and goes to Youtube and Netflix.

Type a message...

summary *in* natural language

Net2Text

question *in*
natural language

Where is the traffic leaving in NEWY coming from?

The traffic enters mostly in PHIL and goes to Youtube and Netflix.

Type a message...

summary *in*
natural language

Finding *a* summary of the
network-wide forwarding state is simple

Finding *a* summary of the
network-wide forwarding state is simple

Traffic is forwarded.

Finding *a* summary of the
network-wide forwarding state is simple

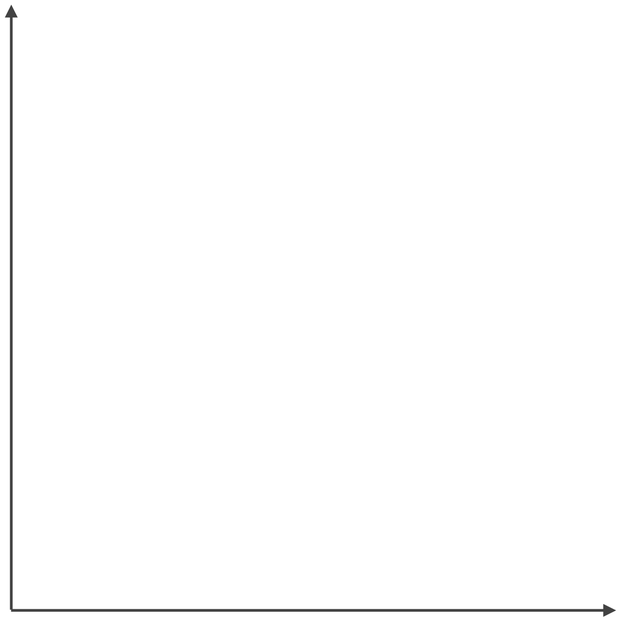
Traffic from LOSA to 35.184.0.0/19,
which is owned by Google,
is leaving the network in CHIC
and takes the path
SUNV, DENV, KSCY, INDI to CHIC.

Finding a good summary of the network-wide forwarding state is hard

Summarization is a multi-objective optimization problem

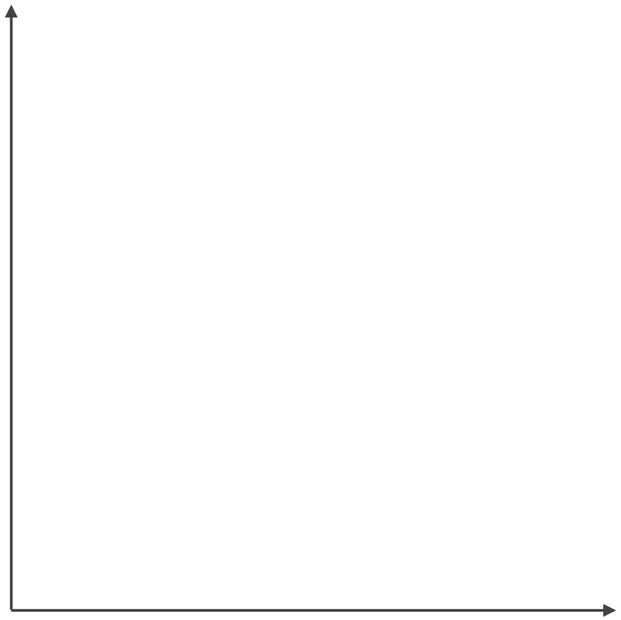
Coverage

amount of data
described by the summary



Coverage

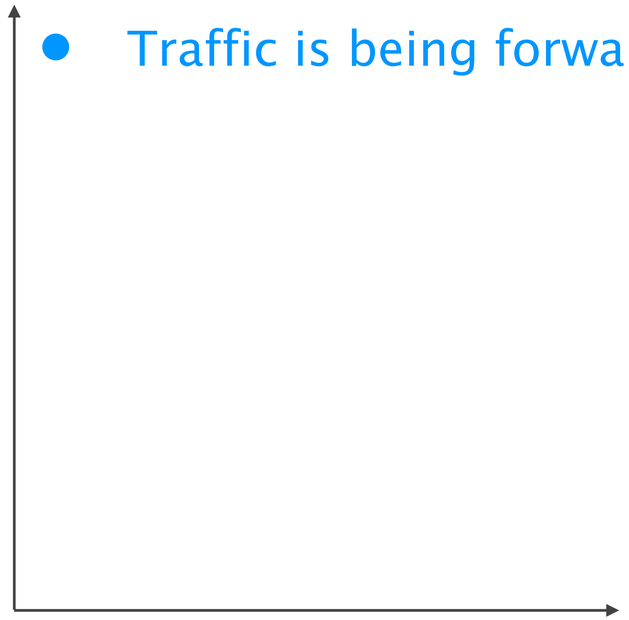
amount of data
described by the summary



Explainability

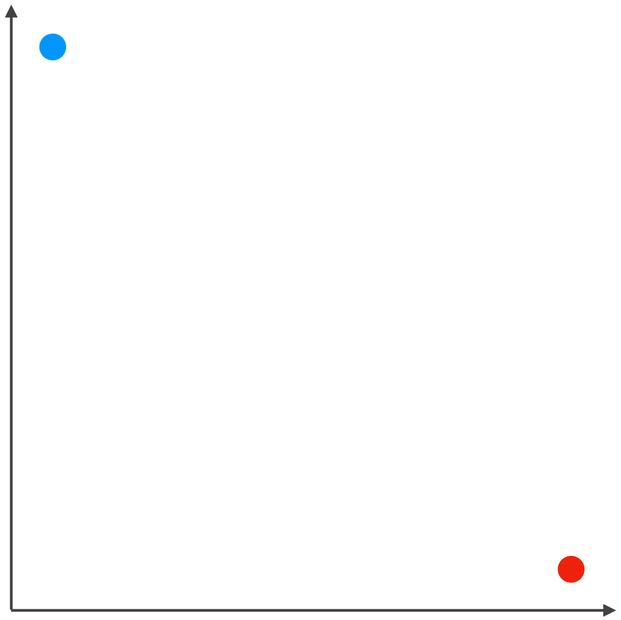
amount of detail
provided by the summary

Coverage



Explainability

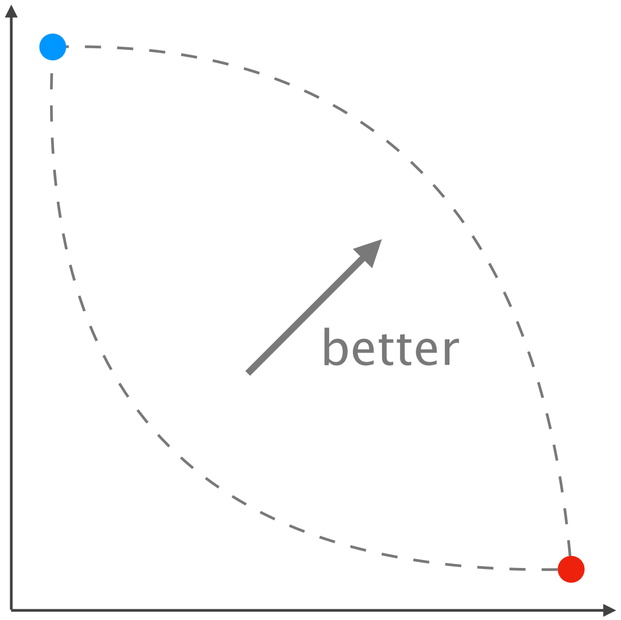
Coverage



Traffic from LOSA to 35.184.0.0/19,
which is owned by Google, …

Explainability

Coverage



Explainability

Summarization is a multi-objective optimization problem

Score Weighted sum of the amount of traffic covered by each path specification in the summary.

Summarization is a multi-objective optimization problem

Coverage

Score Weighted sum of the amount of traffic covered by each path specification in the summary.

Summarization is a multi-objective optimization problem

Explainability

weights based on level of detail
of the path specification

Score

Weighted sum of the amount of traffic covered by each path specification in the summary.

Summarization is a multi-objective optimization problem

Score Weighted sum of the amount of traffic covered by each path specification in the summary.

Goal Find path specifications that maximize the score.

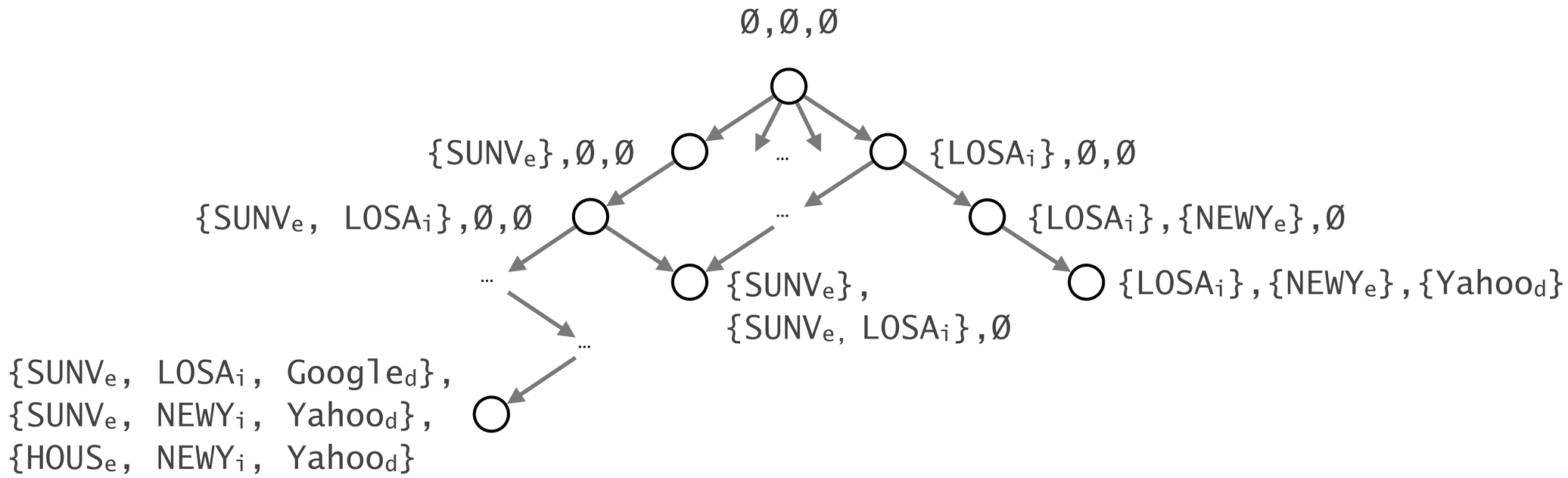
Summarization is a multi-objective optimization problem

Score Weighted sum of the amount of traffic covered by each path specification in the summary.

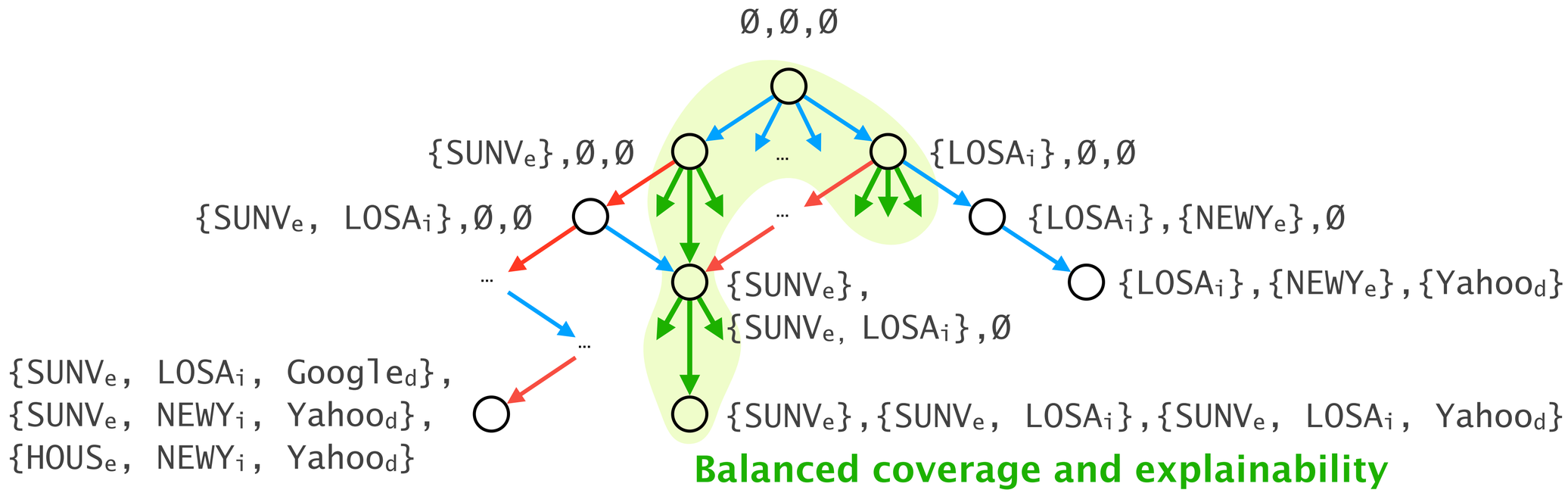
Goal Find ***k*** path specifications each of size at most *t* that maximize the score.

The search space is exponential

in the number of path specifications and feature values



Net2Text reduces the search space to solutions that balance coverage and explainability



Self-driving/monitoring networks

in the age of deep network programmability

1	operator assistance	Part I assisting operators
2	partial automation	
3	conditional automation	Part II taking control
4	high automation	
5	full automation	too futuristic? ☺

Part II: Taking control

applications

frameworks

methods

data-driven convergence

monitoring/inference

some of our challenges lying ahead

Part II: Taking control

applications

frameworks

methods

data-driven

convergence

Blink

Fast Connectivity Recovery Entirely in the Data Plane


Thomas Holterbach


Edgar Costa Molero


Maria Apostolaki

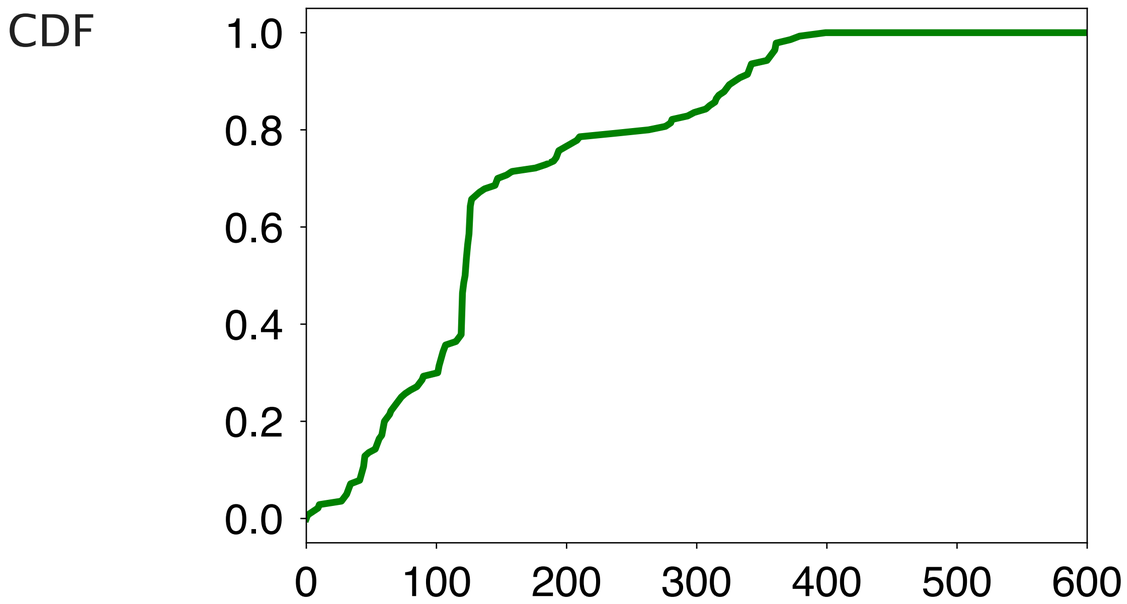

Alberto Dainotti


Stefano Vissicchio


Laurent Vanbever

USENIX Symposium on Networked Systems Design and Implementation. February 2019.

Internet control plane can take minutes to converge upon remote failures



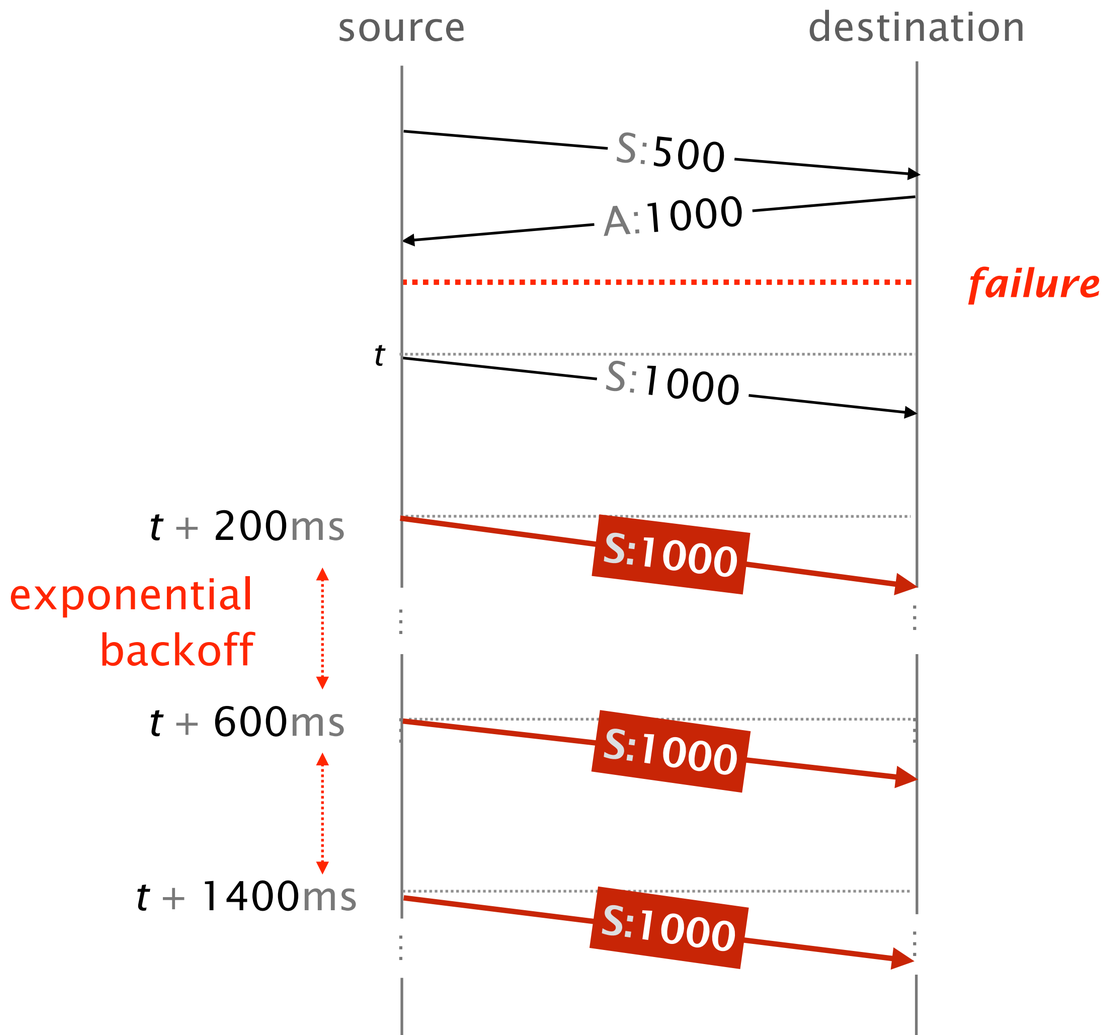
Second elapsed between the data plane failure and the control plane learning about it

While the control plane can take minutes to learn about a failure,

the actual Internet traffic is affected almost instantaneously

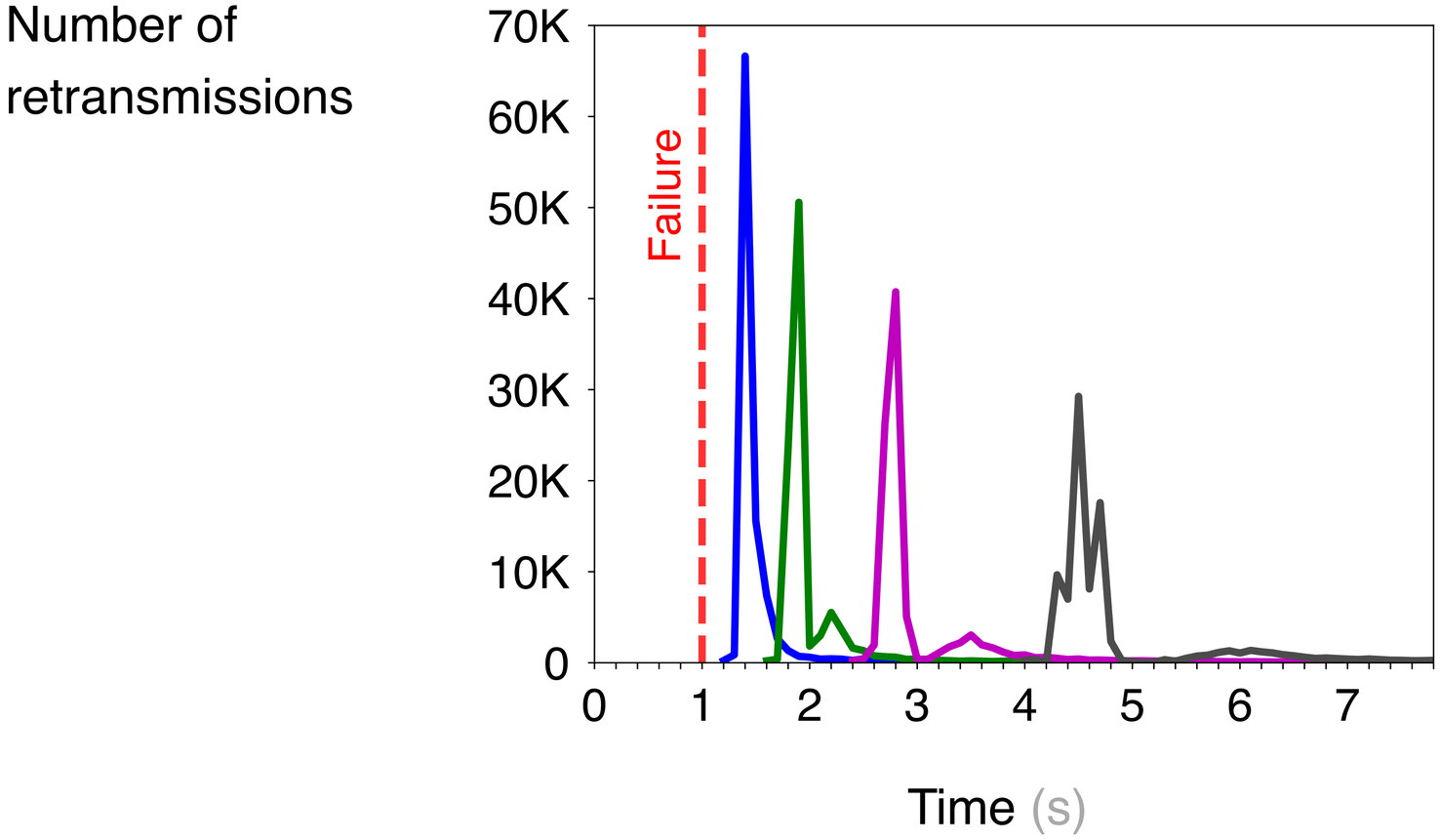
How about we track this signal instead?

Internet traffic end-points retransmit packets upon experiencing packet drops



Many end-points retransmitting leads to

waves of retransmission



Tracking this signal in the data plane is challenging

Challenges

Signal is noisy

packets loss are routinely observed

Signal fades away quickly

due to the exponential backoff

Signal is compounded over many small ones

requires per-connection tracking, which is hard to scale

We solve these challenges by considering a subset of the signal that we carefully craft for maximal signal-to-noise ratio

Solutions

Signal is noisy

Focus on retransmissions caused by bursty losses

Signal fades away quickly

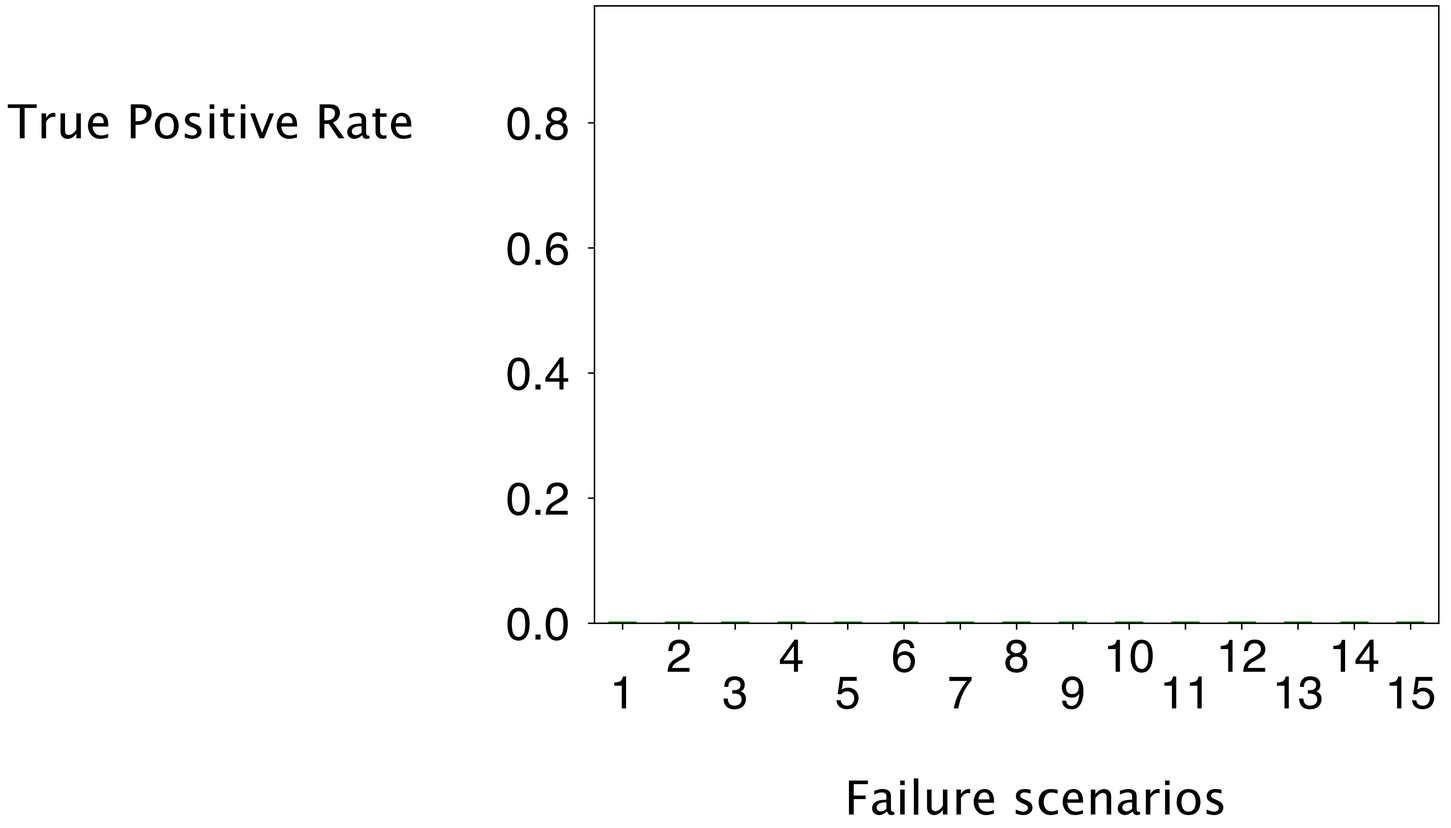
Focus on active flows (fast to retransmit after a failure)

Signal is compounded over many small ones

Rely on scalable data structures and sampling

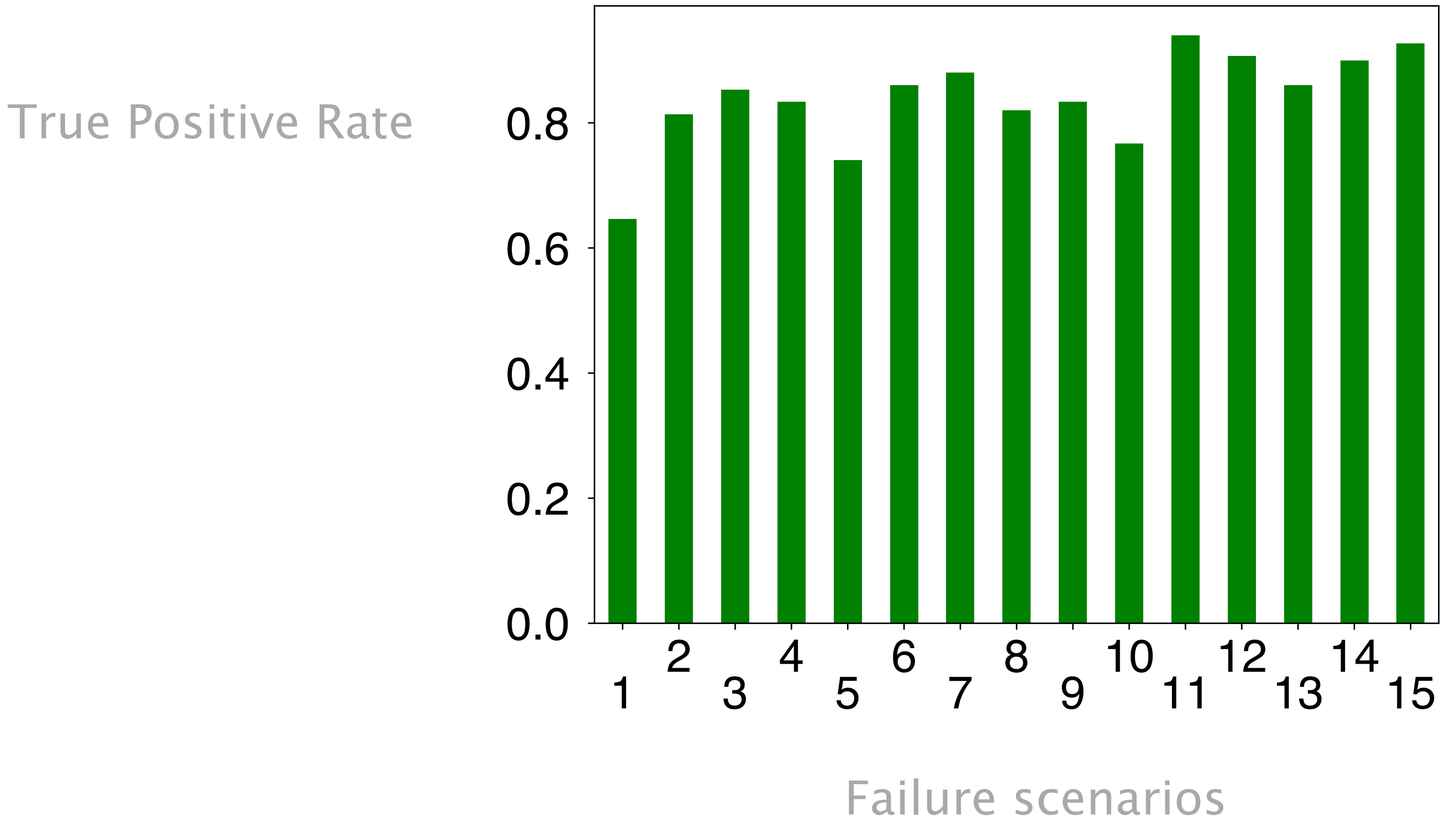
Blink works well in practice,

with an inference accuracy above 80% in most cases



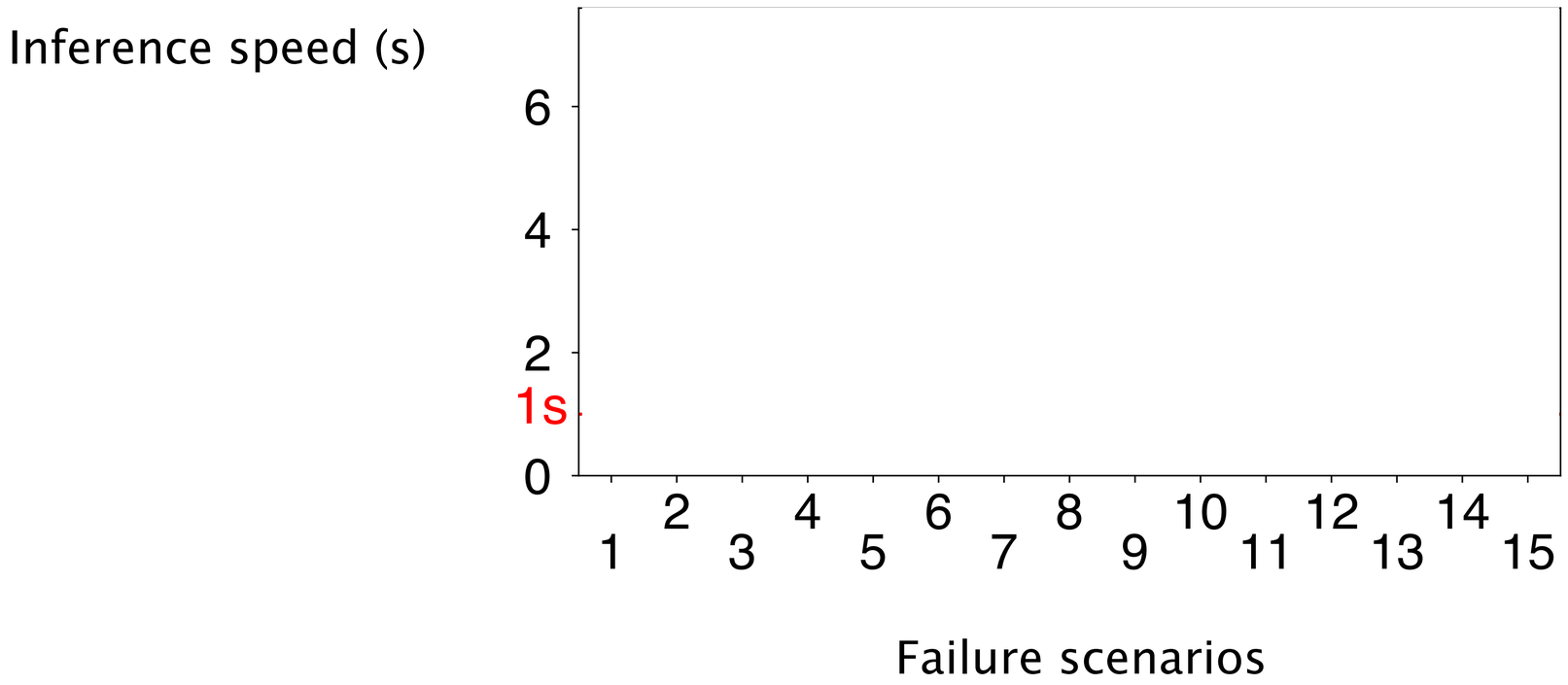
Blink works well in practice,

with an inference accuracy above 80% in most cases



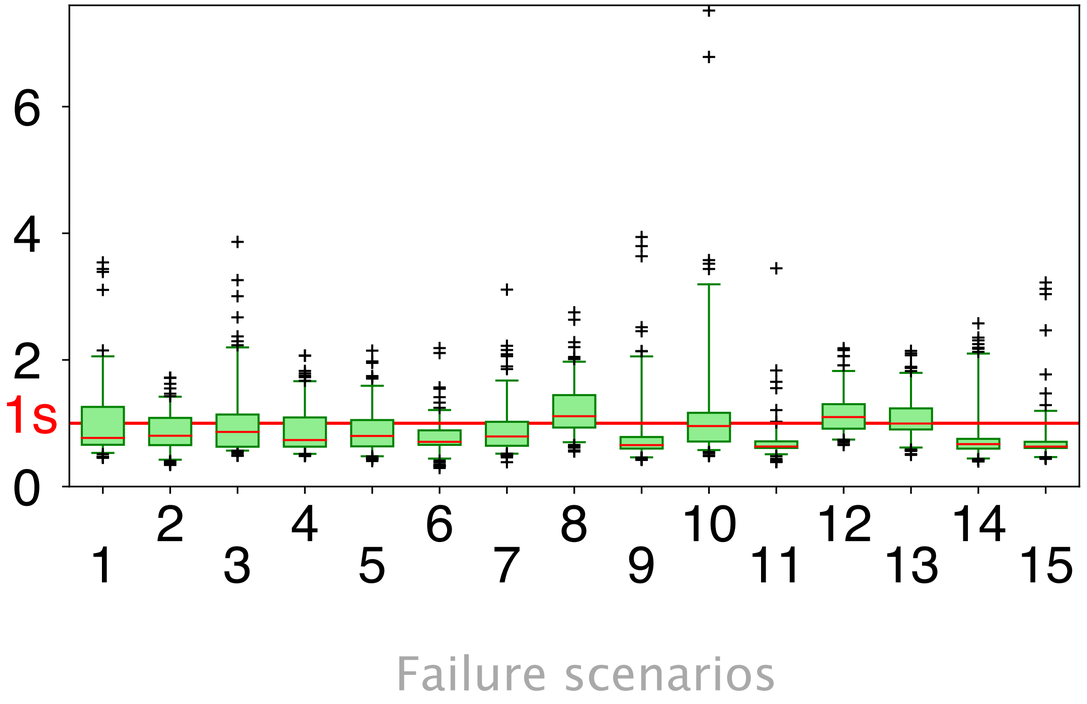
Blink works well in practice,

with an inference speed below 1 second in most cases



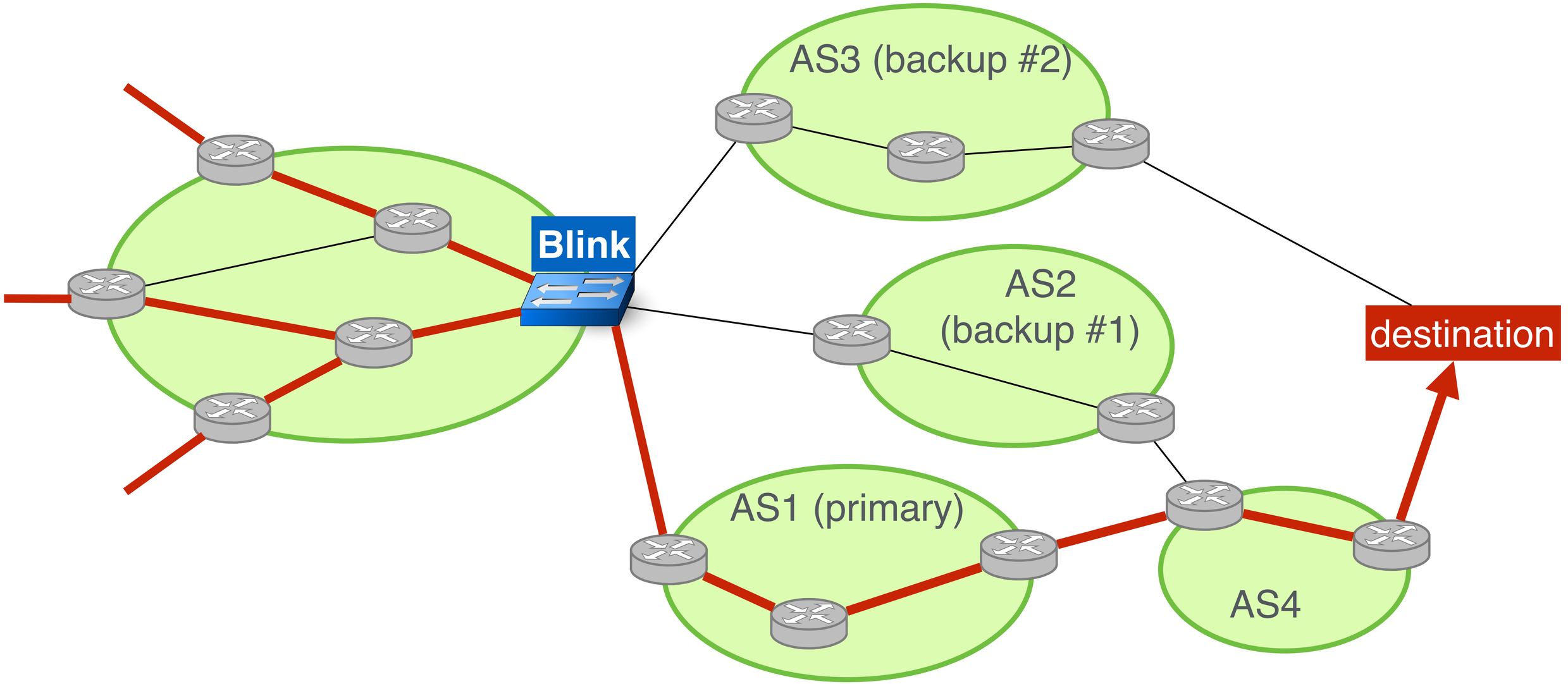
Blink works well in practice,

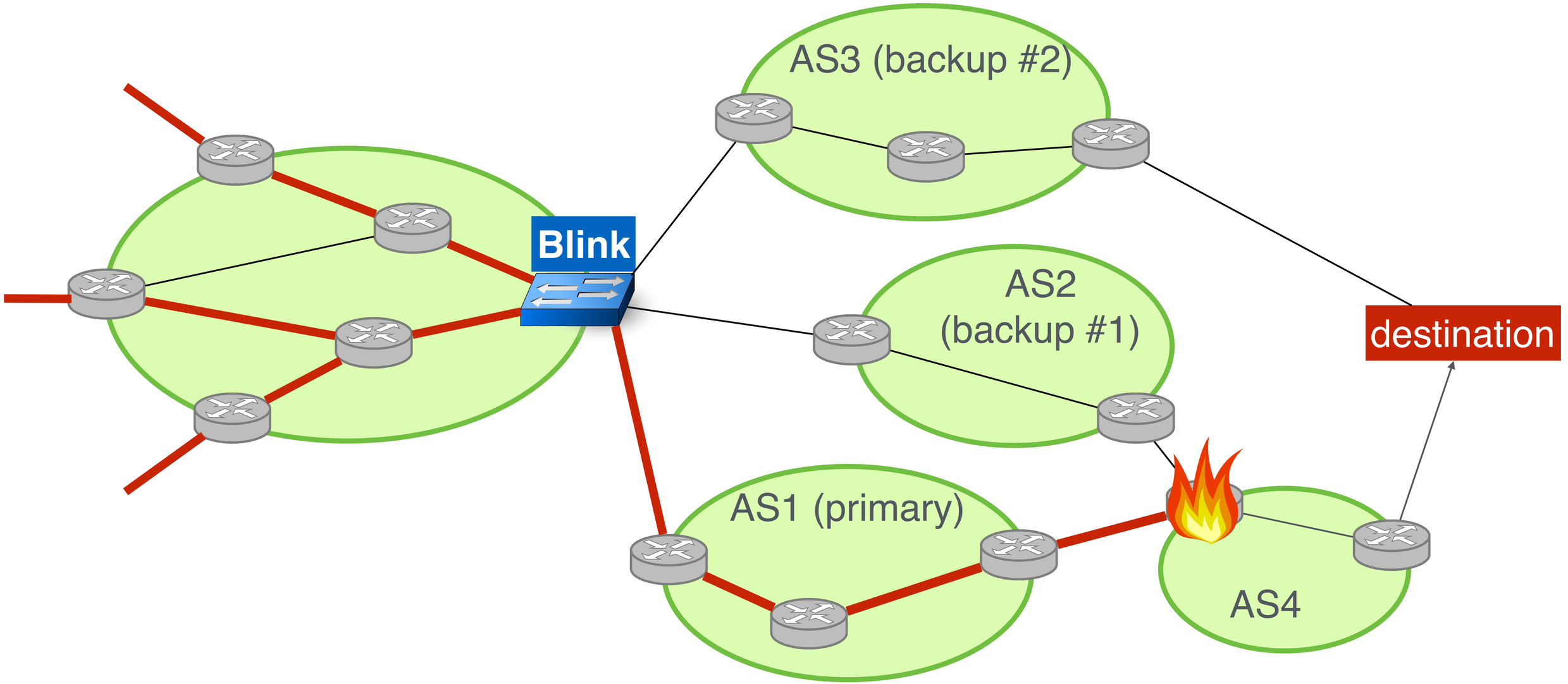
with an inference speed below 1 second in most cases

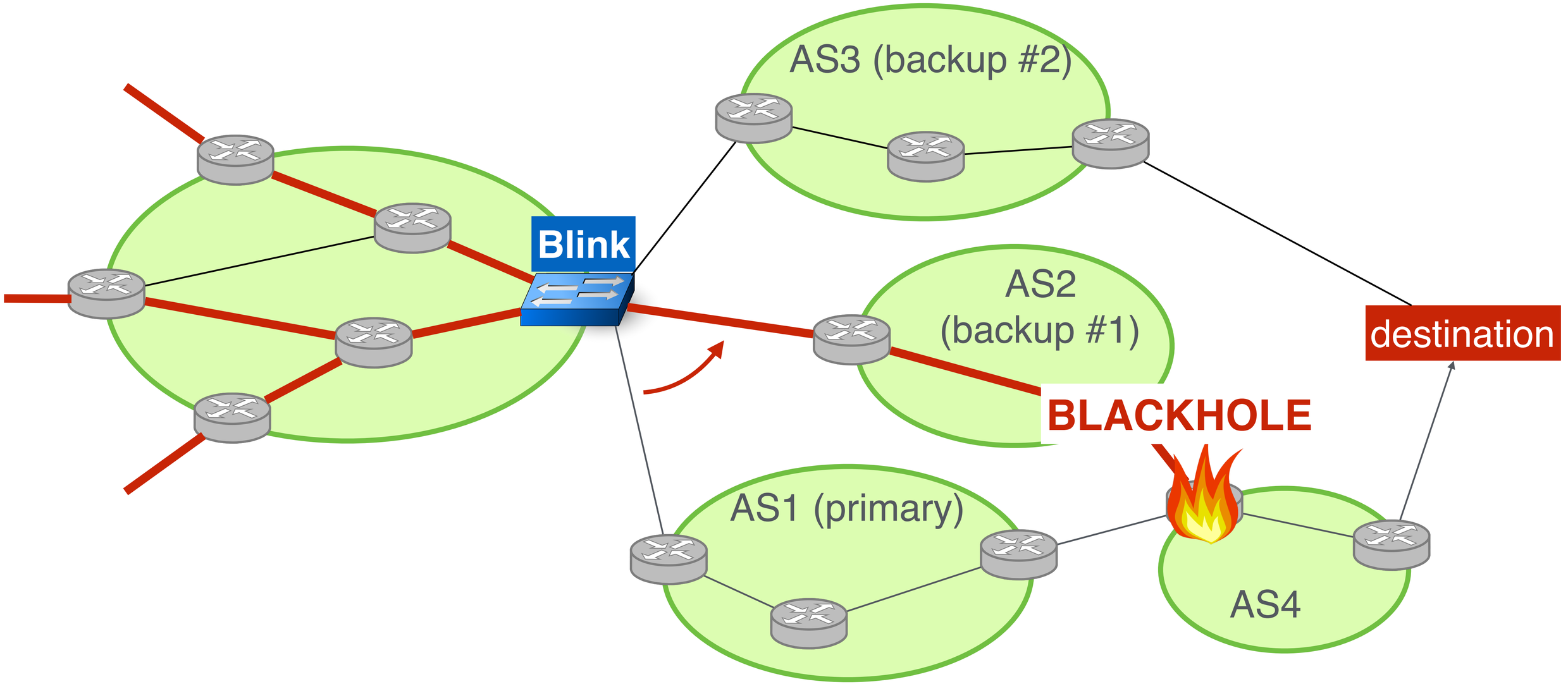


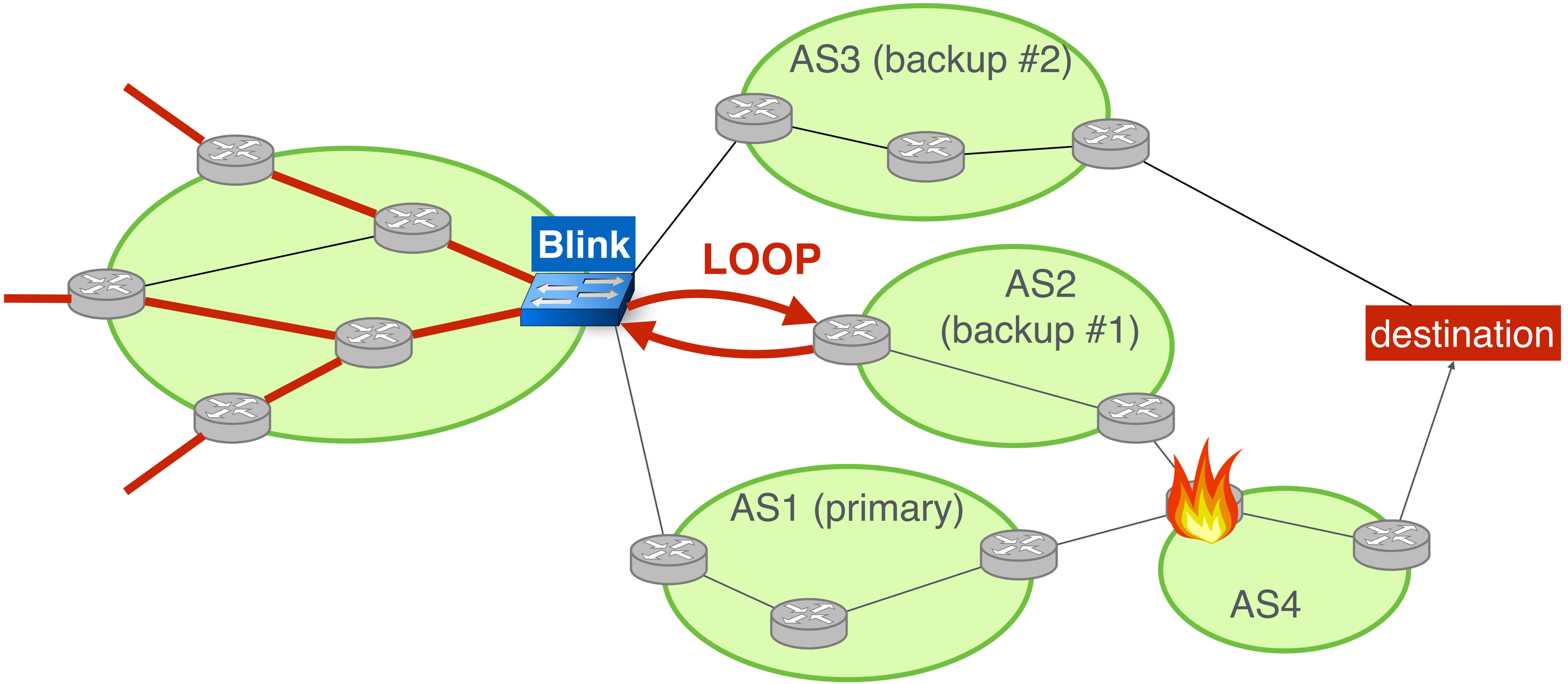
What about actuation?

Where do we send the traffic upon detecting a failure?


AS3 (backup #2)
Blink
AS2
(backup #1)
destination
AS1 (primary)
AS4

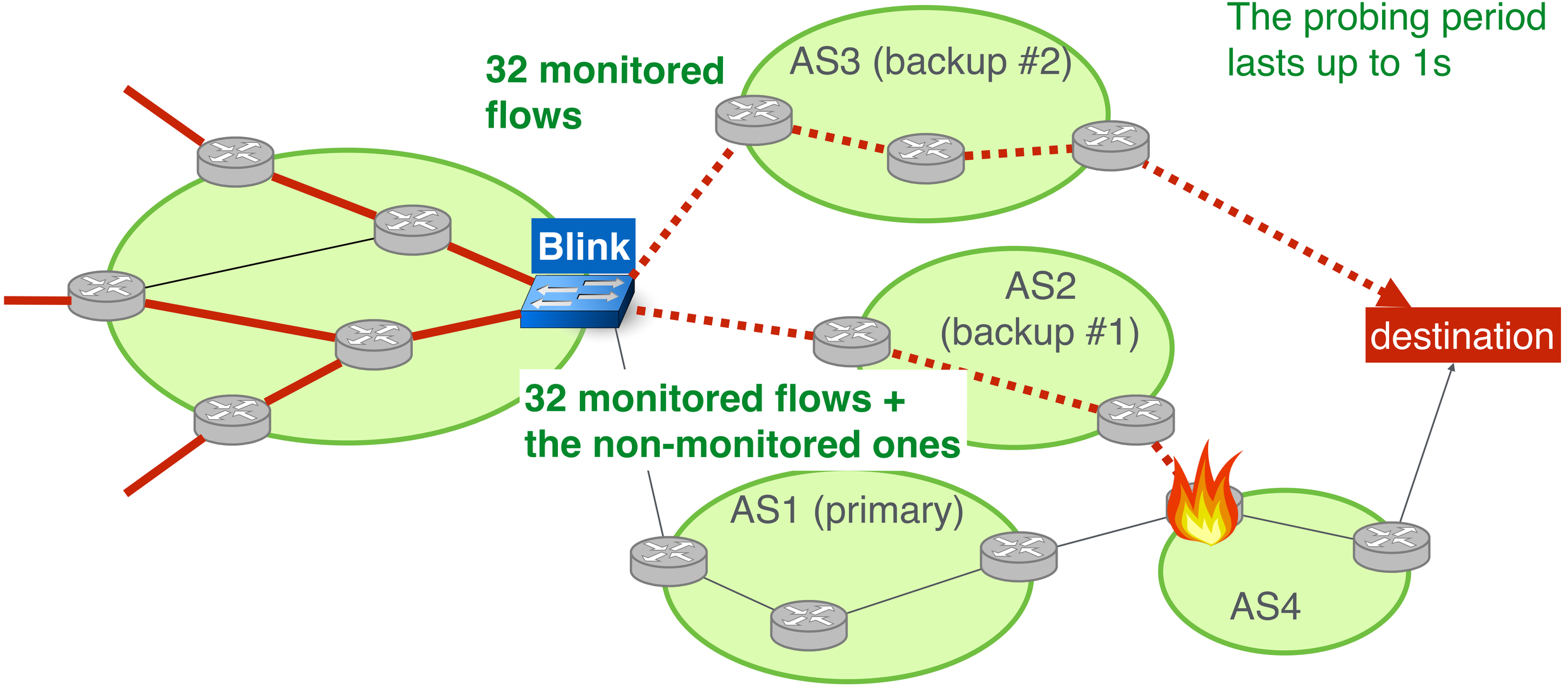

AS3 (backup #2)
Blink
AS2
(backup #1)
destination
AS1 (primary)
AS4

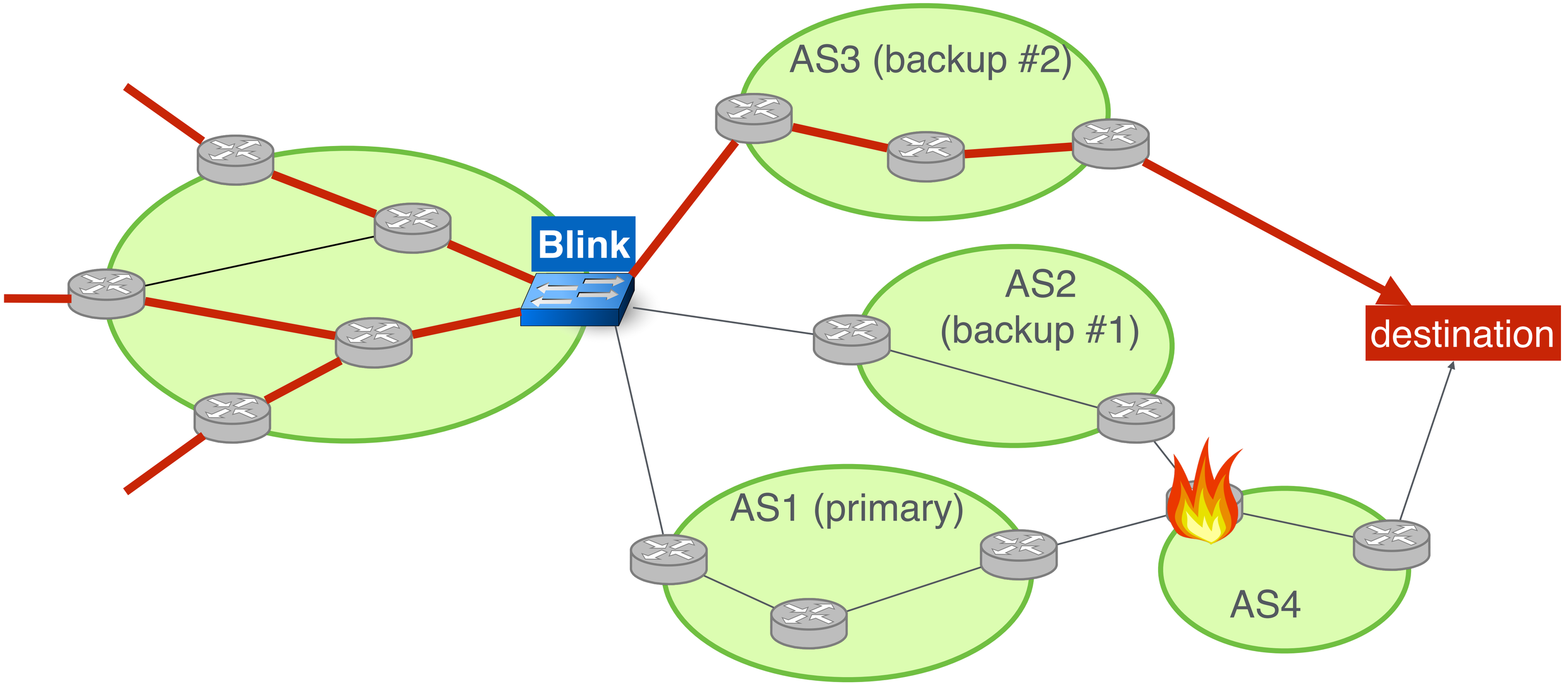

AS3 (backup #2)
Blink
AS2
(backup #1)
destination
BLACKHOLE
AS1 (primary)
AS4


AS3 (backup #2)
Blink
LOOP
AS2
(backup #1)
destination
AS1 (primary)
AS4

As for failures, Blink relies on data plane signals

to pick a working backup path


32 monitored
flows
AS3 (backup #2)
The probing period
lasts up to 1s
Blink
AS2
(backup #1)
destination
32 monitored flows +
the non-monitored ones
AS1 (primary)
AS4


AS3 (backup #2)
Blink
AS2
(backup #1)
destination
AS1 (primary)
AS4

We intend to generalize the signals we track,
for a wider variety of applications

reliability	reroute around fine-grained failures
performance	steer traffic along the most performant path
security	detect unwanted traffic redirection

Part II: Taking control

applications

frameworks

methods

monitoring/inference

Adaptive Network Monitoring

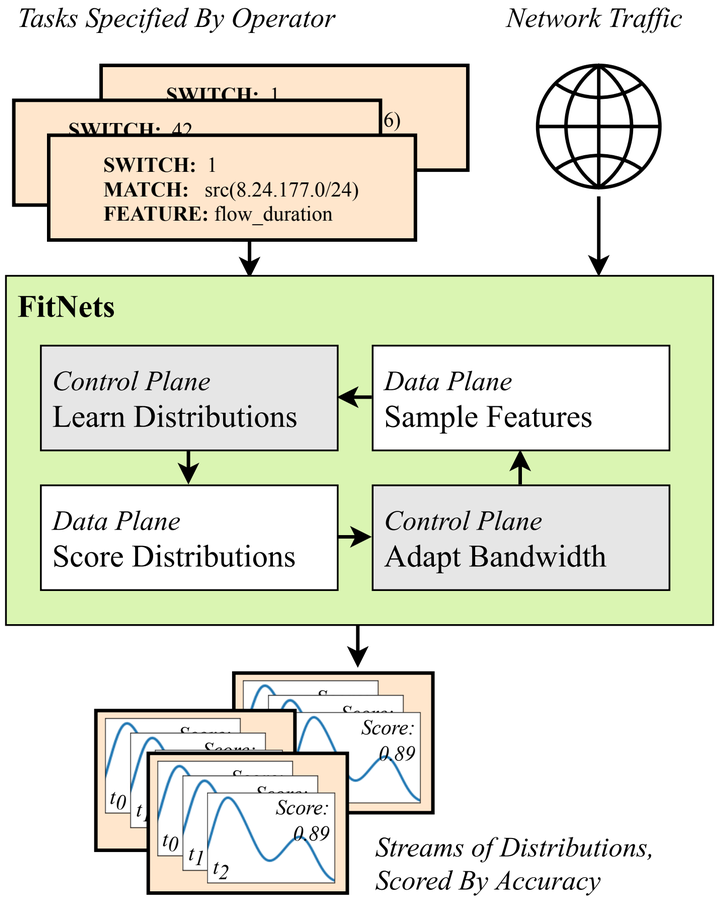
Alexander Dietmüller

Albert
Gran Alcoz

Laurent Vanbever

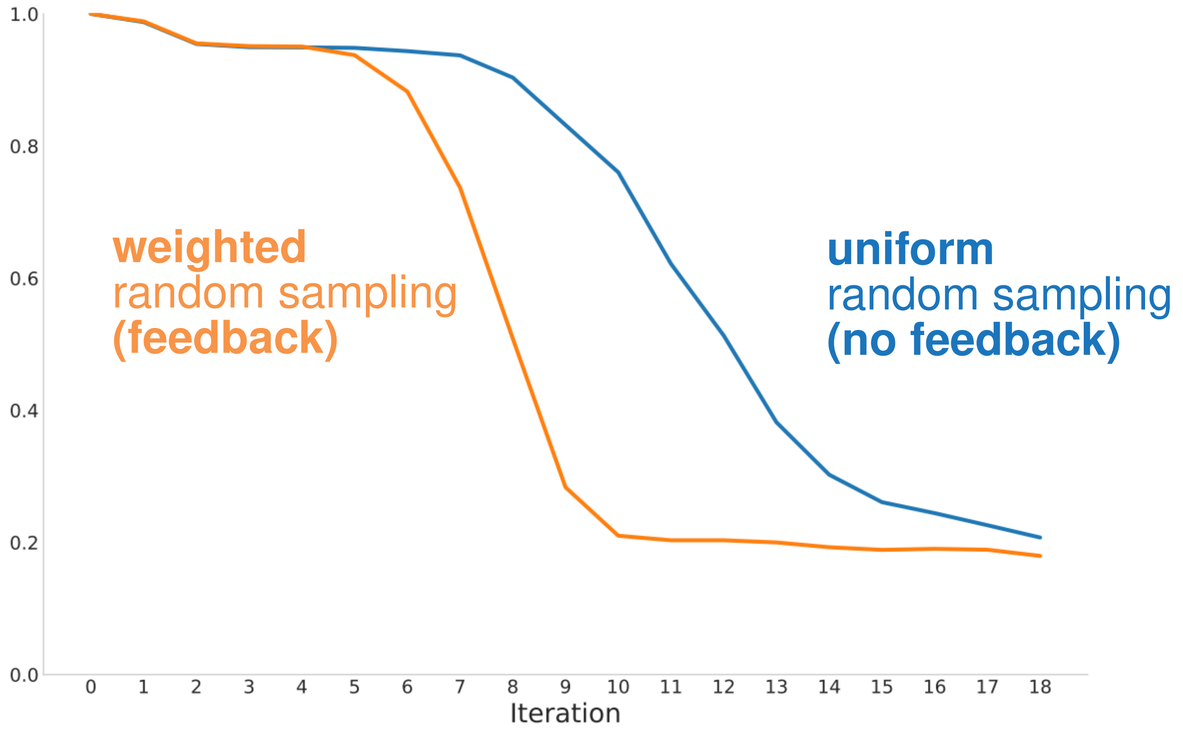
We're building an adaptive monitoring framework that can learn high-quality traffic distributions

- computes traffic distribution
  in the control plane, on traffic samples
- scores these distributions
  in the data plane, on *all* traffic
- adapts the sampling rate
  according to the score



Our framework enables to learn distribution better and faster than simply randomly sampling traffic

Distance from
true distribution
(Jensen-Shannon)



Part II: Taking control

applications

frameworks

methods

monitoring/inference

In-network inference

Coralie Busse-Grawitz

Roland Meier

Alexander Dietmüller

Tobias Bühler

Laurent Vanbever

Why (and How) Networks Should Run Themselves

Nick Feamster and Jennifer Rexford
Princeton University

Abstract

The proliferation of networked devices, systems, and applications that we depend on every day makes managing networks more important than ever. The increasing security, availability, and performance demands of these applications suggest that these increasingly difficult network management

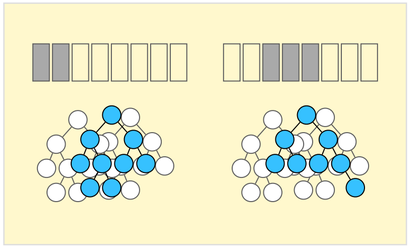
relationships between them and user quality of experience become increasingly complex. Twenty years ago, we had some hope of (and success in) creating clean, closed-form models of individual protocols, applications, and systems [4, 24]; today, many of these are too complicated for closed-form analysis. Prediction problems such as determining how search query response time would vary in response to the placement

Performing automated, real-time inference: The past ten years has demonstrated significant promise in using machine learning to both detect and predict network attacks; we must build on the increasing amount of work in automated infer-

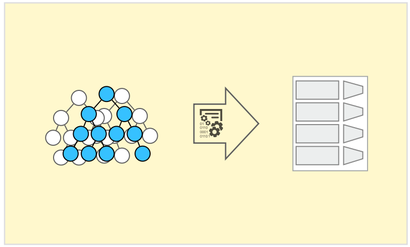
Are programmable network devices powerful enough for machine learning inference?

Yep!

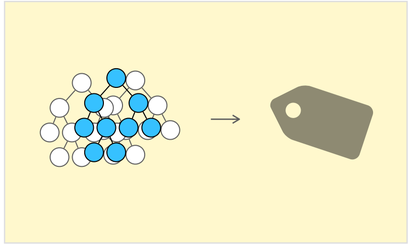
We're building a real-time, in-network inference framework



Optimizing random forest models
for programmable network devices



Compiling random forest models
to programmable network devices



Performing runtime classification
at Tbps

Early, accurate & efficient classification of an ongoing flow as an optimization problem

Given a labeled dataset $\mathcal{F}$ and a threshold τ

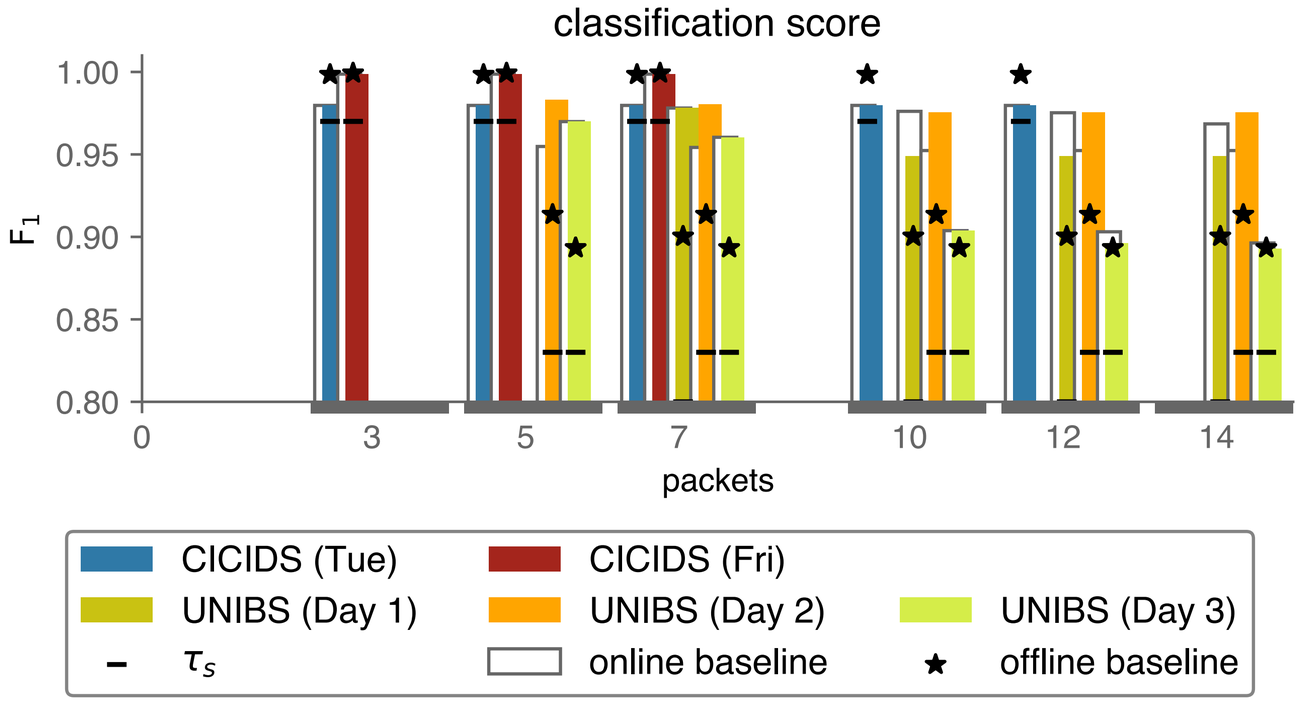
find a classifier C such that

- accuracy(C) >= τ
- C fits in programmable network devices

while

- minimizing memory usage
- maximizing classification speed

Our prototype is already able to classify traffic after few packets, with an accuracy which is on-par with software-based solutions



Part II: Taking control

applications

frameworks

methods

some of our challenges
lying ahead

Building and operating truly self-driving networks require us to overcome fundamental challenges

data

correctness

interpretability

data

correctness

interpretability

data is network-dependent

poor generalization capabilities

interesting events are rare

few interesting examples (if any)

data can easily be polluted

by anyone with a Internet connection…

data

correctness

interpretability

how do we…

ensure correctness?

network should be reachable

guarantee stability?

agents can clash with each other

reason about optimality?

data-driven > non-data-driven (?)

data

correctness

interpretability

how can operators…

reason about self-driving networks?

especially in partial deployment

debug their self-driving networks?

manual override

Self-driving/monitoring networks

in the age of deep network programmability

1 operator assistance

2 partial automation

Part I
assisting operators

3 conditional automation

4 high automation

Part II
taking control

5 full automation

too futuristic? ☺

Huge kudos to my research group!

Check us out at **nsg.ee.ethz.ch**

Ahmed

Roland

Mirja

Alex

Edgar

Rüdiger

Tobias

Albert

Thomas

Maria

Self-driving/monitoring networks

in the age of deep network programmability

Laurent Vanbever

nsg.ee.ethz.ch

TMA

June 19 2019