Blink: Fast Connectivity Recovery Entirely in the Data Plane

Thomas Holterbach
ETH Zürich

NSDI
26th February 2019

https://blink.ethz.ch

Joint work with

Edgar Costa Molero	**ETH Zürich**
Maria Apostolaki	**ETH Zürich**
Stefano Vissicchio	**University College London**
Alberto Dainotti	**CAIDA, UC San Diego**
Laurent Vanbever	**ETH Zürich**

Fire at AT&T facility causes widespread outage in North Texas

TELECOM

Nationwide internet outage affects CenturyLink customers

Time Warner Cable comes back from nationwide Internet outage

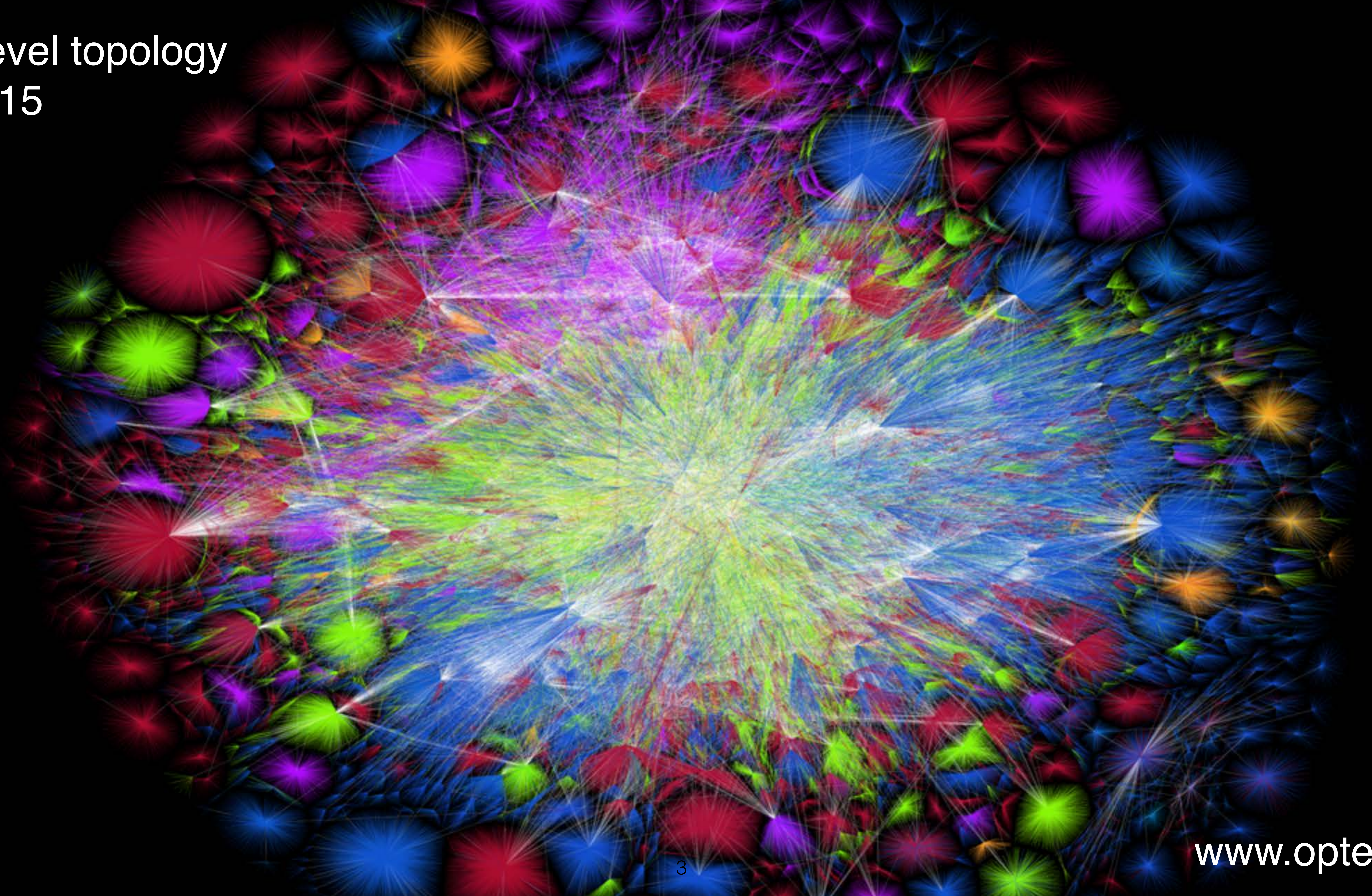
by Brian Stelter @brianstelter

August 27, 2014: 11:07 PM ET

Major internet outage hits the U.S. - Affecting customers of Comcast, Verizon, and AT&T

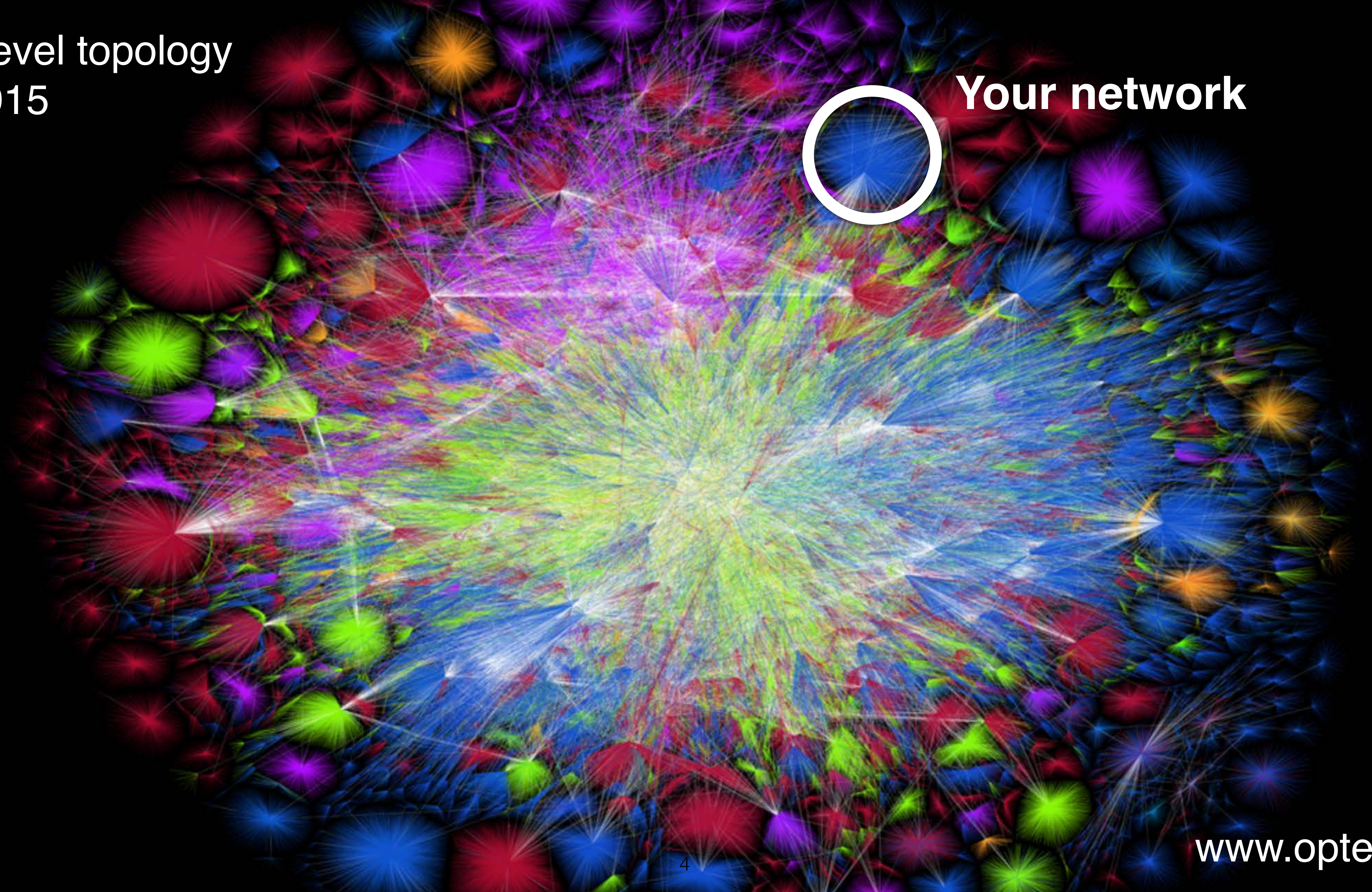
November 6, 2017 | Emerging Threats

AS level topology in 2015



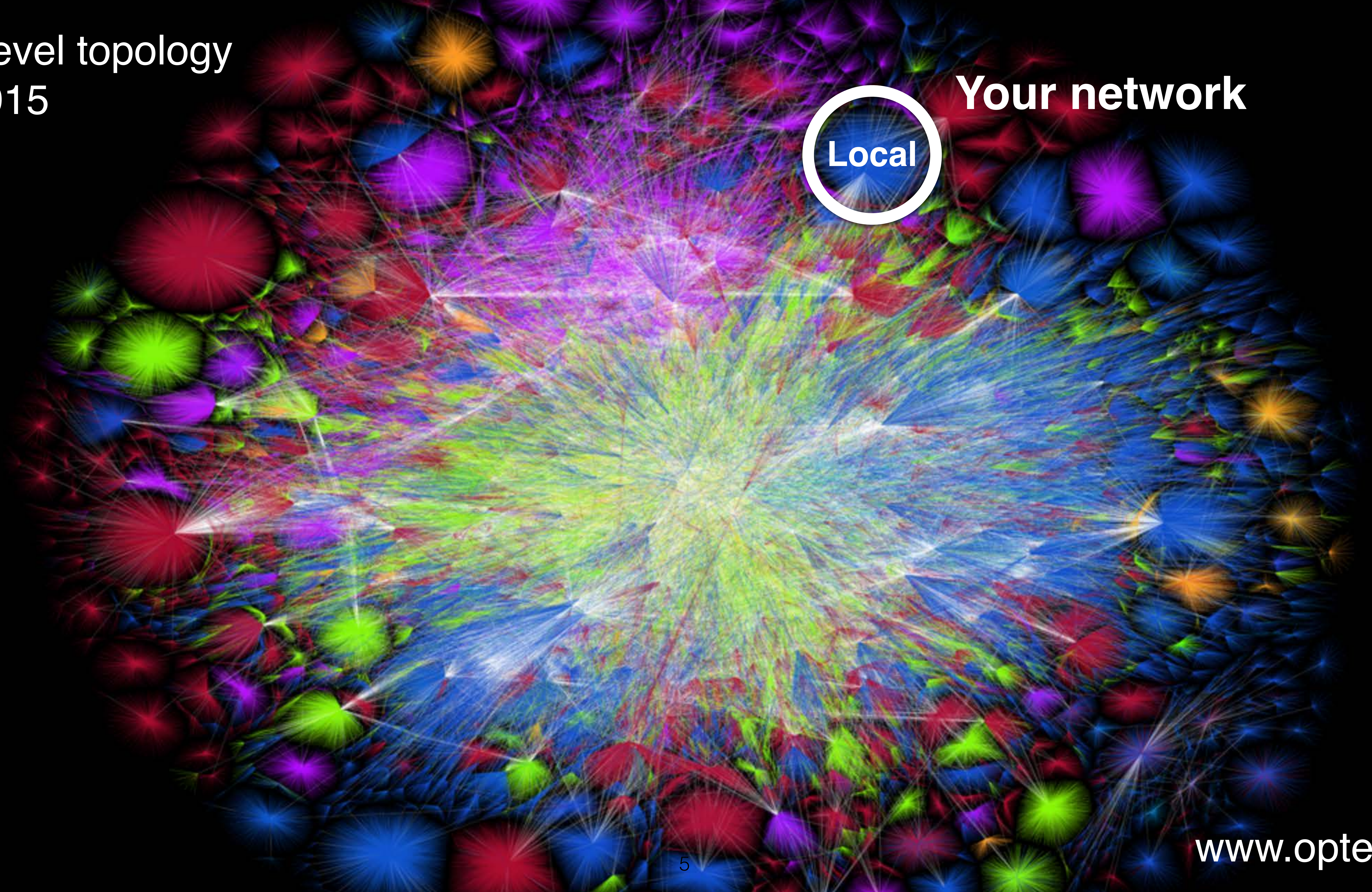
www.opte.org

AS level topology in 2015



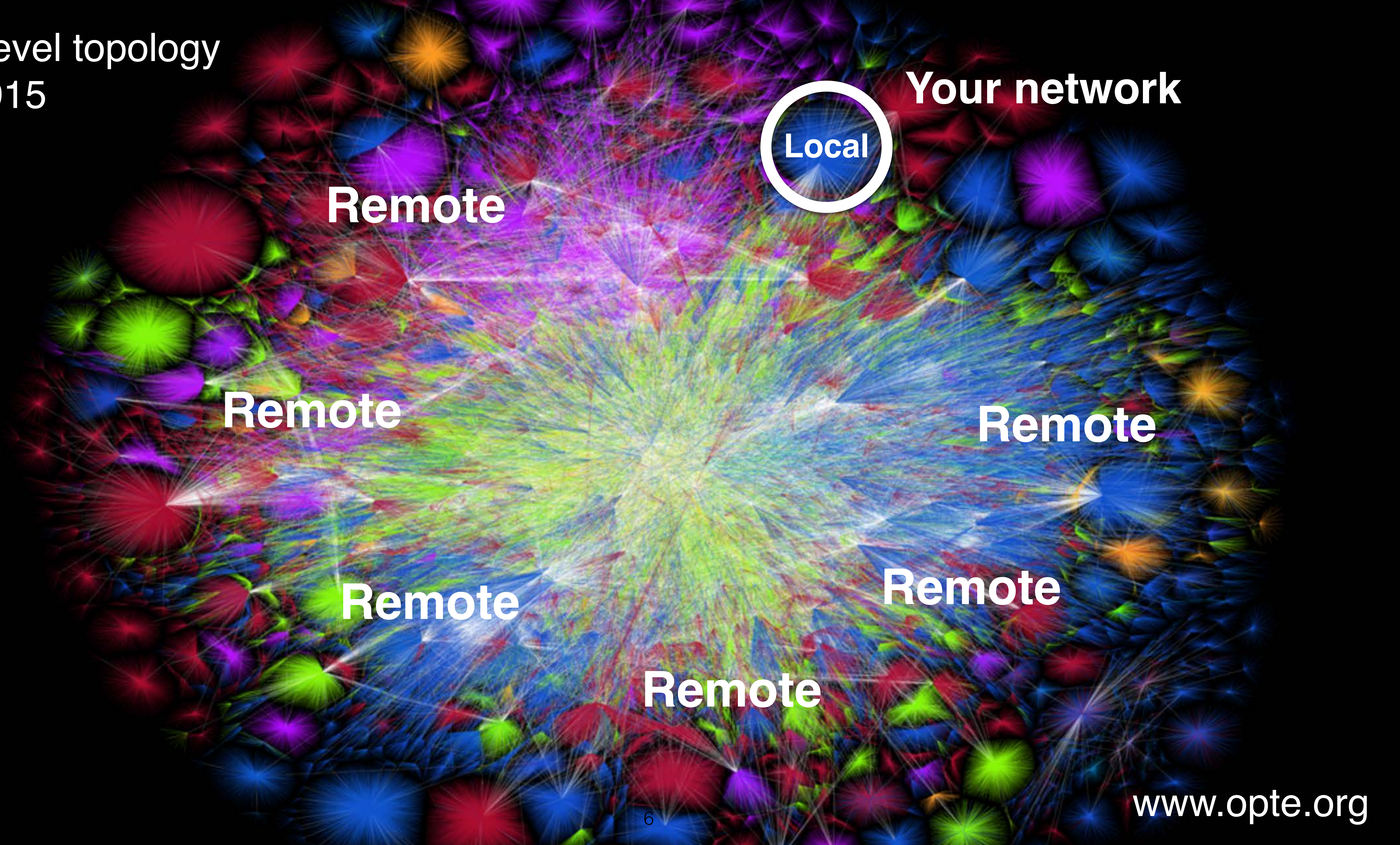
www.opte.org

AS level topology in 2015



www.opte.org

AS level topology in 2015



Upon local failures, connectivity can be quickly restored

Upon local failures, connectivity can be quickly restored

Fast failure detection
using *e.g.*, hardware-generated signals

Fast traffic rerouting
using *e.g.*, Prefix Independent Convergence
or MPLS Fast Reroute

Upon remote failures, the only way to restore connectivity is to wait for the Internet to converge

Upon remote failures, the only way to restore connectivity is to wait for the Internet to converge

… and the Internet converges *very* slowly*

*Holterbach et al. SWIFT: Predictive Fast Reroute ACM SIGCOMM, 2017

Fire at AT&T facility causes widespread outage in North Texas

TELECOM

Nationwide internet outage affects CenturyLink customers

Time Warner Cable comes back from nationwide Internet outage

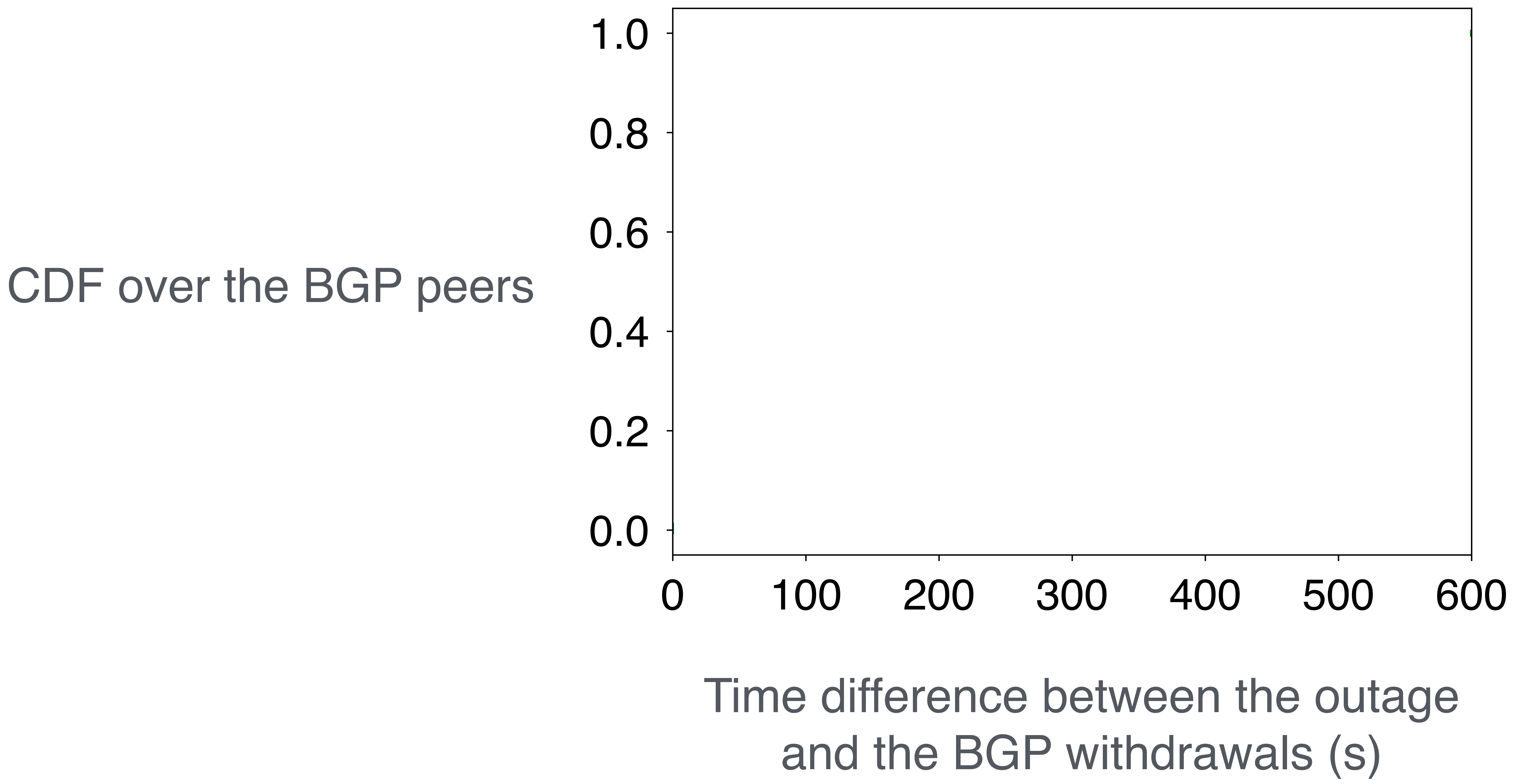
by Brian Stelter @brianstelter

August 27, 2014: 11:07 PM ET

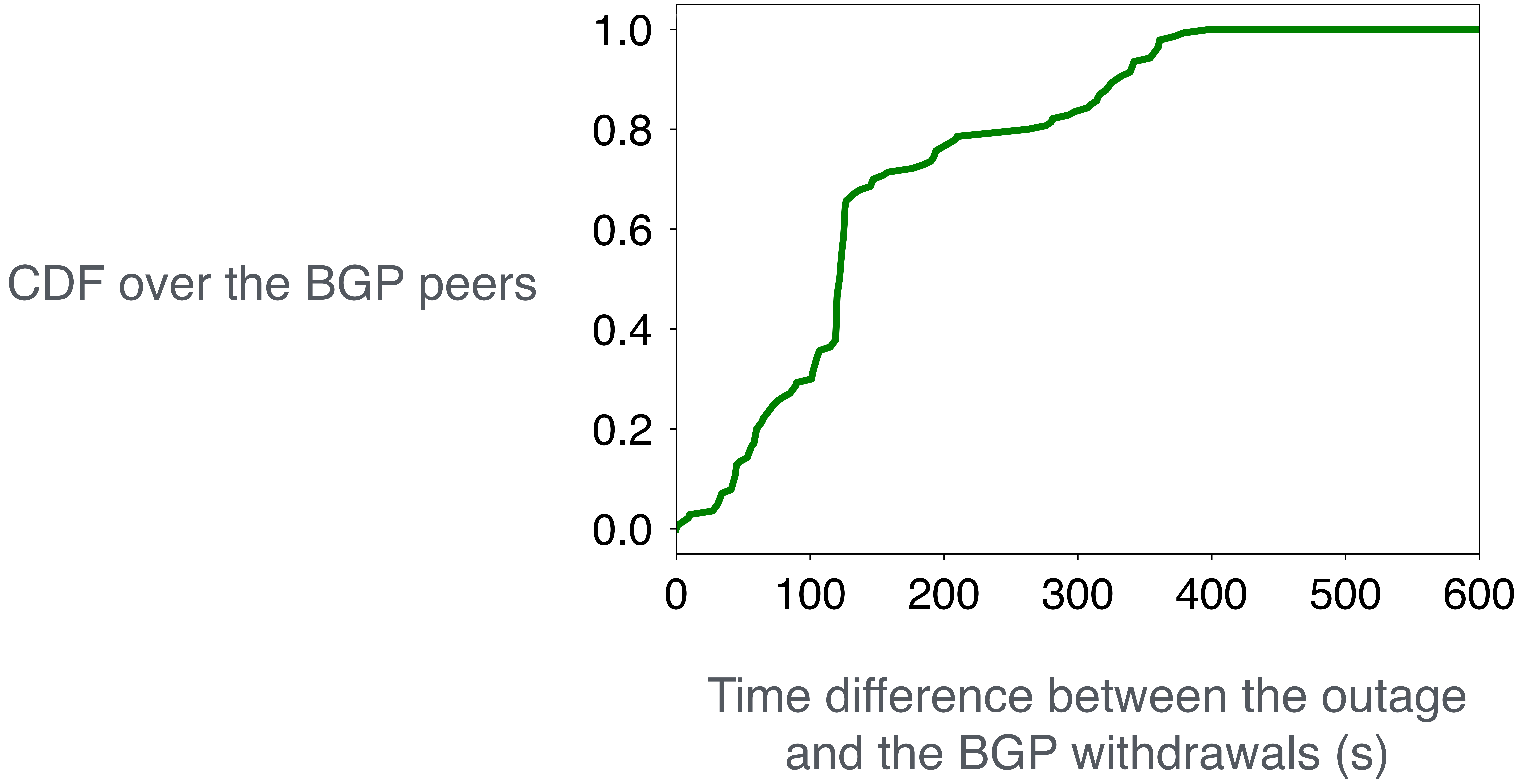
Major internet outage hits the U.S. - Affecting customers of Comcast, Verizon, and AT&T

November 6, 2017 | Emerging Threats

BGP took minutes to converge upon the Time Warner Cable outage in 2014



BGP took minutes to converge upon the Time Warner Cable outage in 2014



Control-plane (*e.g.*, BGP) based techniques typically converge slowly upon remote outages

Control-plane (*e.g.,* BGP) based techniques typically converge slowly upon remote outages

What about using data-plane signals for fast rerouting?

Blink: Fast Connectivity Recovery Entirely in the Data Plane

Thomas Holterbach
ETH Zürich

NSDI
26th February 2019

https://blink.ethz.ch

Outline

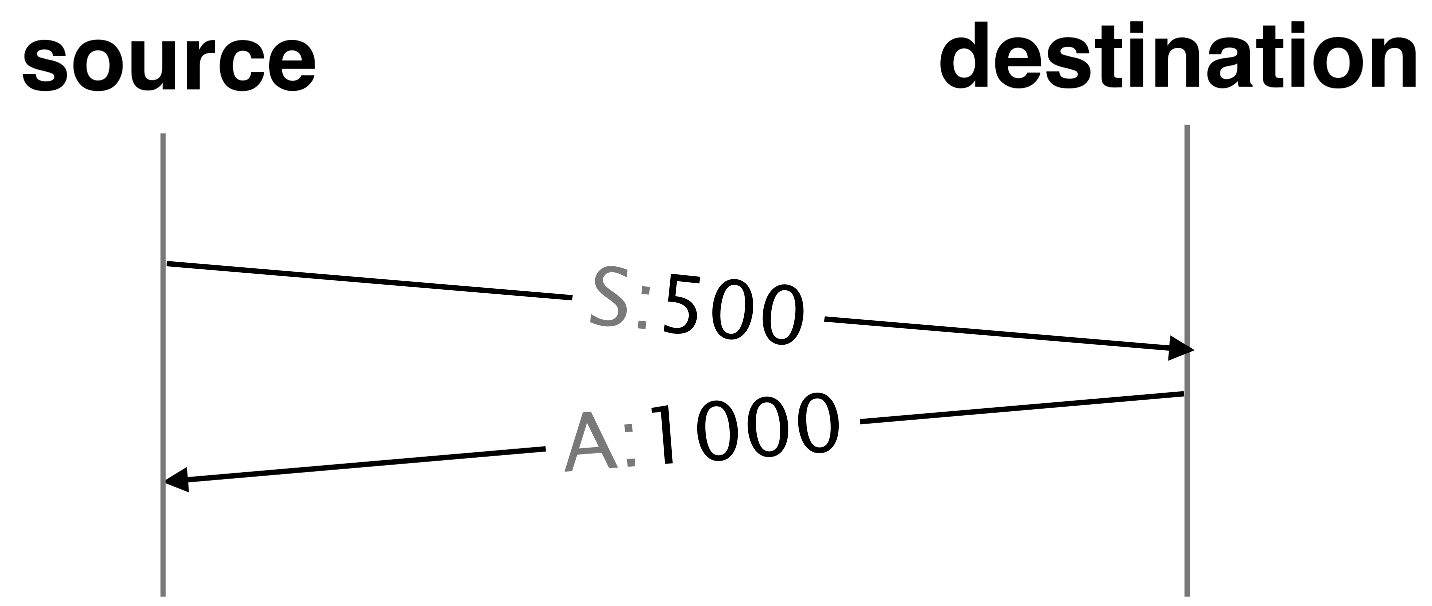
1. Why and how to use data-plane signals for fast rerouting
2. ***Blink*** infers more than 80% of the failures, often within 1s
3. ***Blink*** quickly reroutes traffic to working backup paths
4. ***Blink*** works in practice, on existing devices

Outline

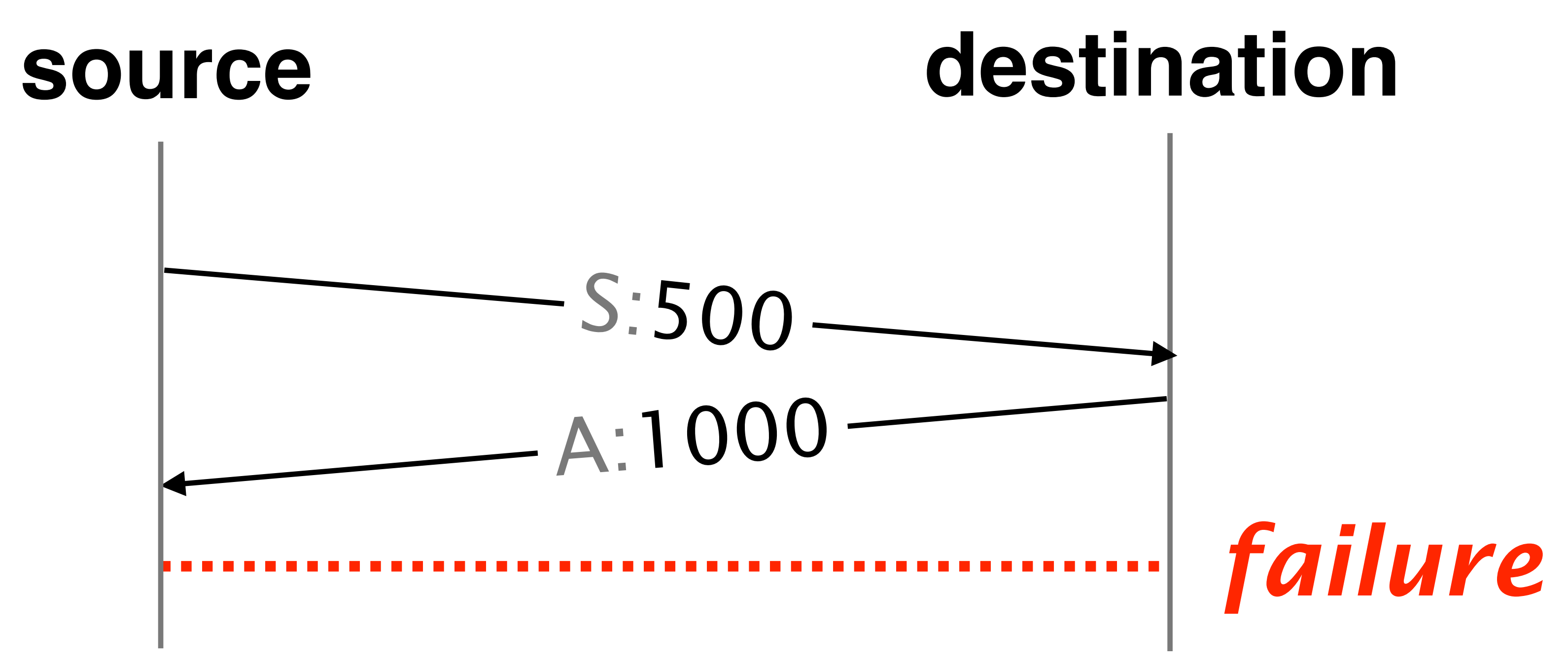
1. Why and how to use data-plane signals for fast rerouting
2. ***Blink*** infers more than 80% of the failures, often within 1s
3. ***Blink*** quickly reroutes traffic to working backup paths
4. ***Blink*** works in practice, on existing devices

TCP flows exhibit the same behavior upon failures

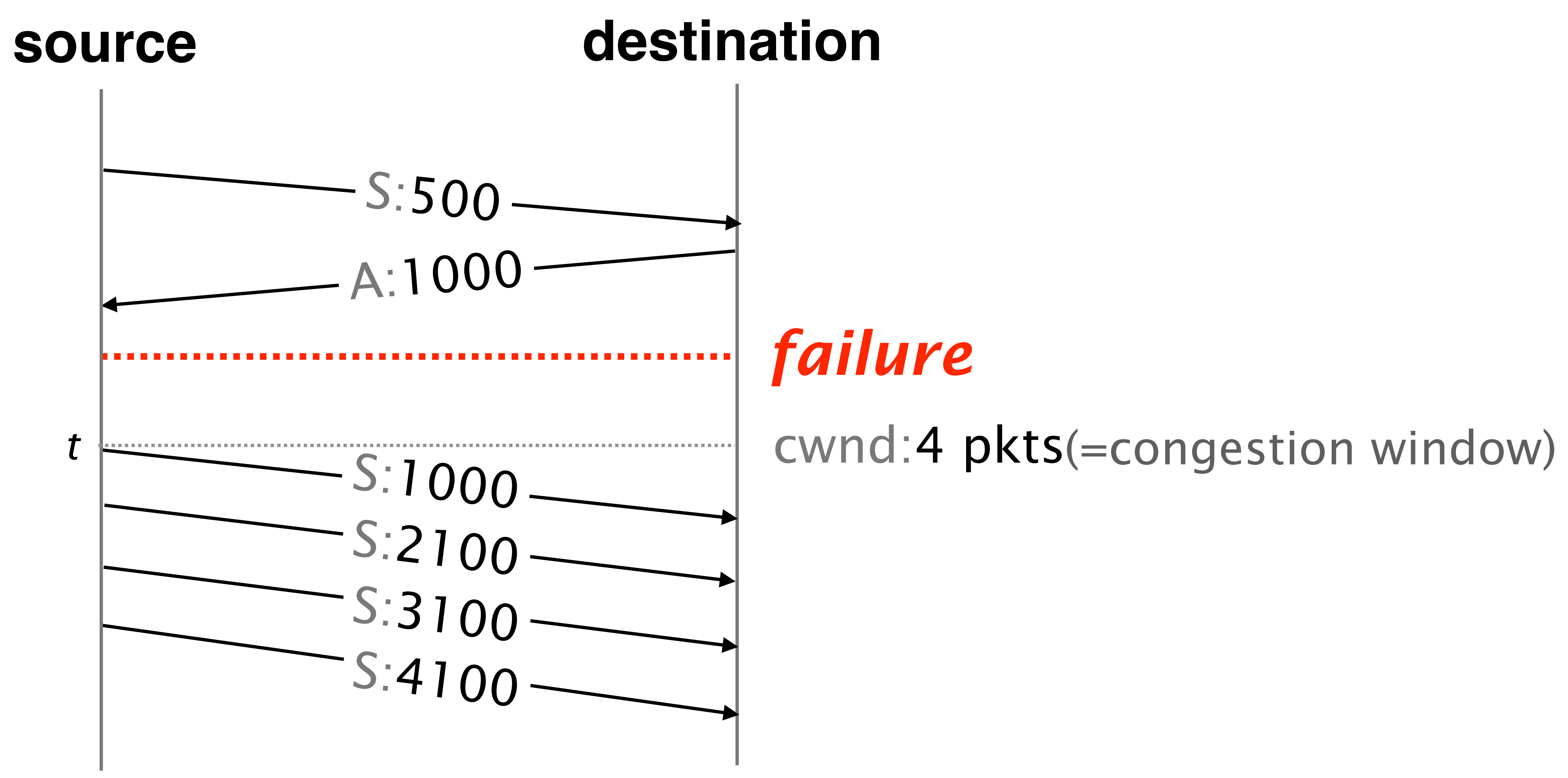
TCP flows exhibit the same behavior upon failures



TCP flows exhibit the same behavior upon failures



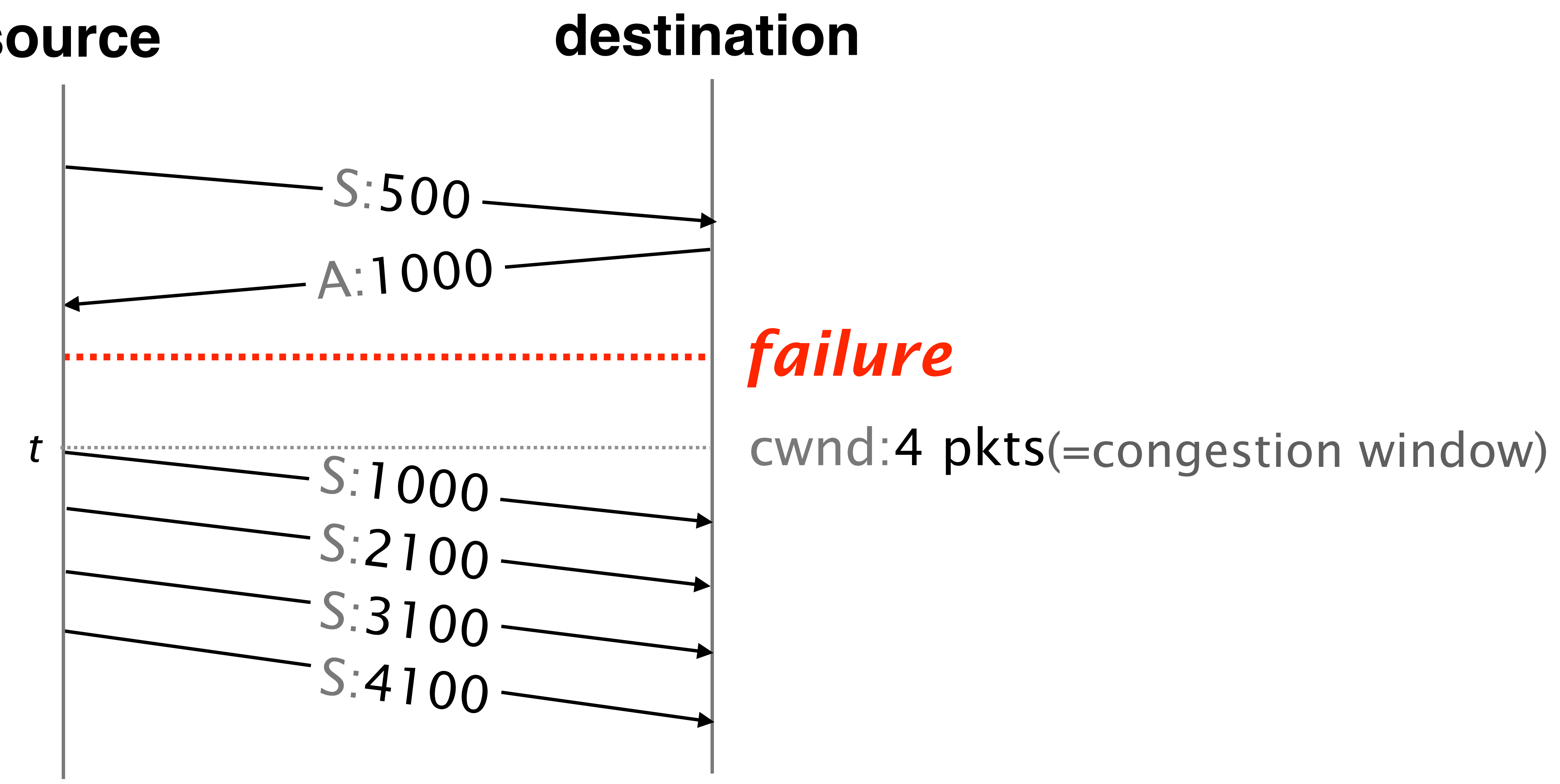
TCP flows exhibit the same behavior upon failures



TCP flows exhibit the same behavior upon failures

Retransmission timeout (RTO)
= SRTT + 4∗RTT_VAR

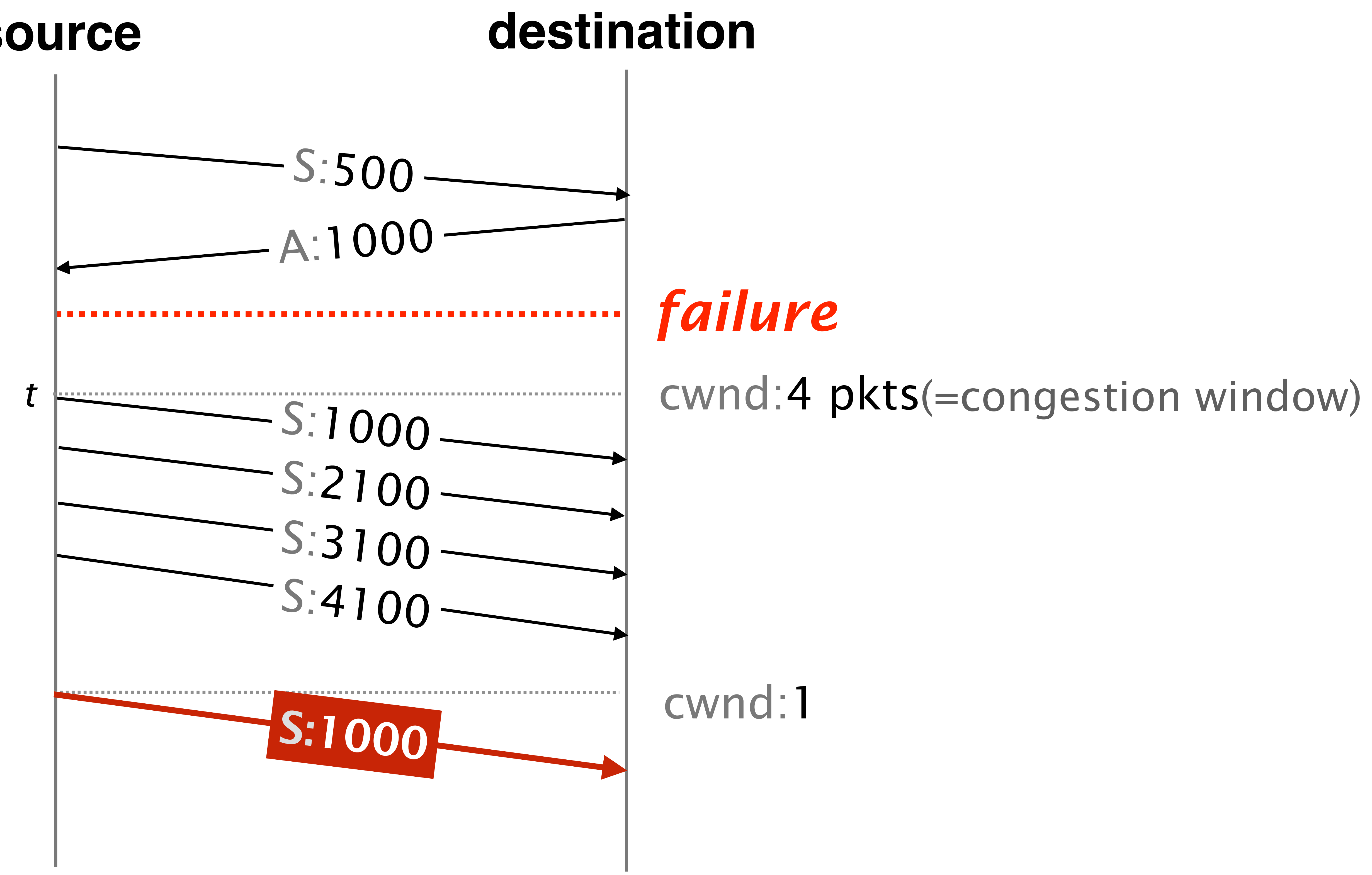
RTO: 200ms



TCP flows exhibit the same behavior upon failures

Retransmission timeout (RTO)
= SRTT + 4∗RTT_VAR

RTO: 200ms

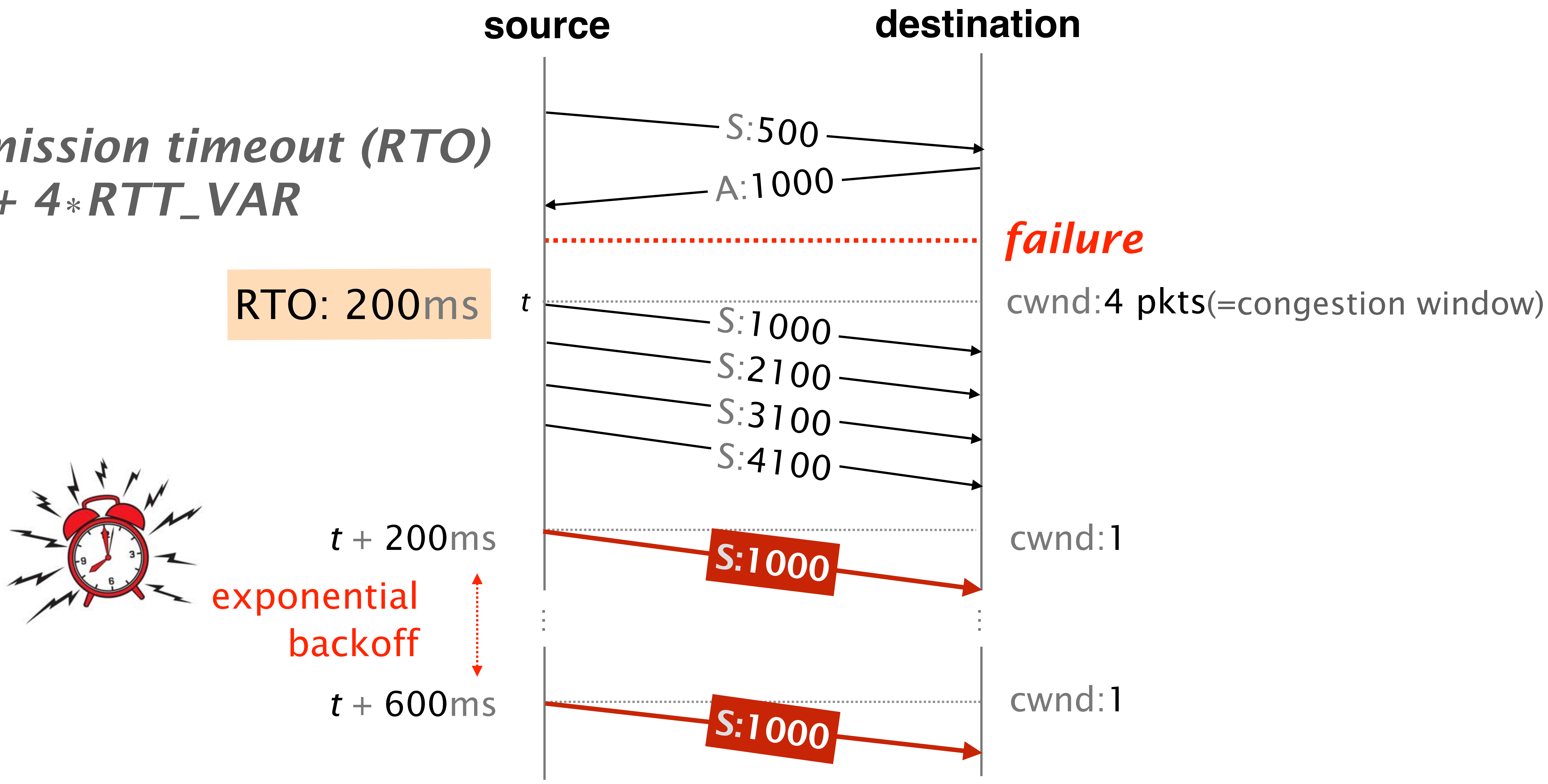


t + 200ms

TCP flows exhibit the same behavior upon failures

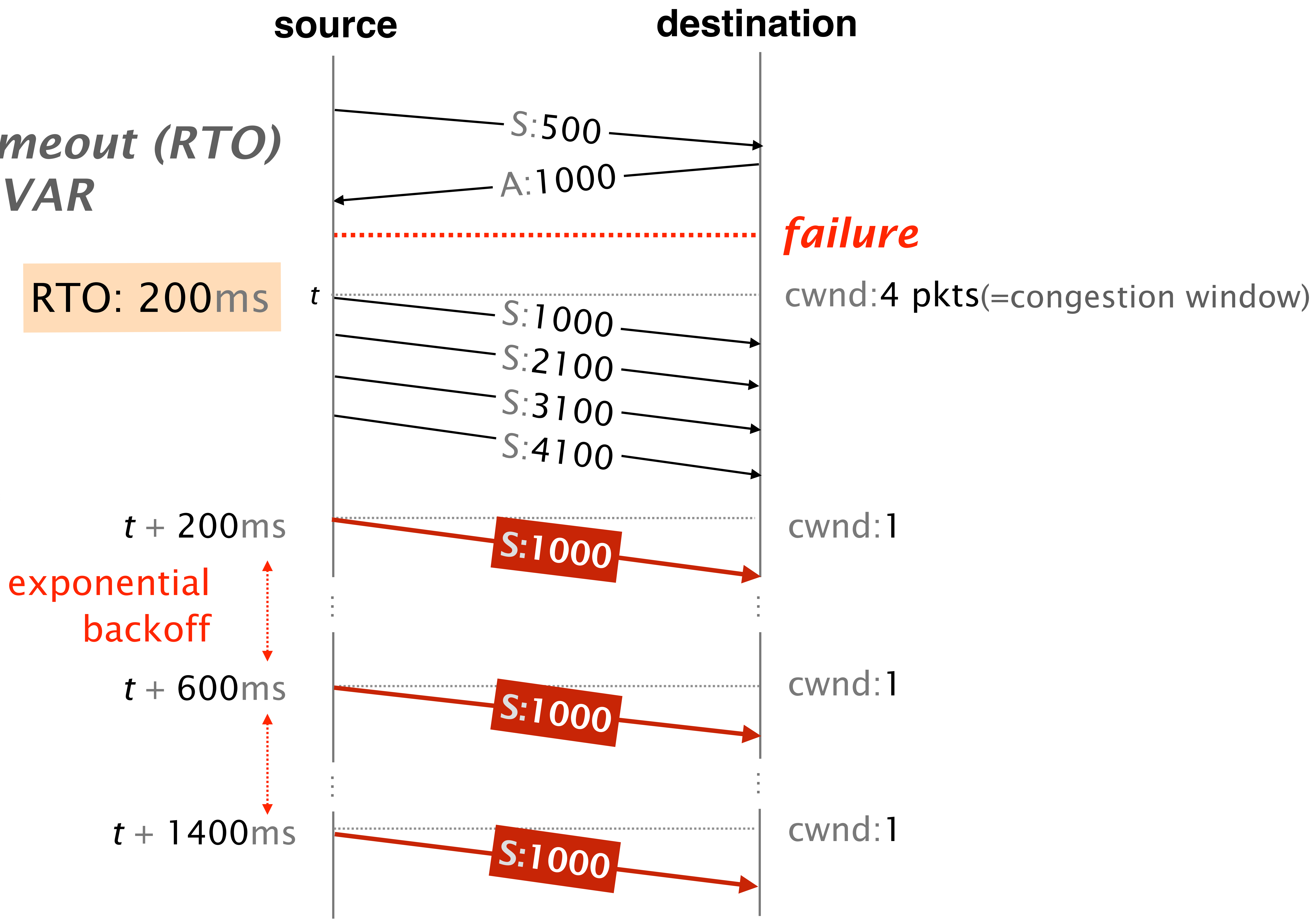
Retransmission timeout (RTO)
= SRTT + 4∗RTT_VAR

RTO: 200ms



TCP flows exhibit the same behavior upon failures

Retransmission timeout (RTO)
= SRTT + 4∗RTT_VAR



When multiple flows experience the same failure
the signal is a wave of retransmissions

When multiple flows experience the same failure
the signal is a wave of retransmissions

We simulated a failure affecting
100k flows with NS3

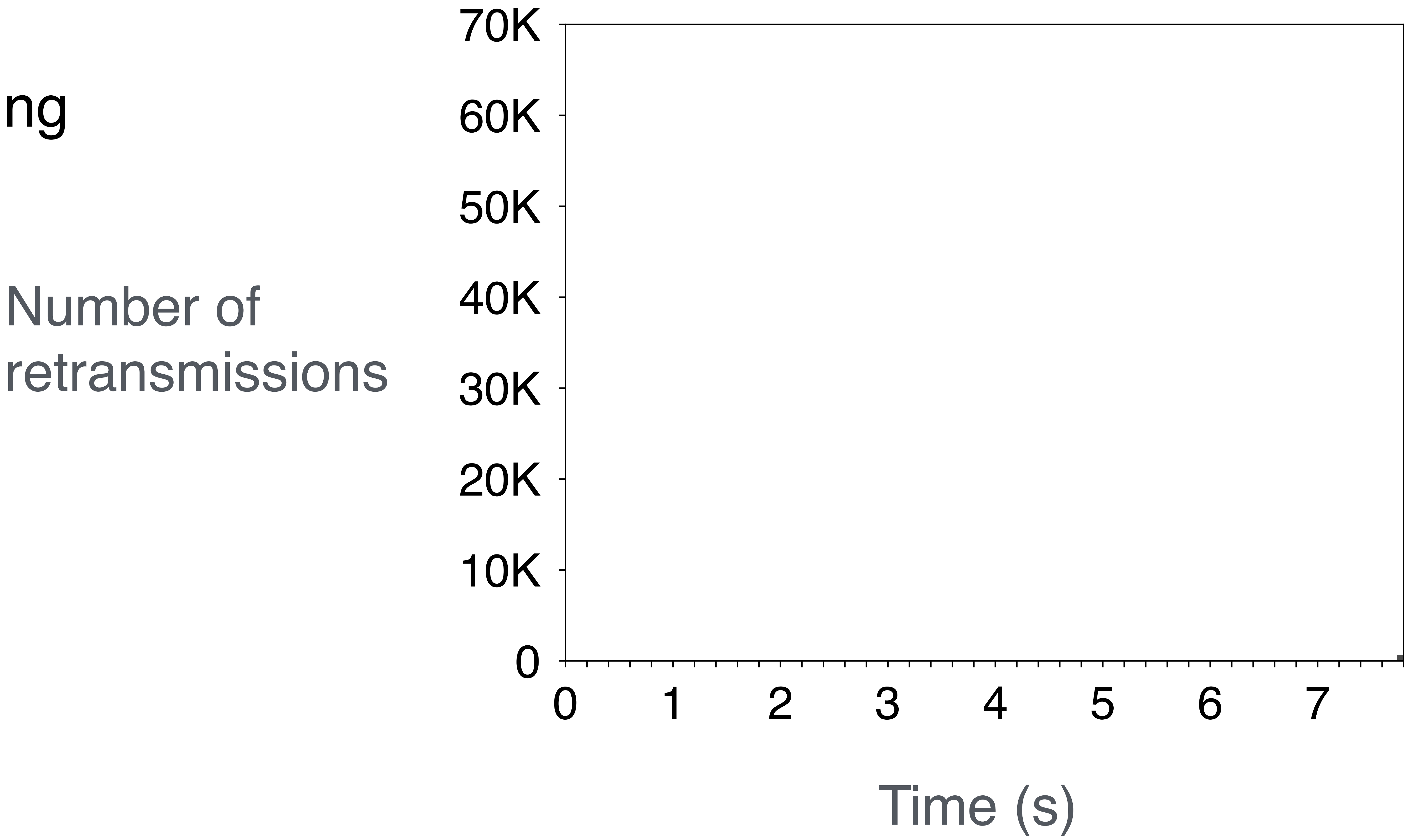
Same RTT distribution
than in a real trace*

*CAIDA equinix-chicago
direction A, 2015

When multiple flows experience the same failure
the signal is a wave of retransmissions

We simulated a failure affecting 100k flows with NS3

Same RTT distribution than in a real trace*

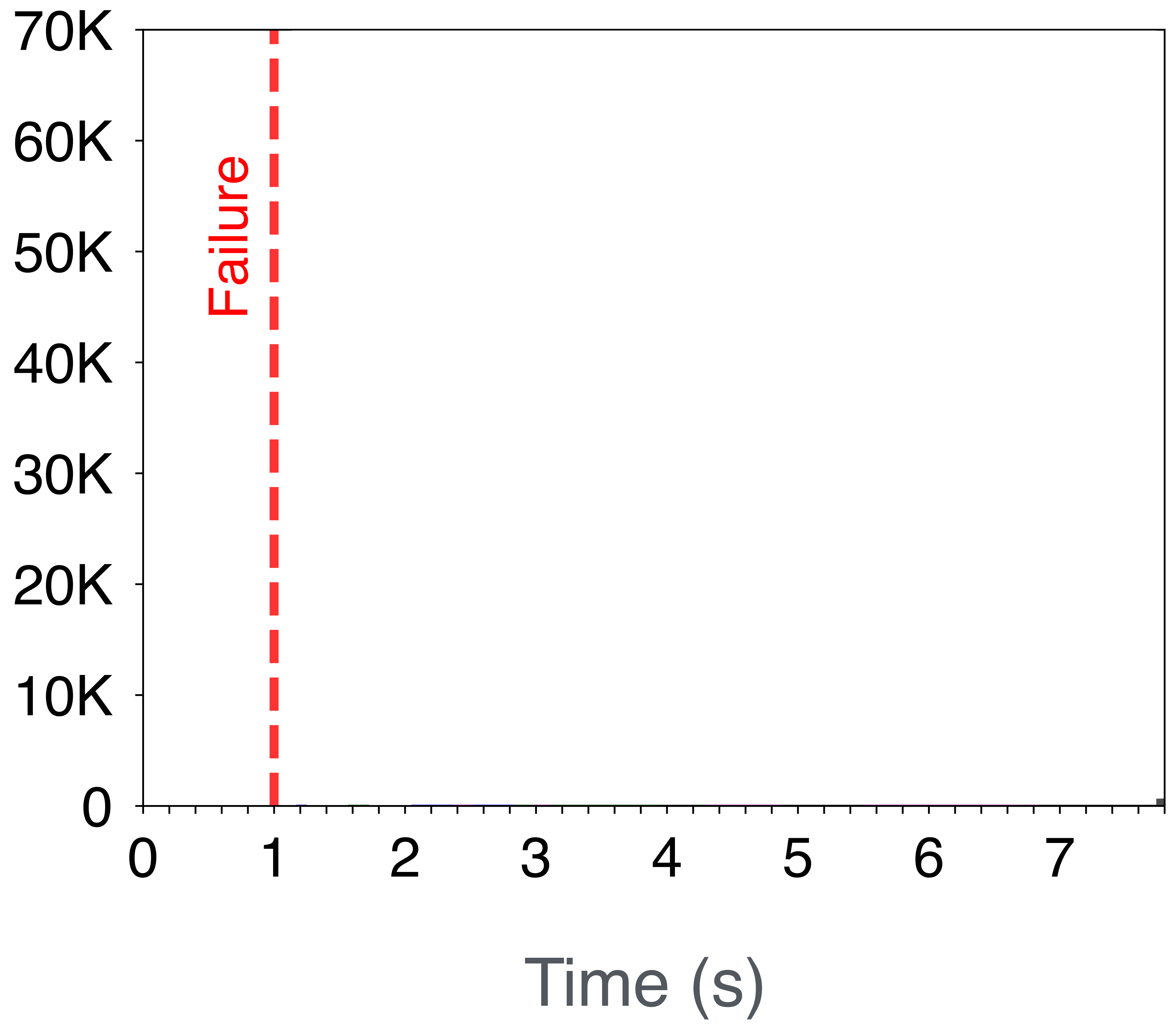


*CAIDA equinix-chicago direction A, 2015

When multiple flows experience the same failure the signal is a wave of retransmissions

We simulated a failure affecting 100k flows with NS3

Same RTT distribution than in a real trace*



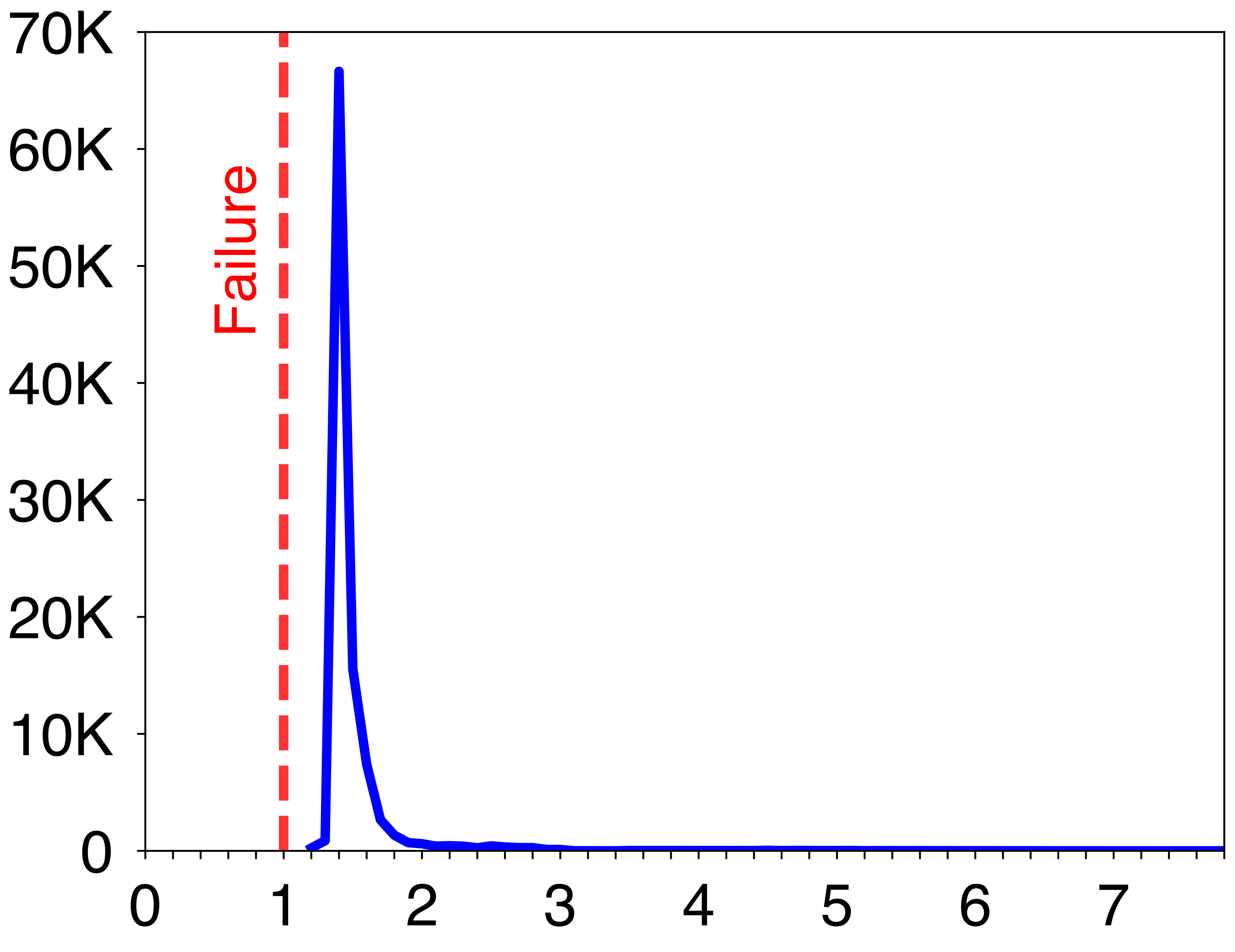
*CAIDA equinix-chicago direction A, 2015

When multiple flows experience the same failure
the signal is a wave of retransmissions

We simulated a failure affecting 100k flows with NS3

Same RTT distribution than in a real trace*

Number of retransmissions



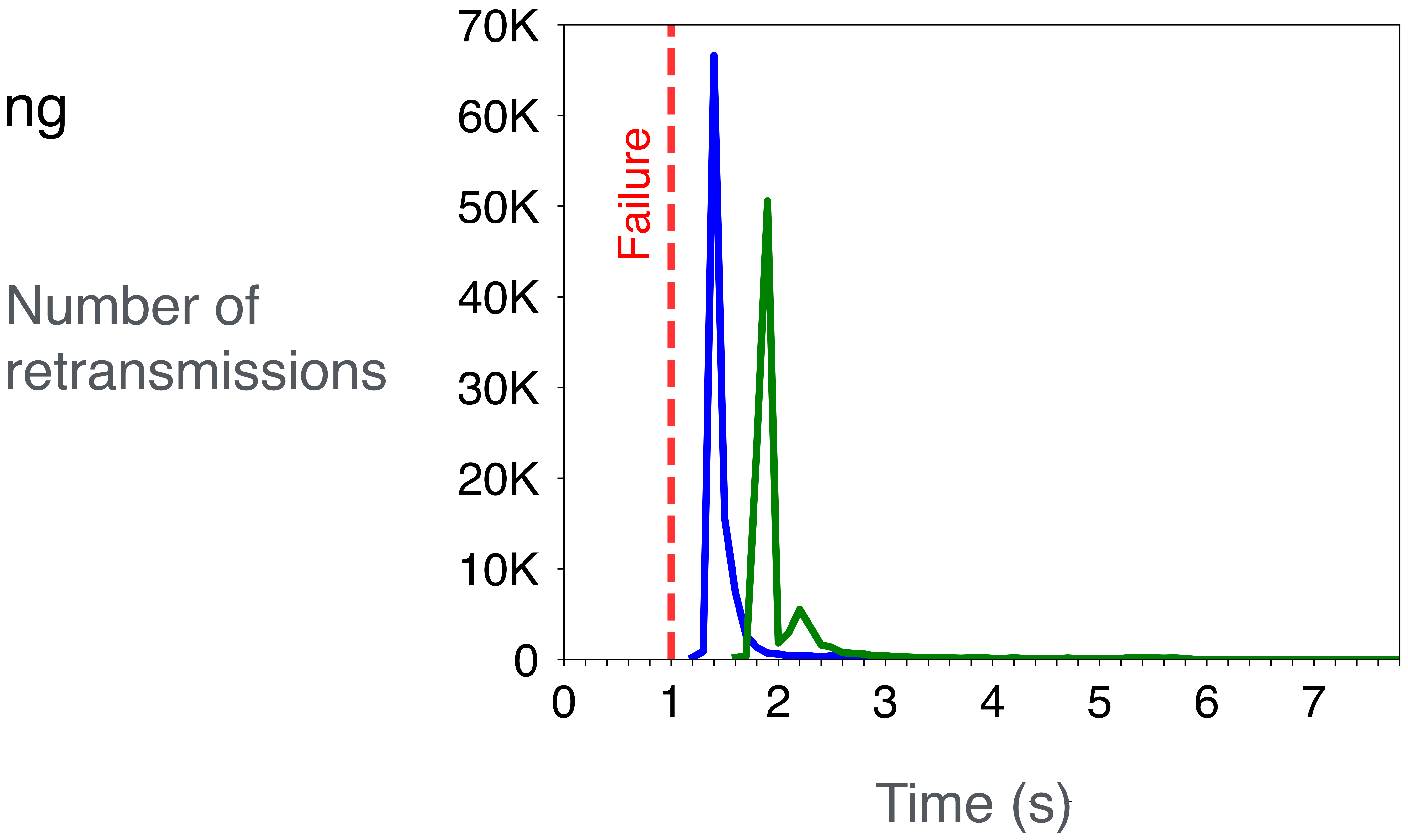
Time (s)

*CAIDA equinix-chicago direction A, 2015

When multiple flows experience the same failure the signal is a wave of retransmissions

We simulated a failure affecting 100k flows with NS3

Same RTT distribution than in a real trace*

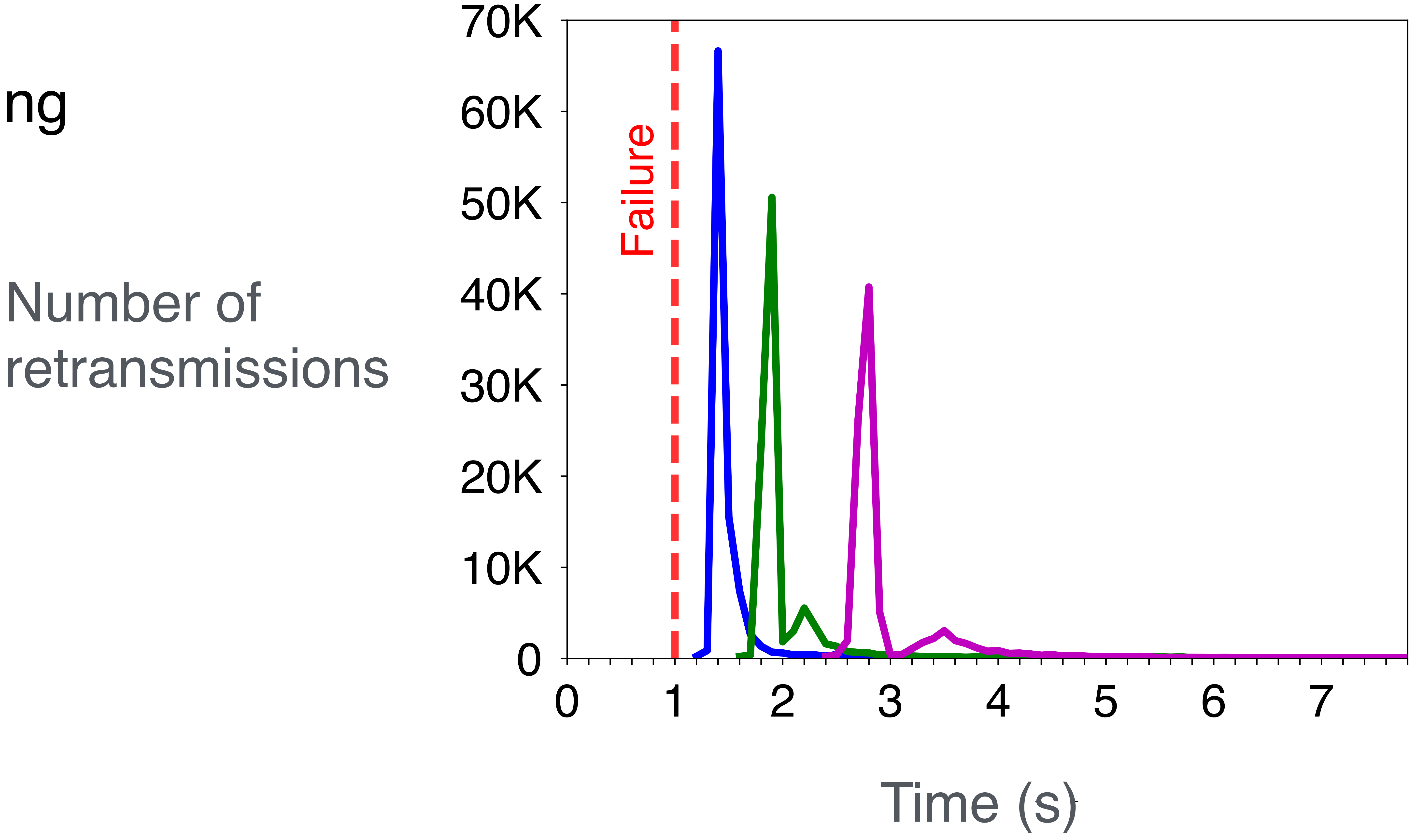


*CAIDA equinix-chicago direction A, 2015

When multiple flows experience the same failure the signal is a wave of retransmissions

We simulated a failure affecting 100k flows with NS3

Same RTT distribution than in a real trace*



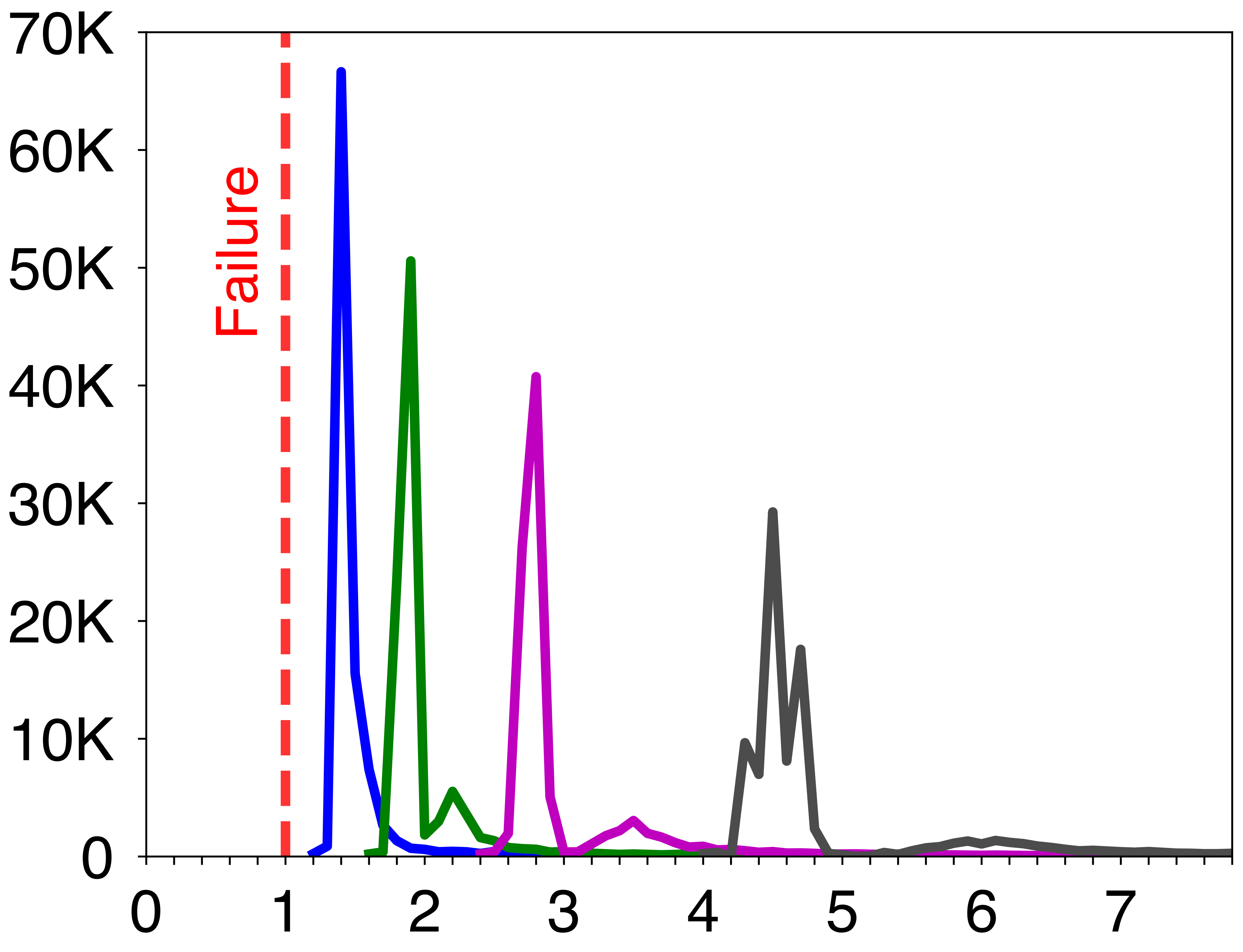
*CAIDA equinix-chicago direction A, 2015

When multiple flows experience the same failure the signal is a wave of retransmissions

We simulated a failure affecting 100k flows with NS3

Same RTT distribution than in a real trace*

Number of retransmissions



Time (s)

*CAIDA equinix-chicago direction A, 2015

Outline

1. Why and how to use data-plane signals for fast rerouting
2. ***Blink*** infers more than 80% of the failures, often within 1s
3. ***Blink*** quickly reroutes traffic to working backup paths
4. ***Blink*** works in practice, on existing devices

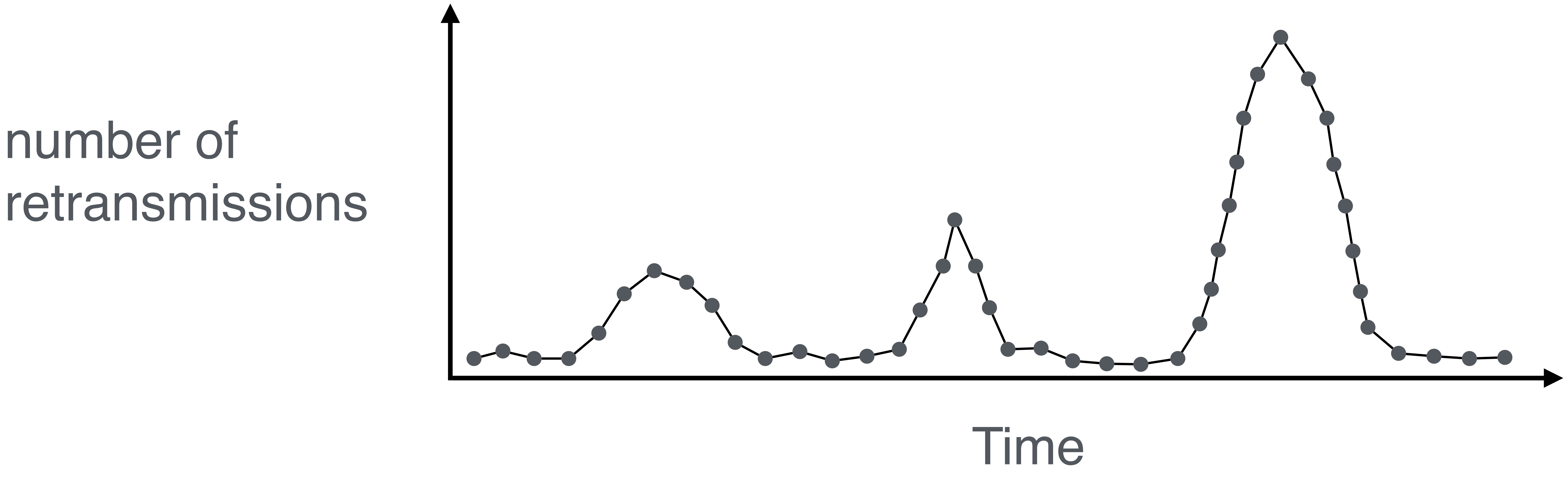
To detect failures, ***Blink*** looks at TCP retransmissions

To detect failures, ***Blink*** looks at TCP retransmissions

Problem: TCP retransmissions can be unrelated to a failure (*i.e.,* noise)

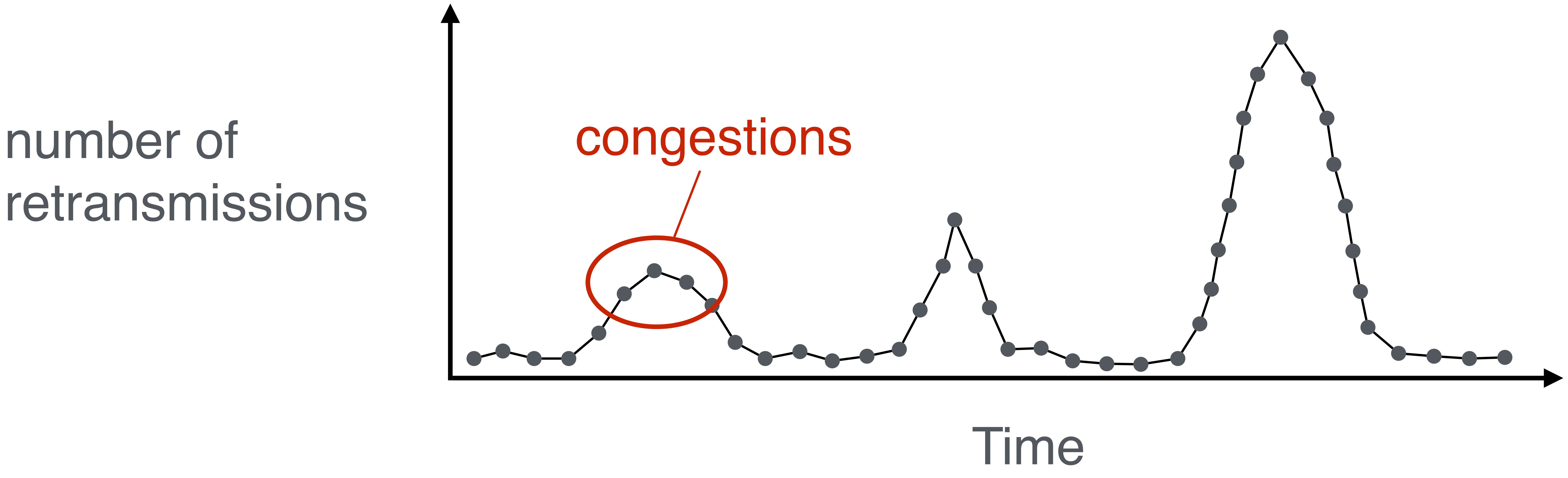
To detect failures, ***Blink*** looks at TCP retransmissions

Problem: TCP retransmissions can be unrelated to a failure (*i.e.*, noise)



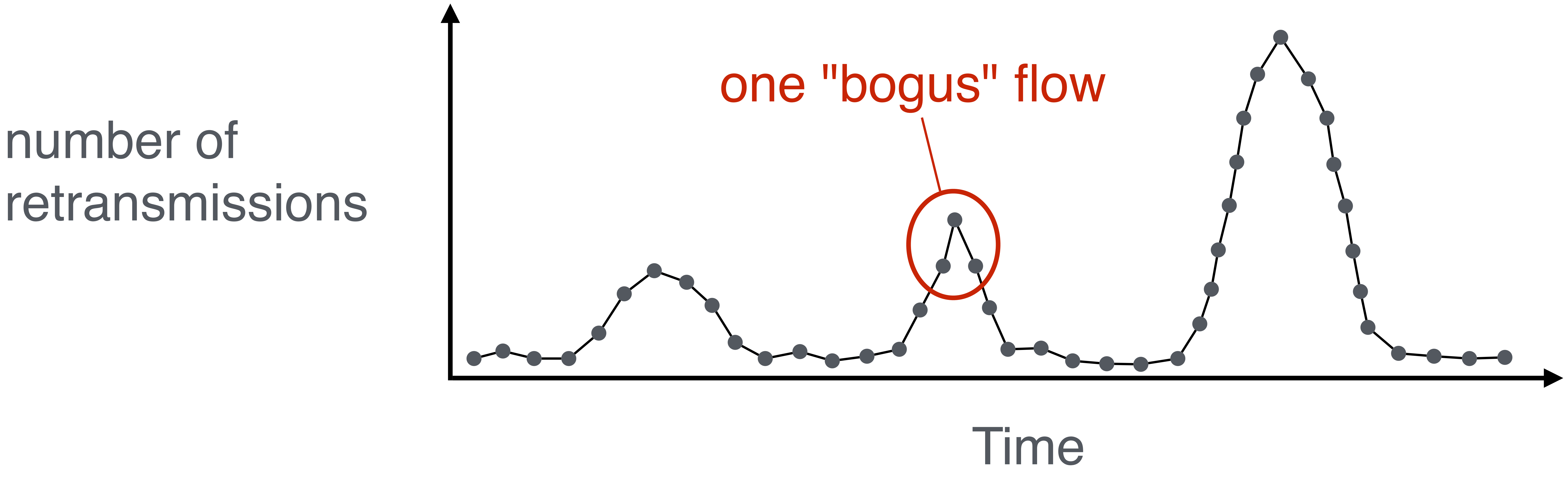
To detect failures, ***Blink*** looks at TCP retransmissions

Problem: TCP retransmissions can be unrelated to a failure (*i.e.*, noise)



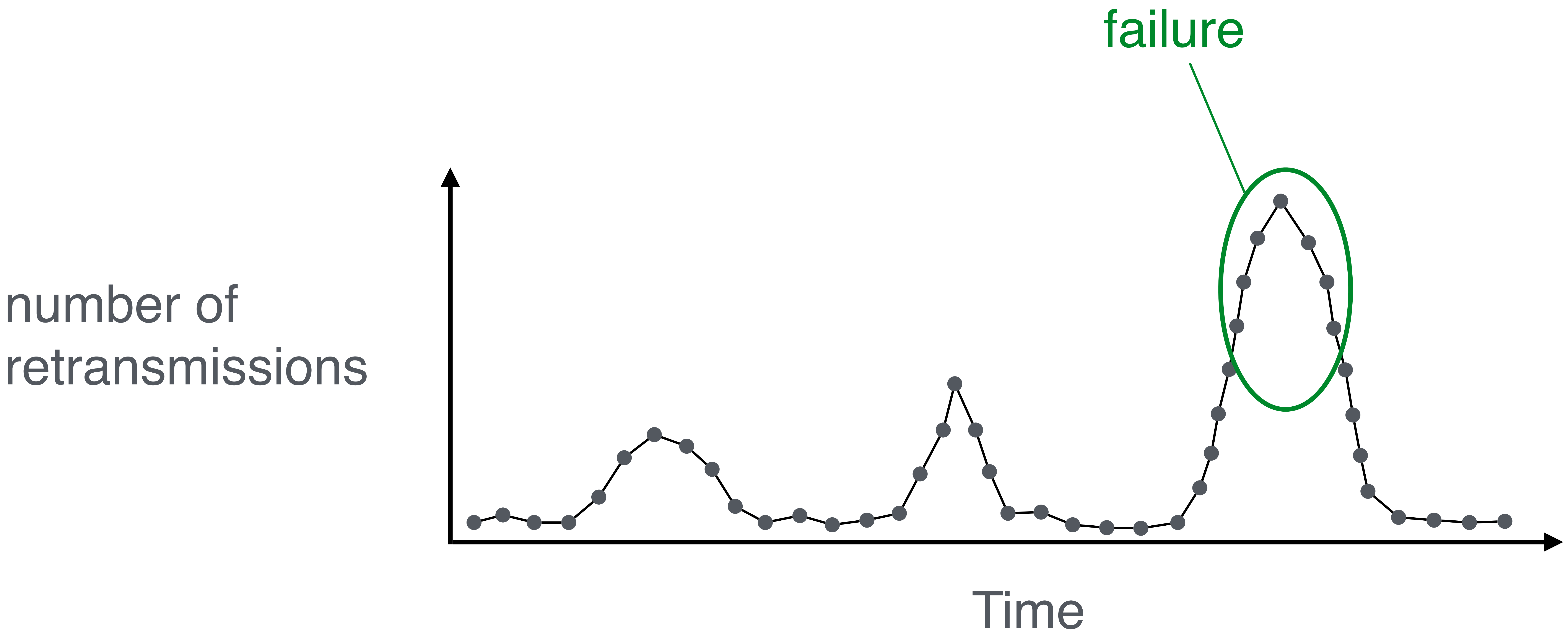
To detect failures, ***Blink*** looks at TCP retransmissions

Problem: TCP retransmissions can be unrelated to a failure (*i.e.*, noise)



To detect failures, ***Blink*** looks at TCP retransmissions

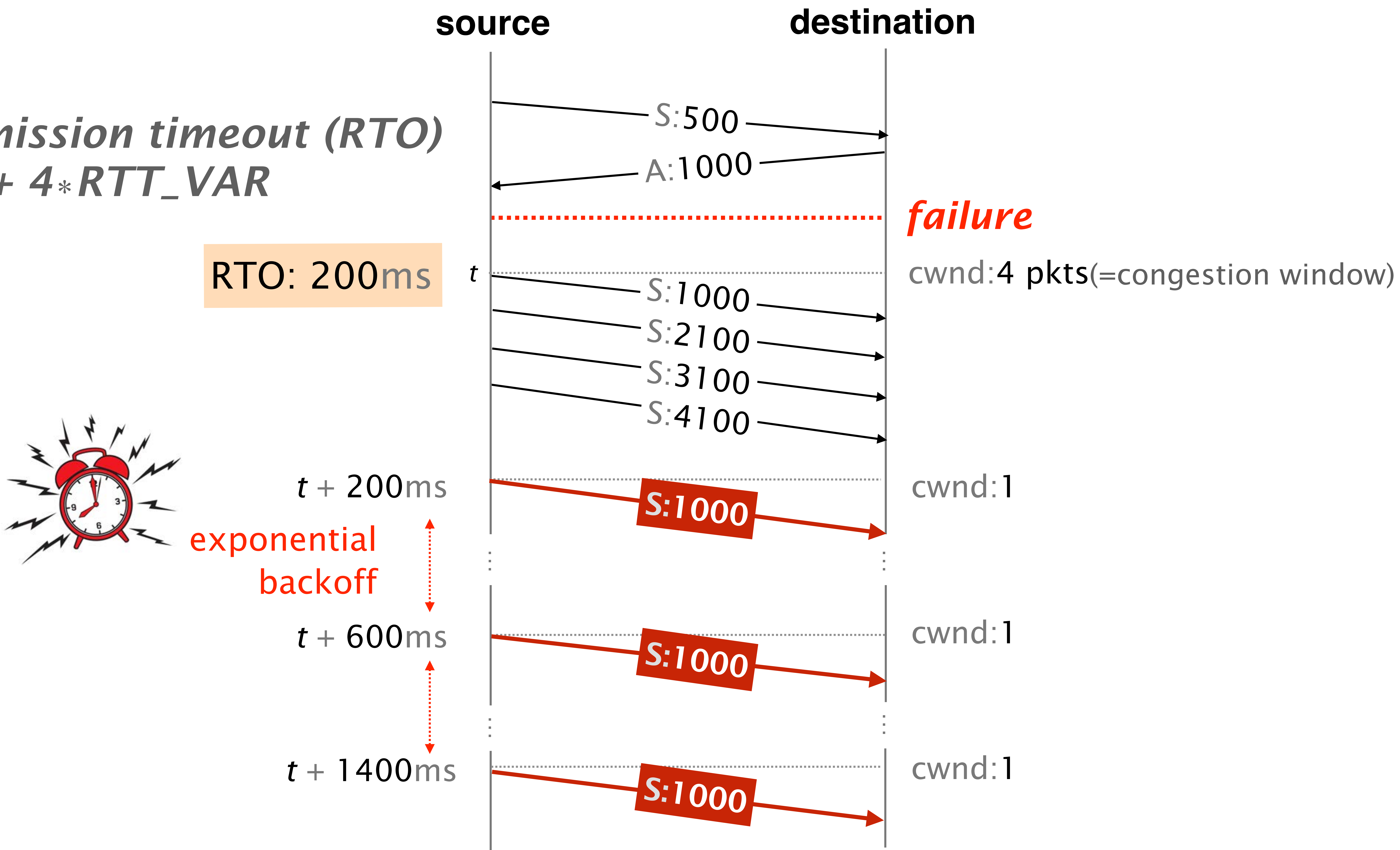
Problem: TCP retransmissions can be unrelated to a failure (*i.e.*, noise)



Solution #1: ***Blink*** looks at consecutive packets with the same sequence number

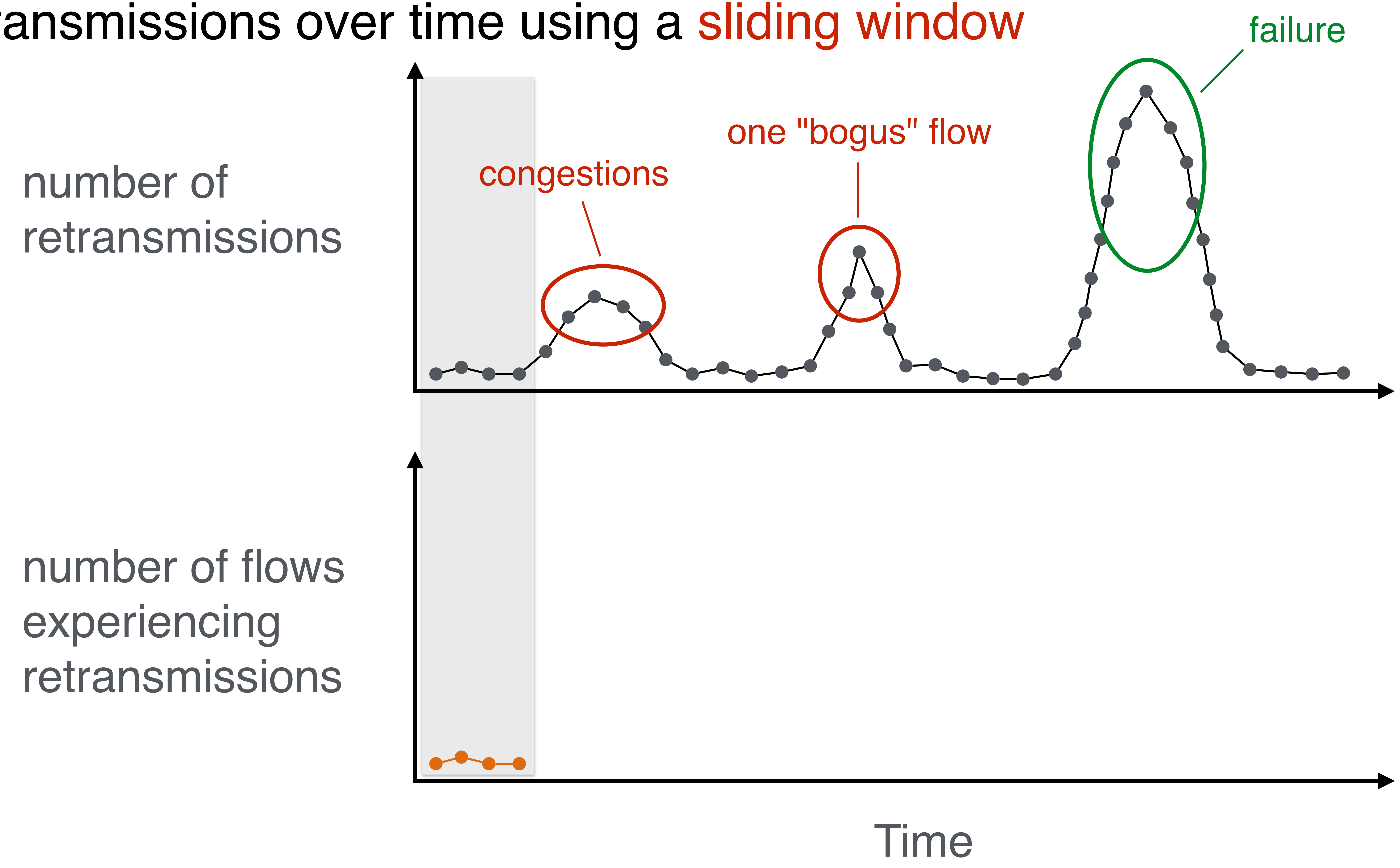
Solution #1: ***Blink*** looks at consecutive packets with the same sequence number

Retransmission timeout (RTO) = SRTT + 4∗RTT_VAR

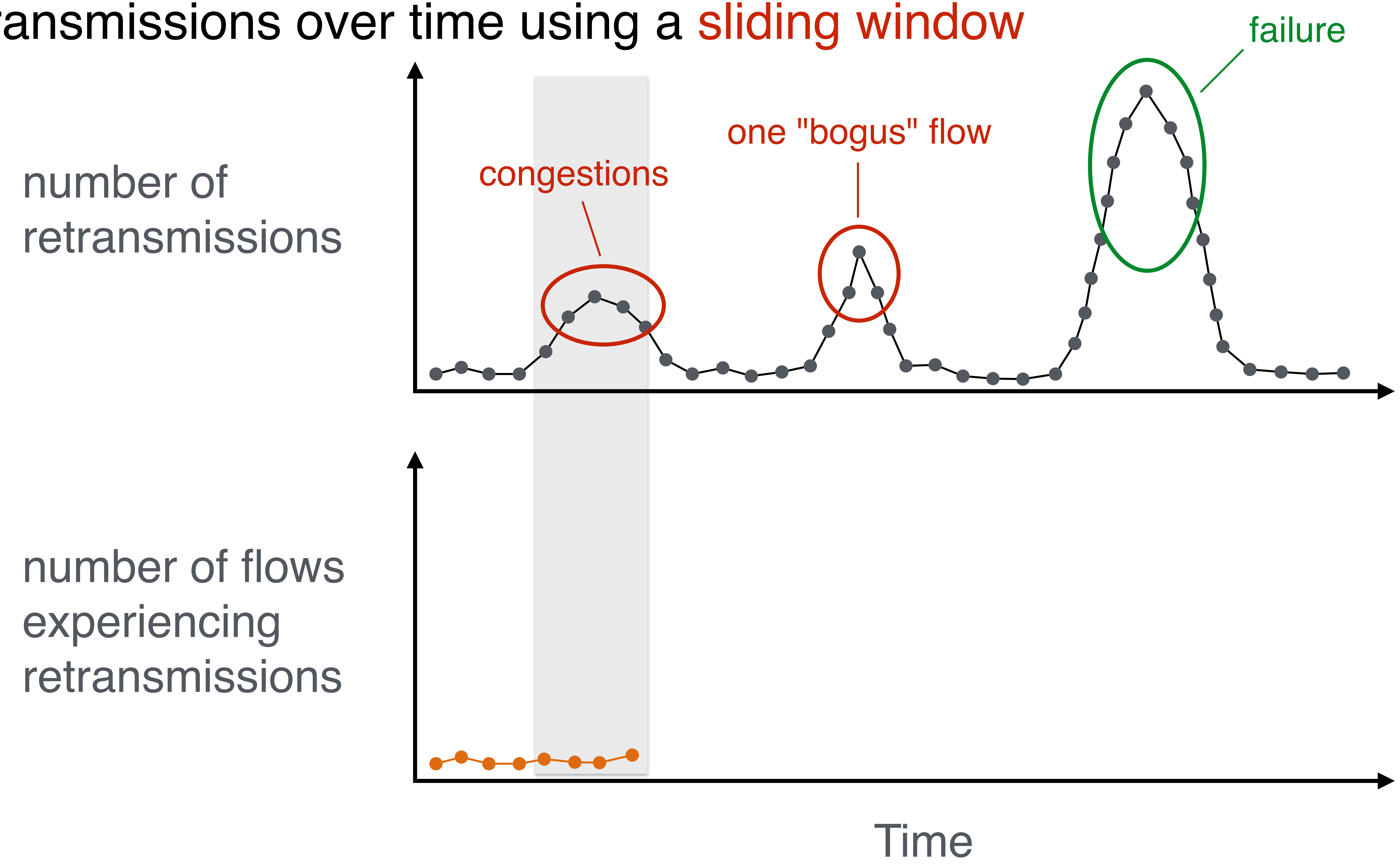


Solution #2: ***Blink*** monitors the number of flows experiencing retransmissions over time using a sliding window

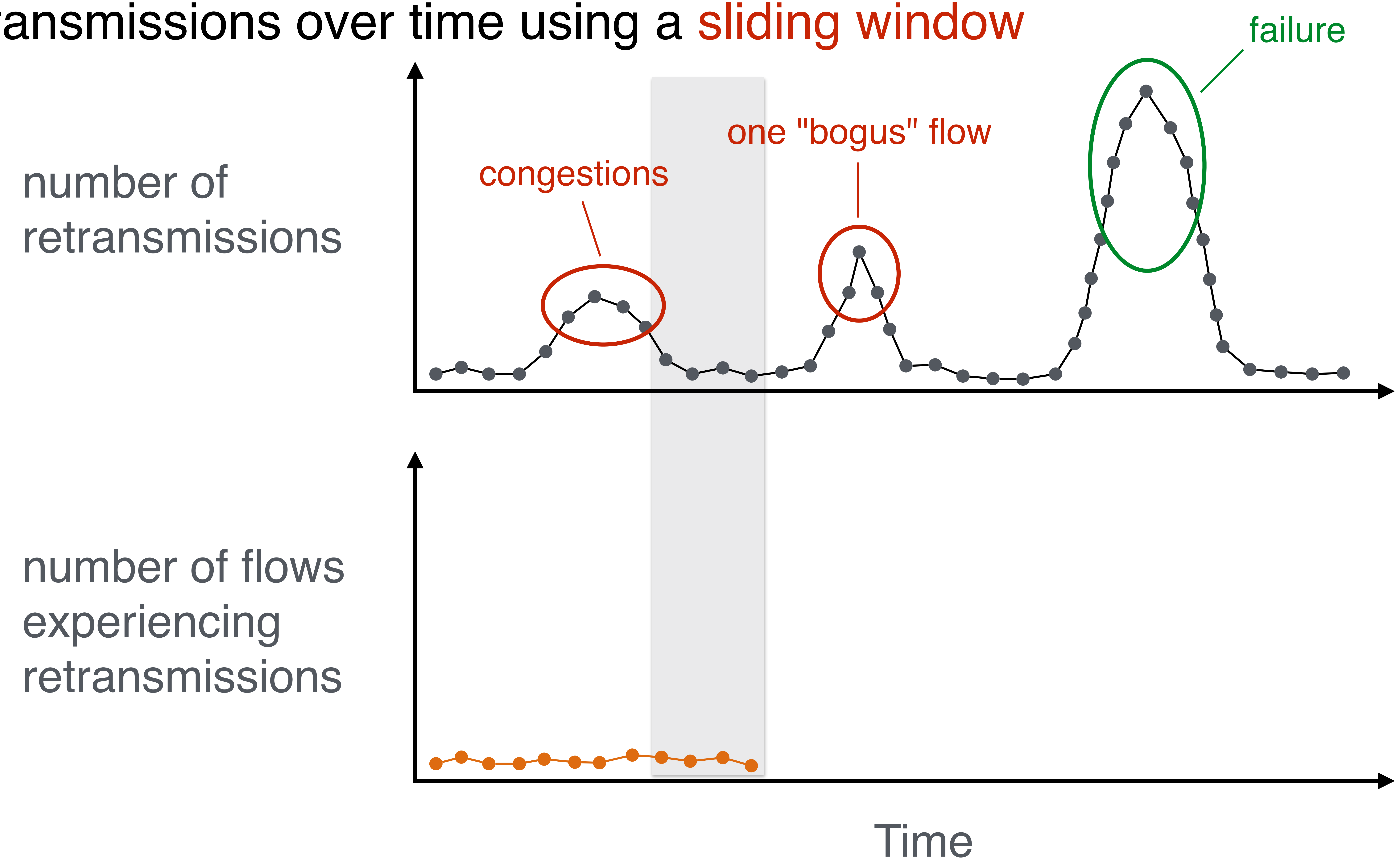
Solution #2: ***Blink*** monitors the number of flows experiencing retransmissions over time using a sliding window



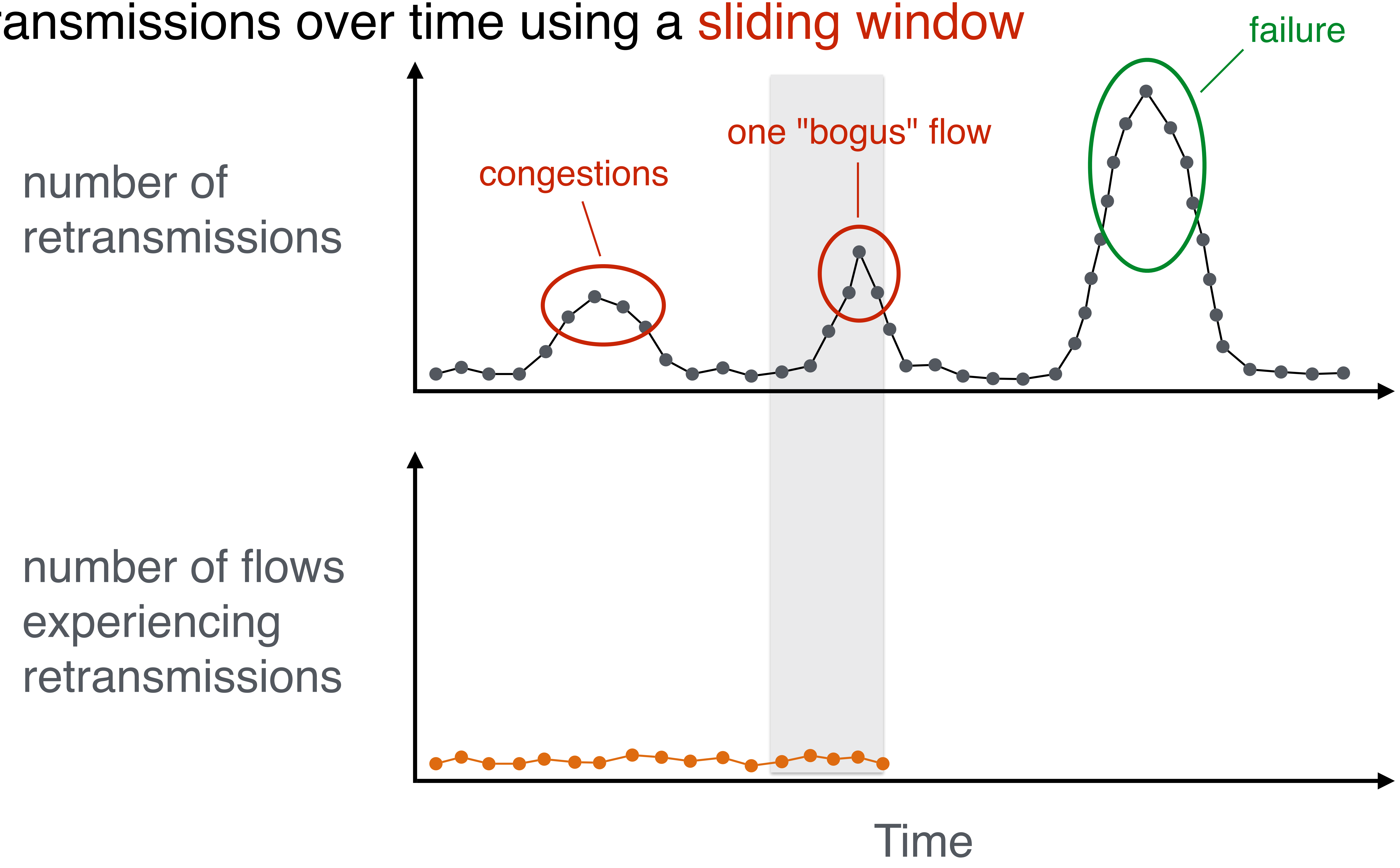
Solution #2: *Blink* monitors the number of flows experiencing retransmissions over time using a sliding window



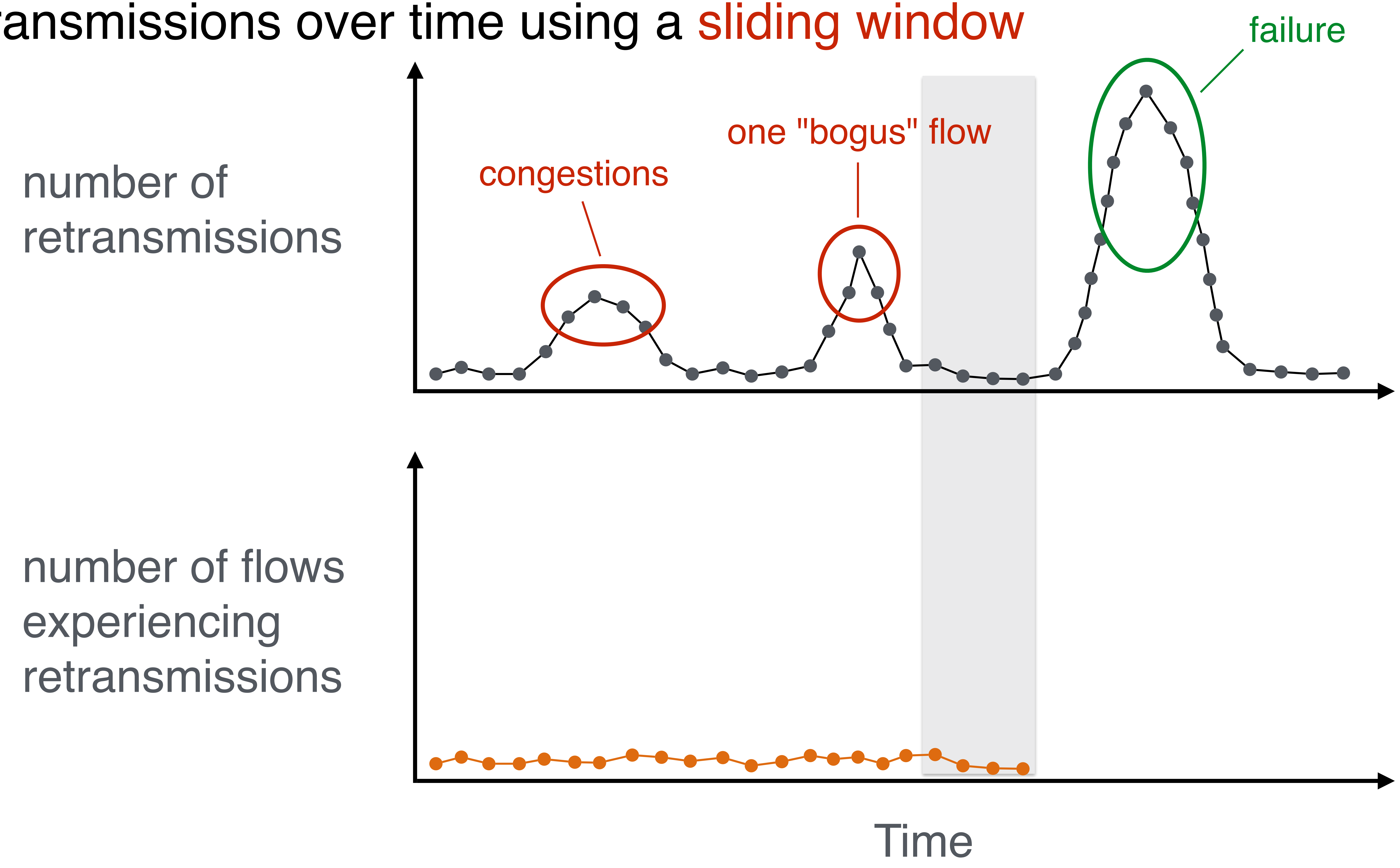
Solution #2: *Blink* monitors the number of flows experiencing retransmissions over time using a sliding window



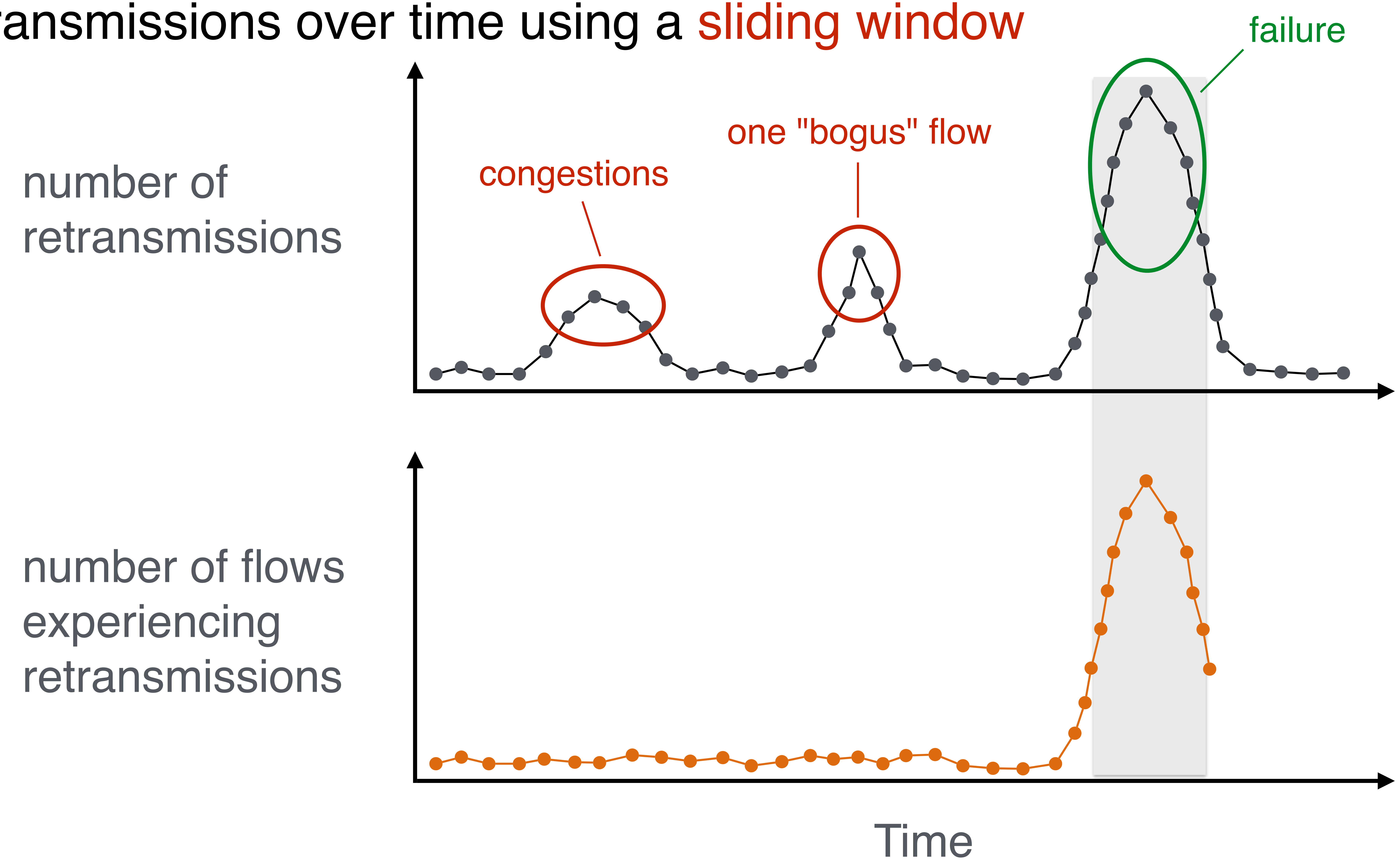
Solution #2: ***Blink*** monitors the number of flows experiencing retransmissions over time using a sliding window



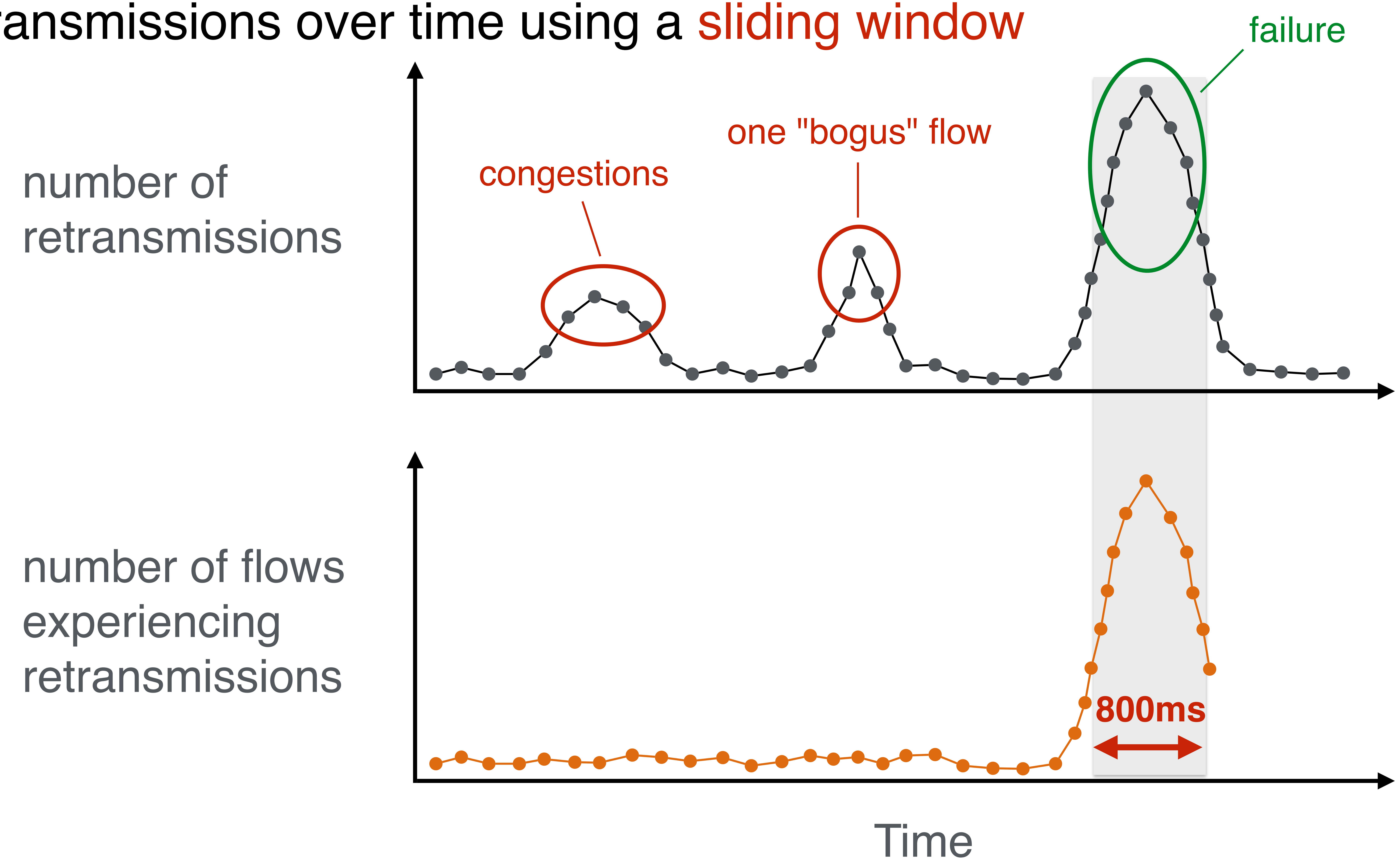
Solution #2: ***Blink*** monitors the number of flows experiencing retransmissions over time using a sliding window



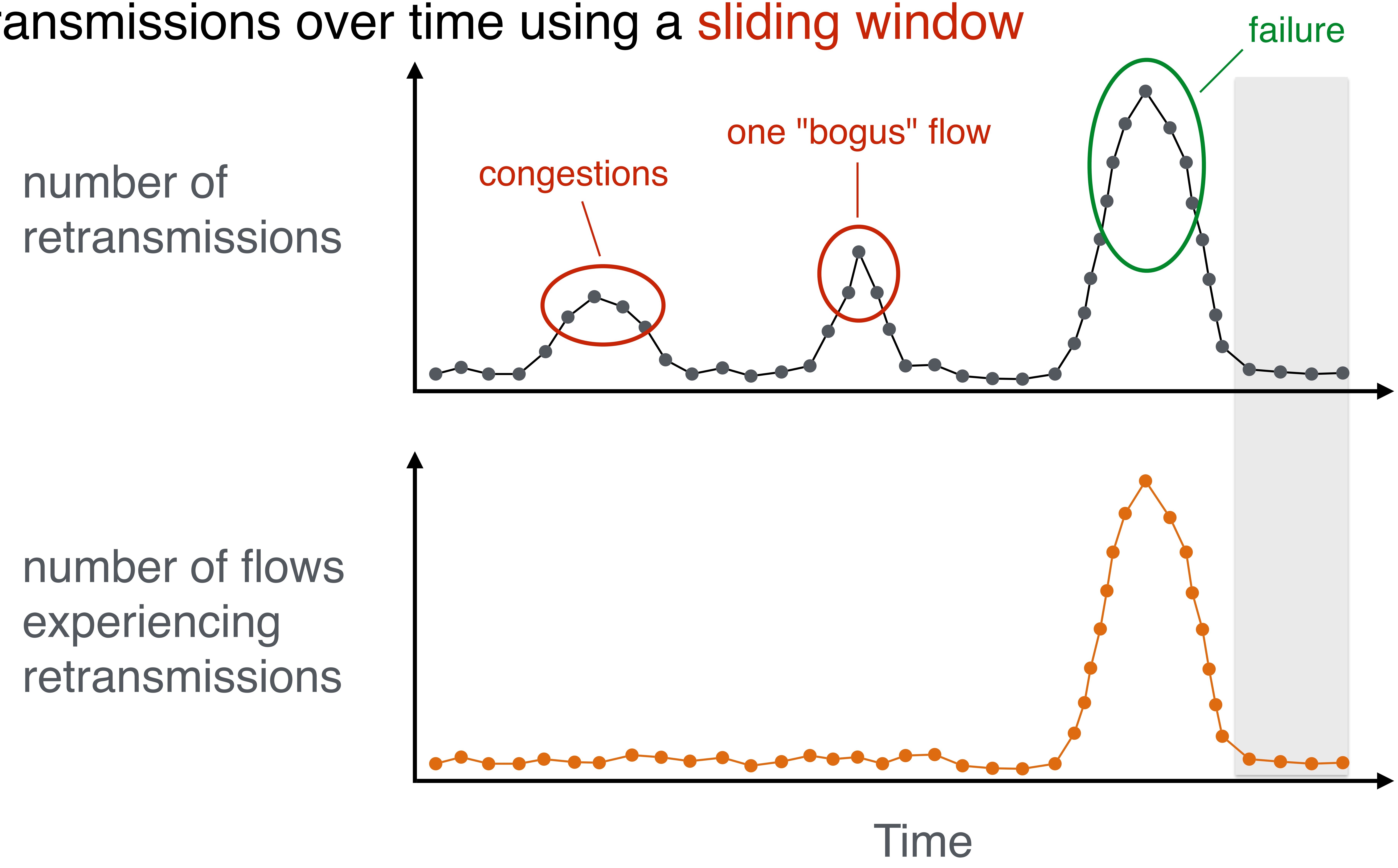
Solution #2: ***Blink*** monitors the number of flows experiencing retransmissions over time using a sliding window



Solution #2: ***Blink*** monitors the number of flows experiencing retransmissions over time using a sliding window



Solution #2: ***Blink*** monitors the number of flows experiencing retransmissions over time using a sliding window



Blink is intended to run in programmable switches

Blink is intended to run in programmable switches

Problem: those switches have very limited resources

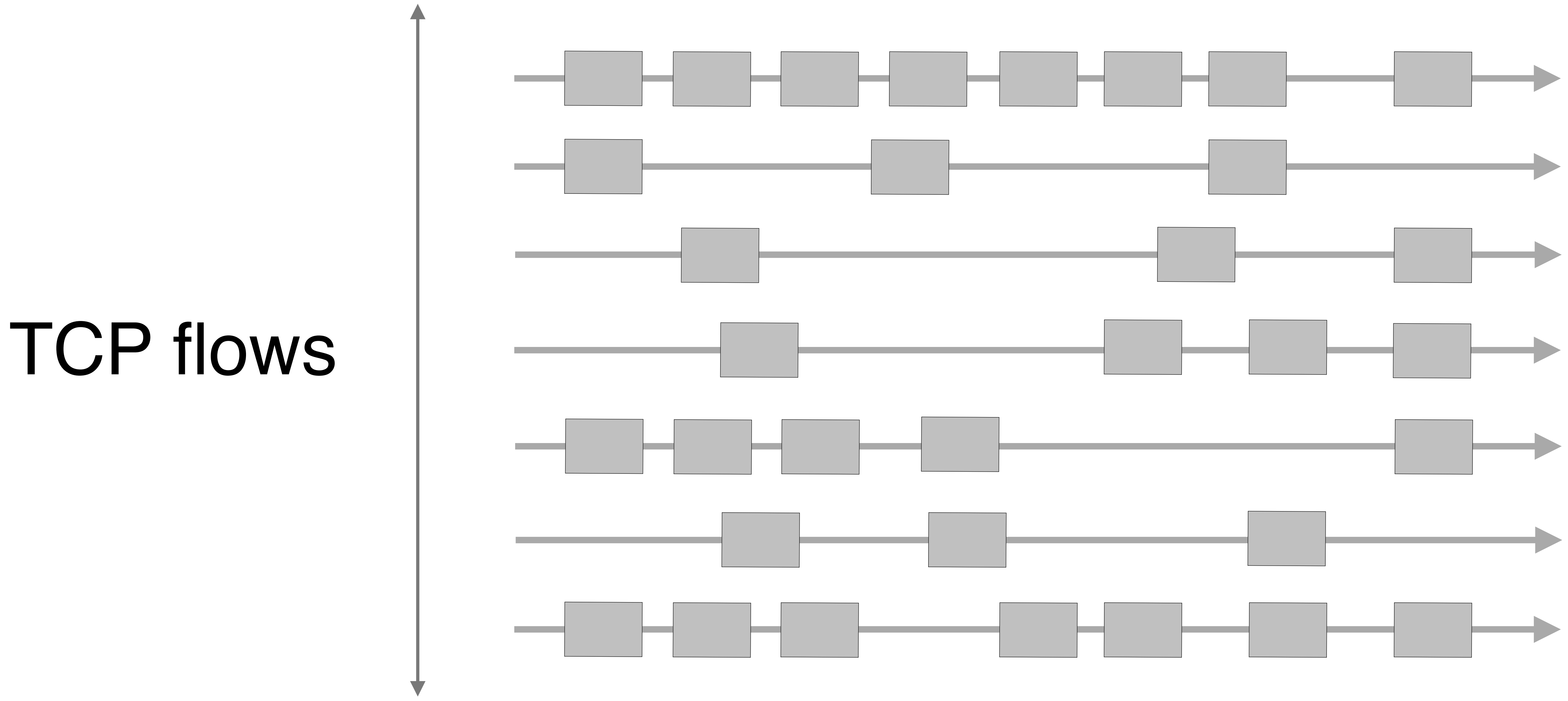
Solution #1: ***Blink*** focuses on the popular prefixes, *i.e.*, the ones that attract data traffic

Solution #1: ***Blink*** focuses on the popular prefixes, *i.e.*, the ones that attract data traffic

As Internet traffic follows a Zipf-like distribution* (1k pref. account for >50%), ***Blink*** covers the vast majority of the Internet traffic

*Sarra et al. Leveraging Zipf's Law for Traffic offloading ACM CCR, 2012

Solution #2: ***Blink*** monitors a sample of the flows for each monitored prefix



Traffic to a destination prefix

Solution #2: ***Blink*** monitors a sample of the flows for each monitored prefix

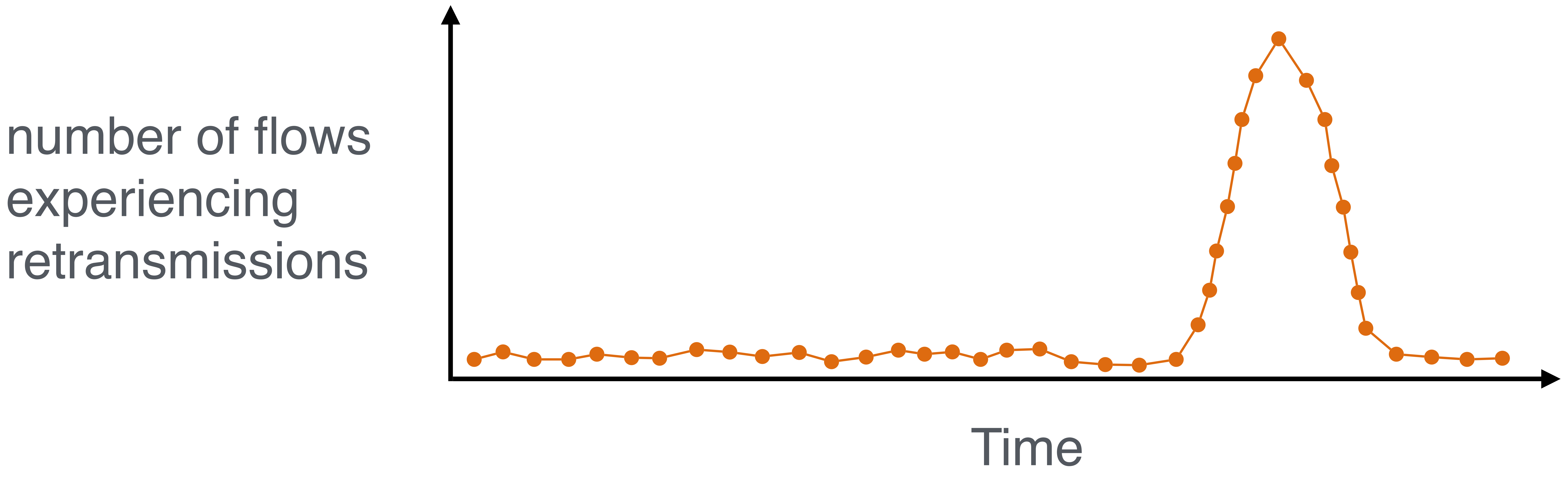
To monitor active flows, ***Blink*** evicts a flow from the sample if it does not send a packet for a given time (default 2s)

To monitor active flows, ***Blink*** evicts a flow from the sample if it does not send a packet for a given time (default 2s)

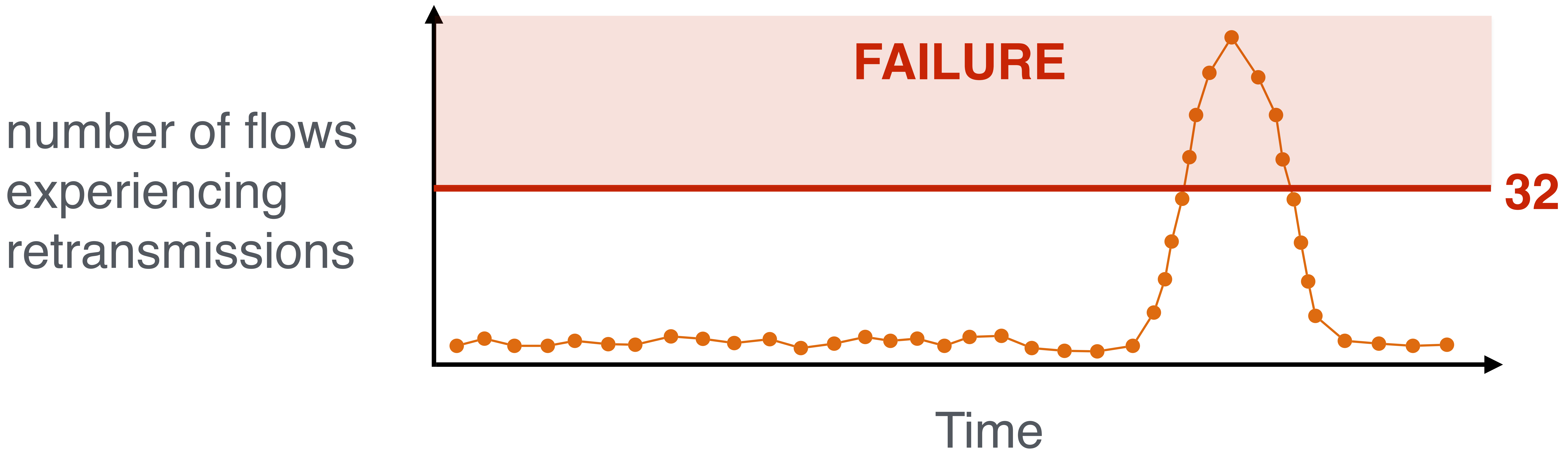
and selects a new one in a *first-seen, first-selected* manner

Blink infers a failure for a prefix when the majority of the monitored flows experience retransmissions

Blink infers a failure for a prefix when the majority of the monitored flows experience retransmissions



Blink infers a failure for a prefix when the majority of the monitored flows experience retransmissions



We evaluated ***Blink*** failure inference using 15 real traces,

13 from CAIDA, 2 from MAWI, covering a total of 15.8 hours

We evaluated ***Blink*** failure inference using 15 real traces,
13 from CAIDA, 2 from MAWI, covering a total of 15.8 hours

We are interested in:

Accuracy: True Positive Rate vs False Positive Rate

Speed: How long does Blink take to infer failures

As we do not have ground truth, we generated synthetic traces following the traffic characteristics extracted from the real traces

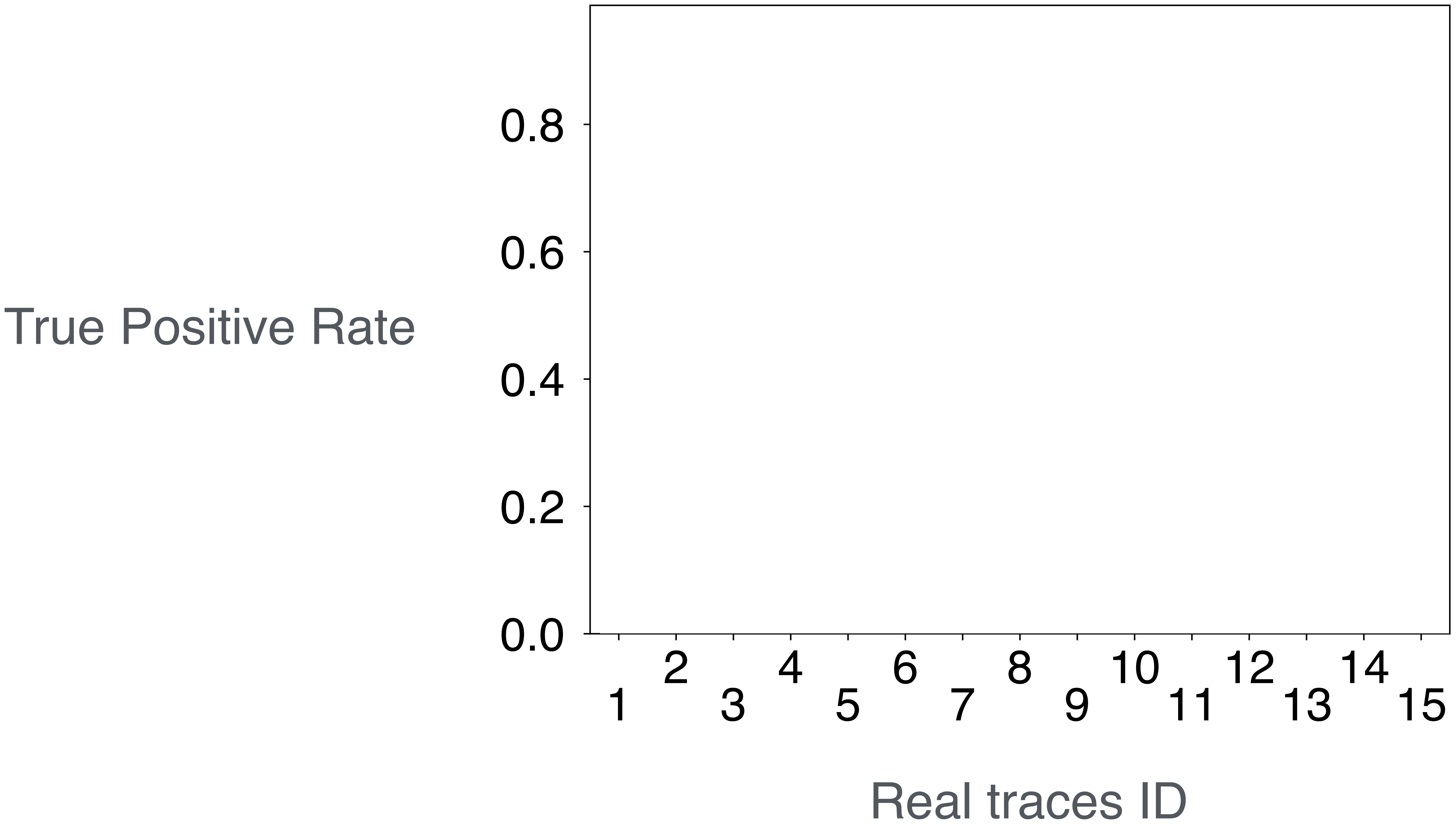
As we do not have ground truth, we generated synthetic traces following the traffic characteristics extracted from the real traces

Step #1 - We extracted the RTT, Packet rate, Flow duration from the real traces

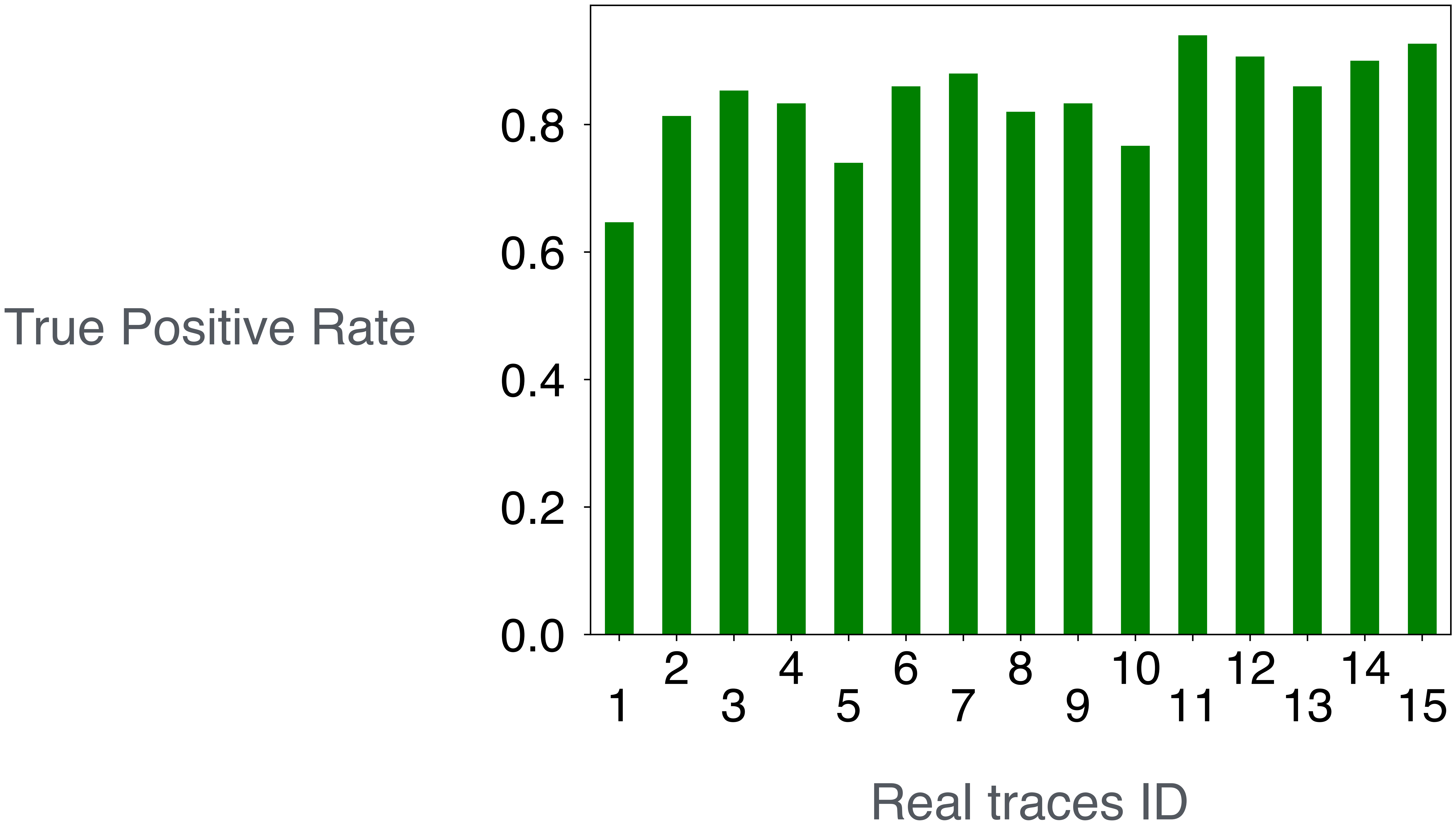
Step #2 - We used NS3 to replay these flows and simulate a failure

Step #3 - We ran a Python-based version of ***Blink*** on the resulting traces

Blink failure inference accuracy is above 80% for 13 real traces out of 15



Blink failure inference accuracy is above 80% for 13 real traces out of 15



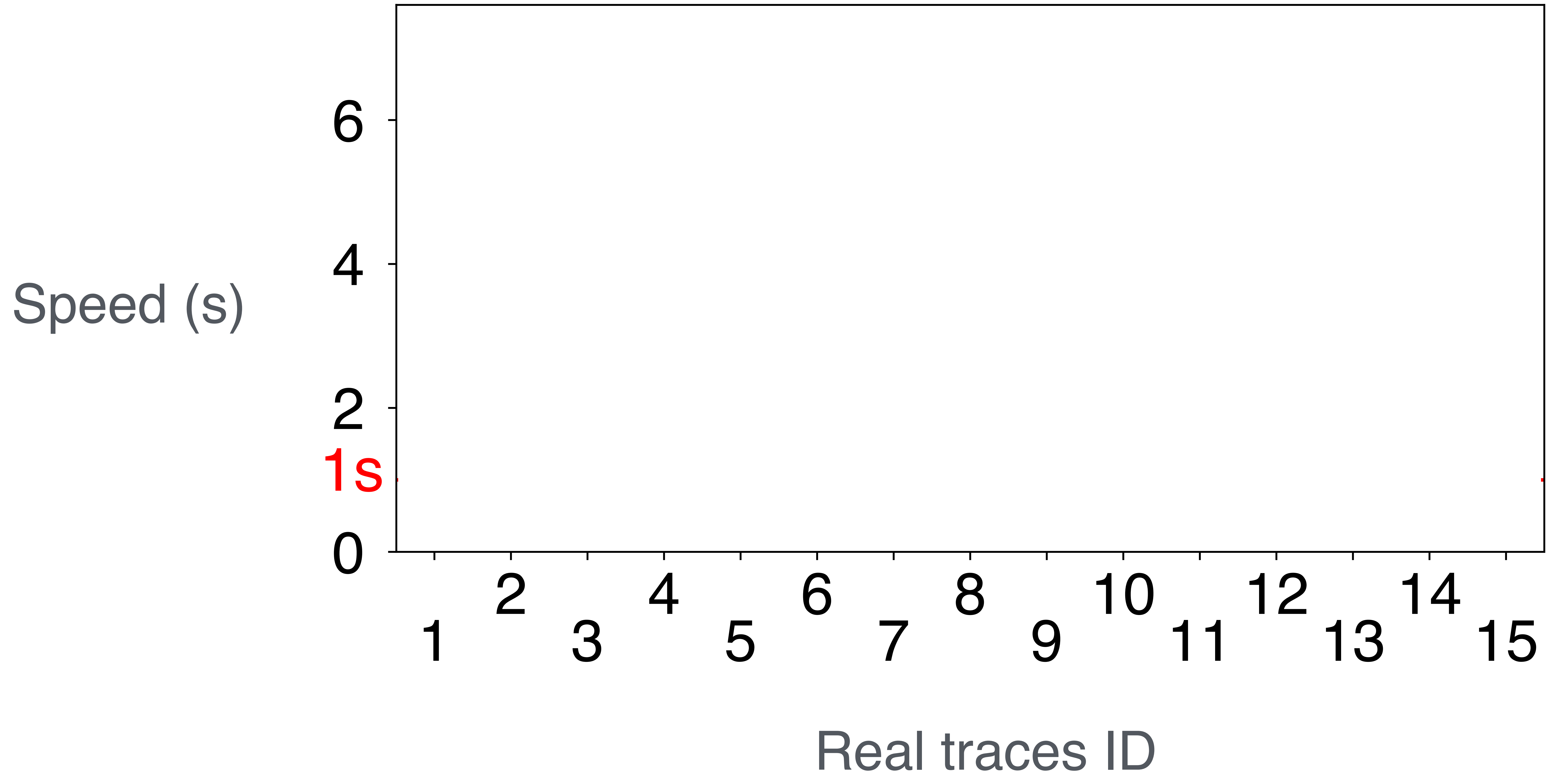
Blink avoids incorrectly inferring failures when packet loss is below 4%

packet loss %	1	2	3	4	5	…	8	9
False Positive Rate								

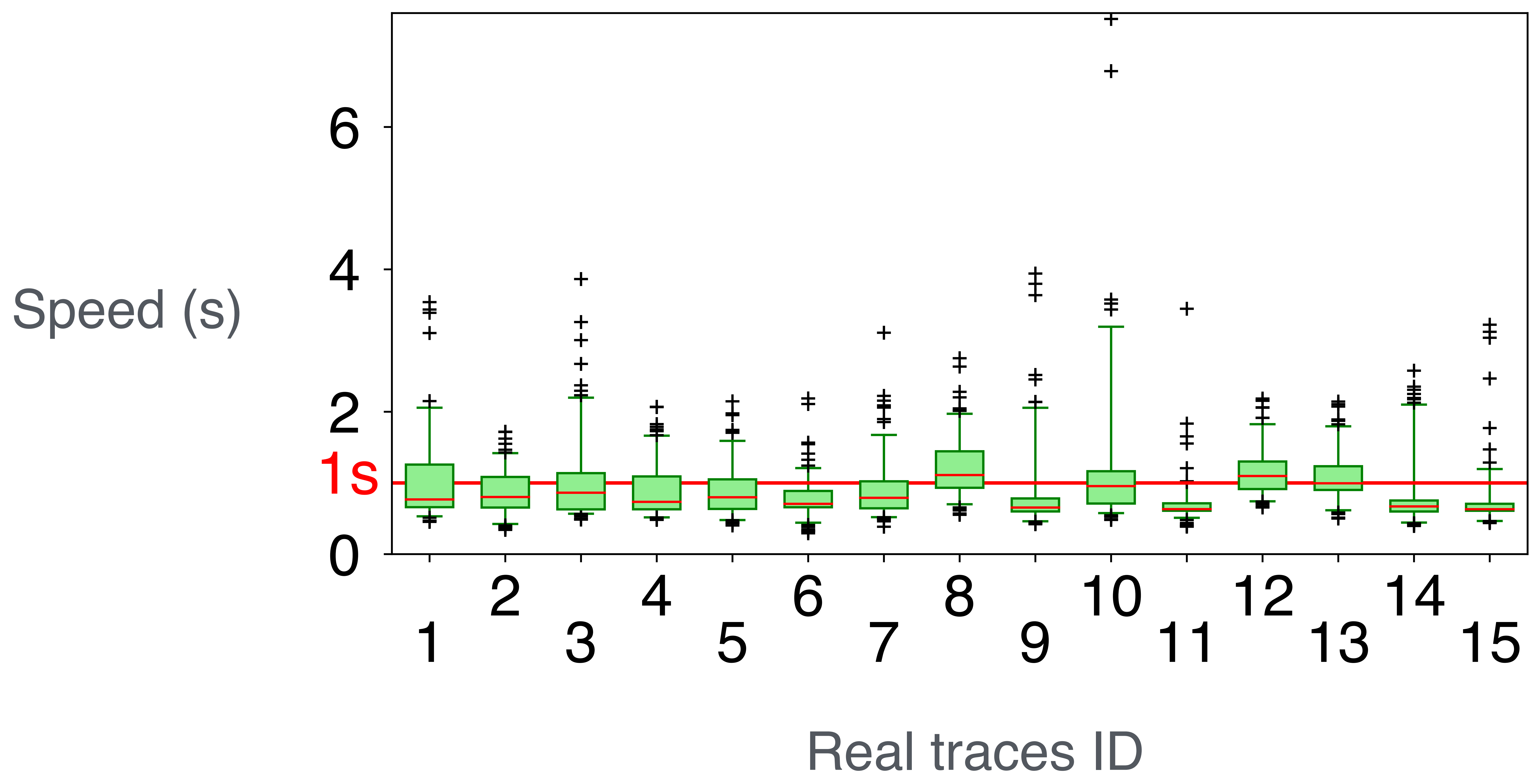
Blink avoids incorrectly inferring failures when packet loss is below 4%

packet loss %	1	2	3	4	5	…	8	9
False Positive Rate	**0**	**0**	**0**	0.67	0.67	…	1.3	2.7

Blink infers a failure within 1s for the majority of the cases



Blink infers a failure within 1s for the majority of the cases



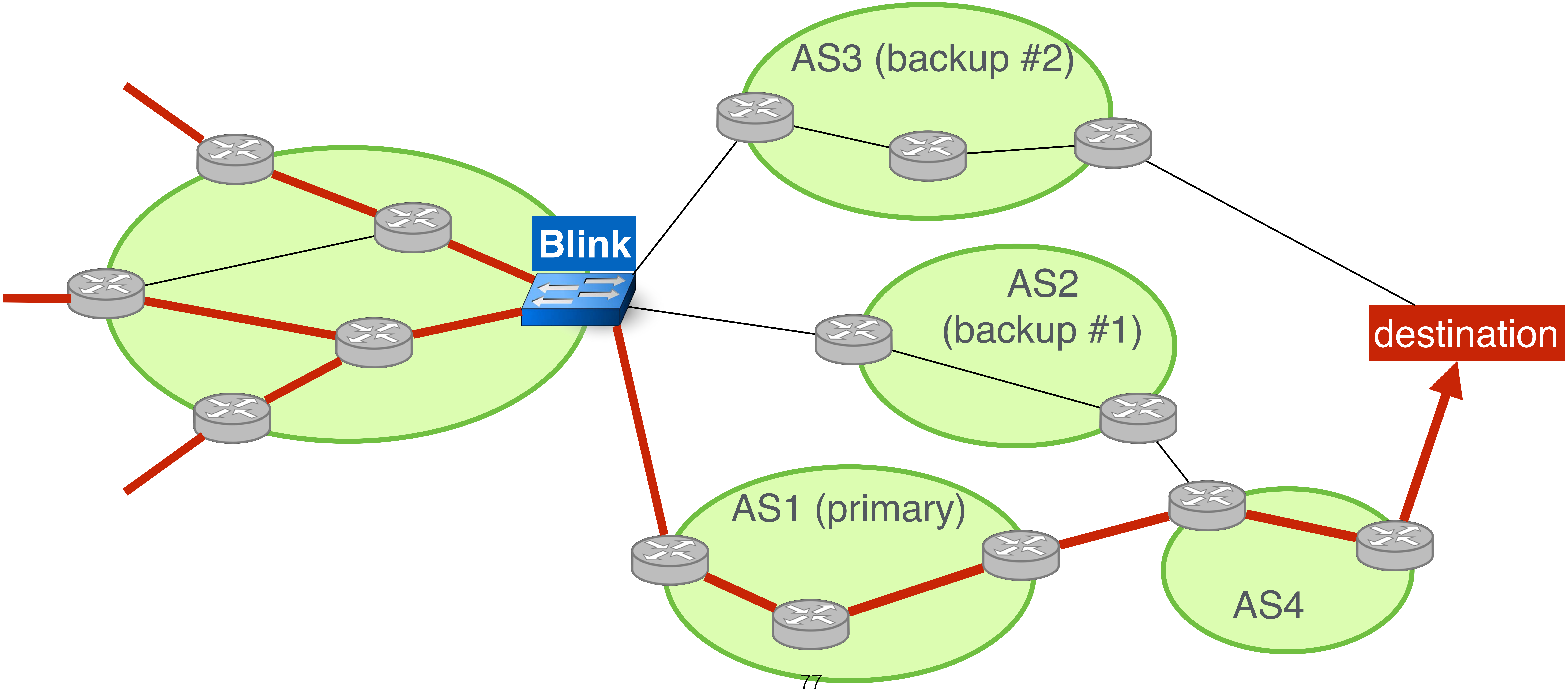
Outline

1. Why and how to use data-plane signals for fast rerouting
2. ***Blink*** infers more than 80% of the failures, often within 1s
3. ***Blink*** quickly reroutes traffic to working backup paths
4. ***Blink*** works in practice, on existing devices

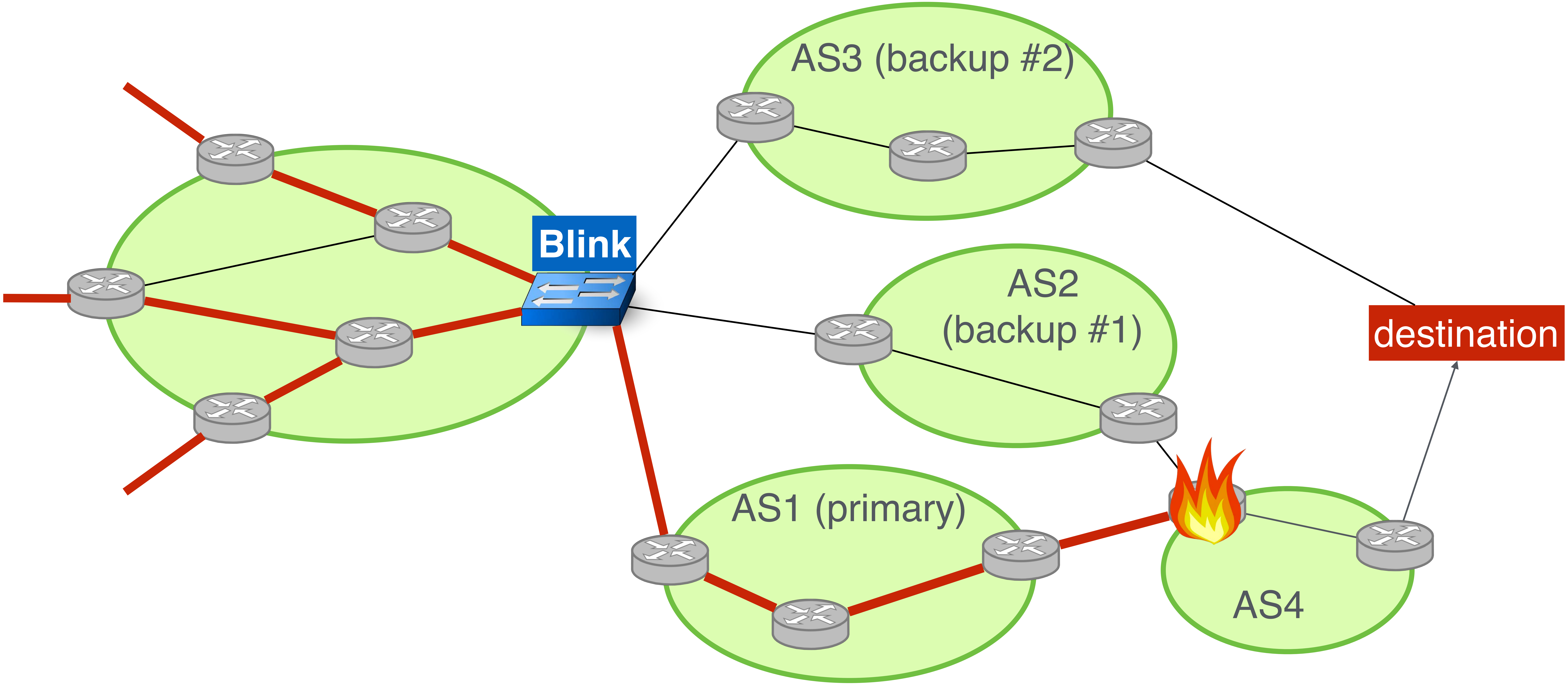
Upon detection of a failure, ***Blink*** immediately activates backup paths pre-populated by the control-plane

Problem: since the rerouting is done entirely in the data-plane, ***Blink*** cannot prevent forwarding issues

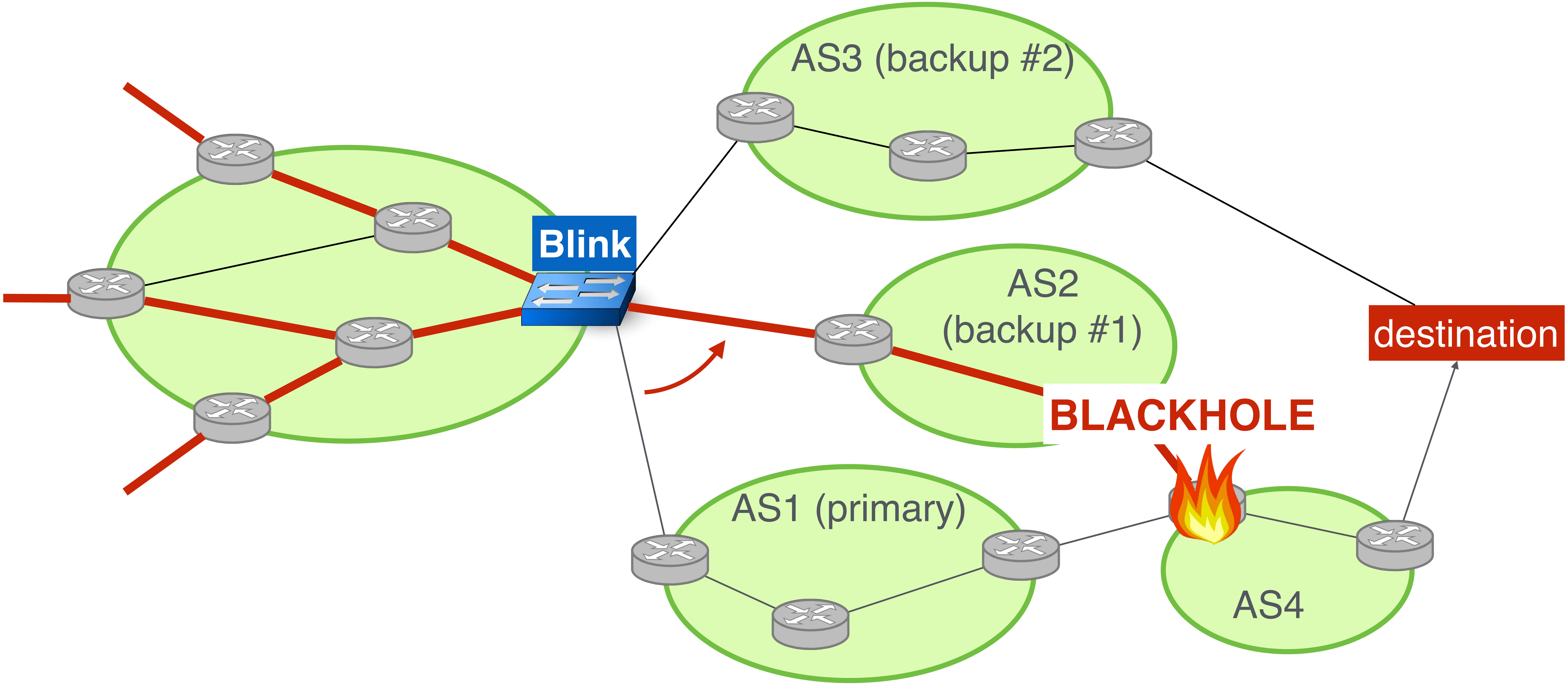
Problem: since the rerouting is done entirely in the data-plane, ***Blink*** cannot prevent forwarding issues



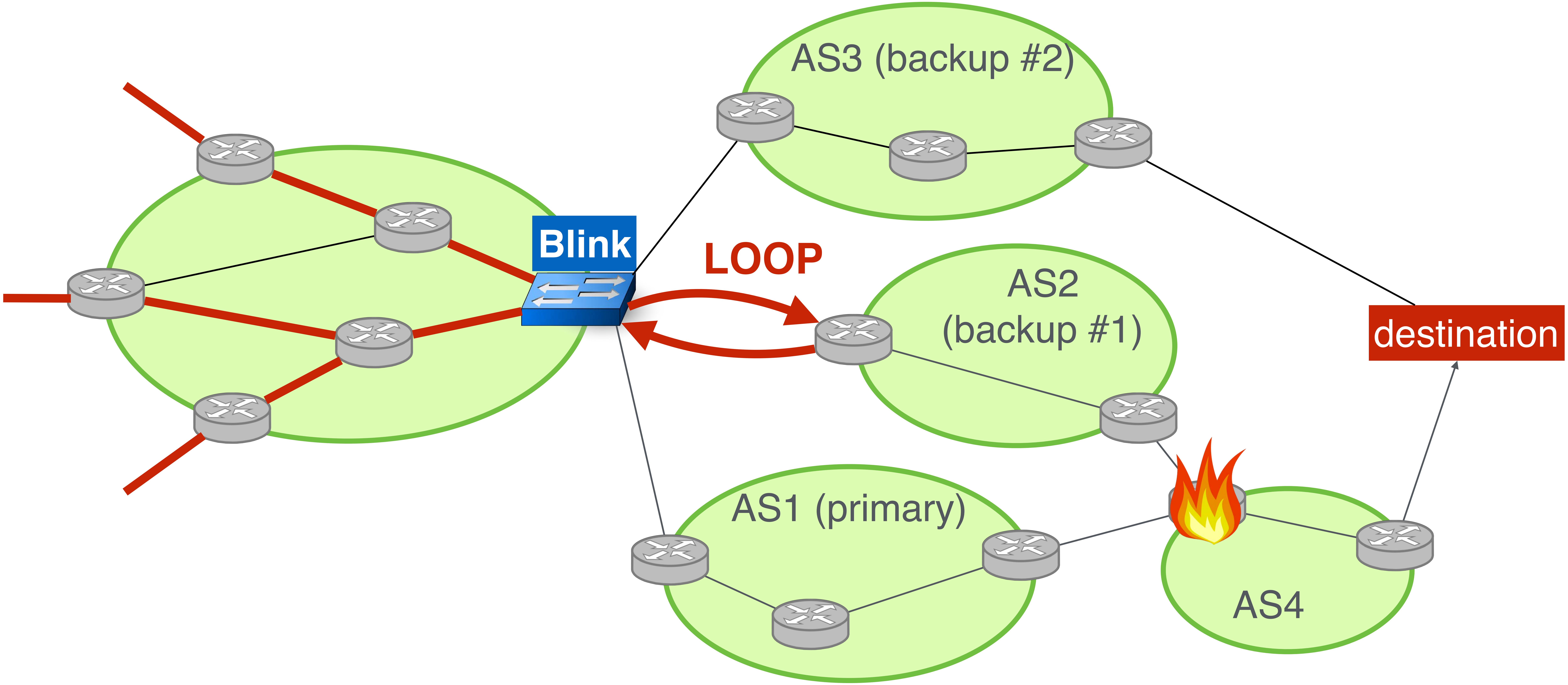
Problem: since the rerouting is done entirely in the data-plane, ***Blink*** cannot prevent forwarding issues



Problem: since the rerouting is done entirely in the data-plane,
Blink cannot prevent forwarding issues

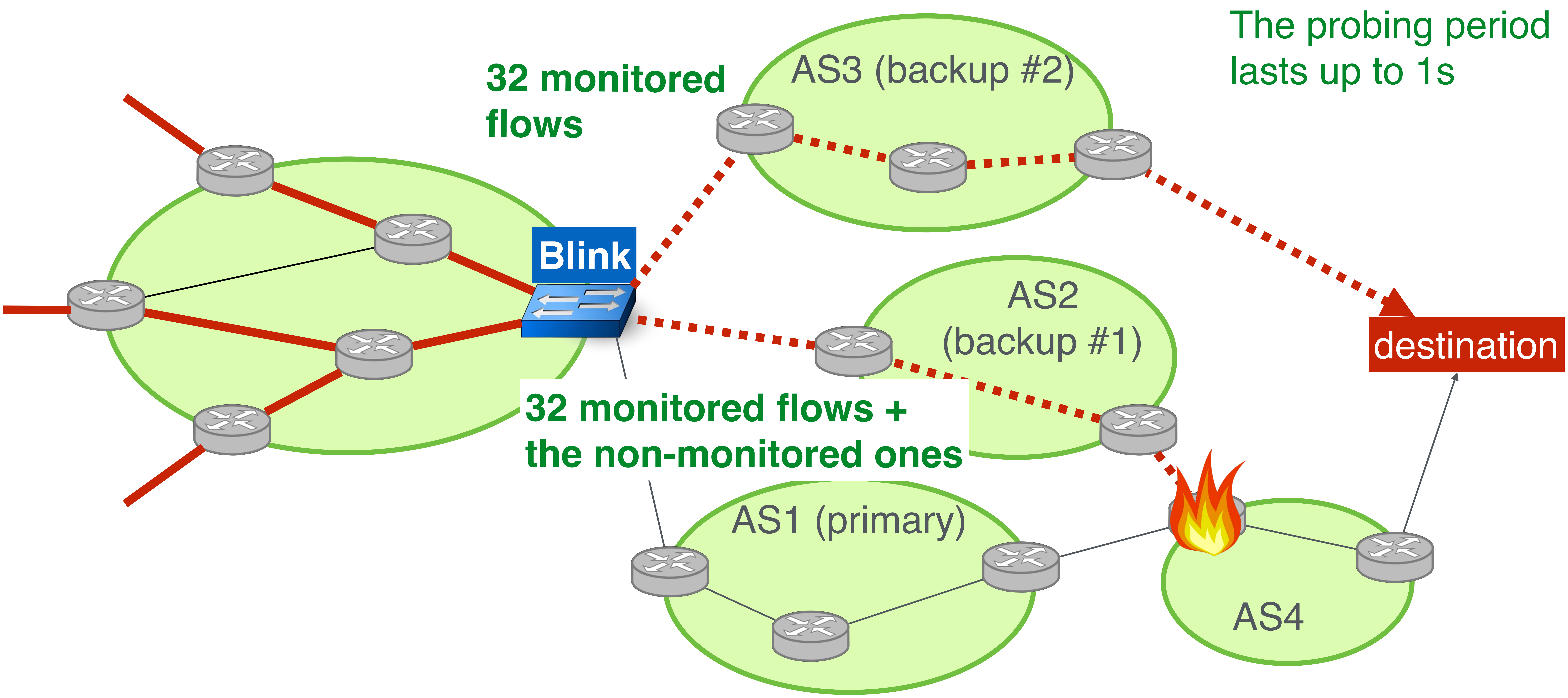


Problem: since the rerouting is done entirely in the data-plane, ***Blink*** cannot prevent forwarding issues

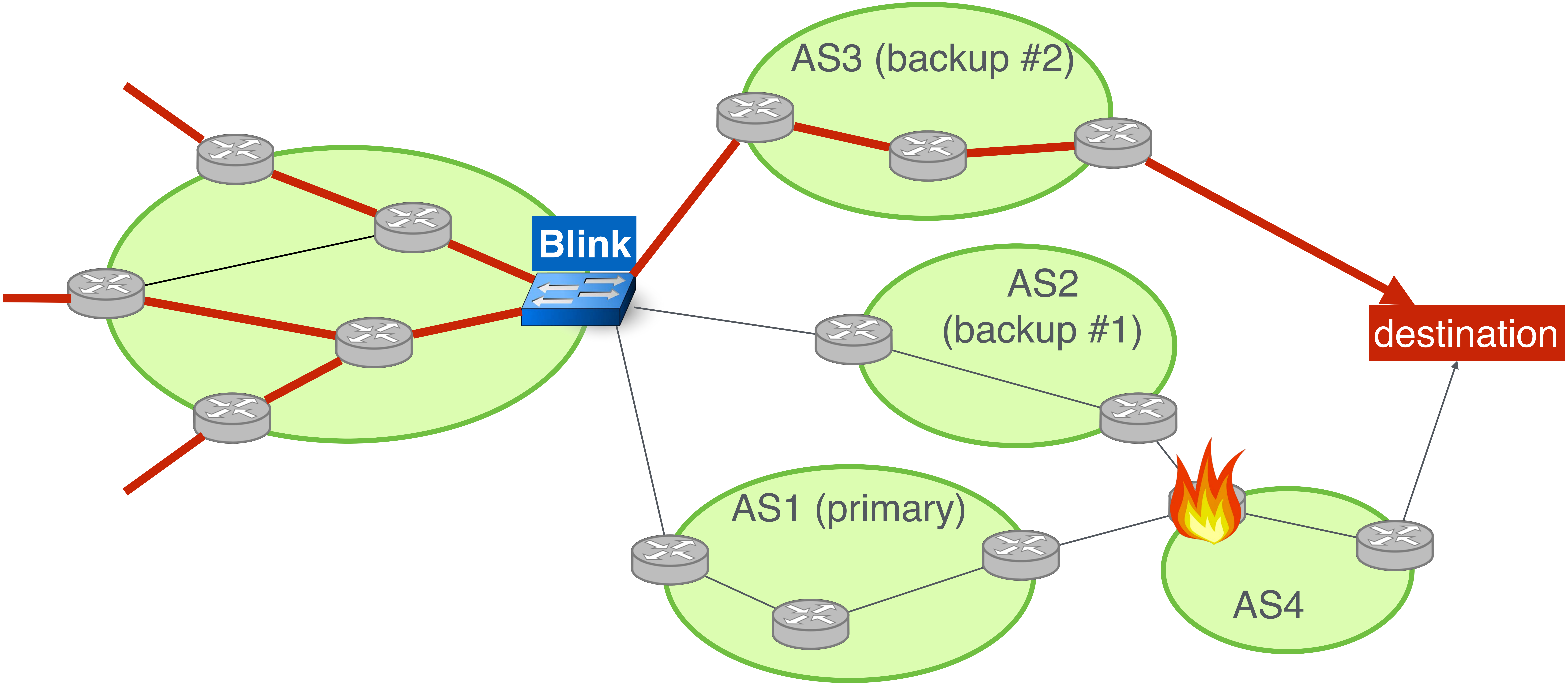


Solution: As for failures, ***Blink*** uses data-plane signals to pick a working backup path

Solution: As for failures, ***Blink*** uses data-plane signals to pick a working backup path



Solution: As for failures, ***Blink*** uses data-plane signals to pick a working backup path



As for failures, ***Blink*** compares the sequence number of consecutive packets to detect blackholes or loops*

*See the paper for an evaluation of the rerouting

Outline

1. Why and how to use data-plane signals for fast rerouting
2. ***Blink*** infers more than 80% of the failures, often within 1s
3. ***Blink*** quickly reroutes traffic to working backup paths
4. ***Blink*** works in practice, on existing devices

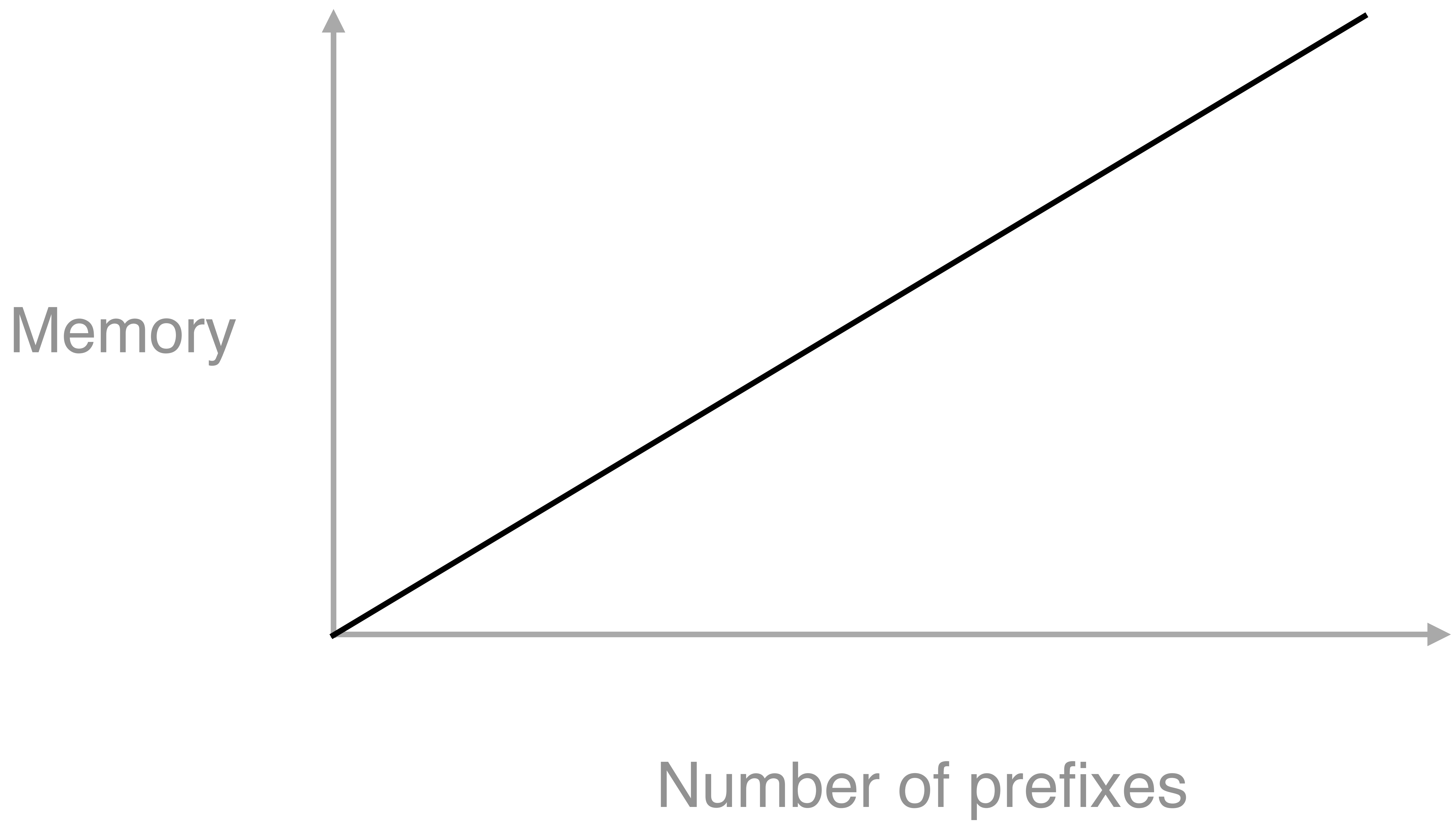
We ran ***Blink*** on the 15 real traces (15.8 hours)

We ran ***Blink*** on the 15 real traces (15.8 hours)

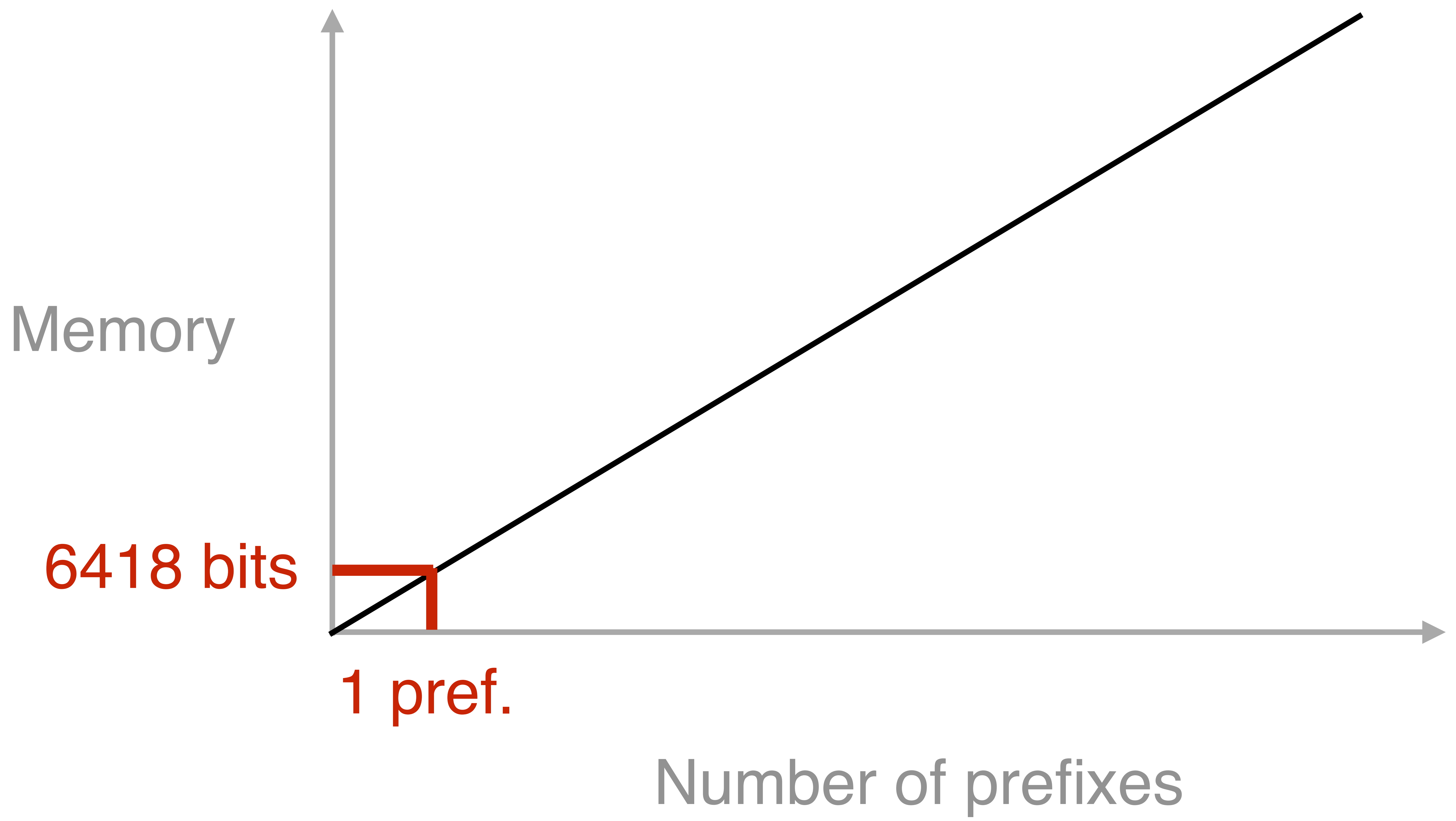
and it detected 6 outages, each affecting *at least* 42% of *all* the flows

On current programmable switches, ***Blink*** supports up to 10k prefixes

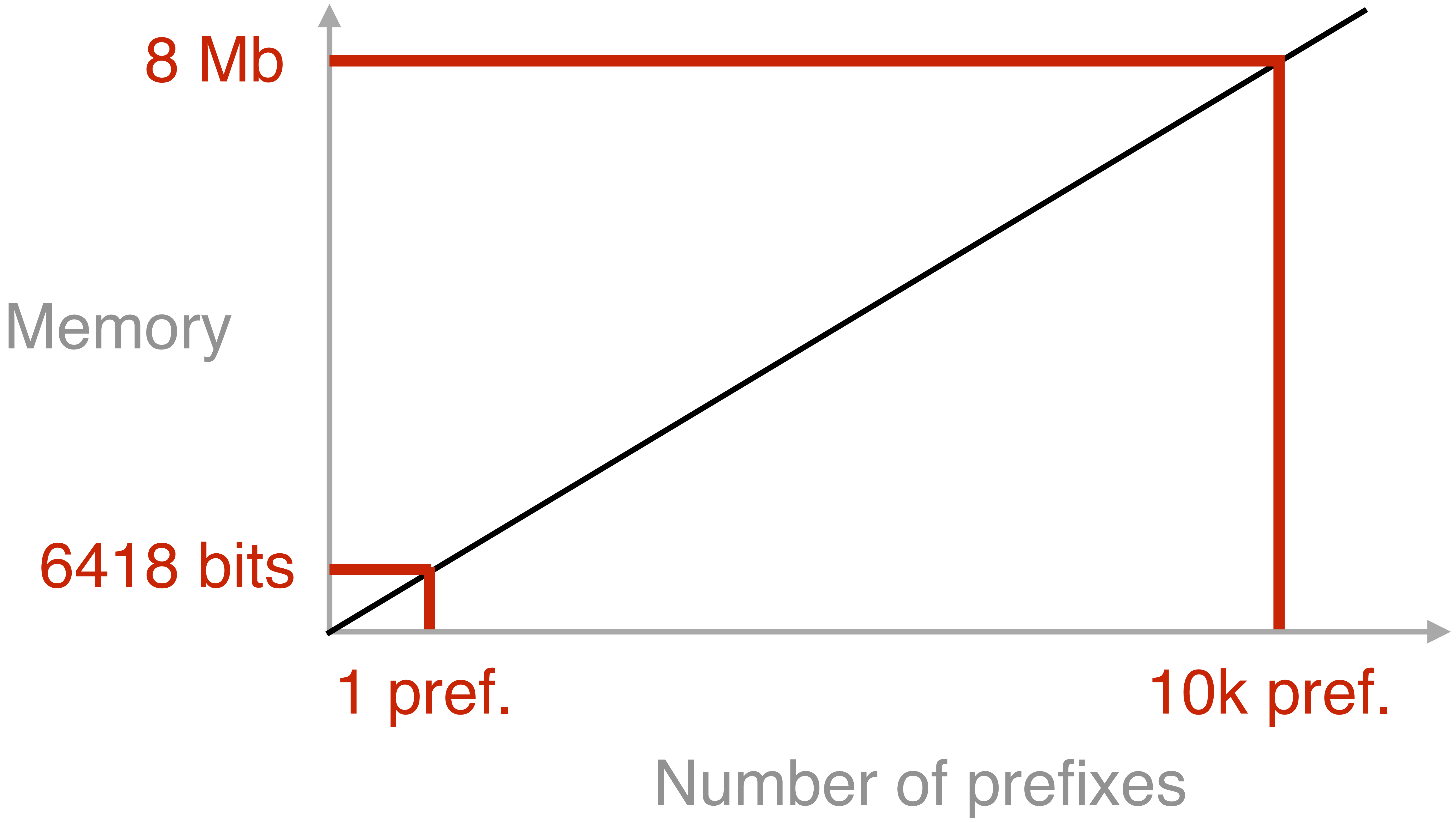
On current programmable switches, ***Blink*** supports up to 10k prefixes



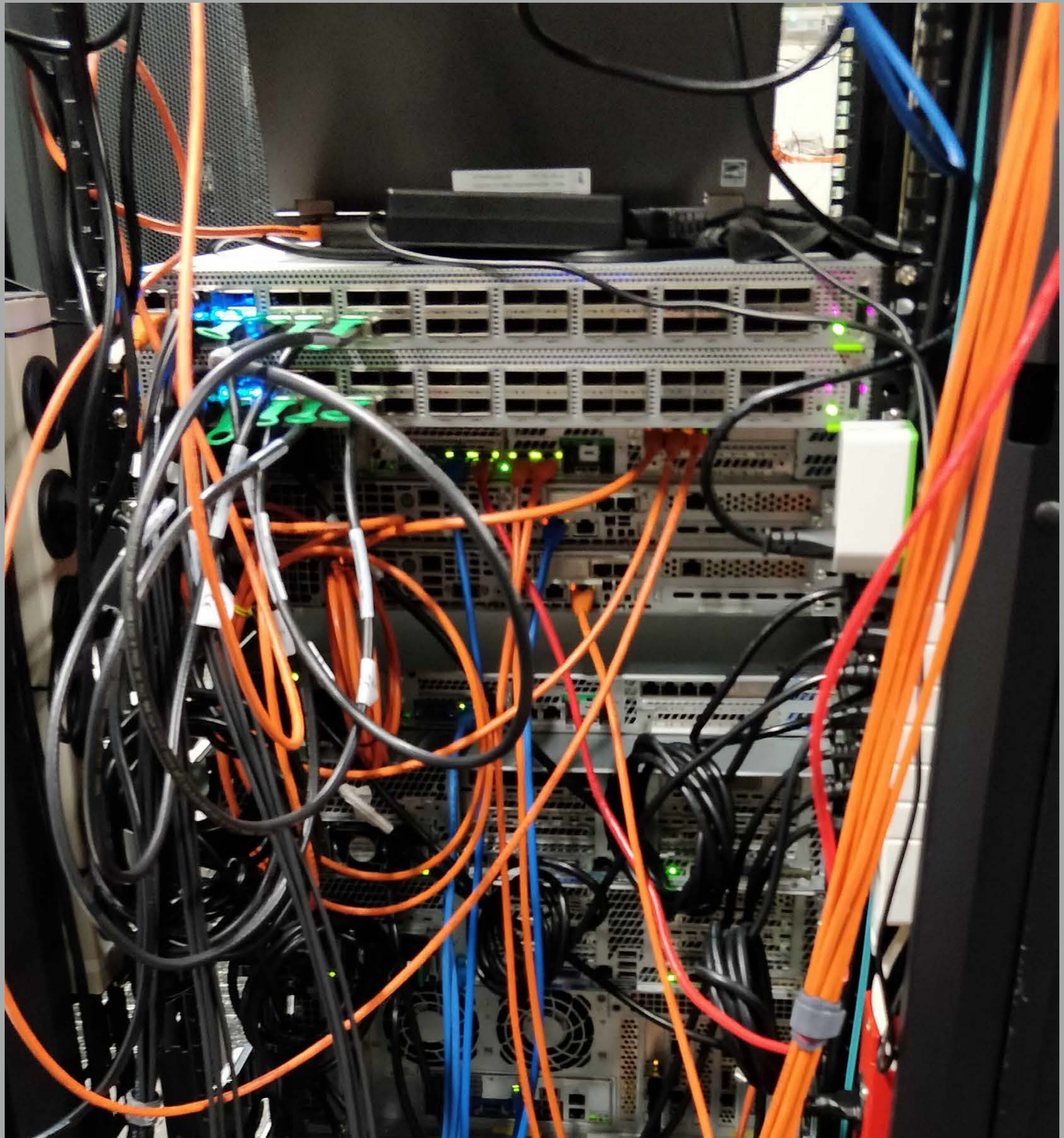
On current programmable switches, ***Blink*** supports up to 10k prefixes



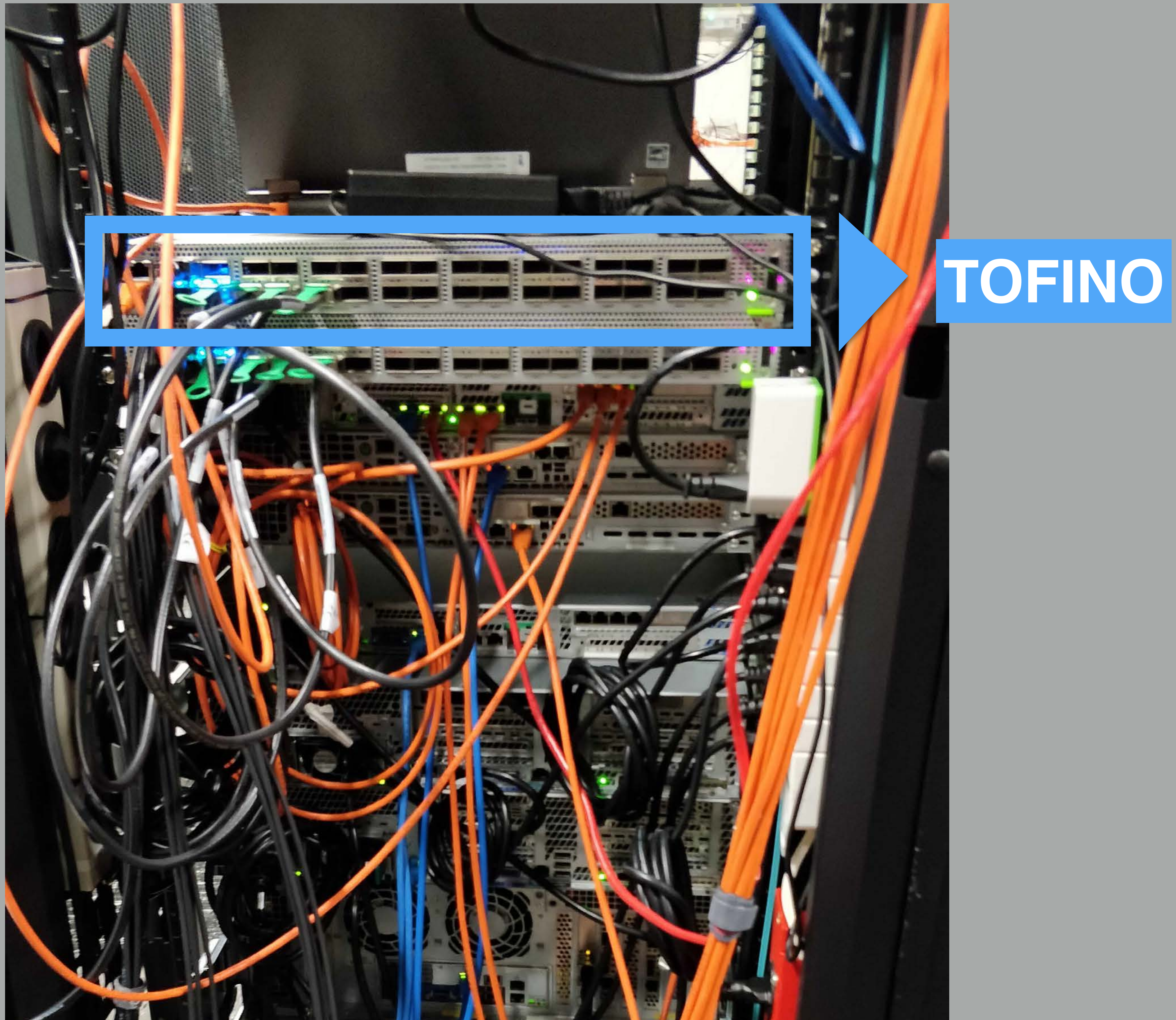
On current programmable switches, ***Blink*** supports up to 10k prefixes



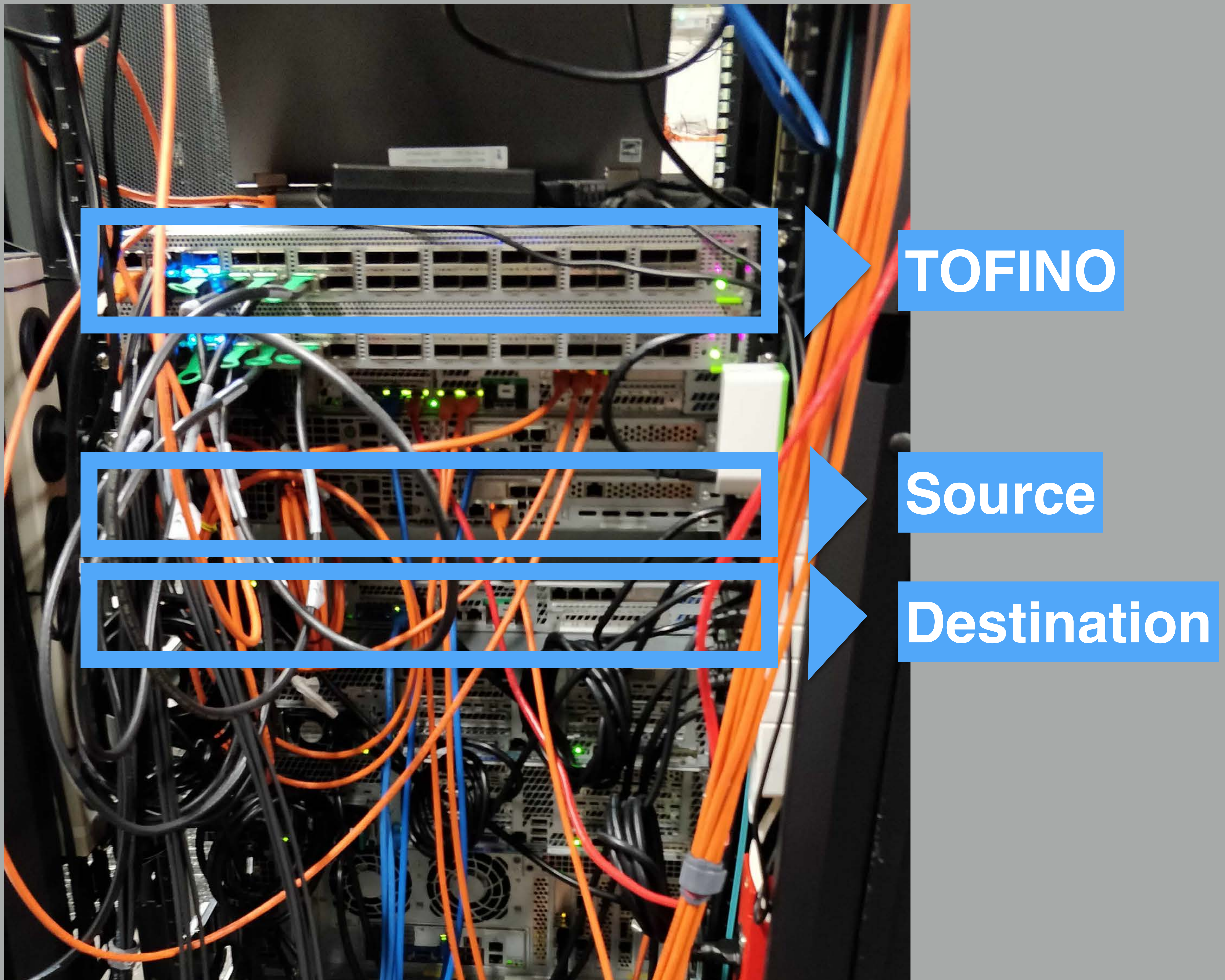
Blink works on a real Barefoot Tofino switch



Blink works on a real Barefoot Tofino switch



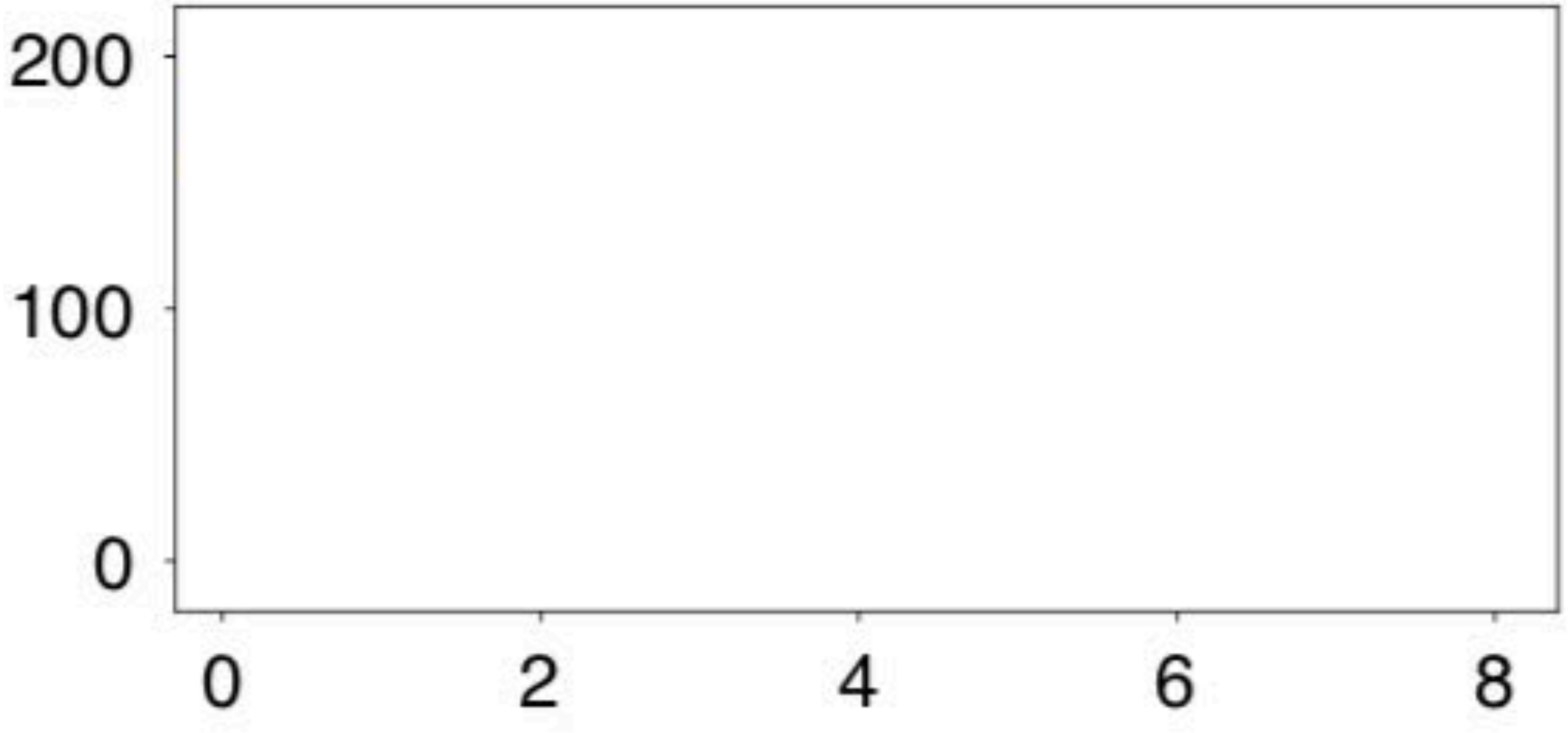
Blink works on a real Barefoot Tofino switch



Blink works on a real Barefoot Tofino switch

RTTs in [10ms; 300ms]

Number of packets every 100ms

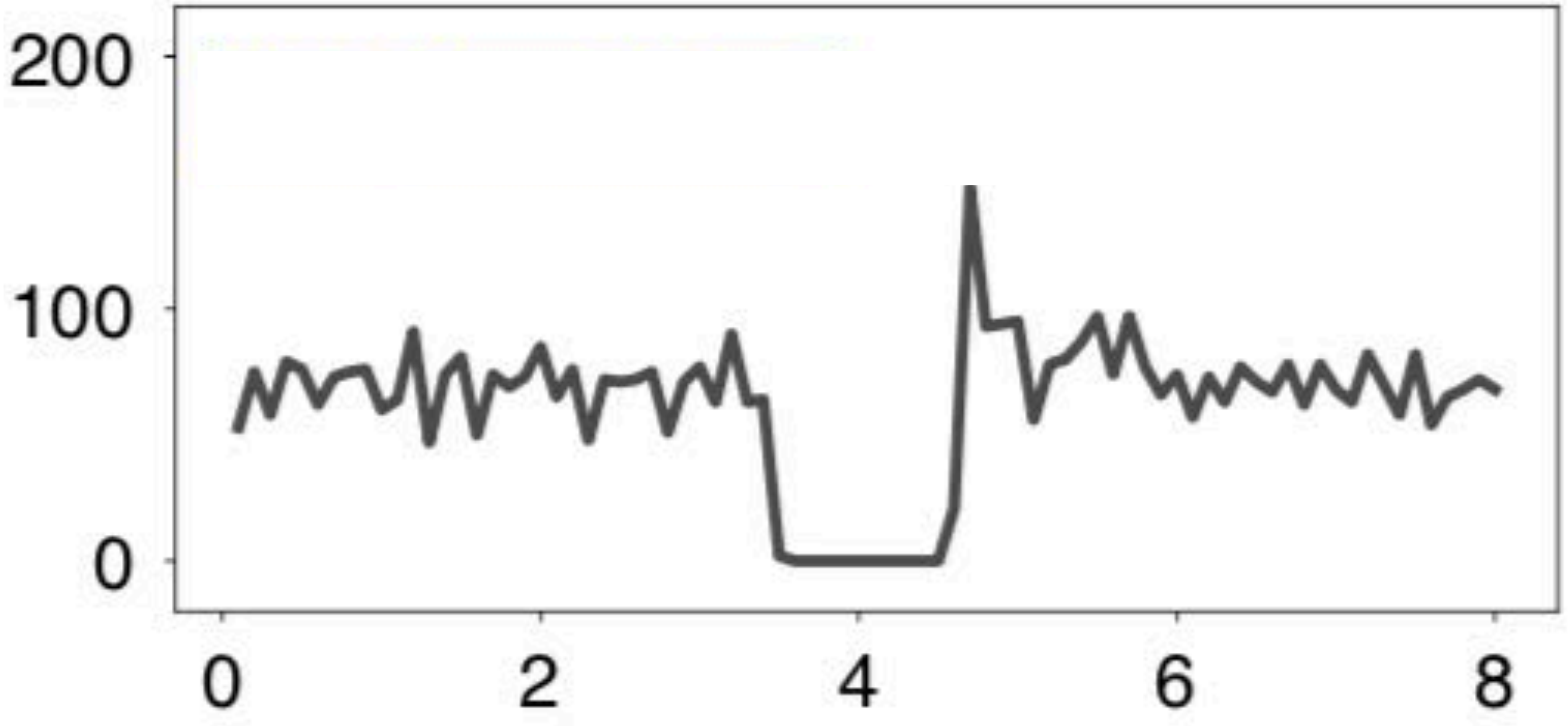


Time (s)

Blink works on a real Barefoot Tofino switch

RTTs in [10ms; 300ms]

Number of packets every 100ms

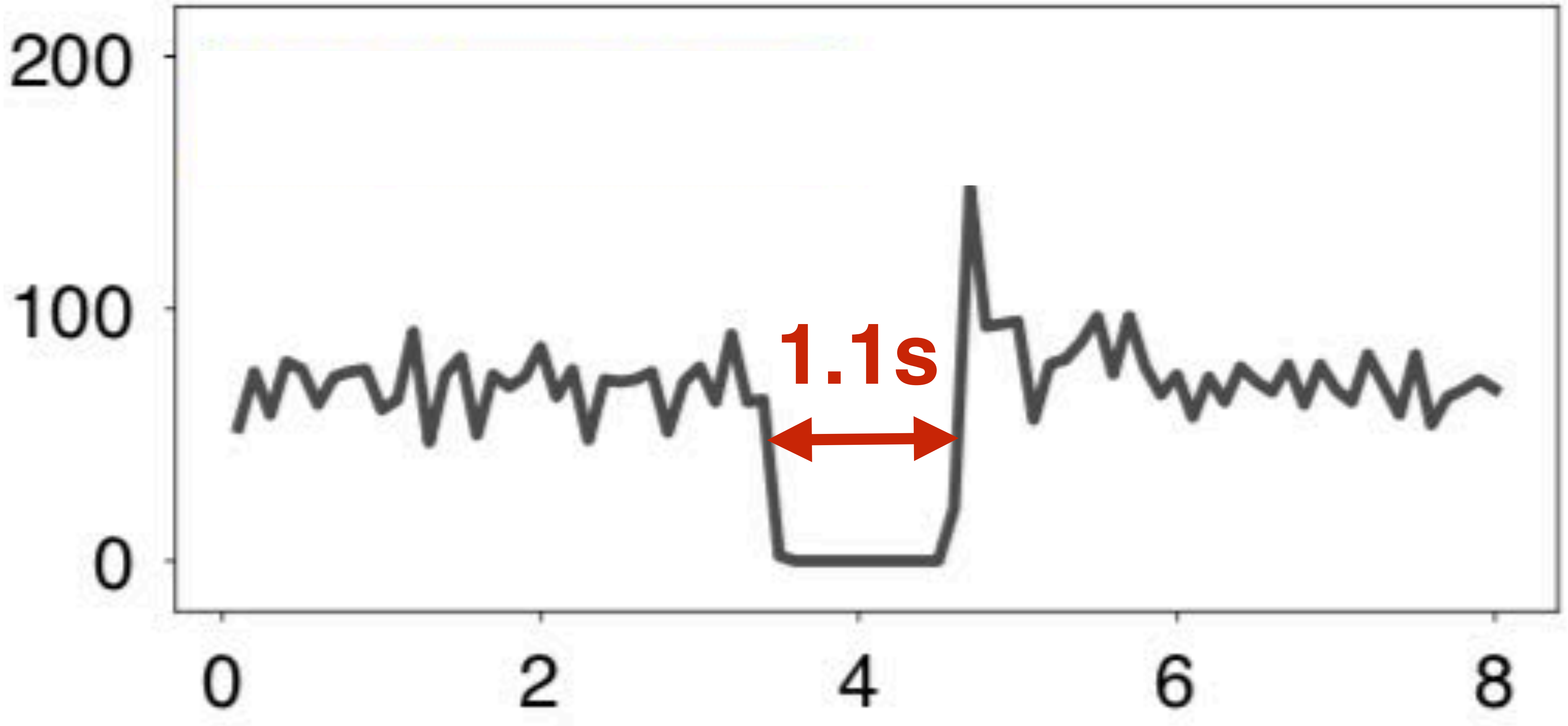


Time (s)

Blink works on a real Barefoot Tofino switch

RTTs in [10ms; 300ms]

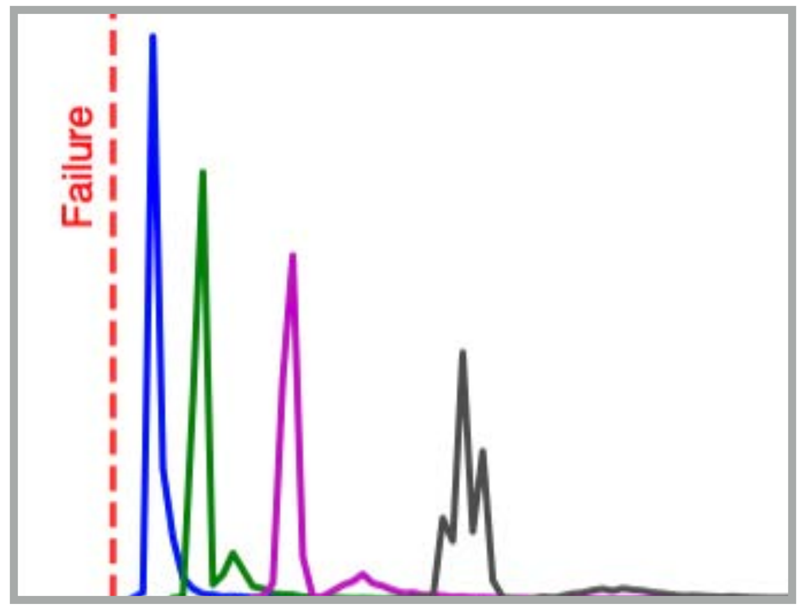
Number of packets every 100ms



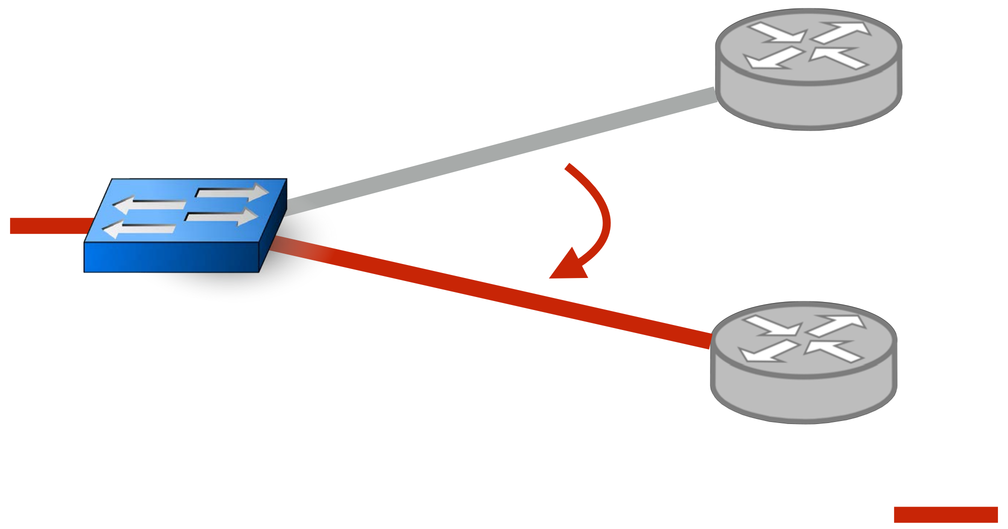
Time (s)

Blink: Fast Connectivity Recovery Entirely in the Data Plane

Infers failures from data-plane signals
with more than 80% accuracy, and often within 1s



Fast reroutes traffic at line rate
to working backup paths



Works on real traffic traces and on existing devices

https://blink.ethz.ch

Blink: Fast Connectivity Recovery Entirely in the Data Plane

Thomas Holterbach
ETH Zürich

NSDI
26th February 2019

Joint work with

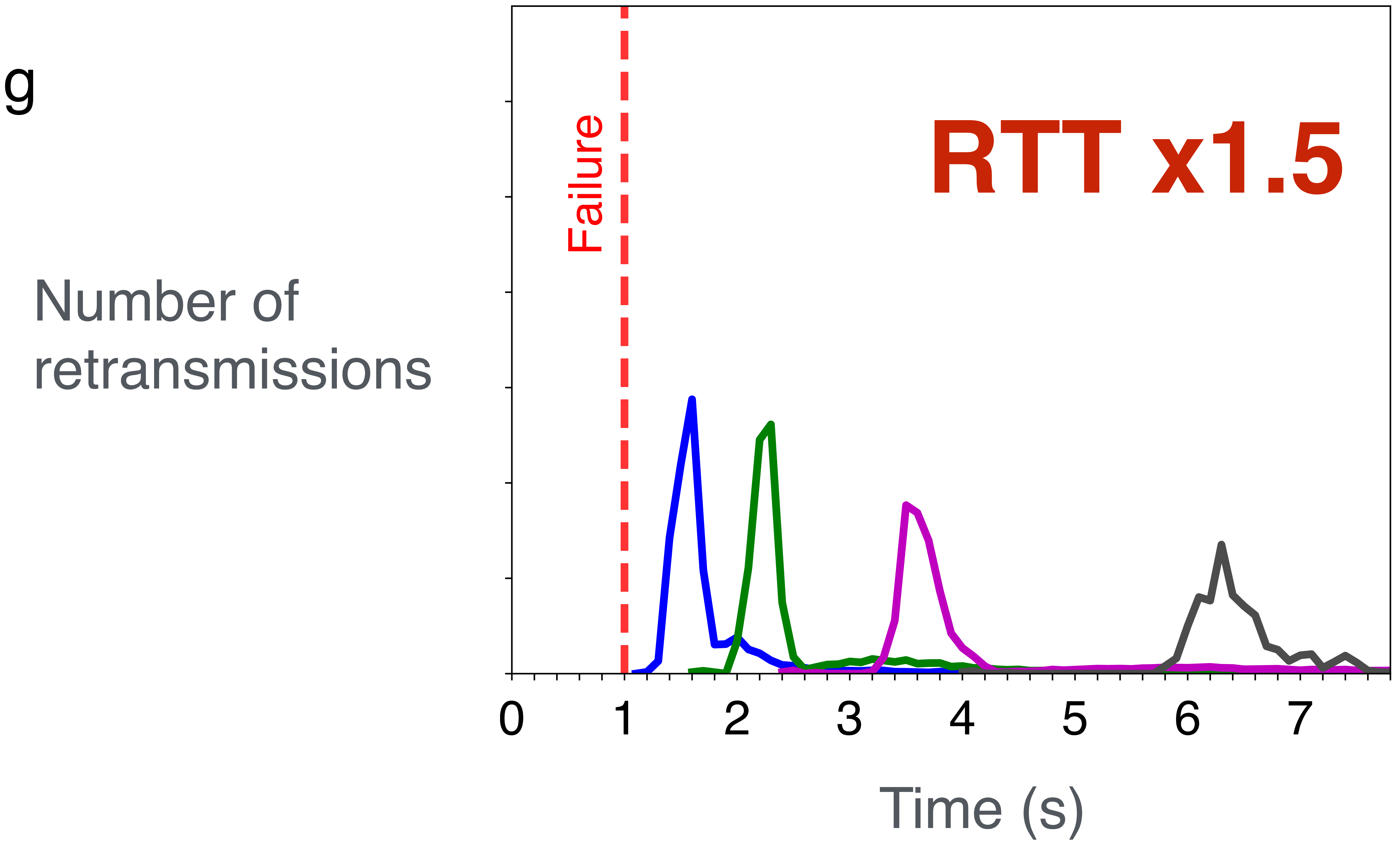
Edgar Costa Molero ETH Zürich
Maria Apostolaki ETH Zürich
Stefano Vissicchio University College London
Alberto Dainotti CAIDA, UC San Diego
Laurent Vanbever ETH Zürich

When multiple flows experience the same failure

the signal is a wave of retransmissions

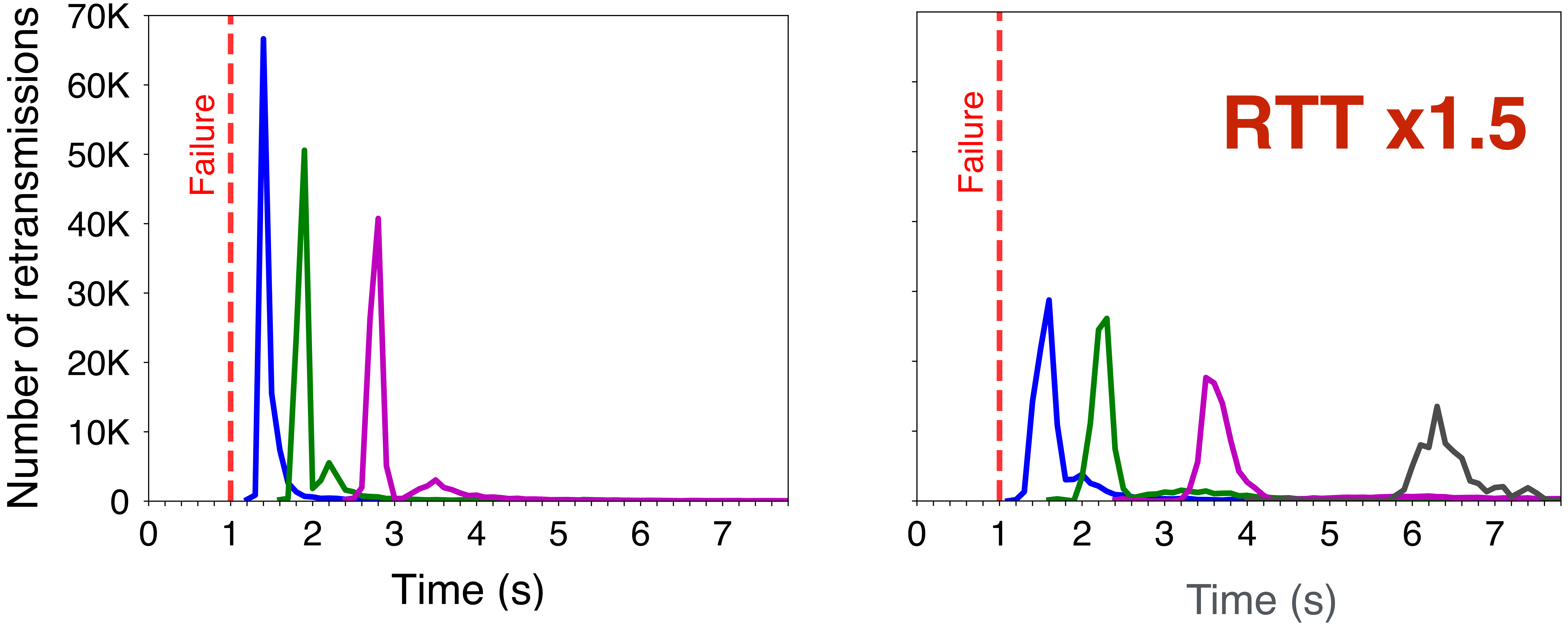
We simulated a failure affecting 100k flows with NS3

Same RTT distribution than in a real trace*

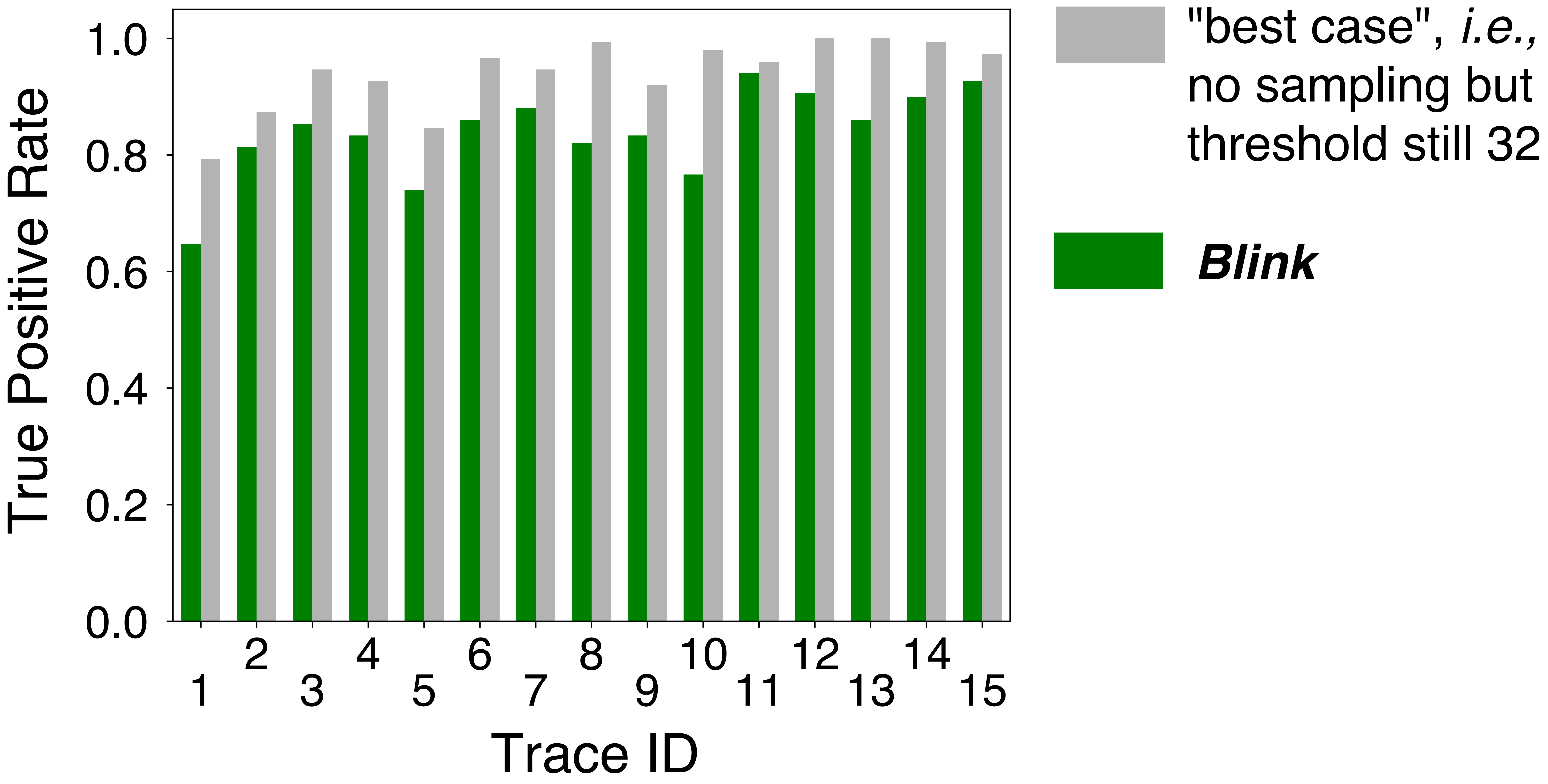


*CAIDA equinix-chicago direction A, 2015

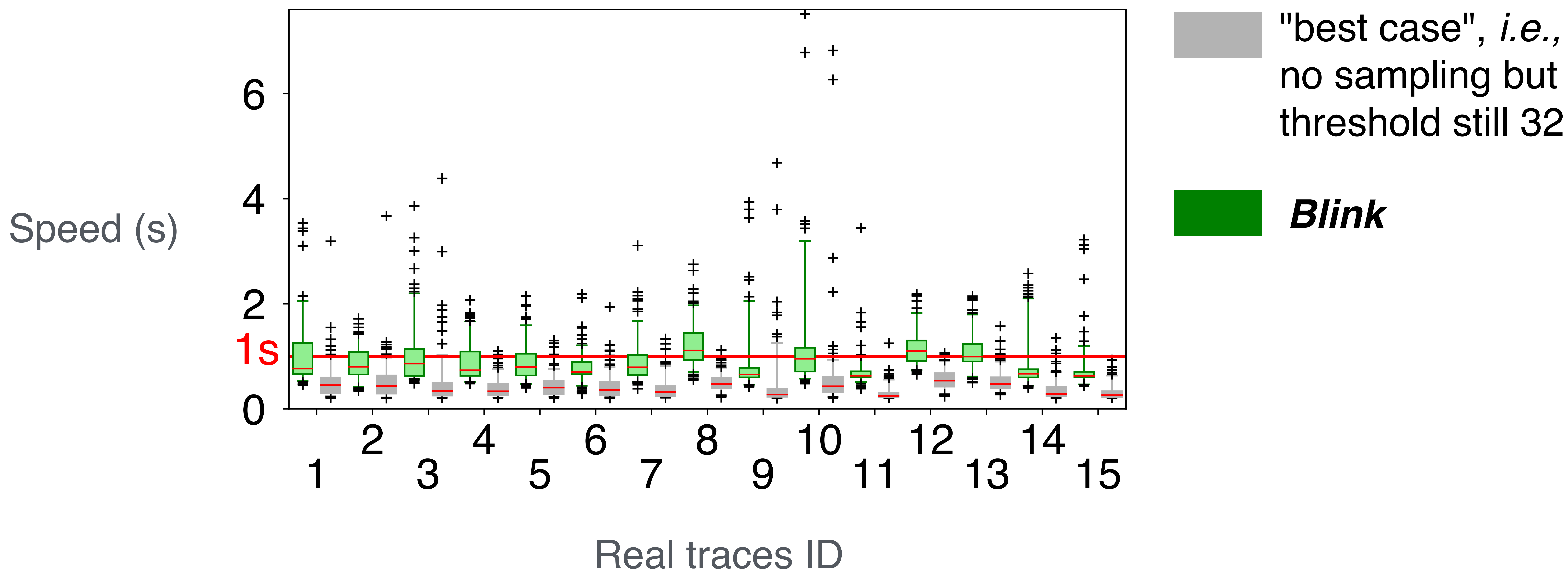
When multiple flows experience the same failure
the signal is a wave of retransmissions



Blink failure inference accuracy is close to a best case scenario, and is above 80% for 13 real traces out of 15



Blink infers a failure within 1s for the majority of the cases



Blink avoids incorrectly inferring failures when packet loss is below 4%

packet loss %	1	2	3	4	5	···	8	9
	False Positive Rate							
Blink	0	0	0	0.67	0.67	···	1.3	2.7
no sampling but threshold still 32	59	85	93	94	95	···	97	98

Blink quickly infers and avoids forwarding loops

Number of packets every 100ms