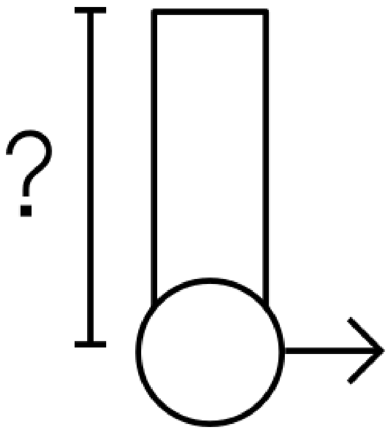
SIGCOMM 2024

Keeping an Eye on Congestion Control in the Wild with Nebby

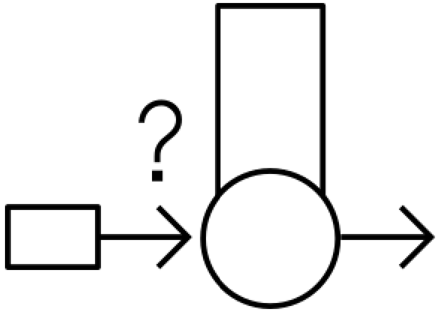
Ayush Mishra[1], Lakshay Rastogi[2], Raj Joshi[3], Ben Leong[1]

[1]National University of Singapore
[2]IIT Kanpur
[3]Harvard University

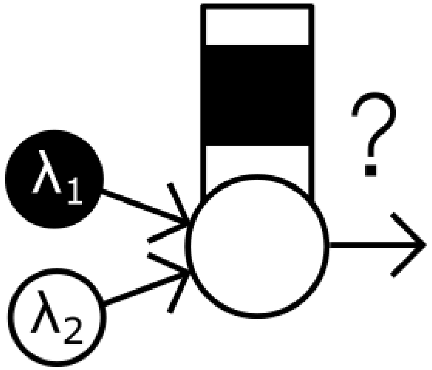
The make up of the Internet's Congestion Control Landscape influences how we think about



Buffer Sizing



AQMs and In-network policing

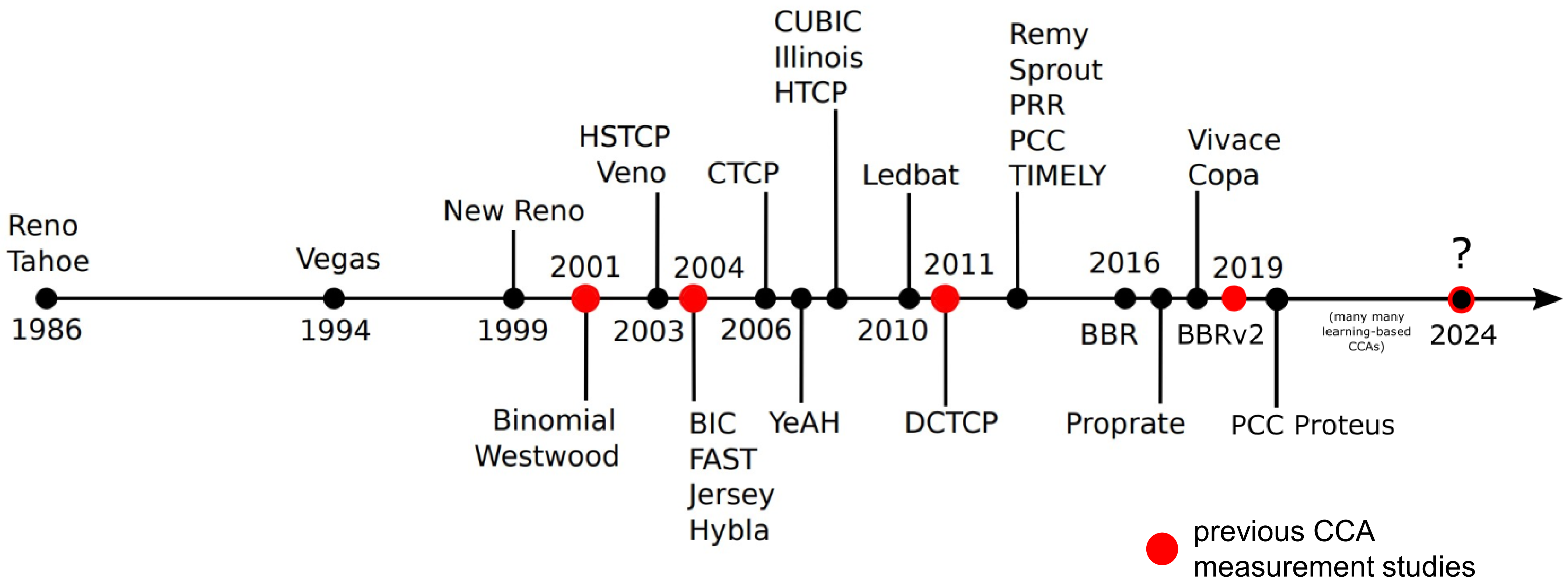


Fairness and Deployability

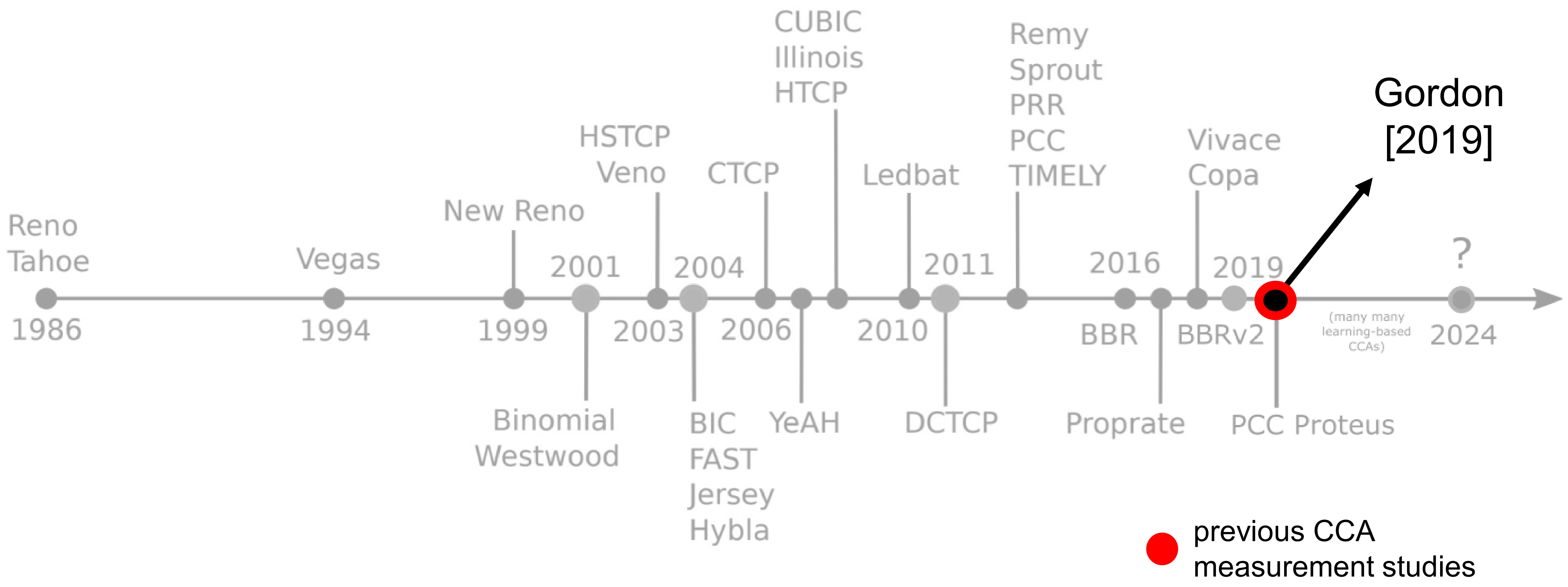
End-host Congestion Control is a unique design space where we expect users to meet their *selfish* goals without causing harm.

We also need to monitor the responsible deployment of CCAs on the Internet.

This is not a new problem.



This is not a new problem.

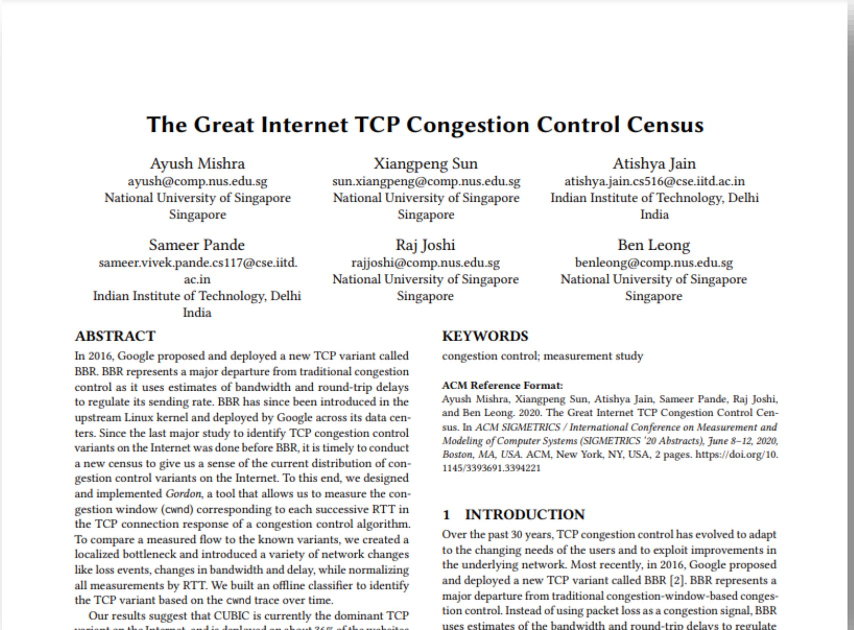


So what does the Internet’s current Congestion Control Landscape look like?

So we decided to re-run the most recent of the measurement tools…

Gordon

[2019]



The Great Internet TCP Congestion Control Census

Ayush Mishra
ayush@comp.nus.edu.sg
National University of Singapore
Singapore

Xiangpeng Sun
sun.xiangpeng@comp.nus.edu.sg
National University of Singapore
Singapore

Atishya Jain
atishya.jain.cs516@cse.iitd.ac.in
Indian Institute of Technology, Delhi
India

Sameer Pande
sameer.vivek.pande.cs117@cse.iitd.ac.in
Indian Institute of Technology, Delhi
India

Raj Joshi
rajjoshi@comp.nus.edu.sg
National University of Singapore
Singapore

Ben Leong
benleong@comp.nus.edu.sg
National University of Singapore
Singapore

ABSTRACT

In 2016, Google proposed and deployed a new TCP variant called BBR. BBR represents a major departure from traditional congestion control as it uses estimates of bandwidth and round-trip delays to regulate its sending rate. BBR has since been introduced in the upstream Linux kernel and deployed by Google across its data centers. Since the last major study to identify TCP congestion control variants on the Internet was done before BBR, it is timely to conduct a new census to give us a sense of the current distribution of congestion control variants on the Internet. To this end, we designed and implemented *Gordon*, a tool that allows us to measure the congestion window (cwnd) corresponding to each successive RTT in the TCP connection response of a congestion control algorithm. To compare a measured flow to the known variants, we created a localized bottleneck and introduced a variety of network changes like loss events, changes in bandwidth and delay, while normalizing all measurements by RTT. We built an offline classifier to identify the TCP variant based on the cwnd trace over time.

Our results suggest that CUBIC is currently the dominant TCP

KEYWORDS

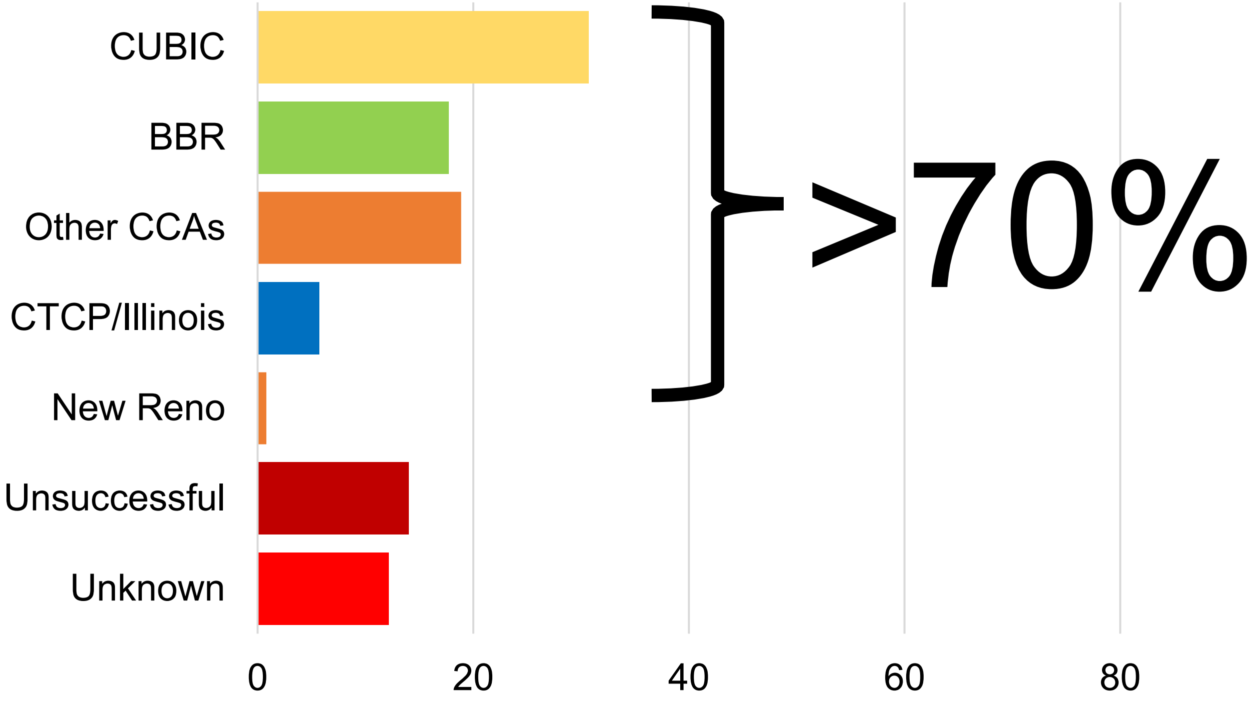
congestion control; measurement study

ACM Reference Format:

Ayush Mishra, Xiangpeng Sun, Atishya Jain, Sameer Pande, Raj Joshi, and Ben Leong. 2020. The Great Internet TCP Congestion Control Census. In *ACM SIGMETRICS / International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '20 Abstracts), June 8–12, 2020, Boston, MA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3393691.3394221

1 INTRODUCTION

Over the past 30 years, TCP congestion control has evolved to adapt to the changing needs of the users and to exploit improvements in the underlying network. Most recently, in 2016, Google proposed and deployed a new TCP variant called BBR [2]. BBR represents a major departure from traditional congestion-window-based congestion control. Instead of using packet loss as a congestion signal, BBR uses estimates of the bandwidth and round-trip delays to regulate

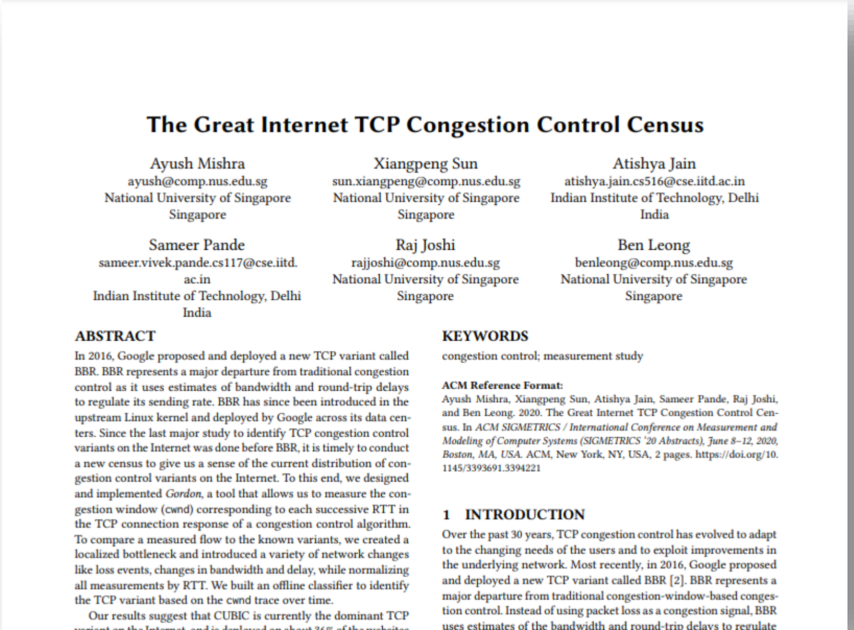


So we decided to re-run the most recent of the measurement tools…

But we were not successful.

Gordon

[2019]



The Great Internet TCP Congestion Control Census

Ayush Mishra
ayush@comp.nus.edu.sg
National University of Singapore
Singapore

Xiangpeng Sun
sun.xiangpeng@comp.nus.edu.sg
National University of Singapore
Singapore

Atishya Jain
atishya.jain.cs516@cse.iitd.ac.in
Indian Institute of Technology, Delhi
India

Sameer Pande
sameer.vivek.pande.cs117@cse.iitd.ac.in
Indian Institute of Technology, Delhi
India

Raj Joshi
rajjoshi@comp.nus.edu.sg
National University of Singapore
Singapore

Ben Leong
benleong@comp.nus.edu.sg
National University of Singapore
Singapore

ABSTRACT

In 2016, Google proposed and deployed a new TCP variant called BBR. BBR represents a major departure from traditional congestion control as it uses estimates of bandwidth and round-trip delays to regulate its sending rate. BBR has since been introduced in the upstream Linux kernel and deployed by Google across its data centers. Since the last major study to identify TCP congestion control variants on the Internet was done before BBR, it is timely to conduct a new census to give us a sense of the current distribution of congestion control variants on the Internet. To this end, we designed and implemented *Gordon*, a tool that allows us to measure the congestion window (cwnd) corresponding to each successive RTT in the TCP connection response of a congestion control algorithm. To compare a measured flow to the known variants, we created a localized bottleneck and introduced a variety of network changes like loss events, changes in bandwidth and delay, while normalizing all measurements by RTT. We built an offline classifier to identify the TCP variant based on the cwnd trace over time.

Our results suggest that CUBIC is currently the dominant TCP

KEYWORDS

congestion control; measurement study

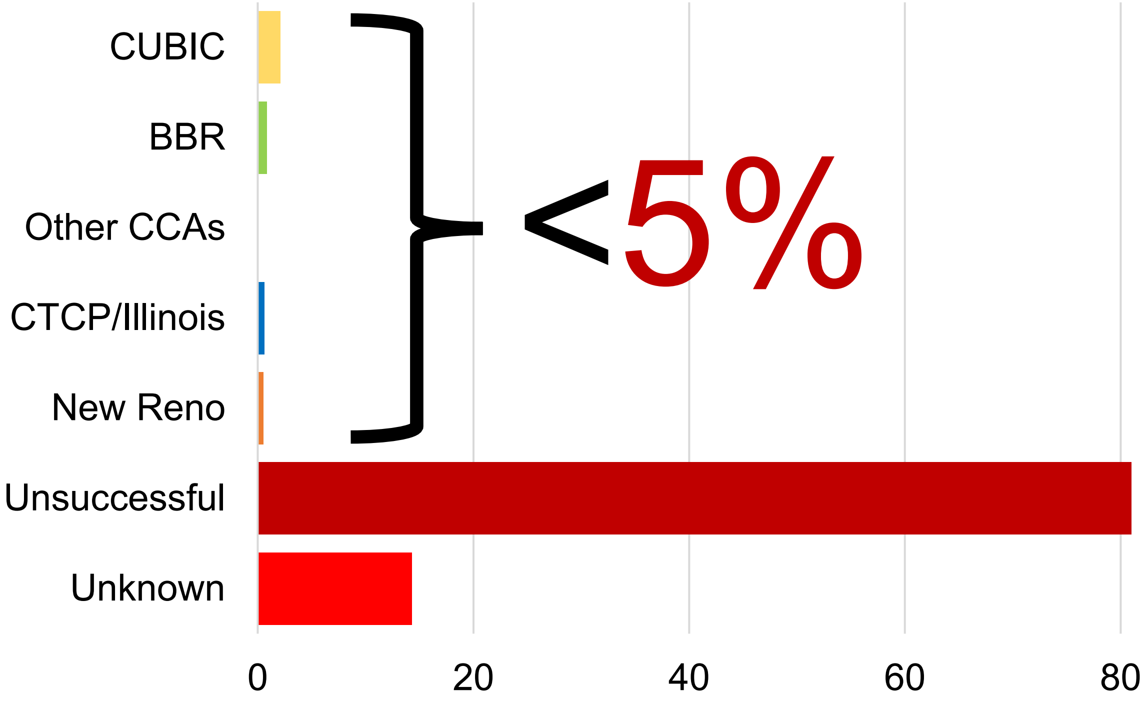
ACM Reference Format:

Ayush Mishra, Xiangpeng Sun, Atishya Jain, Sameer Pande, Raj Joshi, and Ben Leong. 2020. The Great Internet TCP Congestion Control Census. In *ACM SIGMETRICS / International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '20 Abstracts), June 8–12, 2020, Boston, MA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3393691.3394221

1 INTRODUCTION

Over the past 30 years, TCP congestion control has evolved to adapt to the changing needs of the users and to exploit improvements in the underlying network. Most recently, in 2016, Google proposed and deployed a new TCP variant called BBR [2]. BBR represents a major departure from traditional congestion-window-based congestion control. Instead of using packet loss as a congestion signal, BBR uses estimates of the bandwidth and round-trip delays to regulate

2023



This is has been a trend in all previous CCA classification tools.

Let's identify CCAS

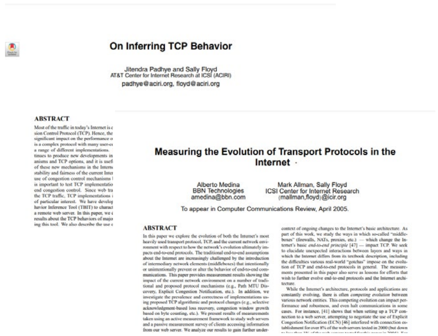
…but **TBIT** doesn't work

TBIT

[2001,2005]

CAAI

[2011]


On Inferring TCP Behavior

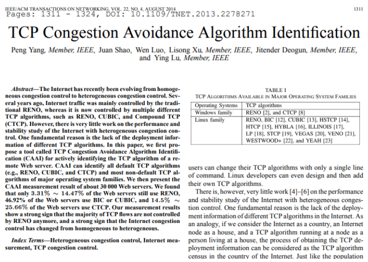
Jitendra Padhye and Sally Floyd
AT&T Center for Internet Research at ICSI (ACIRI)
padhye@aciri.org, floyd@aciri.org

Measuring the Evolution of Transport Protocols in the Internet

Alberto Medina
BBN Technologies
amedina@bbn.com

Mark Allman, Sally Floyd
ICSI Center for Internet Research
{mallman,floyd}@icir.org

To appear in Computer Communications Review, April 2005.


TCP Congestion Avoidance Algorithm Identification

Peng Yang, Member, IEEE, Juan Shao, Wen Luo, Lisong Xu, Member, IEEE, Jitender Deogun, Member, IEEE, and Ying Lu, Member, IEEE

This is has been a trend in all previous CCA classification tools.

Let's identify CCAS

…but **TBIT** doesn't work

…but **CAAI** doesn't work

TBIT
[2001,2005]

CAAI
[2011]

IG,Gordon
[2019]

On Inferring TCP Behavior

Measuring the Evolution of Transport Protocols in the Internet

TCP Congestion Avoidance Algorithm Identification

Inspector Gadget: A Framework for Inferring TCP Congestion Control Algorithms

The Great Internet TCP Congestion Control Census

This is has been a trend in all previous CCA classification tools.

Let's identify CCAS

TBIT

[2001,2005]

…but **TBIT** doesn't work

CAAI

[2011]

…but **CAAI** doesn't work

IG,Gordon

[2019]

but **IG** and **Gordon** don't work

Nebby

[2024]

On Inferring TCP Behavior

Jitendra Padhye and Sally Floyd

Measuring the Evolution of Transport Protocols in the Internet

TCP Congestion Avoidance Algorithm Identification

Inspector Gadget: A Framework for Inferring TCP Congestion Control Algorithms

The Great Internet TCP Congestion Control Census

Keeping an Eye on Congestion Control in the Wild with *Nebby*

Ayush Mishra[†], Lakshay Rastogi[‡], Raj Joshi[†], and Ben Leong[†]

[†]National University of Singapore [‡]Indian Institute of Technology, Kanpur

Crucial design goal: let's also be ✨ Future-proof ✨

Let's identify CCAS

TBIT

[2001,2005]

...but TBIT doesn't work

CAAI

[2011]

...but CAAI doesn't work

IG,Gordon

[2019]

but IG and Gordon don't work

Nebby

[2024]

On Inferring TCP Behavior

Measuring the Evolution of Transport Protocols in the Internet

TCP Congestion Avoidance Algorithm Identification

Inspector Gadget: A Framework for Inferring TCP Congestion Control Algorithms

The Great Internet TCP Congestion Control Census

Keeping an Eye on Congestion Control in the Wild with *Nebby*

Ayush Mishra[†], Lakshay Rastogi[‡], Raj Joshi[†], and Ben Leong[†]

[†]National University of Singapore [‡]Indian Institute of Technology, Kanpur

ABSTRACT

The Internet congestion control landscape is rapidly evolving. Since the introduction of BBR and the deployment of QUIC, it has become increasingly commonplace for companies to modify and implement their own congestion control algorithms (CCAs). To respond effectively to these developments, it is crucial to understand the state of CCA deployments in the wild. Unfortunately, existing CCA identification tools are not future-proof and do not work well with modern CCAs and encrypted protocols like QUIC. In this paper, we articulate the challenges in designing a future-proof CCA identification tool and propose a measurement methodology that directly addresses these challenges. The resulting measurement tool, called *Nebby*, can identify all the CCAs currently available in the Linux kernel and BBRv2 with an average accuracy of 96.7%. We found that among the Alexa Top 20k websites, the share of BBR has shrunk since 2019 and that only 8% of them responded to QUIC requests. Among these QUIC servers, CUBIC and BBR seem equally popular. We show that *Nebby* is extensible by extending it for Copa and an undocumented family of CCAs that is deployed by 6% of the measured websites, including major corporations like Hulu and Apple.

Why is CCA identification hard?

People keep deploying *new* CCAs – we can't be ad hoc

Work well with a wider range of applications and application traffic – be client-agnostic

We can't appear hostile – we need to be as passive as possible

Why is CCA identification hard?

People keep deploying *new* CCAs – we **can't be ad hoc**

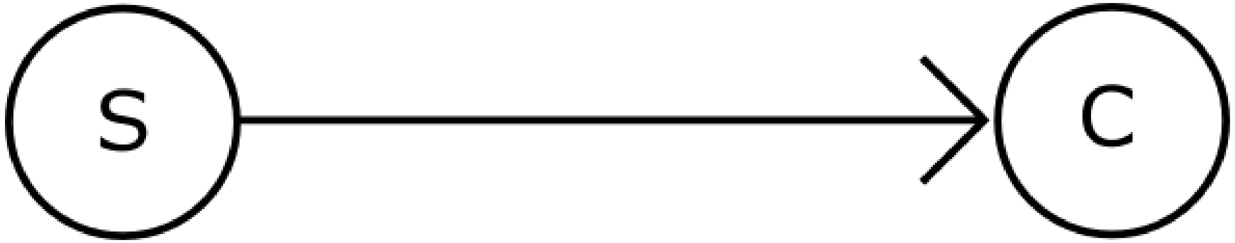
Work well with a wider range of applications and application traffic – be **client-agnostic**

We can't appear hostile – we need to be as **passive** as possible

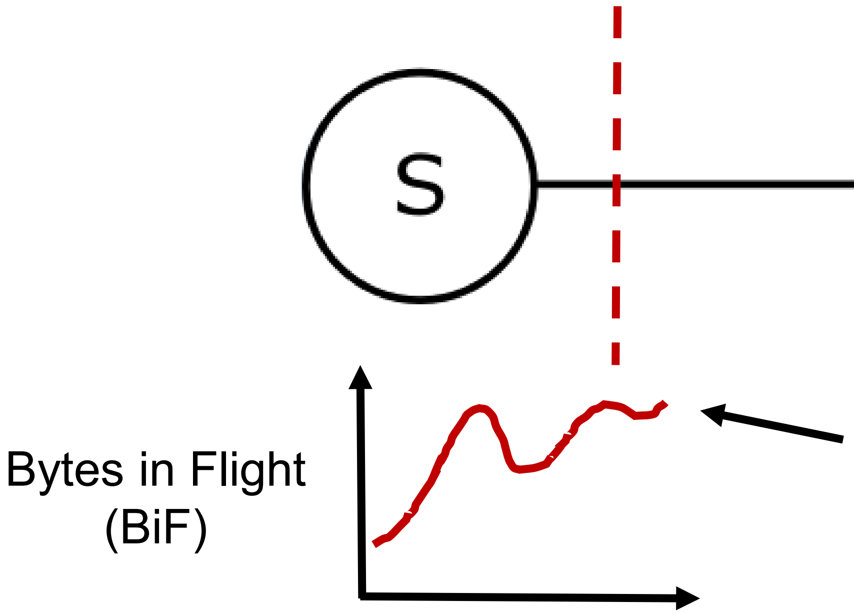
There *is* an obvious way to identify CCAs while meeting all these criteria, albeit in the controlled setting.

Let's review the task we have at hand:

Identify the Congestion Control Algorithm run by a server

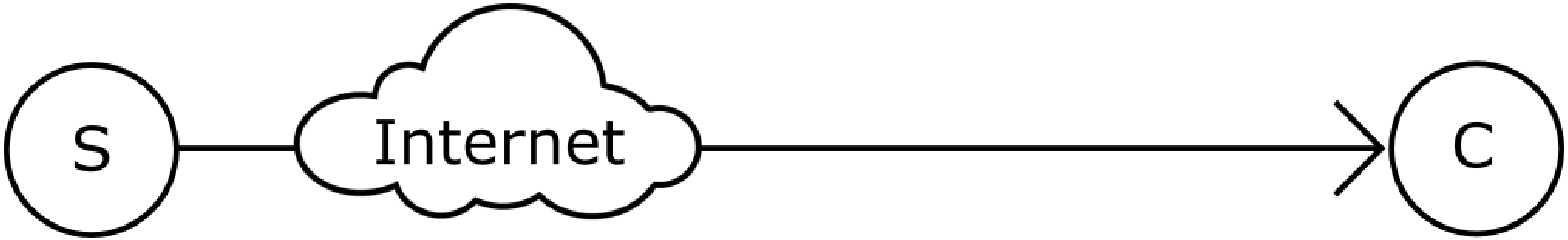


Identify the Congestion Control Algorithm run by a server



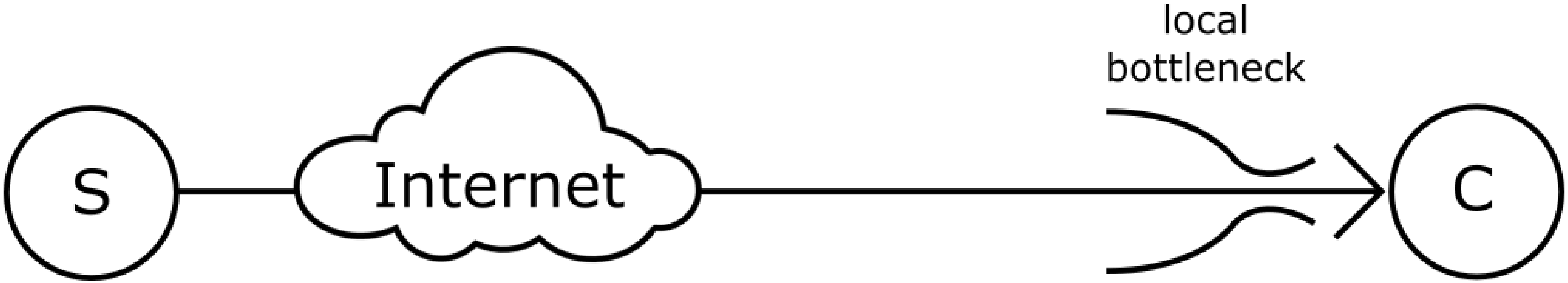
Unique for a CCA in response to a set of some specific network conditions

Identify the Congestion Control Algorithm run by a remote server on the Internet

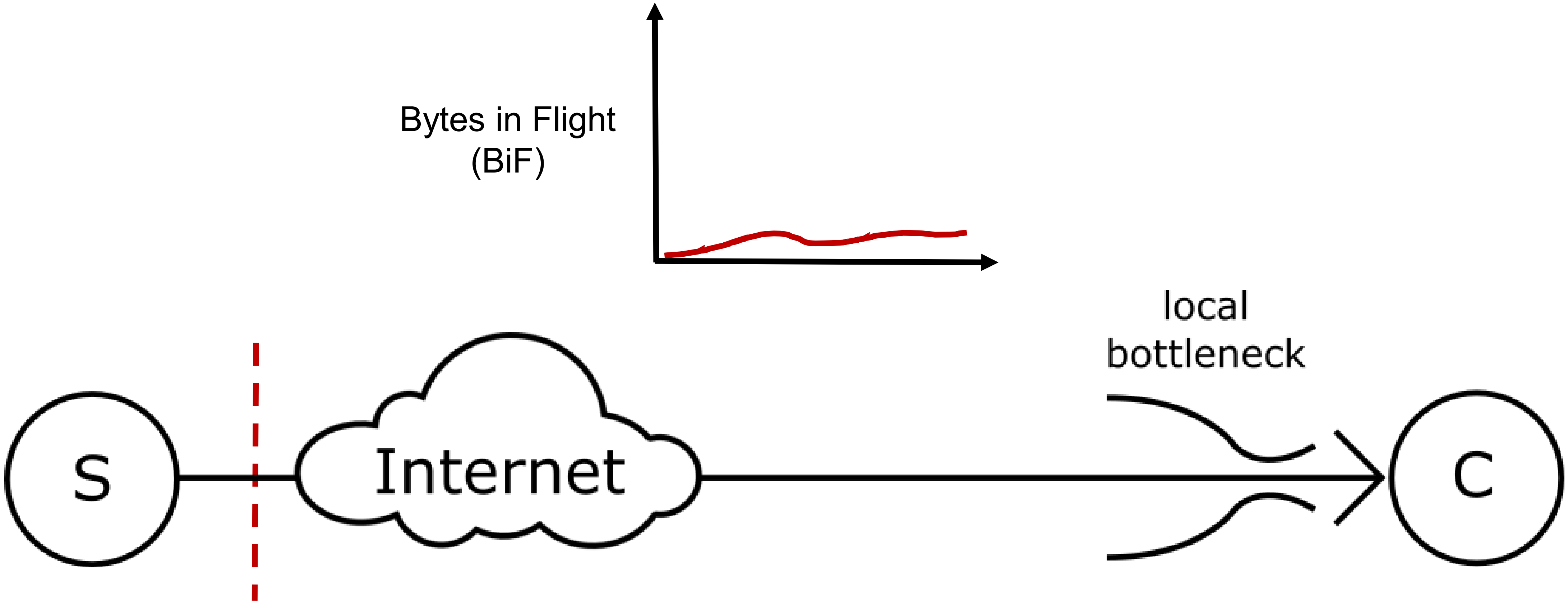


We no more have control over the network conditions

Identify the Congestion Control Algorithm run by a remote server on the Internet

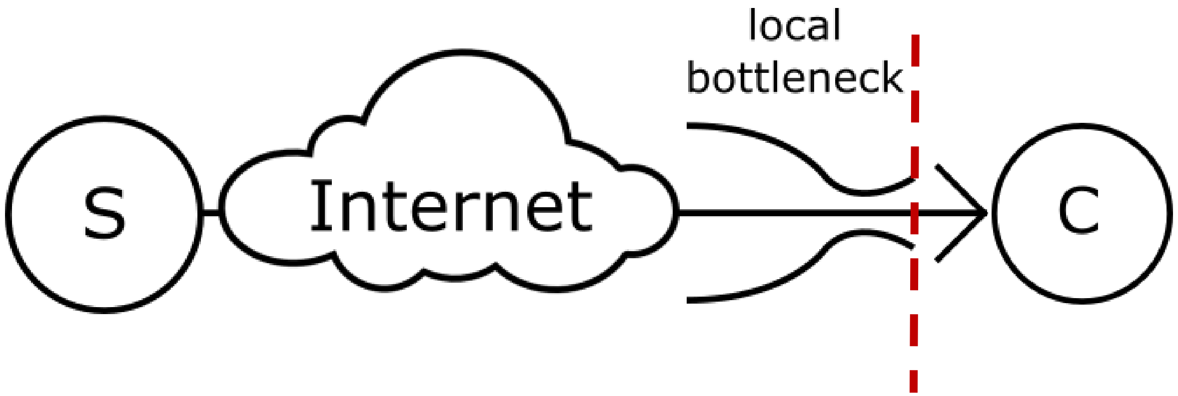


We can solve this how we have solved it before: via a localized bottleneck

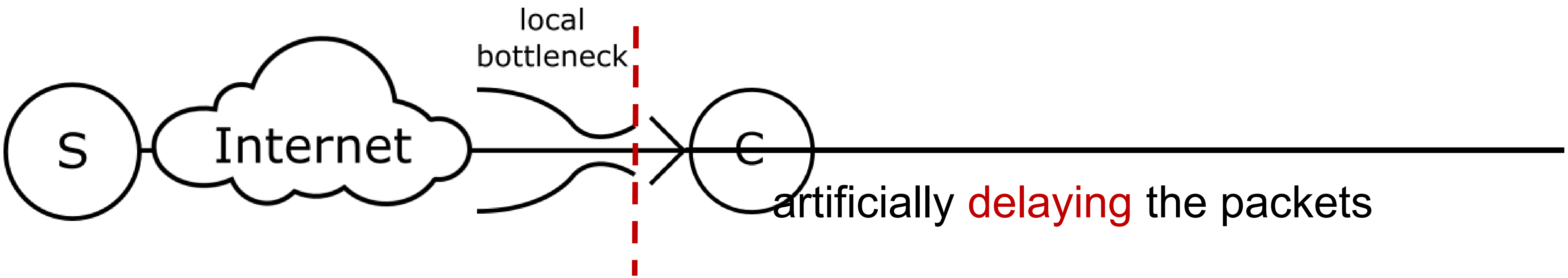

Bytes in Flight
(BiF)
local
bottleneck
S
Internet
C

What's the solution here?

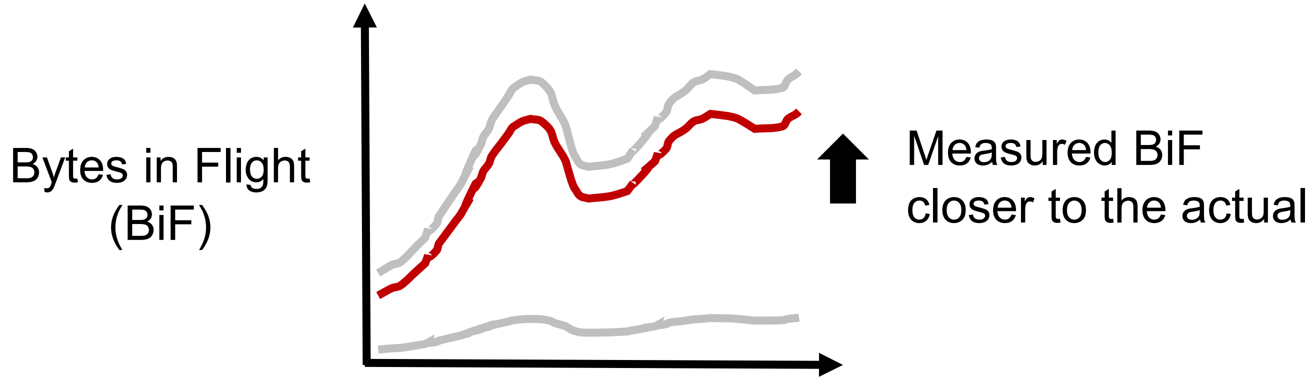
We can't go any nearer to the server…

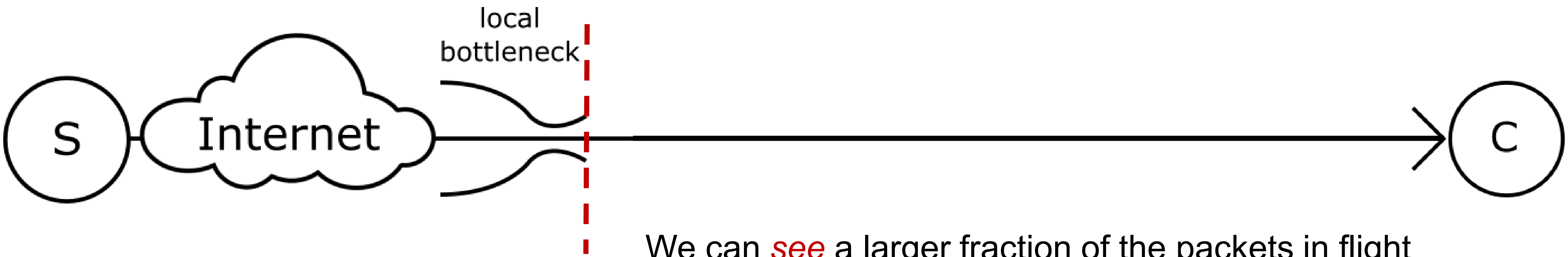


…but we can get further from the client



artificially delaying the packets

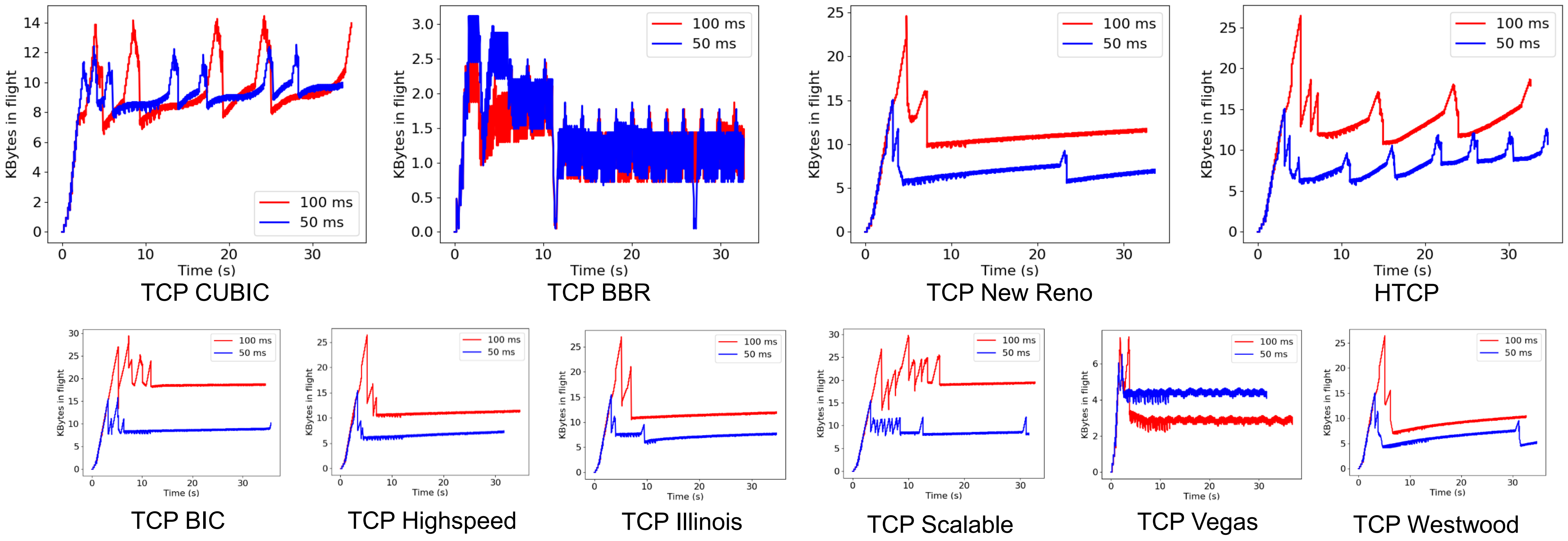




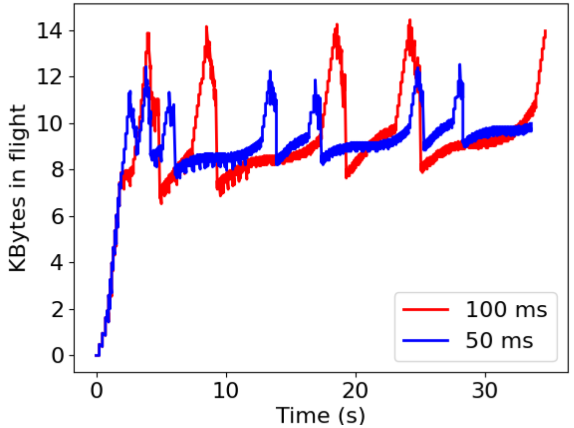
We can *see* a larger fraction of the packets in flight

Nebby's measurement methodology is built on this key insight

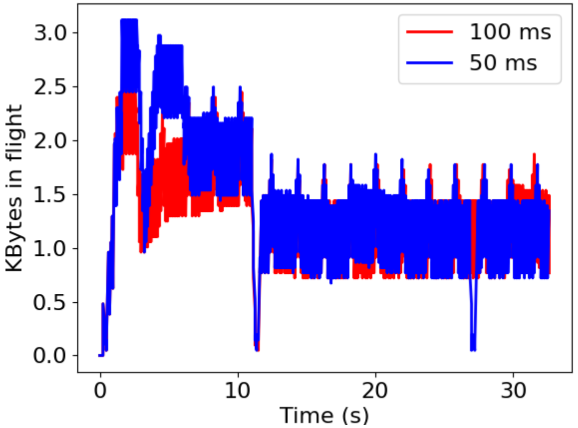
This simple strategy is good enough to capture distinct BiF traces for most CCAs



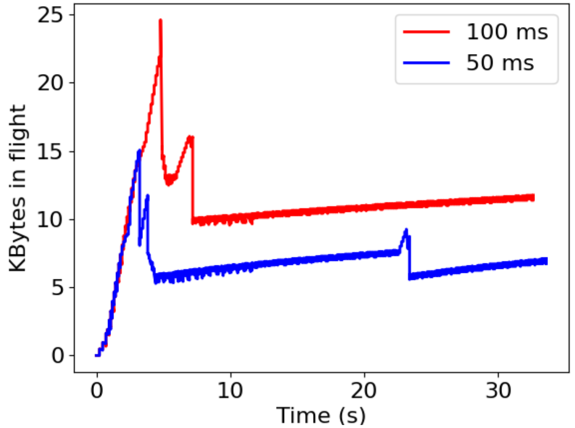
This simple strategy is good enough to capture distinct BiF traces for most CCAs



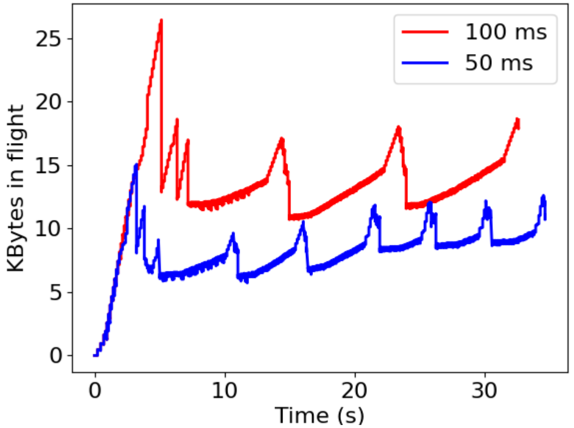
TCP CUBIC



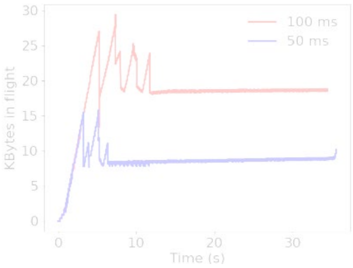
TCP BBR



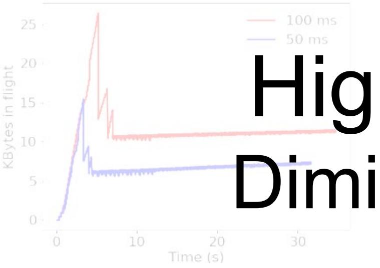
TCP New Reno



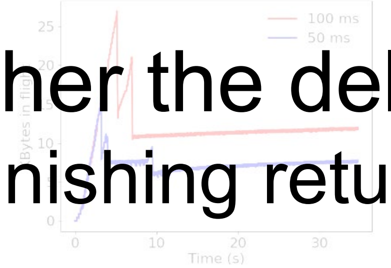
HTCP



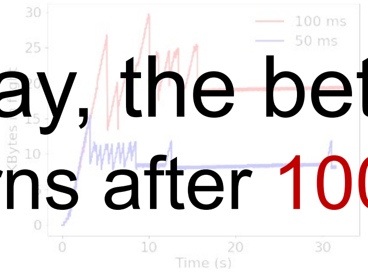
TCP BIC



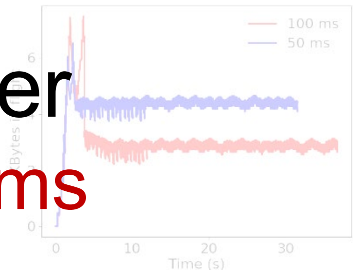
TCP Highspeed



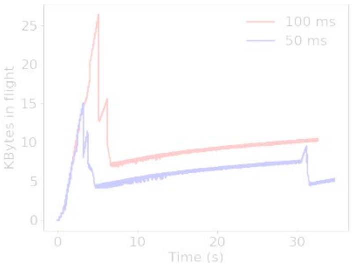
TCP Illinois



TCP Scalable



TCP Vegas

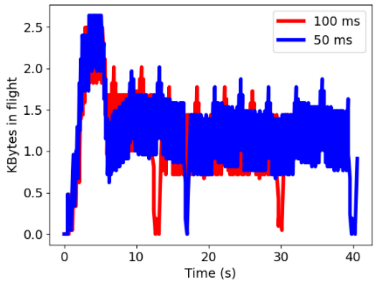


TCP Westwood

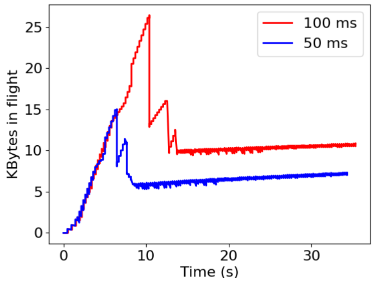
Higher the delay, the better

Diminishing returns after 100ms

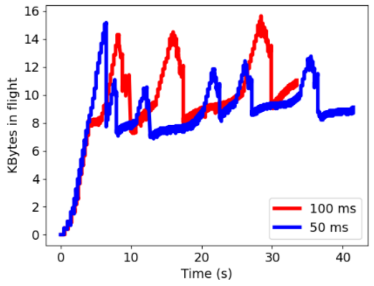
Moreover, since Nebby is mostly passive, it can work with QUIC and real browser traffic too



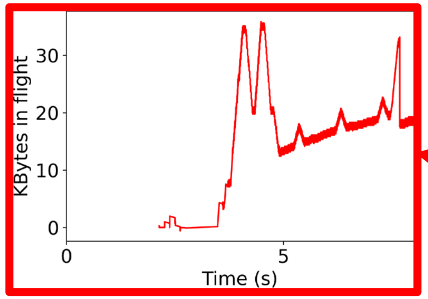
mvfst BBR

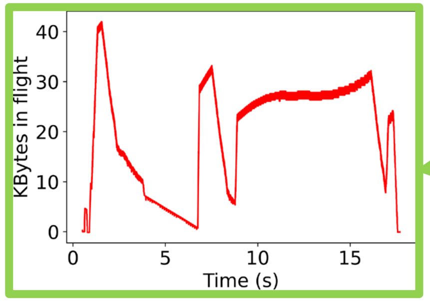


quicgo BBR

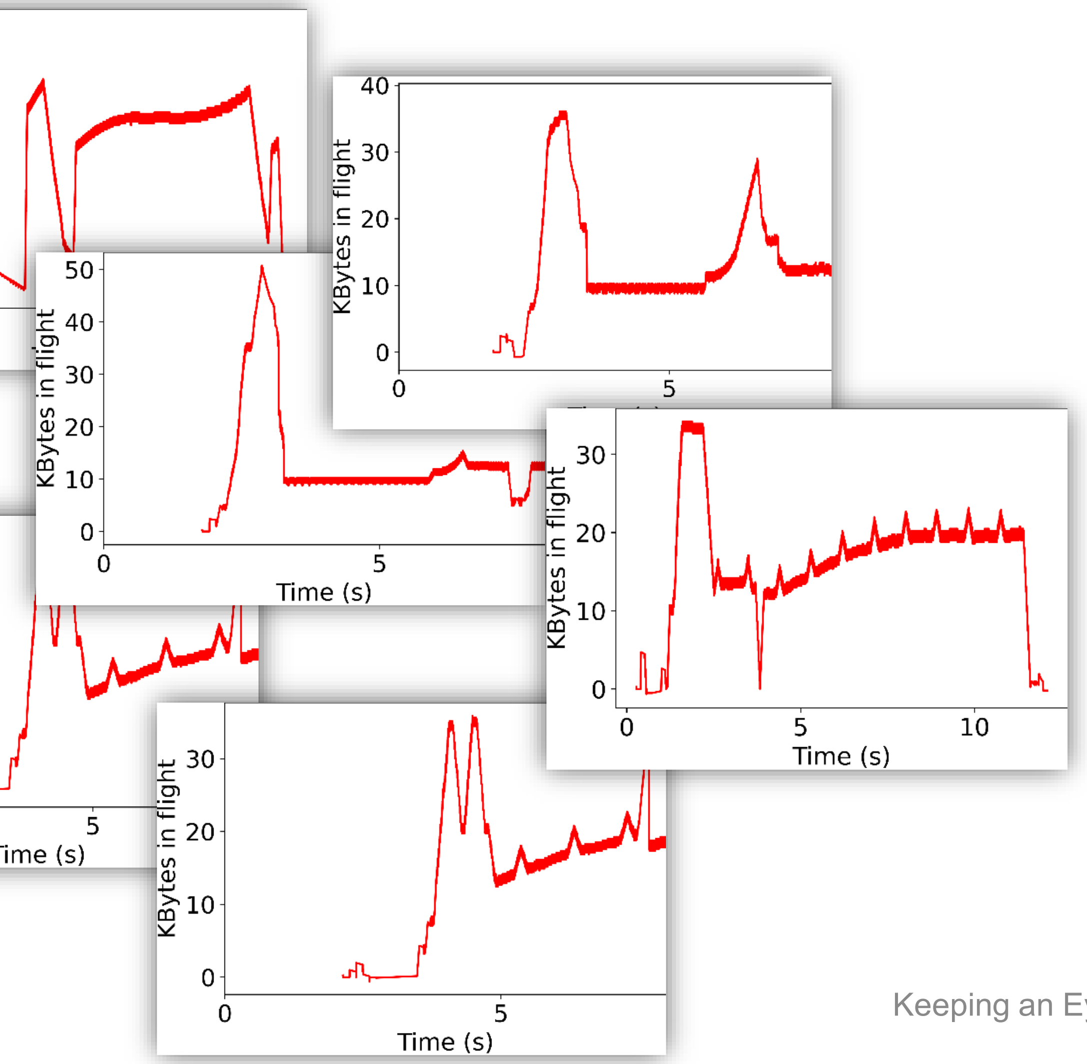


quiche CUBIC





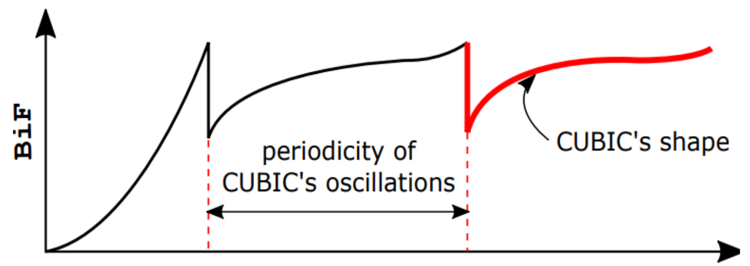
How do we build a classifier for all these traces?



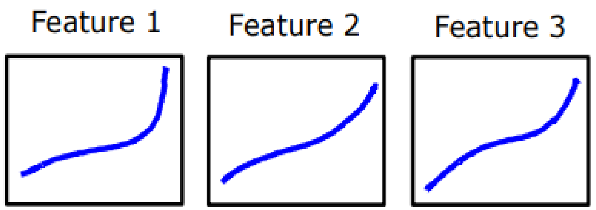
Avoiding a common pitfall: ML-based classifier

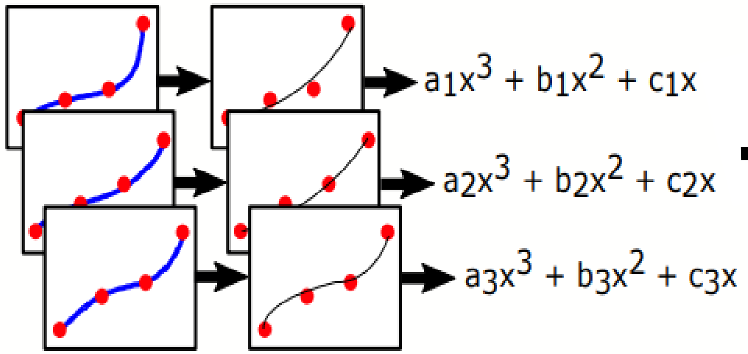
Key insight:
All *reasonable* CCAs are feedback loops with periodic probes and oscillations in the *congestion avoidance* phase.

Using a shape-based Classifier

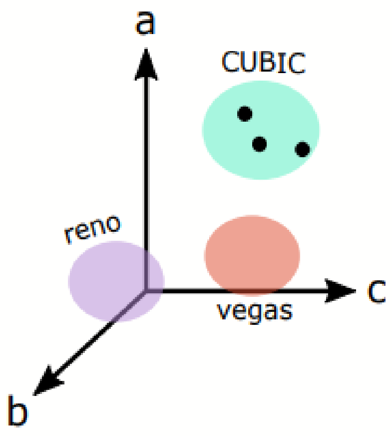


Extract these periodic regions: duration and frequency





Fit Polynomials and compare them with the polynomials for oscillations of known CCAs

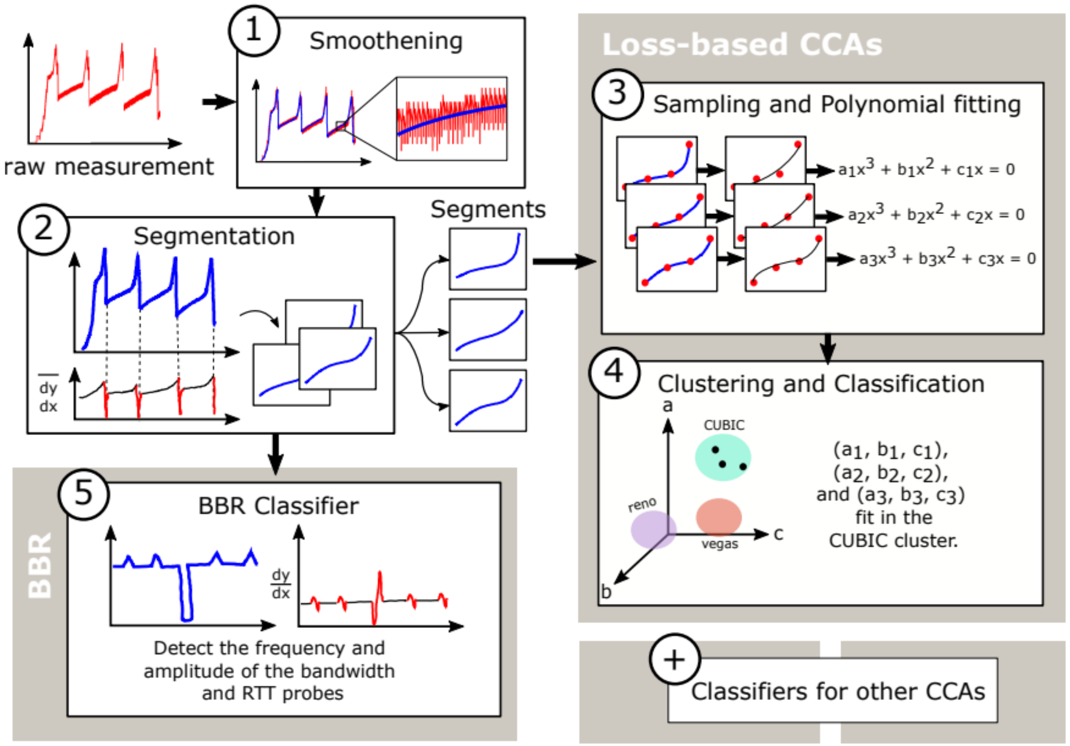


(a_1, b_1, c_1), (a_2, b_2, c_2), and (a_3, b_3, c_3) fit in the CUBIC cluster.

Using a shape-based Classifier

Shape-based classifier can successfully classify all CCAs in the Linux kernel and BBRv2 with an average accuracy of 96%

Nebby is demonstrably extensible, with support for CCAs like Copa and PCC. Can detect unknown CCAs deployed by popular websites.



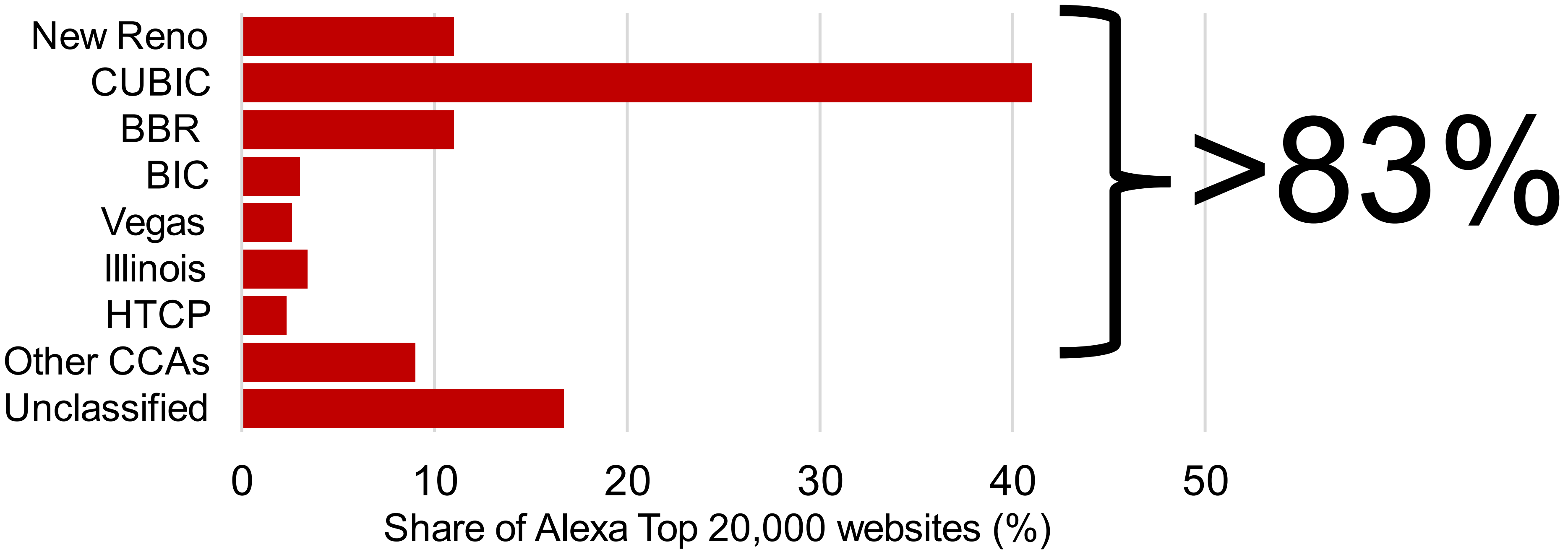
Internet Measurement Results

We made our measurements from `aws` instances in Ohio, Paris, Mumbai, Singapore, and Sao Paulo

We measured the Alexa Top 20,000 websites over a single TCP connection (`wget`) and a single QUIC connection (`quiche`)

We also measured a selection of websites that stream video, audio, and other dynamic content via a chromium web browser.

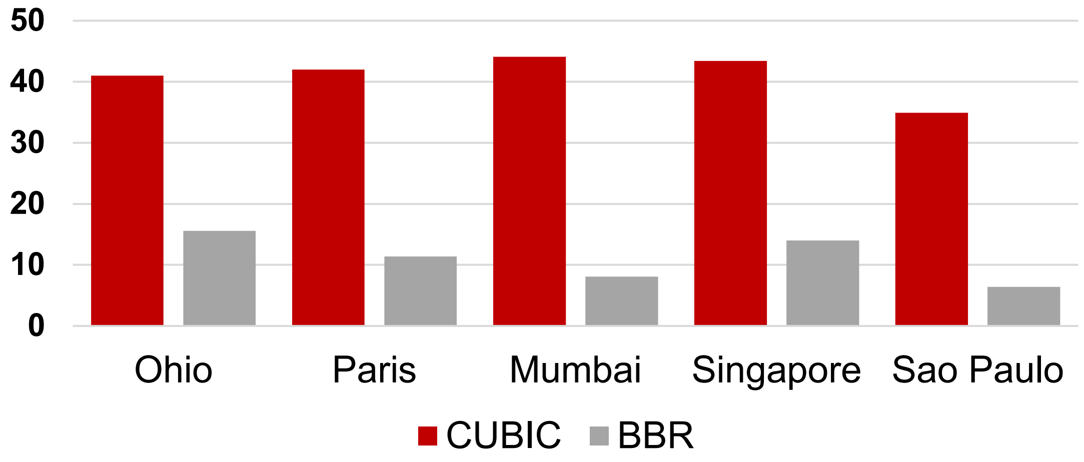
Internet Measurement Results

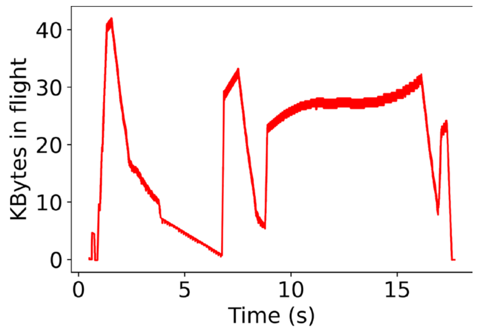


5 key findings

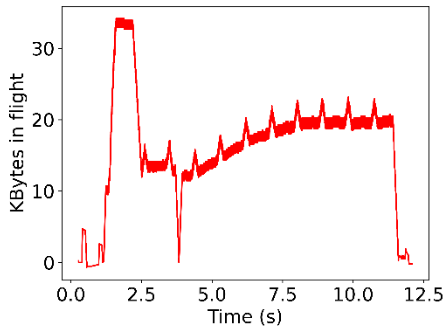
…and many more nuggets covered in the paper

#1 CCA Deployment differs by region





Mumbai



Ohio

While CUBIC and BBR remain the two most dominant CCAs on the Internet, their deployment differs by region

Some websites deploy different CCAs in different regions. For example, `amazon.in` uses CUBIC in Mumbai and BBRv1 in Ohio

#2 Slow Migration to BBRv2

Since the last measurement study, Google has proposed BBRv2, a more fairness-conscious alternative to BBRv1

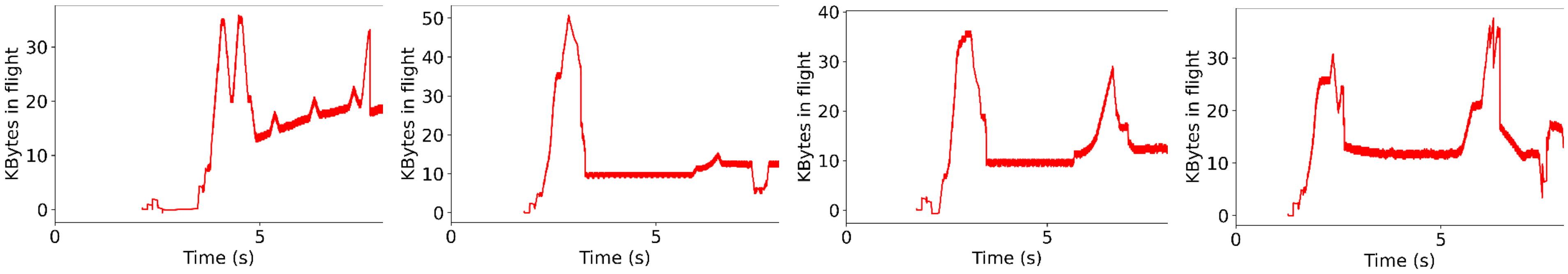
However, despite this, about 98% of websites that deployed BBRv1 in 2019 have either stuck to BBRv1 or switched to CUBIC

Most websites that deploy BBRv2 are new adopters of BBR

#3 Unknown CCAs on the Internet

Nebby found about 24% of websites deploy CCAs that did not resemble any CCAs in the Linux kernel or BBRv2

About 200 of these websites were Google domains(!)

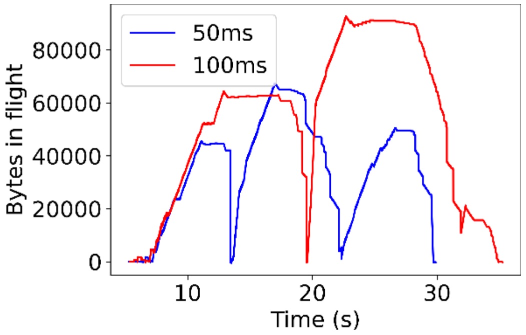


BBRv1 | BBRv2 | `google.com` (June 2023) | BBRv3 (released Aug 2023)

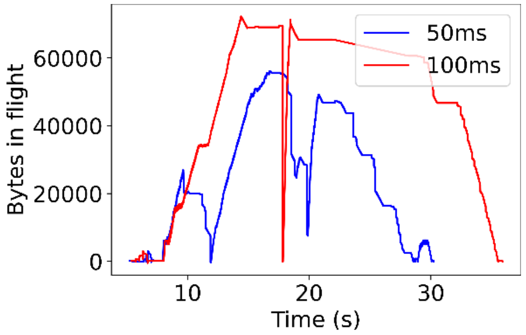
#3 Unknown CCAs on the Internet

Nebby found about 24% of websites deploy CCAs that did not resemble any CCAs in the Linux kernel or BBRv2

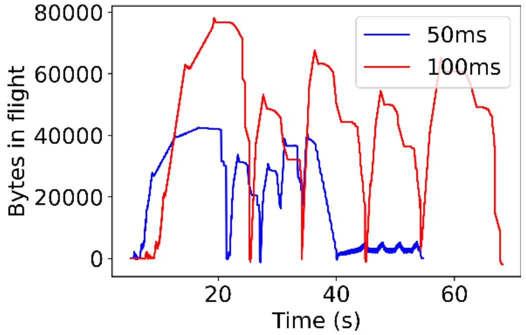
A large proportion of these websites hosted on Akamai were also deploying their own CCAs



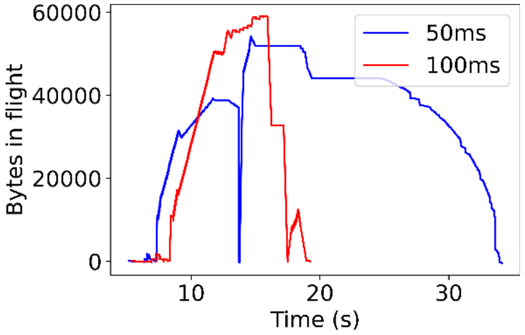
apple.com



hulu.com



pornhub.com



tiktok.com

#4 QUIC still has a long way to go

We saw a surprisingly small number of websites in the Alexa Top 20,000 websites respond to QUIC

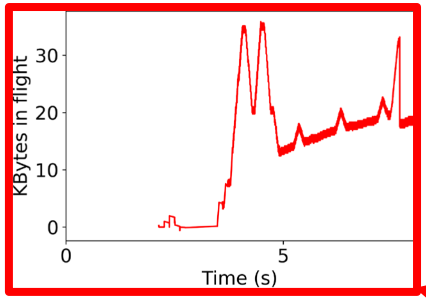
About only 8% of measured websites deployed QUIC

CUBIC and BBR were equally popular amongst websites deploying QUIC

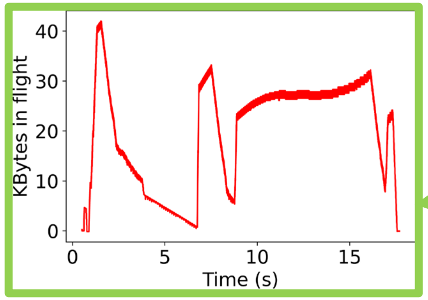
#5 Deployment differs by asset type

Websites like `twitch.tv` and `appletv.com` chose to deploy different CCAs for serving different content

In general, BBR is more popular with websites serving video



BBR



CUBIC

Nebby's Limitations

The current weakest link is the classifier – even though it is modular, it requires a new module for new CCAs

The classifier is also only as good as the selected network profile. However, there is scope for generating specific network profiles for new CCAs

In Summary…

We introduce a fresh methodology for studying and identifying CCAs on the Internet for TCP, QUIC, and live clients.

We show that while BBR's adoption has slowed down, most bandwidth intensive applications still opt for BBR.

BBRv3 and AkamaiCC are case studies in using Nebby to catch the deployment of unknown congestion control algorithms.

Nebby is open source and available on GitHub!