FlowTrace: A Framework for Active Bandwidth Measurements using In-band Packet Trains

Adnan Ahmed (University of Iowa)

Ricky K. P. Mok (CAIDA, UCSD)

Zubair Shafiq (University of Iowa)

Email: adnan-ahmed@uiowa.edu
Website: http://cs.uiowa.edu/~aahmed1/

Outline

Background and Motivation

Measurement tools
Prior art and limitations

FlowTrace

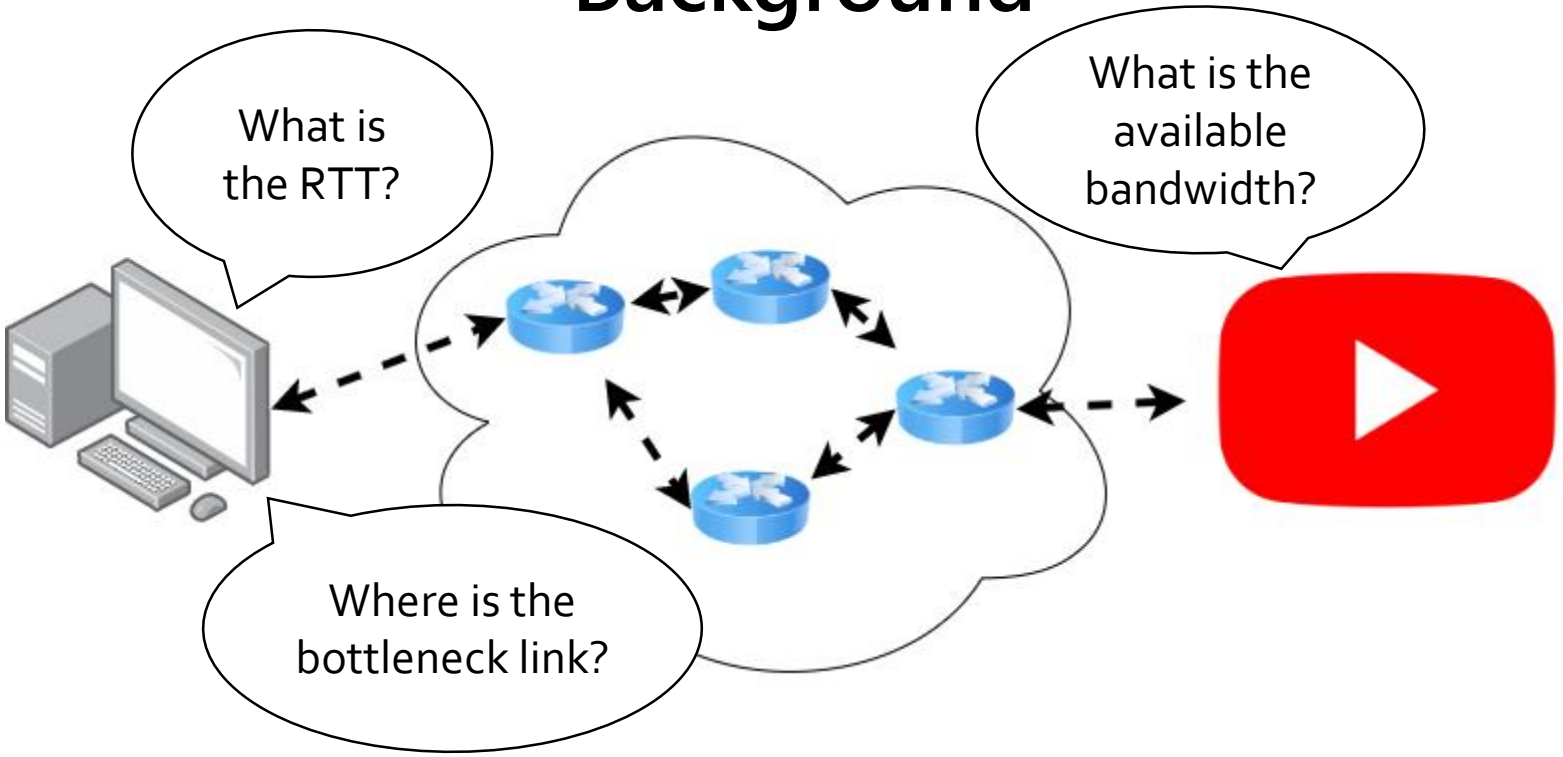
Design framework
pathneck implementation

Evaluation

Emulab testbed

Conclusion and Future work

Background


What is the RTT?
What is the available bandwidth?
Where is the bottleneck link?

Background

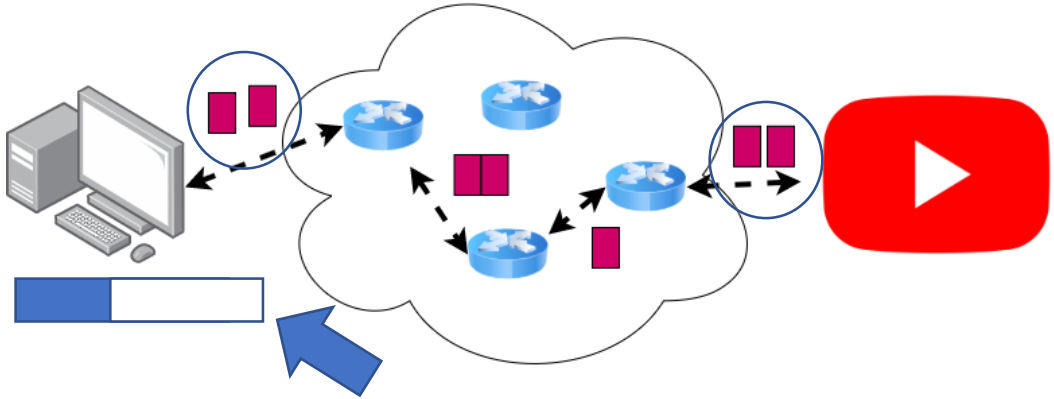
Passive measurements

Active measurements

Passive measurements

Monitor ongoing traffic and local state to infer available bandwidth

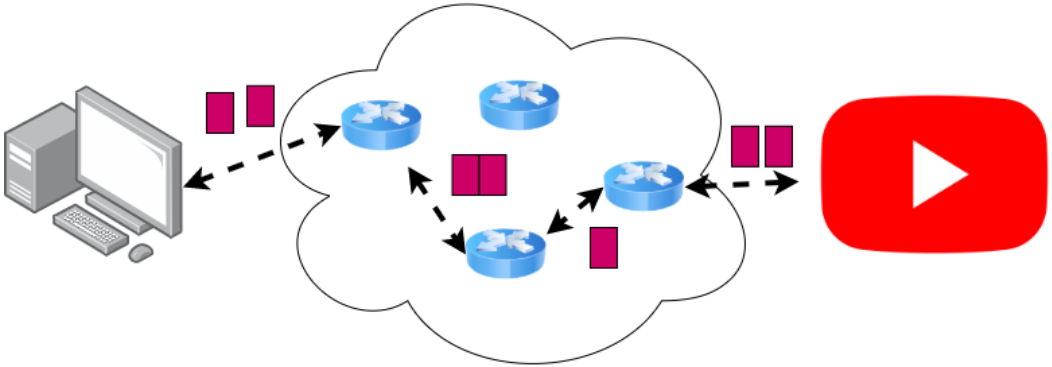
- Video playback buffer
- Packet arrival rate



Passive measurements

Drawbacks

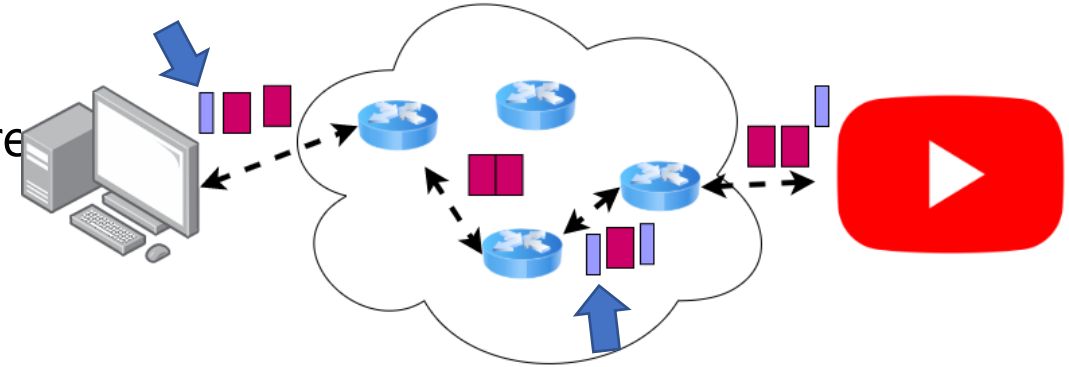
Cannot detect when available bandwidth increases



Active measurements

Inject measurement probes into the network and measure network response to estimate available bandwidth (and choke points)

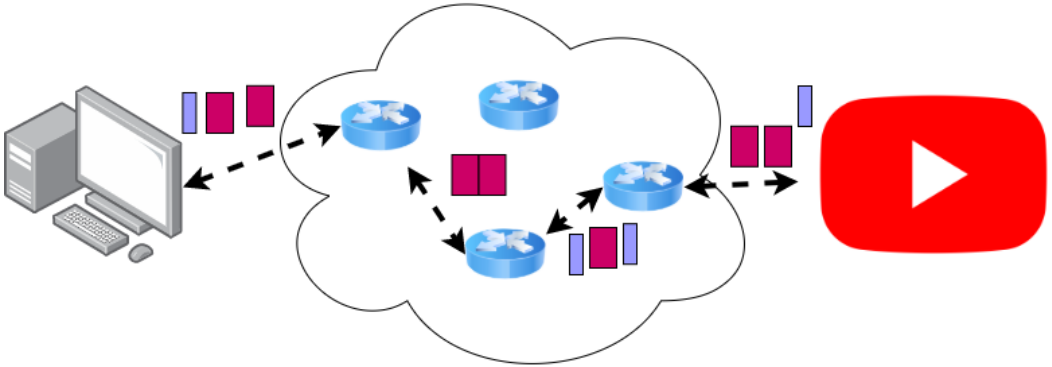
- iPerf
- pathload
- pathneck
- … and many more



Active measurements

Drawbacks

- Measurement traffic competes with the application traffic
- Incurs additional congestion overhead on to the network



Prior Research

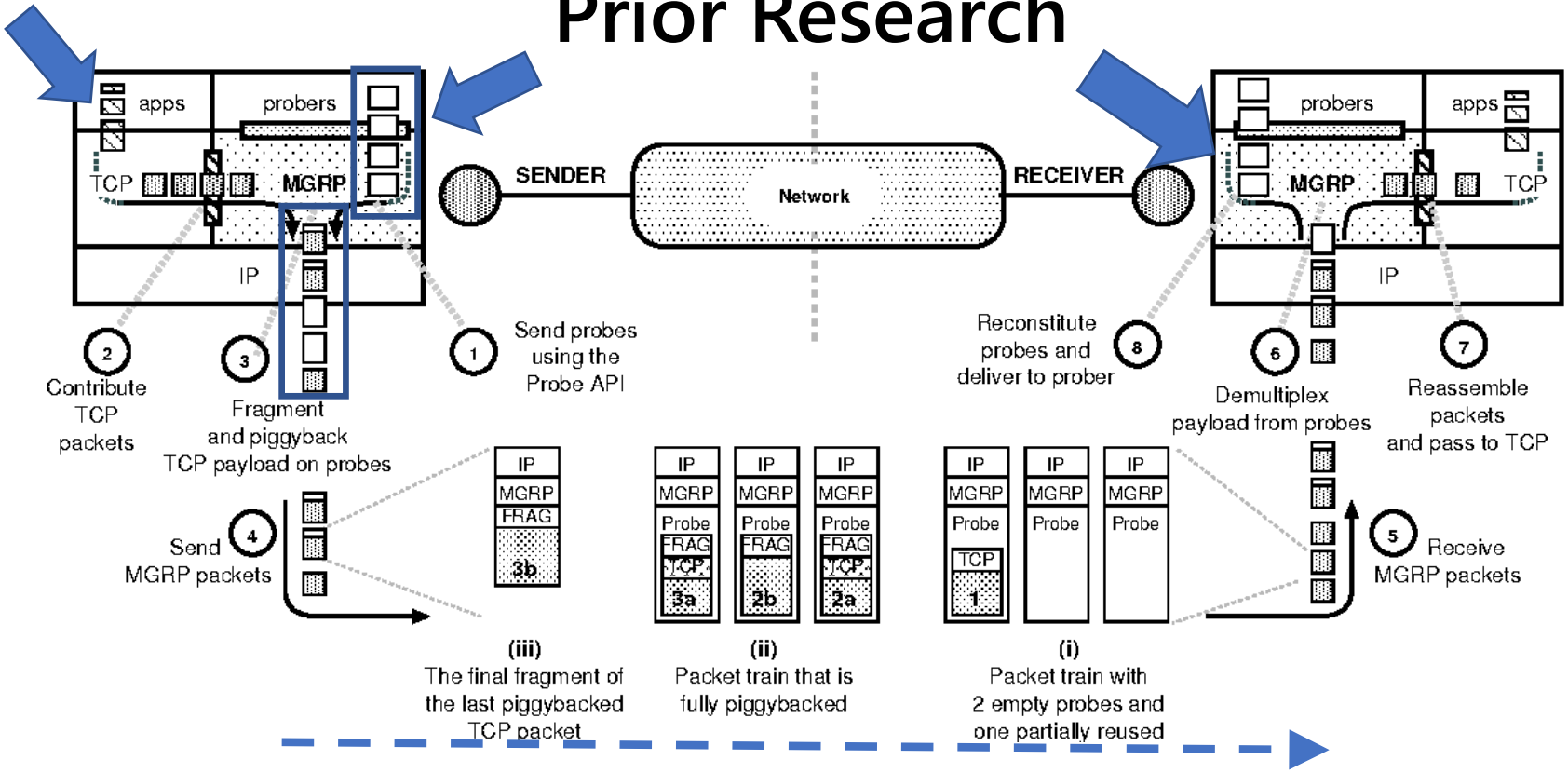
Active bandwidth measurement techniques transfer significant *dummy* measurement payloads over the network

Leverage *useful* application data to construct measurement traffic

MGRP (*SIGCOMM* '09), minProbe (*IMC* '14)

Prior solutions require Kernel changes or specialized hardware

Prior Research



*source: http://www.cs.umd.edu/~pavlos/papers/sigcomm-mgrp-slides.pdf

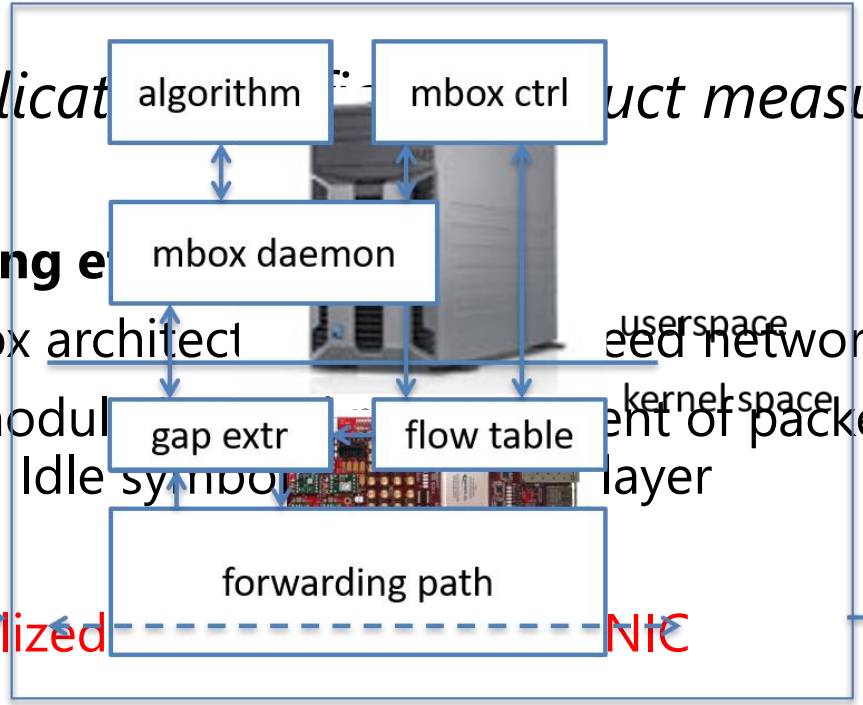
Prior Research

Use applicat… …uct measurements

MinProbe (Wang e…

Middlebox architect… …eed networks

Precise modul… …ent of packet timings by accessing Idle sy…bol… …layer



Requires specialized … NIC

*courtesy: https://www.cs.cornell.edu/courses/cs5413/2014fa/lectures/22-network-measurements-avail-bandwidth.pptx

Outline

Background and Motivation

Measurement tools
Prior art and limitations

FlowTrace

Design framework
pathneck implementation

Evaluation

Emulab testbed

Conclusion and Future work

FlowTrace

Implement measurement algorithms in userspace

Modify the userspace FlowTrace program to implement the desired measurement algorithms

Does not require any specialized hardware or Kernel changes

Design FlowTrace by using basic Linux utilities easing deployment in the wild

Design framework

Allow in-band implementation of active measurement algorithms to decrease the associated overheads

Identify flows and intercept flow packets

Collect packets in userspace

Wait for the desired number of packets to arrive

Construct and transmit the *in-band* measurement traffic at the desired rate

Challenges

Lack of Kernel-level visibility and packet control in userspace

Control transmission rates of the packets

Minimize the impact of FlowTrace on application performance

Challenges

iptables

Configures firewall to allow or block application traffic

libnetfilter_queue and NFQUEUE

NFQUEUE is an iptables target

Delegates verdicts on intercepted packets to userspace

Userspace programs issue verdicts via libnetfilter_queue API

Challenges

libnetfilter_queue and NFQUEUE (contd.)

Packets are delegated from the head of the queue

Packets are popped from the queue upon verdict

Packet buffering

Buffer packets in userspace and drop from the queue

Challenges

Lack of Kernel-level visibility and packet control in user-space

Intercept packets using iptables and libnetfilter_queue

Control transmission rates of the packets

Buffer packets in memory and respawn at the desired rate

Minimize the impact of FlowTrace on application performance

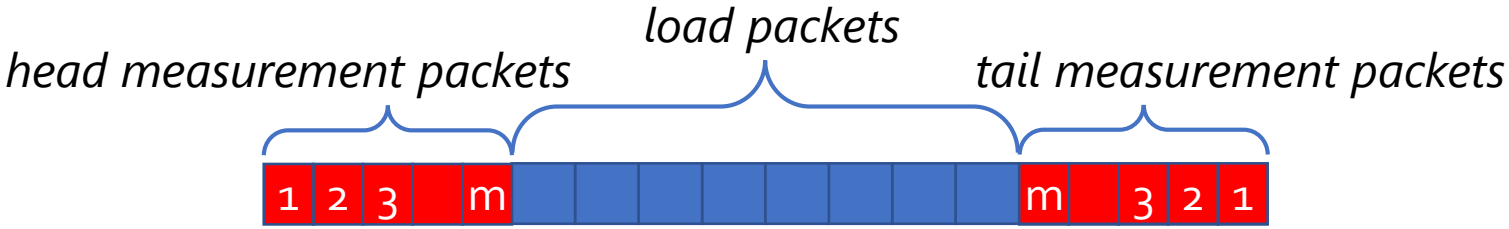
Per-packet wait timers (t_{ipa}) to avoid latency overheads

pathneck (Hu et al., *SIGCOMM '04*)

An active probing technique to locate choke points and bottleneck along an Internet path

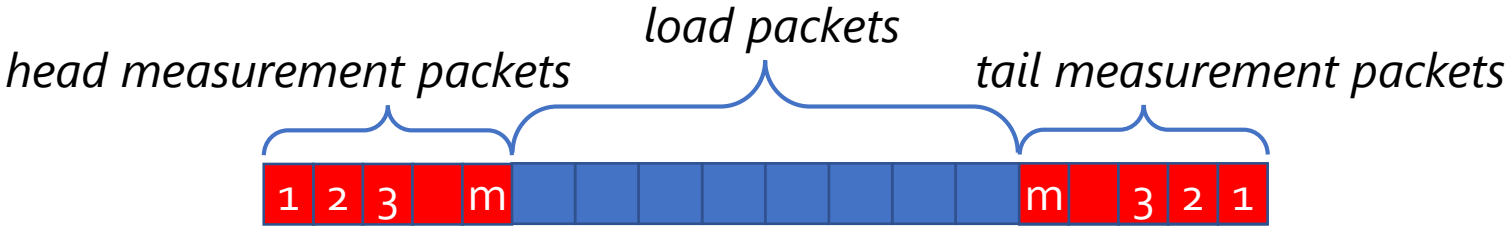
Recursive Packet Trains (RPT) consisting of load packets and TTL-limited measurement packets

Locates choke points by measuring gap values in the returning ICMP messages from each hop

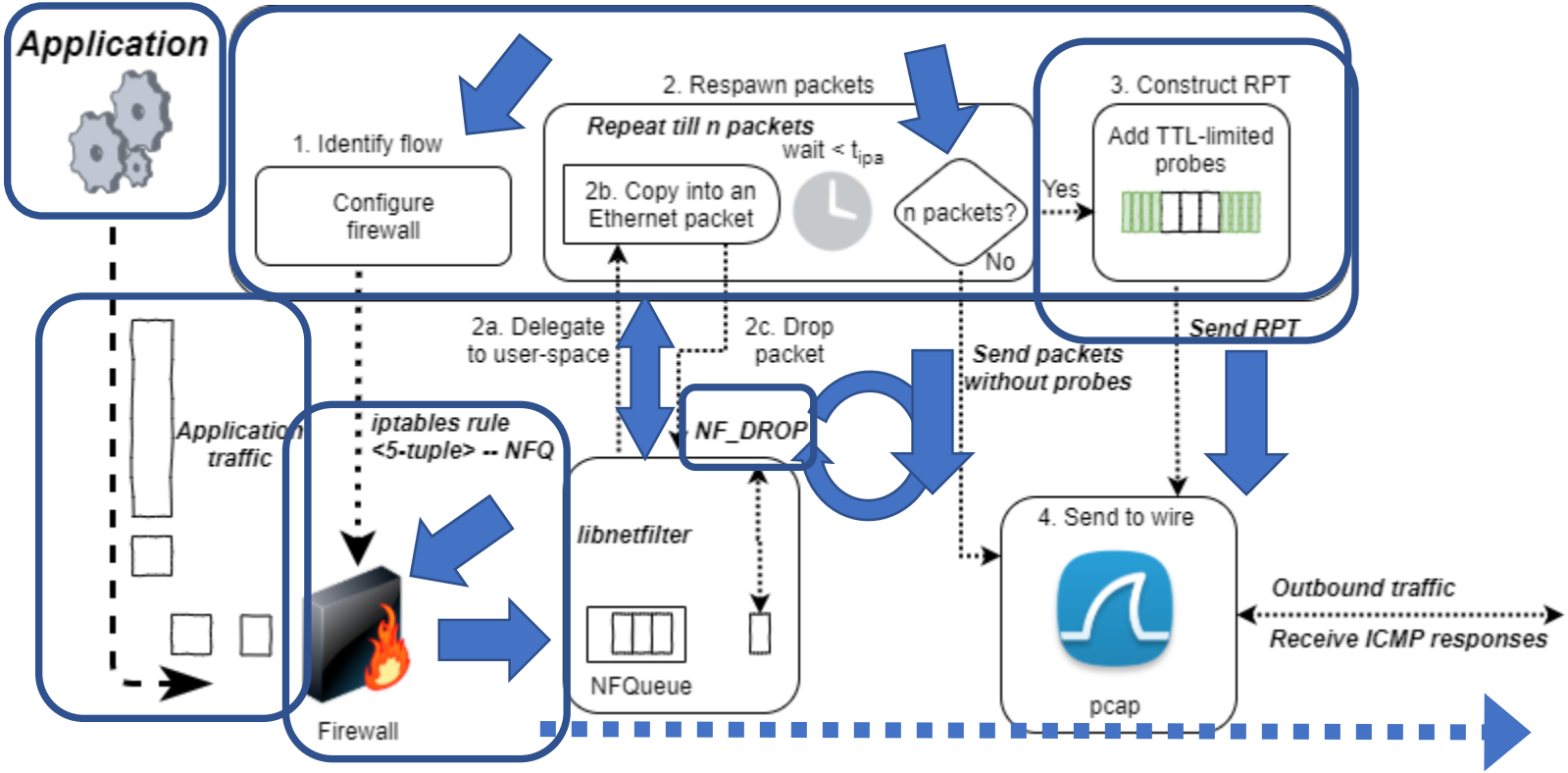


pathneck

Use application packets as load packets in the RPT



Implementing pathneck



Outline

Background and Motivation

Measurement tools
Prior art and limitations

FlowTrace

Design framework
pathneck implementation

Evaluation

Emulab testbed

Conclusion and Future work

Testbed Evaluation

Are the measurements done with pathneck implemented on FlowTrace close to the measurements done with pathneck?

How does cross-traffic affect the measurements done with both pathneck and FlowTrace?

How high are the latency overheads introduced by FlowTrace in the application layer flow?

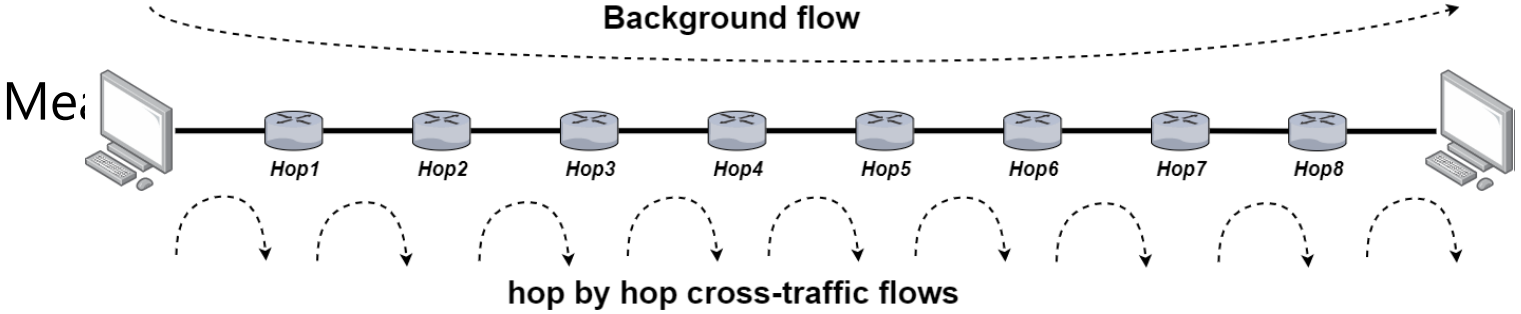
Testbed Evaluation

Emulab testbed

Linear topology

Varying bottleneck location and size

Background flow, hop-by-hop varying cross-traffic flows



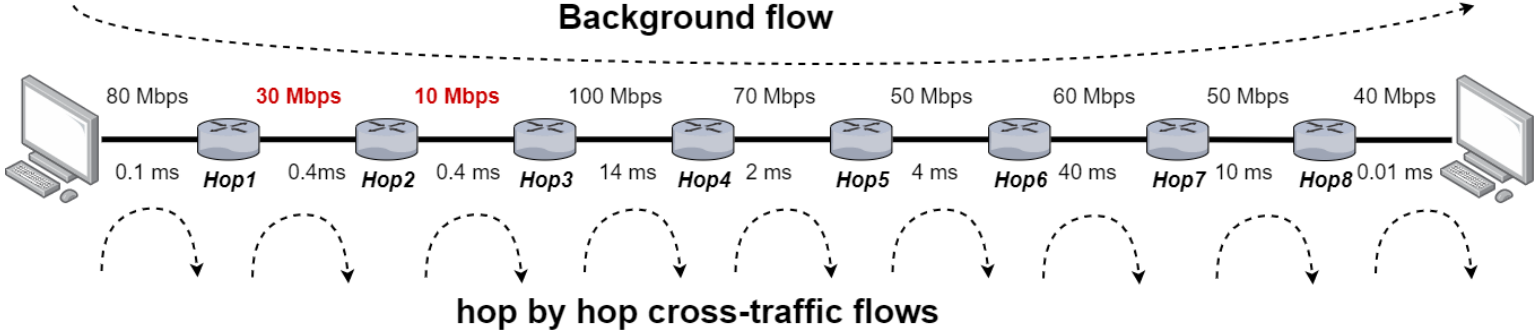
Testbed Evaluation

Two choke points scenario

Choke points at Hop2 and Hop3

With and without cross traffic

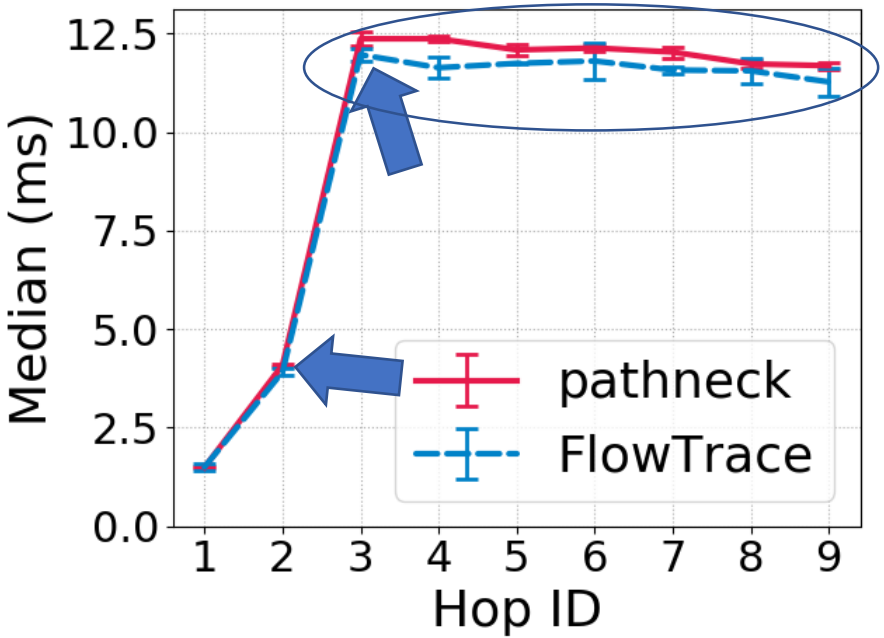
Measure with both pathneck and FlowTrace



Testbed Evaluation

Without cross traffic

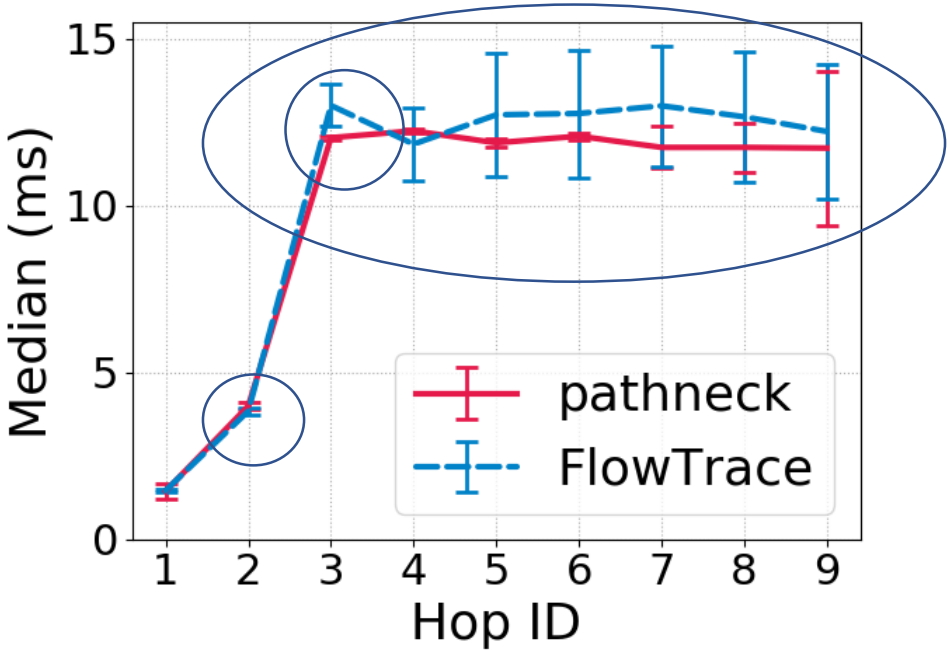
Both pathneck and FlowTrace identify the gap between hop 2 and hop 3 as the bottleneck



Testbed Evaluation

With cross traffic

Both pathneck and FlowTrace identify similar increases in gap values along the path



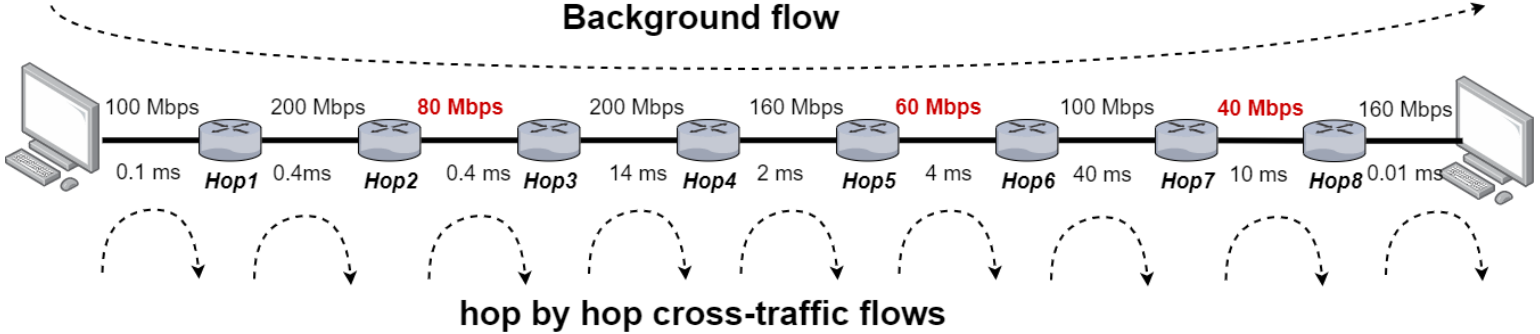
Testbed Evaluation

Three choke points scenario

Choke points at Hop3, Hop6, and Hop8

With and without cross traffic

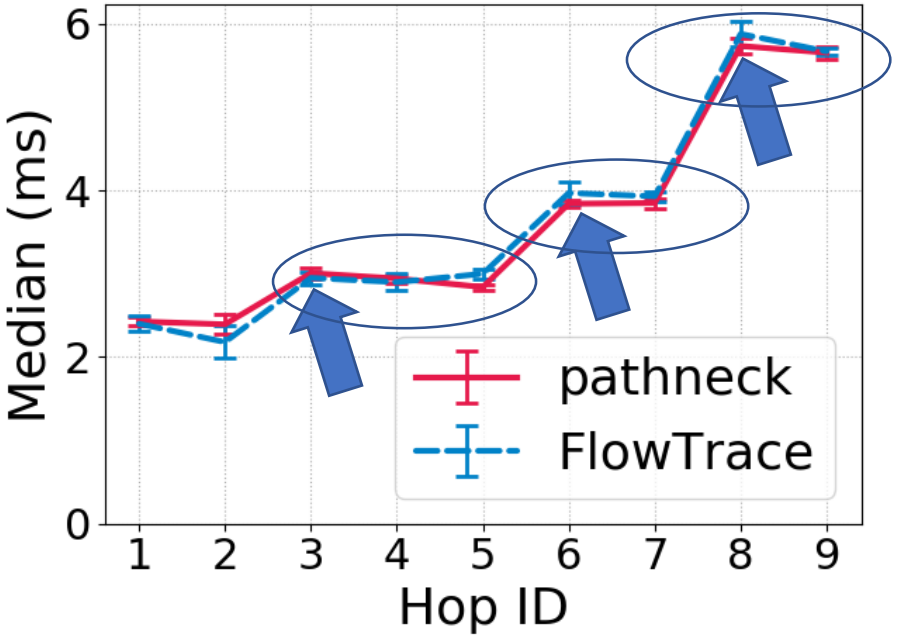
Measure with both pathneck and FlowTrace



Testbed Evaluation

Without cross traffic

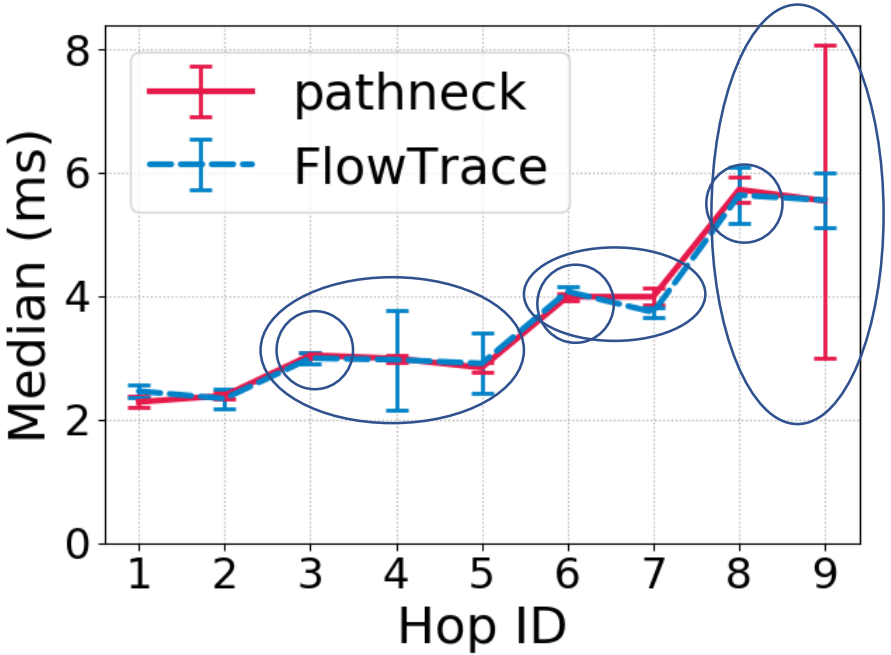
Both pathneck and FlowTrace identify the gap between Hop 6 and Hop 8 as the choke points



Testbed Evaluation

With cross traffic

Both pathneck and FlowTrace identify Hop 3, Hop 6, and Hop 8 as choke points



Testbed Evaluation

Latency overheads introduced by FlowTrace

Measure the additional latency experienced by each application packet due to buffering by FlowTrace

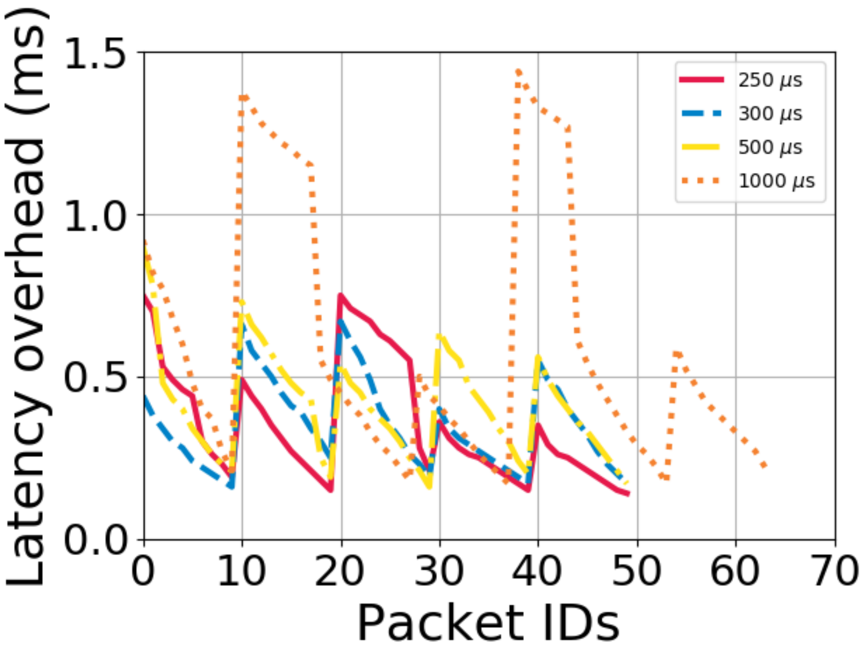
Study the effect of the per-packet wait timers (t_{ipa}) on the latency overheads

Testbed Evaluation

Latency overheads

As T_{ipd} increases, packets tend to exhibit potentially periodic behavior with longer, increasing the latency overheads

Approximately 0.75ms latency overhead can be buffered for the last packet in the RPT experiencing least additional latency due to buffering



Outline

Background and Motivation

Measurement tools
Prior art and limitations

FlowTrace

Design framework
pathneck implementation

Evaluation

Emulab testbed

Conclusion and Future work

Conclusion

We presented FlowTrace—an in-band framework to implement active network measurement algorithms using basic Linux utilities

We implemented pathneck in FlowTrace and analyzed the measurements conducted by both pathneck and FlowTrace

The latency overheads introduced by FlowTrace are relatively insignificant, and increase with the value of inter-packet arrival threshold, t_{ipa}.

Future Work

Implement other active measurement algorithms such as pathload, pathchirp, on FlowTrace

Conduct Internet-wide experiments to measure network characteristics such as available bandwidth and bottleneck location

Study the impact of FlowTrace on the performance of different protocols and applications such as web browsing, video streaming.

Questions?

Email: adnan-ahmed@uiowa.edu
Website: http://cs.uiowa.edu/~aahmed1/
Twitter: @_adnan_ahmed_