Measuring End-to-End Bulk Transfer Capacity

Mark Allman
NASA GRC/BBN
mallman@grc.nasa.gov

SIGCOMM Internet Measurement Workshop
November 2001

Overview

- Background on Bulk Transfer Capacity (BTC)
- Tools and Methodology
- Preliminary Validation Results

Background

- BTC is defined in RFC 3148 as:
  - Roughly, Bulk Transfer Capacity (BTC) is a measure of the throughput that a standards-compliant implementation of TCP's congestion control algorithms would obtain over a given path at a given time.
- But, TCP allows implementers a bit of slack in some of the details in the CC algorithms.
- However, BTC metrics must nail down all these details.

Related Work

- Lots of work on measuring the raw bandwidth and the available bandwidth of links and network paths.
  - pathchar, cprobe, bprobe, pchar, clink, etc., etc.

- The BTC does not attempt to measure either raw or available bandwidth.
- But, we hope BTC is a better predicter of what a user might experience when using the network.

BTC Motivation

- BTC has been envisioned as a user-level process that would implement CC according to the TCP specification.
- Possible uses for such a tool:
  - Find and diagnose problems in a given network path.
  - Measure BTC uniformly -- without relying on underlying operating system quirks.
    - (Can’t completely factor out the OS, but we can try to minimize its impact.)
  - Attractive for researching new congestion control mechanisms and tweaks.
    - Development is likely easier.
    - Deployment for wide-scale testing is easier.

BTC Motivation (cont.)

- BTC uses (cont.):
  - Provides a way to probe the network for various details on the same timescales as apps are likely to observe these characteristics.
    - e.g., loss
    - e.g., reordering
    - e.g., packet duplication
  - The rate at which we send traffic to determine this is "safe"

Goal

- The goal of this work is to see how the performance measured by a BTC application compares to the performance of a stock TCP implementation.

Methodology

- We developed a BTC tool called cap.
  - Uses two programs (cap and capd) that send and sink data respectivly.
    - Really just an exchange of UDP packets
  - CC algorithms written to the specification (RFC 2581) not necessarily attempting to mimic any particular TCP implementation.
- cap was deployed on the NIMI mesh of measurement hosts.
- At the time the measurements were taken the NIMIs were all some form of FreeBSD or NetBSD.

Methodology (cont.)

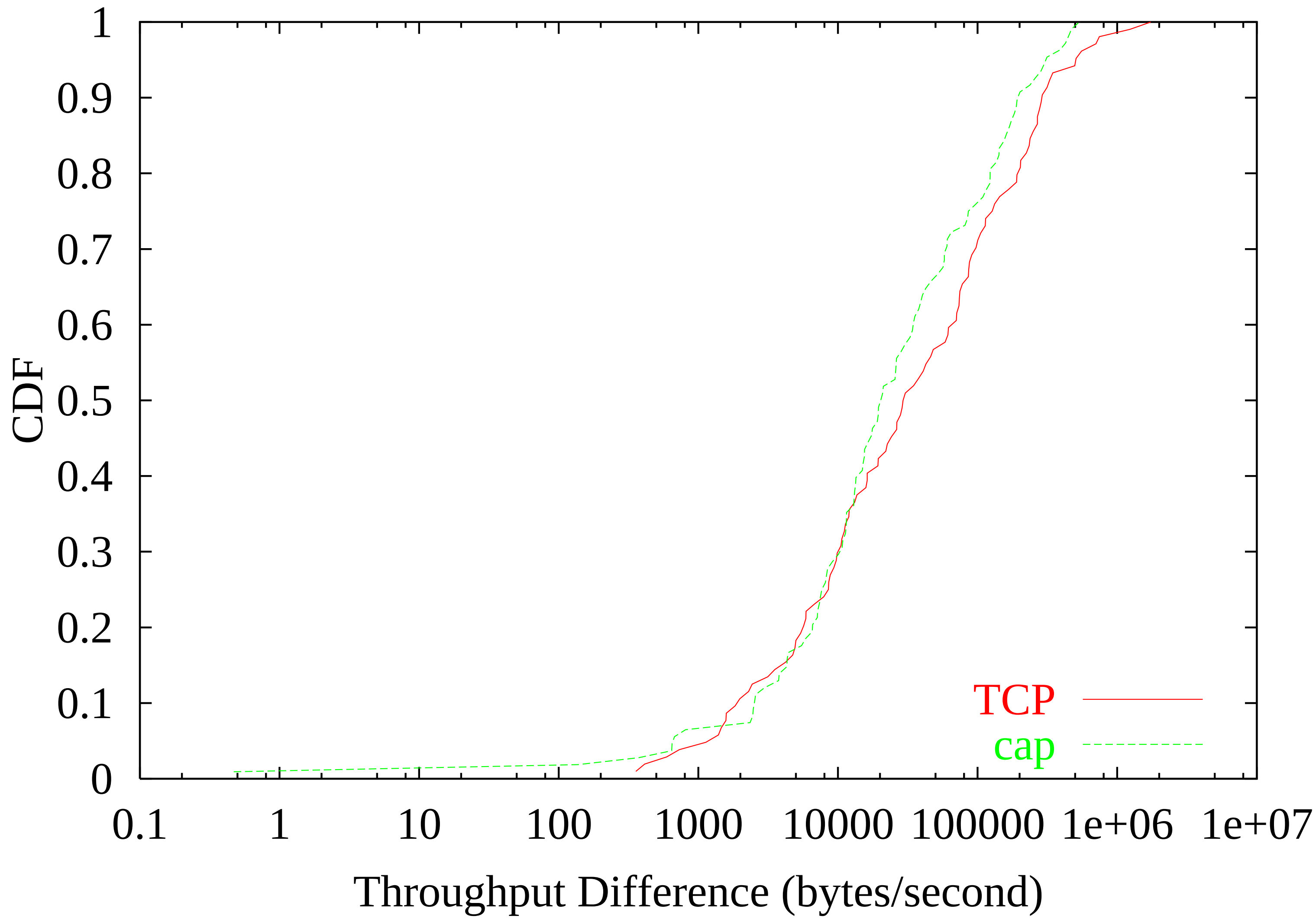
- We scheduled measurements at random times between random NIMIs.
- Each measurement consists of 2 back-to-back transfers of roughly 1 MB o data.
  - We hope that these two transfers behave about the same since they are closely spaced.
    - (And, we understand that this is bogus!)
- The method (cap or TCP) for each transfer is chosen randomly (with replacement).

Methodology (cont.)

- After scrubbing the data we ended up with over 100 measurements for each of these categories:
  - TCP / TCP
  - cap / cap
  - cap / TCP or TCP / cap
- (This is not enough and we plan to collect a new dataset whenever we get all the tools working under NIMI again.)

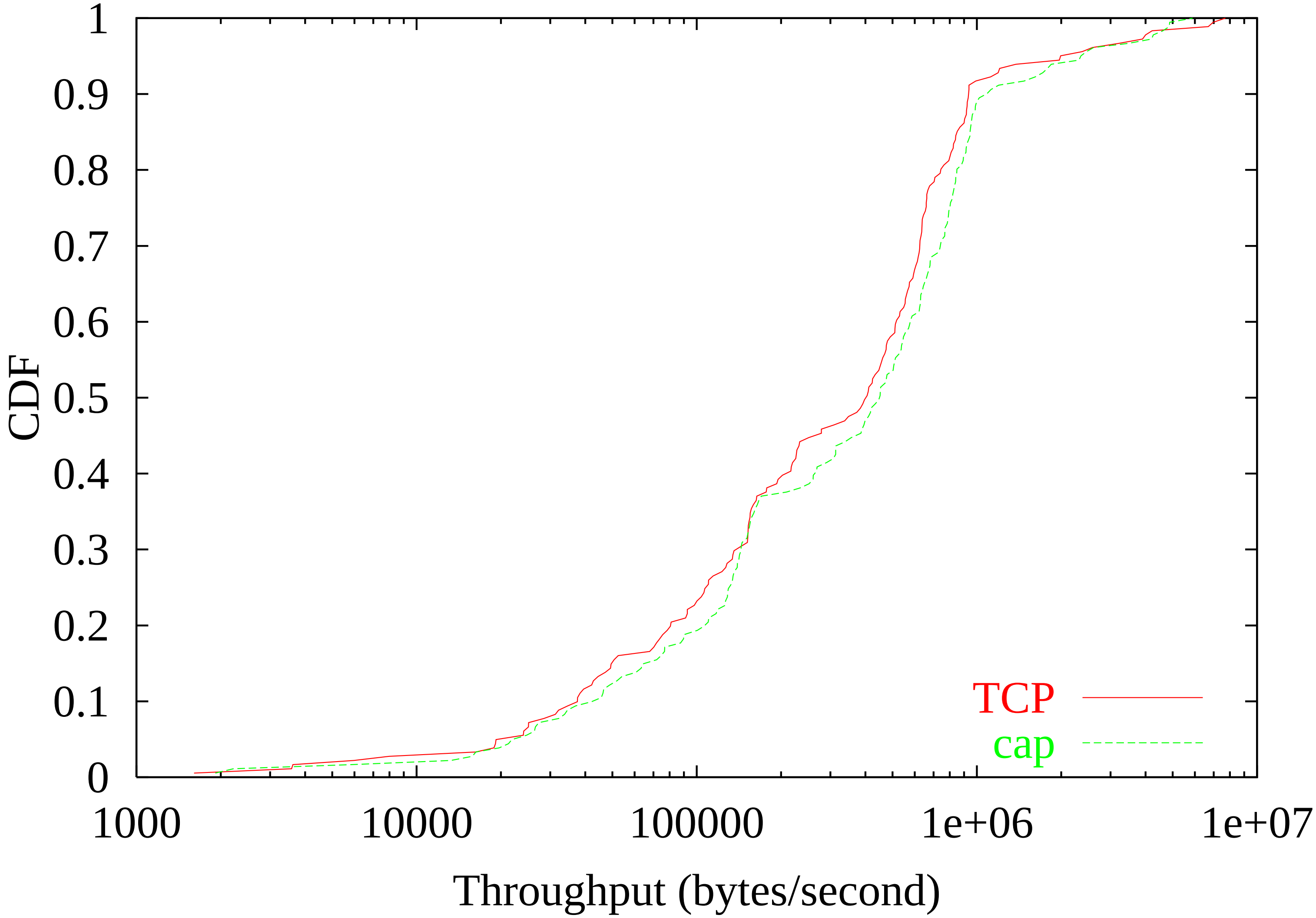
Results

Stability of back-to-back transfers:



Results (cont.)

Throughput distribution:



Conclusions

- A tool that implements BTC is attractive for several reasons.
- We have some preliminary results that show that accuratly measuring BTC with an application layer process seems possible.
- Future work includes:
  - Validating these prelminary results with more tests over more network paths.
  - Digging a bit deeper into the data to make sure things like loss rates and RTT measurements are being done in a similar fashion to those obtained by TCP.