CS514: Intermediate Course in Computer Systems

Lecture 6: Sept. 17, 2003

"Performance of the Internet WAN"

What performance are we interested in?

CS514

- Latency
  - Small web access
  - Interactive voice/video or gaming
- Jitter (variation in inter-packet arrival)
  - Interactive voice/video
  - Streaming could benefit (less buffering)
- Throughput
  - Large web access
  - Interactive or streaming video
- Problems
  - Packet loss, outages, out-of-order packets, packet corruption

This lecture

CS514

- Shows why measuring performance is difficult
- Shows how some of the measurement takes place
- Tries to give some sense of how well the internet works
  - Though really nobody knows!!!

Email with Vern Paxson*

CS514

Q: What are the best papers for describing the performance of the internet more generally?

A: There's no broad-perspective available like that. Very hard to do, given the immense diversity and difficulty of attaining sufficient measurement perspectives.

* God of internet measurement

Paxson 1994-1995 Study

CS514

- 35 sites
- Measured TCP bulk transfers
  - Idea is that measurements of TCP can be applied to understanding TCP as well as understanding the Internet
  - Problem though is that TCP backs-off, so don't have complete control over your measuring tool
- Built filter to measure exact times
  - Some packets missed by filter, but this is detectable

TCP-centric measurements

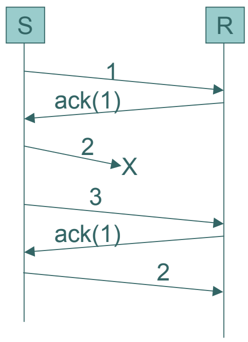
CS514

Simple UDP or ICMP-based measurements don't tell you how well TCP will perform

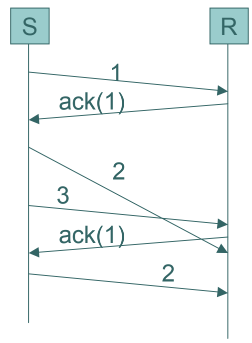
- You have to look at things that will hurt TCP performance:
  - Out of order delivery
  - Replication (rare, it turns out)
  - Packet corruption

Out of order delivery: Why does TCP care?

CS514



If packet lost, best strategy is to retransmit immediately after duplicate ACK



If packet reordered, best strategy is to wait for another ACK

Out of order delivery

CS514

- It does happen (~0.2% in 1995)
  - Route changes? Path splitting?
- It varies a lot (up to 15% seen)
  - Thus nice if TCP could measure it dynamically
- Different in either direction
  1. Asymmetric paths
  2. Data more frequently reordered than ACKs
  - Therefore can't determine sending reorder rate from measuring received reorder rate

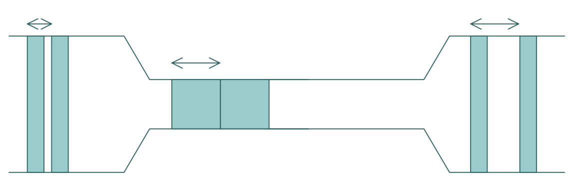
Packet corruption

CS514

- Happens (1 in 5000 data packets)
  - Since 16-bit TCP checksum misses 1/65000 errors, 1/300M corrupted packets go undetected
  - Not a lot, but if your data is important, check at higher layers
  - Encryption will do this for you
- Happens much less for short packets (TCP ACKs)
  - Suggests that corruption happens in routers
  - Because link-layer checksum shouldn't care about packet length

Measuring bottleneck bandwidth: principle

CS514



Send two packets within a short time space

Bottleneck link will increase the spacing

Measured receive spacing tells you the size of the bottleneck link

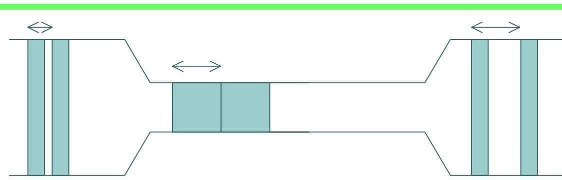
Why do we care?

CS514

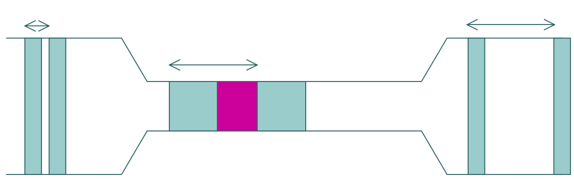
- If we want to understand effect of queuing delays, processing delays, etc., we must know bottleneck bandwidth
  - For instance, difference between bottleneck BW and available BW tells what fraction of the bottleneck link we got
- TCP never wants to send faster than the bottleneck bandwidth
  - Though often TCP wants to send much slower because of competing traffic (available bandwidth)

Why is it hard?

CS514



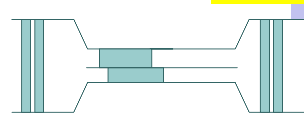
In part, because any competing traffic changes the spacing



Other complications

CS514

- Multi-channel effects



- Bottleneck may change
  - Route change
  - Dynamic bandwidth allocation (ISDN)
- Poor clock resolution
- Out-of-order delivery

Hard to do at one end

CS514

- If ICMP echo, can’t tell in which direction bottleneck occurs
- If TCP, ACK spacing may be compressed on return path
  - Queue emptying effect

Basic approach to bottleneck bandwidth est.

CS514

- Send stream of evenly spaced packets
  - Look for smallest spacing
- If possible link bandwidths known apriori, then look for peaks near these speeds
  - (modem speeds, T1, E1, multiples of these, Ethernet, etc.)
- This actually works pretty well
  - But impossible to do good bottleneck bandwidth estimation without loading the network

Measuring packet loss

CS514

- Not real hard
  - Though can't use TCP data to measure loss rate, because TCP backs off in order to prevent loss
  - But can use ACKs
    - If loss in both directions is not correlated, which is the case
- Lots of variation
- Lots of correlation
  - Likelihood packet is lost if predecessor was lost
  - 25%-50%
  - But this is pre-RED

One-way Transit Time (Latency)

CS514

- To do with absolute accuracy, required synchronized clocks
  - GPS, which is fairly inexpensive these days (<$500)
- Paxson didn't have this
  - But with analysis could determine relative skew between clocks
  - Basically by sampling only packet round trips with low and similar delay, and tracking timestamps against these
- He really only measured relative OTT

Other Paxson '97 factoids

CS514

- Queuing time scales
  - Period of time over which a queue's delay changes
  - We care because if time is too small, no point in trying to adapt
  - Typically 0.1 – 1 sec, but can be much larger
    - 60 second spike, but due to routing oscillations

Other Paxson '97 factoids

CS514

- Available bandwidth
  - Percentage of bottleneck bandwidth allocated to a given connection
  - Essentially actual_bw/bottleneck_bw
  - Wide range:
    - From 5% to 100%
    - Less as bottleneck BW grows

Large scale measurement now common

CS514

- Many ongoing measurement projects
  - NIMI (Paxson), RIPE TTM, Caida skitter, etc.
- Standard measurement metrics
  - IP Performance Metrics (IPPM)
  - RFC 2330: IPPM Framework
  - Basic concepts and terms
  - Allows results from different measurement infrastructures to be meaningfully compared and combined

IPPM Metrics

CS514

- RFC 2678: IPPM Metrics for Measuring Connectivity
- RFC 2679: A One-way Delay Metric for IPPM
- RFC 2680: A One-way Packet Loss Metric for IPPM
- RFC 2681: A Round-trip Delay Metric for IPPM
- Series ended in 1999, with metrics clearly missing
  - Bandwidth especially, also jitter, packet order, packet corruption, …

Internet Flow Rates

CS514

- Zhang, Breslau, Paxson, Shenker
- Study of flow rates and sizes
- Particular interest in causes of different flow characteristics
- Traced flows at ISPs and campus access links
- Developed a tool (T-RAT) to analyze cause of rate limiting

  http://www.research.att.com/projects/T-RAT/

Subsequent slides taken from Sigcomm 2002 presentation

Internet Flow Rates Data Set

CS514

- Packet traces at ISP backbones and campus access links
  - 8 datasets; each lasts 0.5 – 24 hours; over 110 million packets
- Summary flow statistics collected at 19 backbone routers
  - 76 datasets; each lasts 24 hours; over 20 billion packets

Subsequent slides taken from Sigcomm 2002 presentation

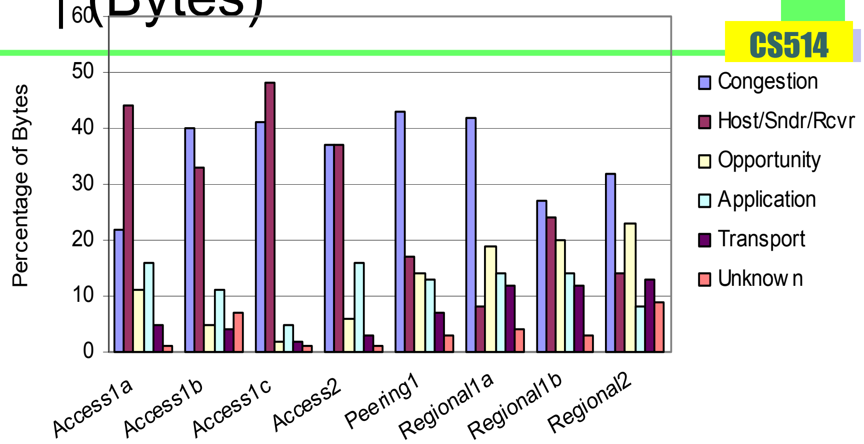
Flow Rate Characteristics

CS514

- Rate distribution
  - Most flows are *slow,* but most bytes are in *fast* flows
  - Distribution is skewed
    - Not as skewed as size distribution
    - Consistent with log-normal distribution [BSSK97]
- Correlations
  - Rate and size are strongly correlated
  - Not due to TCP slow-start
    - Removed initial 1 second of each connection; correlations increase
  - What users download is a function of their bandwidth

Rate Limiting Factors (Bytes)

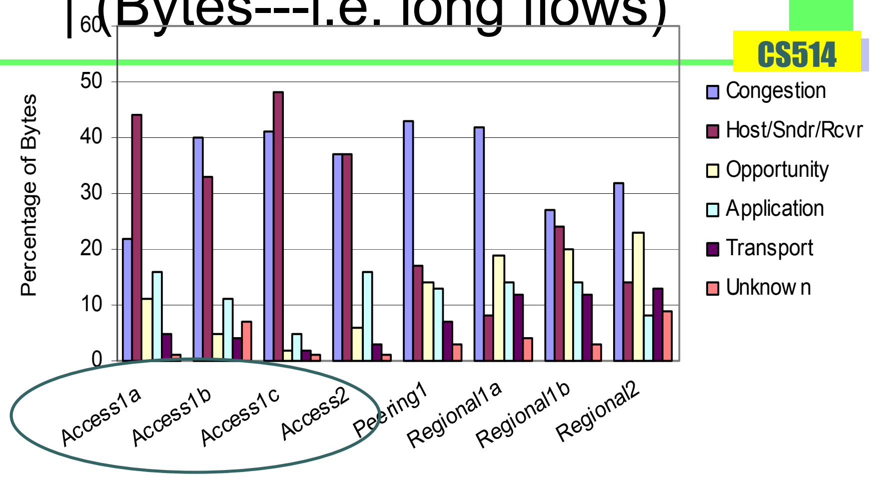
CS514



Dominant causes by bytes: Congestion, Receiver

Rate Limiting Factors (Bytes---i.e. long flows)

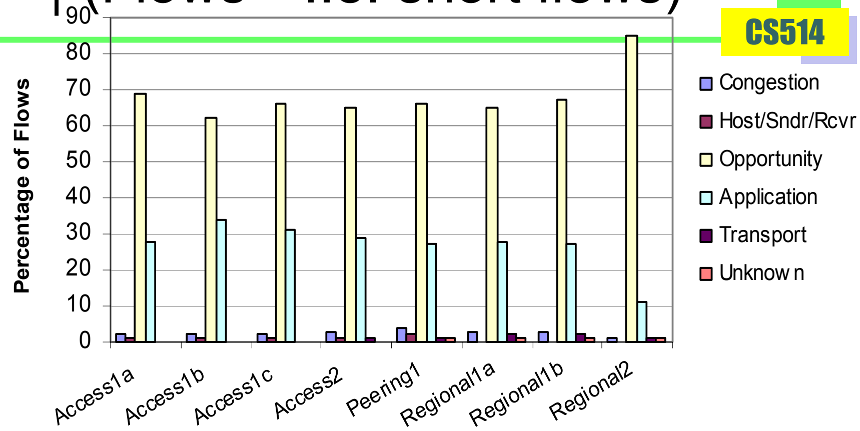
CS514



This population tends to have better internet connectivity, and so is more often receiver limited

Rate Limiting Factors (Flows---I.e. short flows)

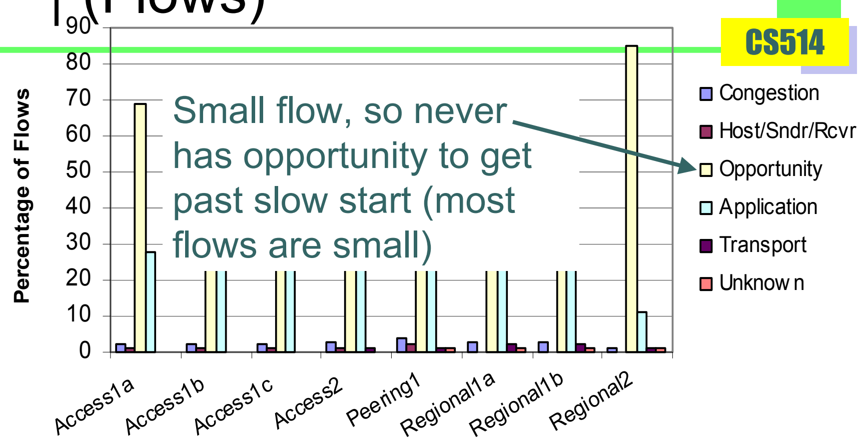
CS514



Dominant causes by flows: Opportunity, Application

Rate Limiting Factors (Flows)

CS514

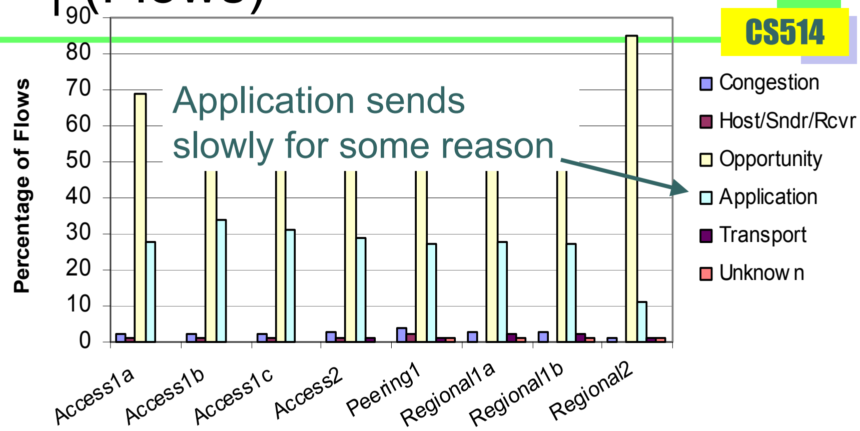


Small flow, so never has opportunity to get past slow start (most flows are small)

Dominant causes by flows: Opportunity, Application

Rate Limiting Factors (Flows)

CS514



Dominant causes by flows: Opportunity, Application

Flow Characteristics by Cause

CS514

- Different causes are associated with different performance for users
  - Rate distribution
    - Highest rates: Receiver, Transport
    - (Don't experience congestion)
  - Size distribution
    - Largest sizes: Receiver
    - (Size and rate a correlated)
  - Duration distribution
    - Longest duration: Congestion
    - (Congested flows take longer)

Some flow rate study conclusions (mine)

CS514

- Latency matters (most flows are small)
  - Slow start performance dominated by round trip time
- Congestion matters, but most congestion is at the edge
  - Recall that more congestion seen for users with thinner access pipes
  - Web service provider can't do anything about edge congestion

End-to-end effects of Internet Path Selection

CS514

- Savage et. al., Univ of Washington Seattle
- Compared path found by internet routing with alternates
  - Alternates composed by gluing together two internet-routed paths
- Roundtrip time, loss rate, bandwidth
- Data sets: Paxson, plus new ones from UW

Results

CS514

- Round-trip time
  - 30% - 55% of paths had better alternate paths
    - Mostly within 30ms
    - 10% had 50% or better latency
- Loss rate
  - 75% - 80% of paths had better alternate paths
  - 5% - 50% of paths had 5% or better drop rates

Results

CS514

- Bandwidth
  - 70% - 80% of paths have better alternates
  - 10% - 20% by a factor of three
- Results held for different times of day
- Results were not due to only a small number of hosts
- Shorter propagation delay and avoidance of congestion both contributed to better alternate paths

Some conclusions (mine)

CS514

- These results probably apply more to p2p than to popular web services
  - Web services well connected, so almost always a good path
- In cases where congestion was avoided, benefit derives from fact that few flows were going through alternate path
  - If many users took advantage of alternate paths, the alternate paths would no longer be better!
  - MIT RON, Sockeye (global routing service)

E2E WAN Service Availability

CS514

- Chandra et. al., U Texas at Austin
- Understand how network failures effect service availability
- Help web service designers make best use of available tools
- Evaluate likely value of techniques like:
  - replication of active objects
  - overlay objects

Data sets and limitations

- Traceroute data sets
  - Paxson NIMI data sets ('94-'95)
    - Measures middle of network but not the typical edge (NIMI probes well connected)
  - Savage UW data sets ('99)
    - All from Univ. Washington
- HTTP data sets from squid proxies
  - Not reflective of either typical connectivity or typical user
- Authors think they underestimate failure
  - Mainly because of flakeyness of many clients

Model derived from data sets

Parameter	Default Value	Comment
Rate	1.5% (all) 1.25% (>30s)	Varies from 0.4% to 7.4% in different data sets
Location	Src Mid Dst 25% 50% 25%	All locations significant (vaguely defined though)
Duration	avg. = 609 sec	Heavy tailed
Interarrival	avg. = 13 hr.	

Key Findings (but I don't trust them)

CS514

- Failure distributions are heavy-tailed
  - Long failures account for significant fraction of failure duration
- Data caching techniques will have little positive effect
  - Because still uncached content at failure time
- Prefetching and shipping mobile extension code to clients may have order-of-magnitude benefit
  - Prefetch everything before failure
  - Mobile extension: ship code and data to client---not yet practical

What sounds right to me

CS514

- Engineer web services for reliability
- Connect web services to many ISPs to minimize effects of routing failures in the middle
  - On the theory that fewer AS hops means fewer chances for failure
- There isn't much web service can do about failures near the client

Conclusions

CS514

- Network performance varies tremendously
  - WAN, MAN, LAN . . .
  - Even in different WAN settings, performance varies
- You'll need to either make your own measurements, or understand how to interpret those claimed by the ISP
  - Either way, the issues are subtle