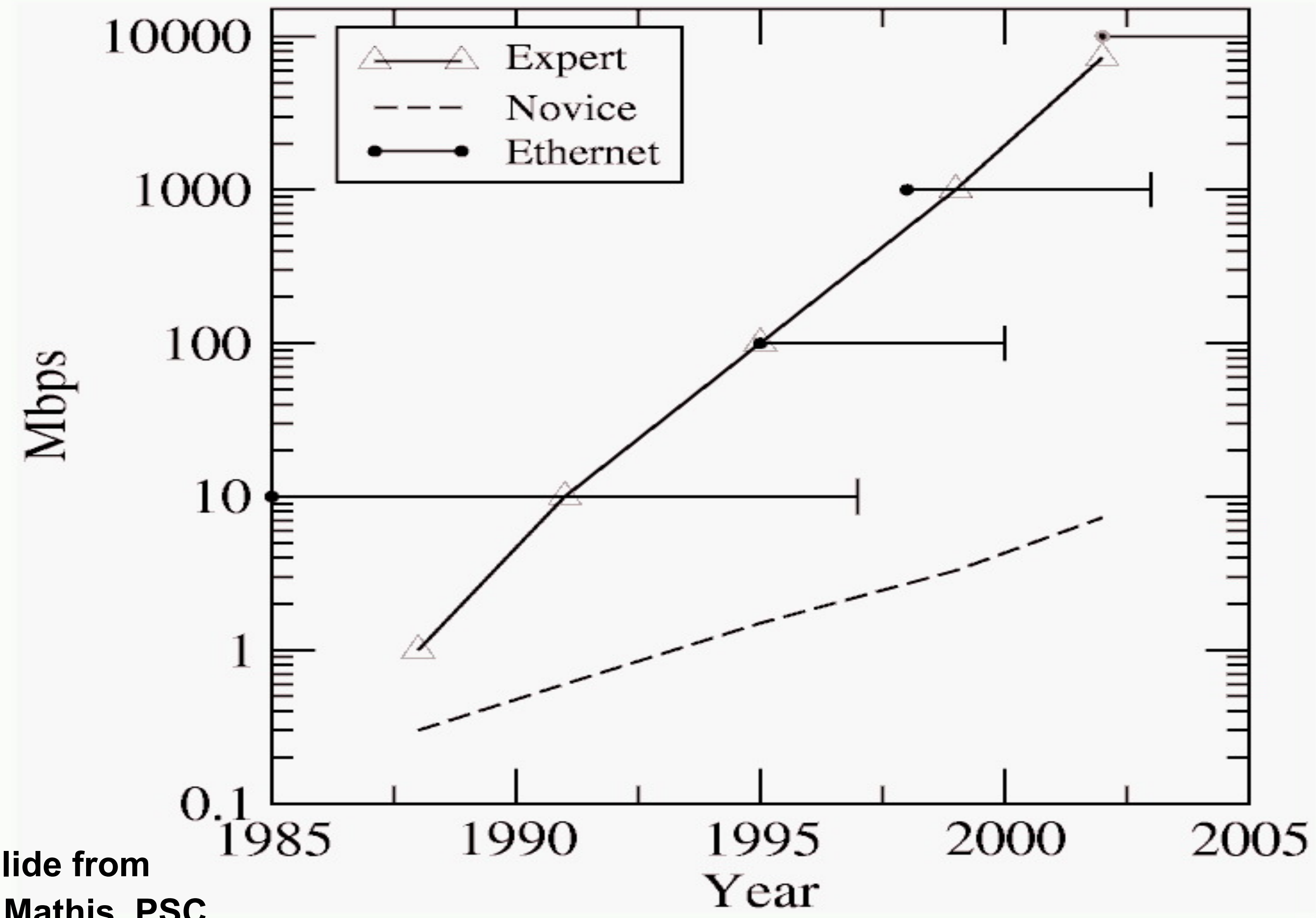
High Performance Bulk Data Transfer

Brian Tierney and Joe Metzger, ESnet

Joint Techs, Columbus OH, July, 2010

Why does the network seem so slow?

Wizard Gap



Slide from Matt Mathis, PSC

Today’s Talk

This talk will cover:

- Some Information to help you become a “wizard”
- Work being done so you don’t have to be a wizard

Goal of this talk:

- Help you fully optimize wide area bulk data transfers
  - or help your users do this

Outline

- TCP Issues
- Bulk Data Transfer Tools
- Router Issues
- Network Monitoring/Troubleshooting Tools

Time to Copy 1 Terabyte

10 Mbps network : 300 hrs (12.5 days)

100 Mbps network : 30 hrs

1 Gbps network : 3 hrs (are your disks fast enough?)

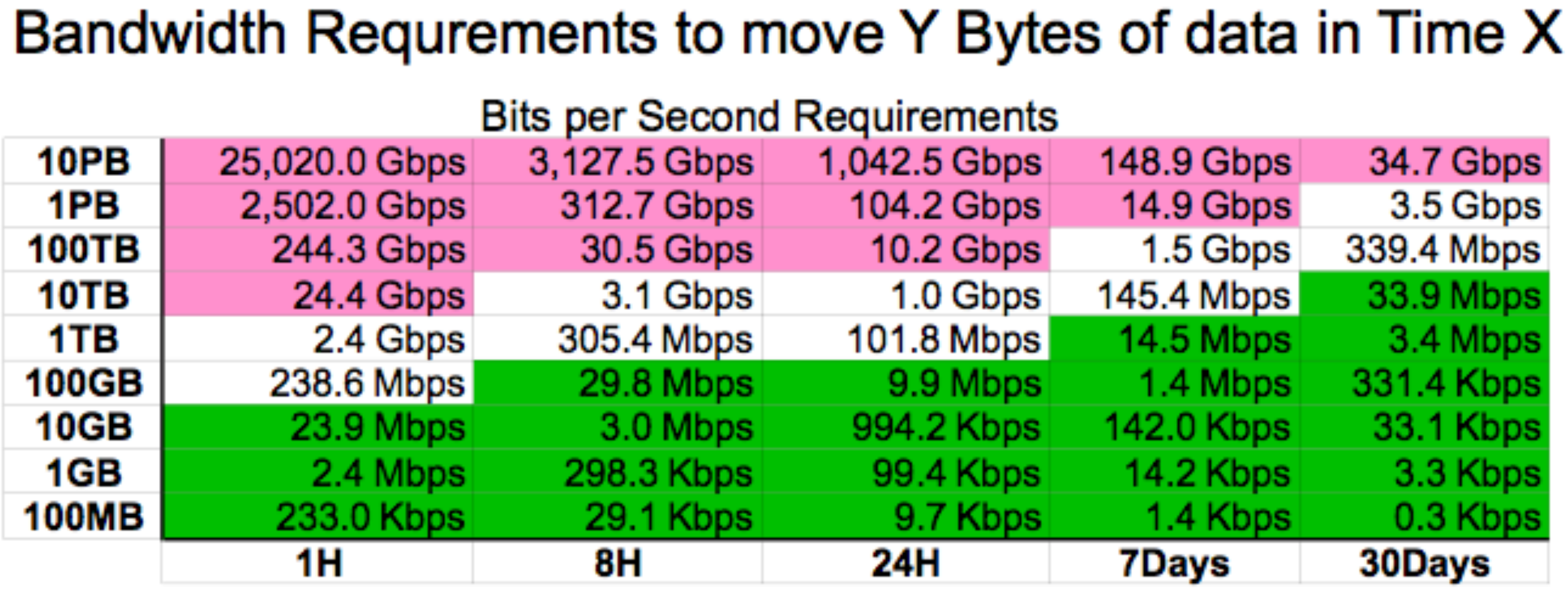
10 Gbps network : 20 minutes (need really fast disks and filesystem)

These figures assume some headroom left for other users

Compare these speeds to:

- USB 2.0 portable disk
  - 60 MB/sec (480 Mbps) peak
  - 20 MB/sec (160 Mbps) reported on line
  - 5-10 MB/sec reported by colleagues
  - 15-40 hours to load 1 Terabyte

Bandwidth Requirements



Bandwidth Requrements to move Y Bytes of data in Time X

Bits per Second Requirements

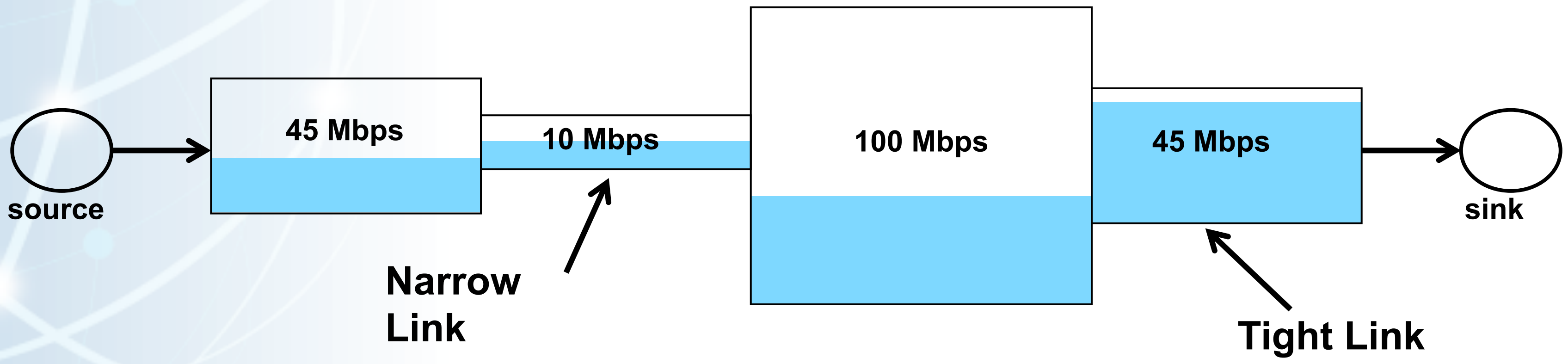
10PB	25,020.0 Gbps	3,127.5 Gbps	1,042.5 Gbps	148.9 Gbps	34.7 Gbps
1PB	2,502.0 Gbps	312.7 Gbps	104.2 Gbps	14.9 Gbps	3.5 Gbps
100TB	244.3 Gbps	30.5 Gbps	10.2 Gbps	1.5 Gbps	339.4 Mbps
10TB	24.4 Gbps	3.1 Gbps	1.0 Gbps	145.4 Mbps	33.9 Mbps
1TB	2.4 Gbps	305.4 Mbps	101.8 Mbps	14.5 Mbps	3.4 Mbps
100GB	238.6 Mbps	29.8 Mbps	9.9 Mbps	1.4 Mbps	331.4 Kbps
10GB	23.9 Mbps	3.0 Mbps	994.2 Kbps	142.0 Kbps	33.1 Kbps
1GB	2.4 Mbps	298.3 Kbps	99.4 Kbps	14.2 Kbps	3.3 Kbps
100MB	233.0 Kbps	29.1 Kbps	9.7 Kbps	1.4 Kbps	0.3 Kbps
	1H	8H	24H	7Days	30Days

This table available at http://fasterdata.es.net

Throughput? Bandwidth? What?

The term “throughput” is vague

- Capacity: link speed
  - Narrow Link: link with the lowest capacity along a path
  - Capacity of the end-to-end path = capacity of the narrow link
- Utilized bandwidth: current traffic load
- Available bandwidth: capacity – utilized bandwidth
  - Tight Link: link with the least available bandwidth in a path
- Achievable bandwidth: includes protocol and host issues



(Shaded portion shows background traffic)

More Terminology

Latency: time to send 1 packet from the source to the destination

RTT: Round-trip time

Bandwidth*Delay Product = BDP

- The number of bytes in flight to fill the entire path
- Example: 100 Mbps path; ping shows a 75 ms RTT
  - BDP = 100 * 0.075 = 7.5 Mbits (940 KBytes)

LFN: Long Fat Networks

- A network with a large BDP

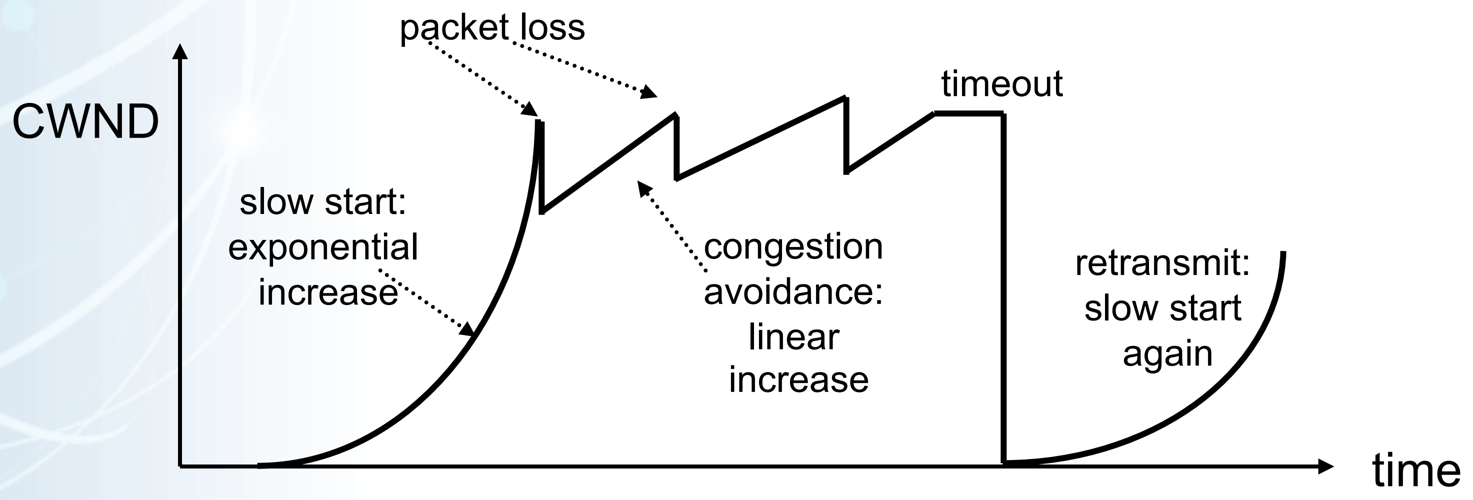
How TCP works: A very short overview

Congestion window (CWND) = the number of packets the sender is allowed to send

- The larger the window size, the higher the throughput
  - Throughput = Window size / Round-trip Time

TCP Slow start

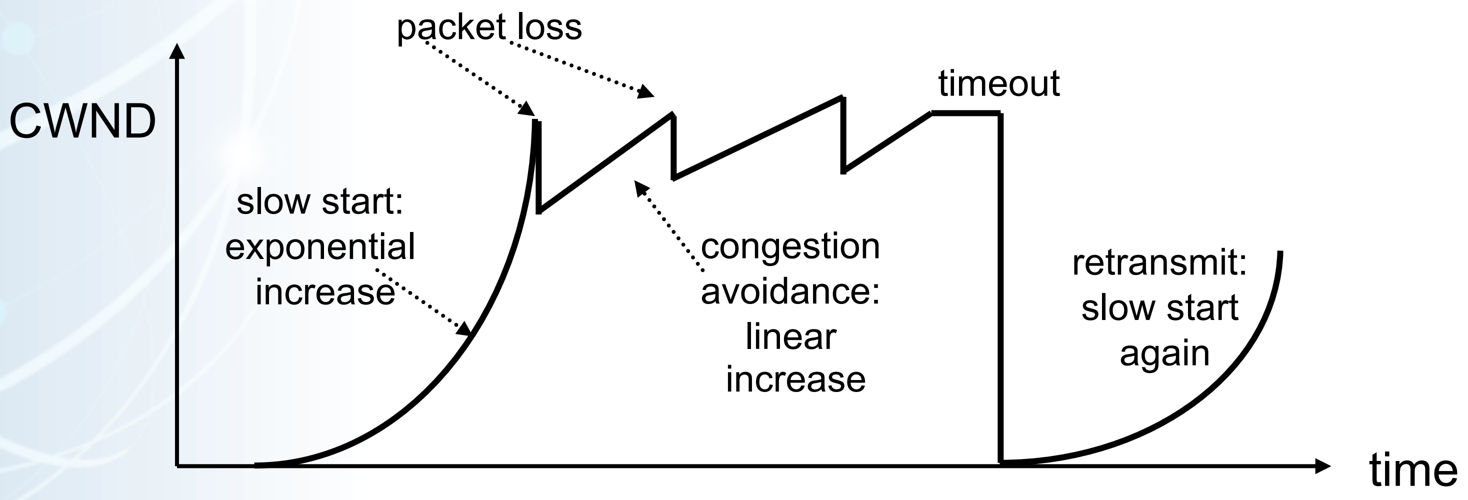
- exponentially increase the congestion window size until a packet is lost
  - this gets a rough estimate of the optimal congestion window size


packet loss
timeout
CWND
slow start: exponential increase
congestion avoidance: linear increase
retransmit: slow start again
time

TCP Overview

Congestion avoidance

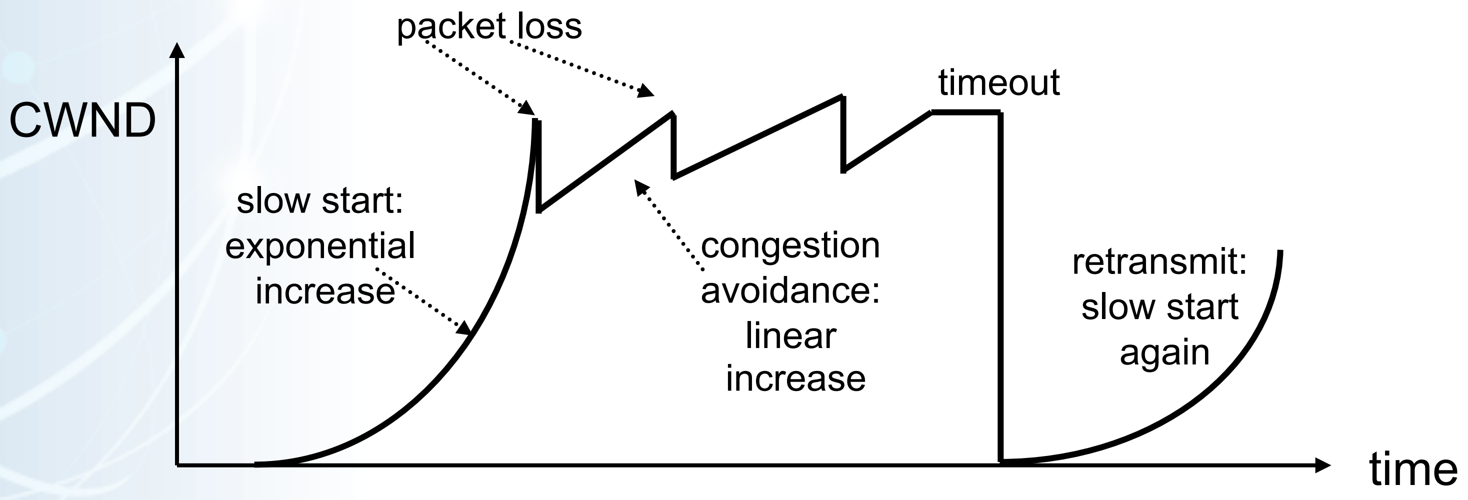
- additive increase: starting from the rough estimate, linearly increase the congestion window size to probe for additional available bandwidth
- multiplicative decrease: cut congestion window size aggressively if a timeout occurs



packet loss
timeout
CWND
slow start: exponential increase
congestion avoidance: linear increase
retransmit: slow start again
time

TCP Overview

- Fast Retransmit: retransmit after 3 duplicate acks (got 3 additional packets without getting the one you are waiting for)
  - this prevents expensive timeouts
  - no need to go into “slow start” again
- At steady state, CWND oscillates around the optimal window size
- With a retransmission timeout, slow start is triggered again



TCP Tuning

Host Tuning – TCP

TCP tuning commonly refers to the proper configuration of buffers that correspond to TCP windowing

Historically TCP tuning parameters were host-global, with exceptions configured per-socket by applications

- Applications had to understand the network in detail, and know how far away clients were
- Some applications did this – most did not

Solution: auto-tune TCP connections within pre-configured limits

Setting the TCP buffer sizes

It is critical to use the optimal TCP send and receive socket buffer sizes for the link you are using.

- Recommended size to fill the pipe
  - 2 x Bandwidth Delay Product (BDP)
- Recommended size to leave some bandwidth for others
  - around 20% of (2 x BPB) = .4 * BDP

Default TCP buffer sizes can be way too small for today's high speed networks

- Until recently, default TCP send/receive buffers were typically 64 KB
  - Only way to change this was for the application to call the 'setsockopt()' system call
- tuned buffer to fill CA to NY link: 10 MB
  - 150X bigger than the default buffer size

Buffer Size Example

Optimal Buffer size formula:

- buffer size = 20% * (bandwidth * RTT)
- (assuming your target is 20% of the narrow link)

Example:

- ping time (RTT) = 50 ms
- Narrow link = 1000 Mbps (125 MBytes/sec)
- TCP buffers should be:
  - .05 sec * 125 * 20% = 1.25 MBytes

Socket Buffer Autotuning

To solve the buffer tuning problem, based on work at LANL and PSC, Linux OS added TCP Buffer autotuning

- Sender-side TCP buffer autotuning introduced in Linux 2.4
- Receiver-side autotuning added in Linux 2.6

Most OS's now include TCP autotuning

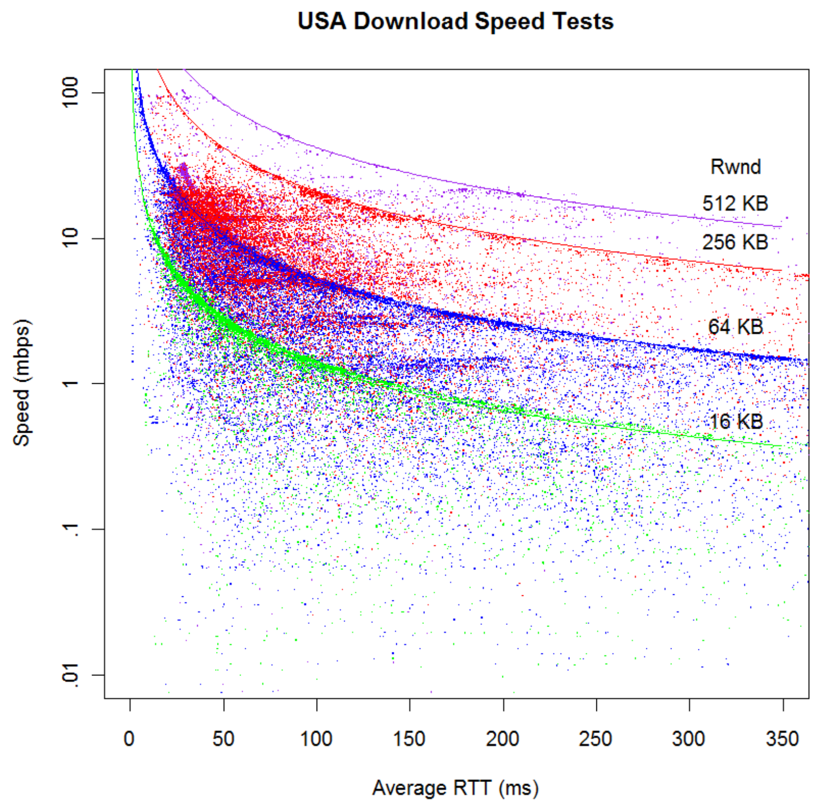
- TCP send buffer starts at 64 KB
- As the data transfer takes place, the buffer size is continuously re-adjusted up max autotune size

Current OS Autotuning default maximum buffers

- Linux 2.6: 256K to 4MB, depending on version
- Windows Vista: 16M
- Mac OSX 10.5-10.6: 4M
- FreeBSD 7: 256K

Huge number of hosts are still receive window limited

- Results from M-Lab NDT tests, Feb-Sept, 2009 (over 400K tests)
- 34% of tests had no congestion events
  - i.e.: limited by RWIN, and fixable via buffer tuning
- See white paper “Understanding broadband speed measurements”, by Steve Bauer, David Clark, William Lehr, Massachusetts Institute of Technology” (URL)
  - http://mitas.csail.mit.edu/papers/Bauer_Clark_Lehr_Broadband_Speed_Measurements.pdf



Autotuning Settings: http://fasterdata.es.net/TCP-Tuning/

Linux 2.6: add to /etc/sysctl.conf

```
net.core.rmem_max = 16777216
net.core.wmem_max = 16777216
# autotuning min, default, and max number of bytes to use
net.ipv4.tcp_rmem = 4096 87380 16777216
net.ipv4.tcp_wmem = 4096 65536 16777216
```

FreeBSD 7.0+: add to /etc/sysctl.conf

```
net.inet.tcp.sendbuf_max=16777216
net.inet.tcp.recvbuf_max=16777216
```

Mac OSX: add to /etc/sysctl.conf

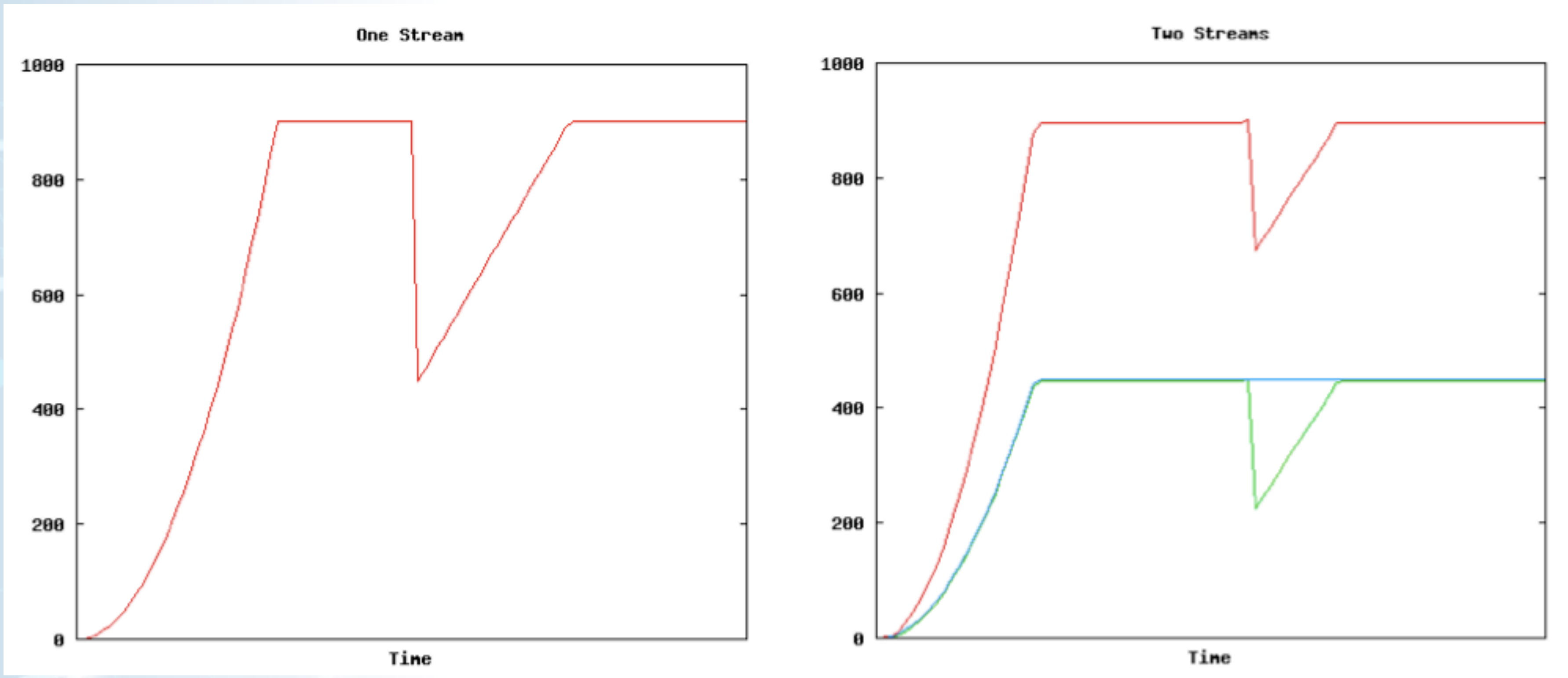
```
kern.ipc.maxsockbuf=16777216
net.inet.tcp.sendspace=8388608
net.inet.tcp.recvspace=8388608
```

Windows XP

- use “DrTCP” (http://www.dslreports.com/drtcp/) to modify registry settings to increase TCP buffers

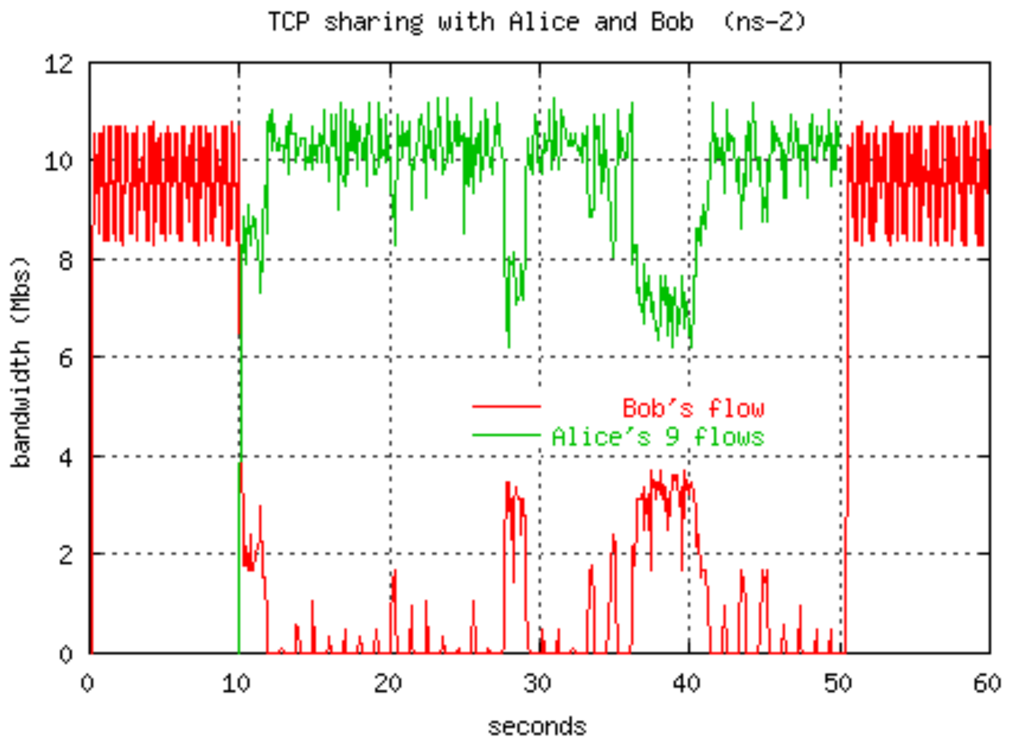
Windows Vista/Windows 7: autotunes by default, 16M Buffers

Parallel Streams Can Help



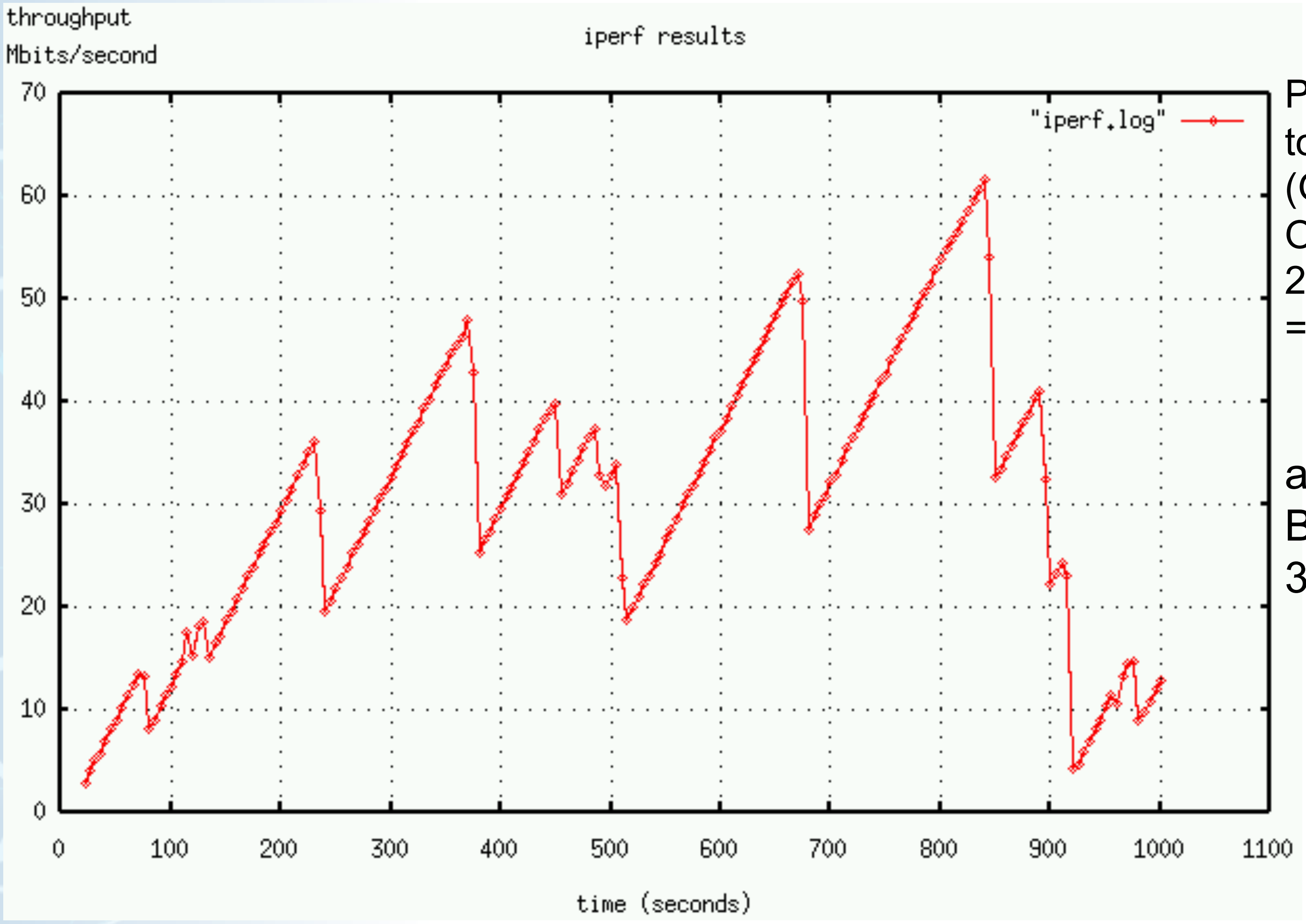
Parallel Streams Issues

- Potentially unfair
- Places more load on the end hosts
- But they are necessary when you don't have root access, and can't convince the sysadmin to increase the max TCP buffers
- Most web browsers will open around 4 parallel streams by default



graph from Tom Dunigan, ORNL

Another Issue: TCP congestion control



Path = LBL to CERN (Geneva) OC-3, (in 2000), RTT = 150 ms

average BW = 30 Mbps

TCP Response Function

Well known fact that TCP Reno does not scale to high-speed networks

Average TCP congestion window = $1.2/\sqrt{p}$ segments

- p = packet loss rate
- see: http://www.icir.org/floyd/hstcp.html

What this means:

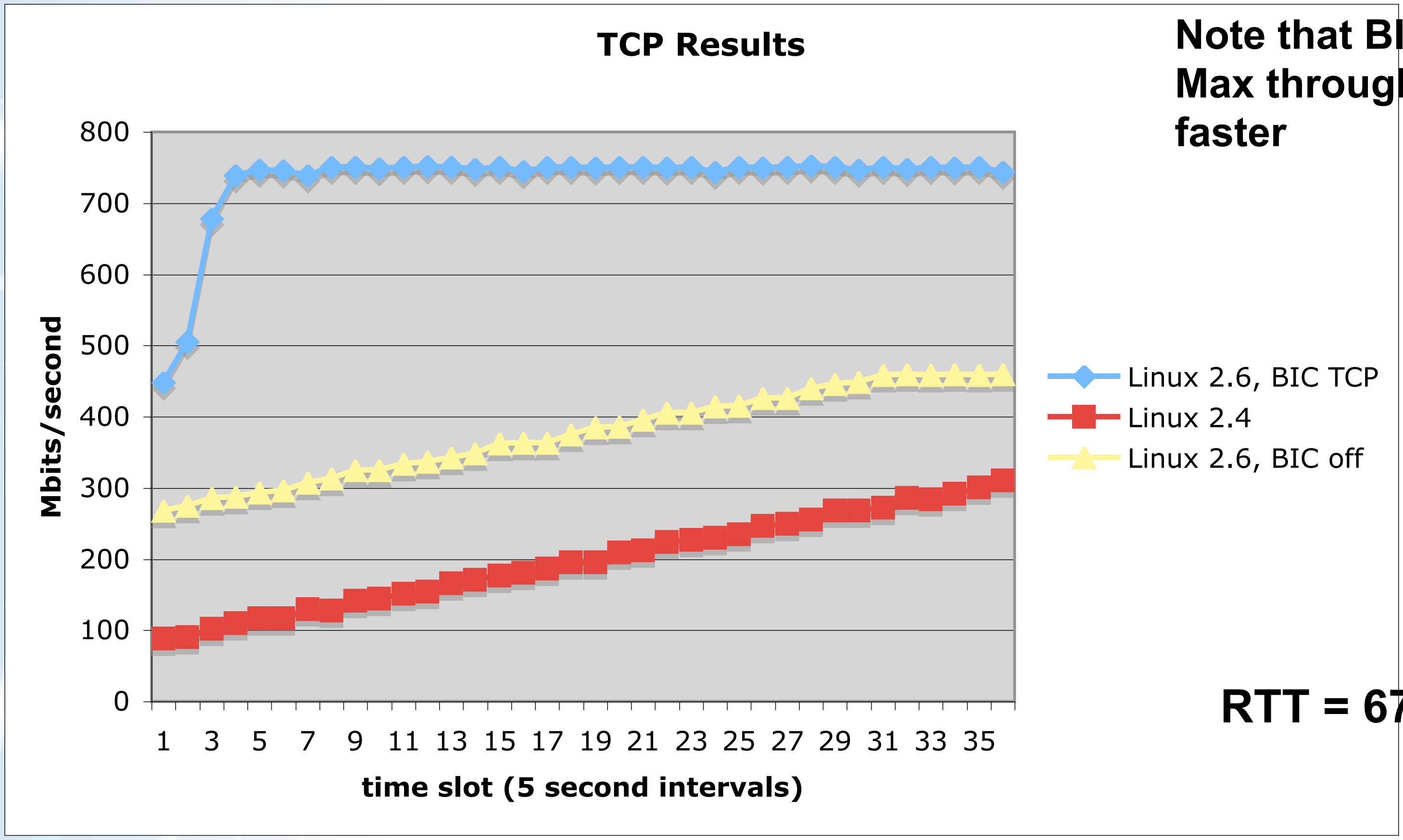
- For a TCP connection with 1500-byte packets and a 100 ms round-trip time
  - filling a 10 Gbps pipe would require a congestion window of 83,333 packets,
  - a packet drop rate of at most one drop every 5,000,000,000 packets.
- requires at most one packet loss every 6000s, or 1h:40m to keep the pipe full

Proposed TCP Modifications

Many proposed alternate congestion control algorithms:

- BIC/CUBIC
- HTCP: (Hamilton TCP)
- Scalable TCP
- Compound TCP
- And several more

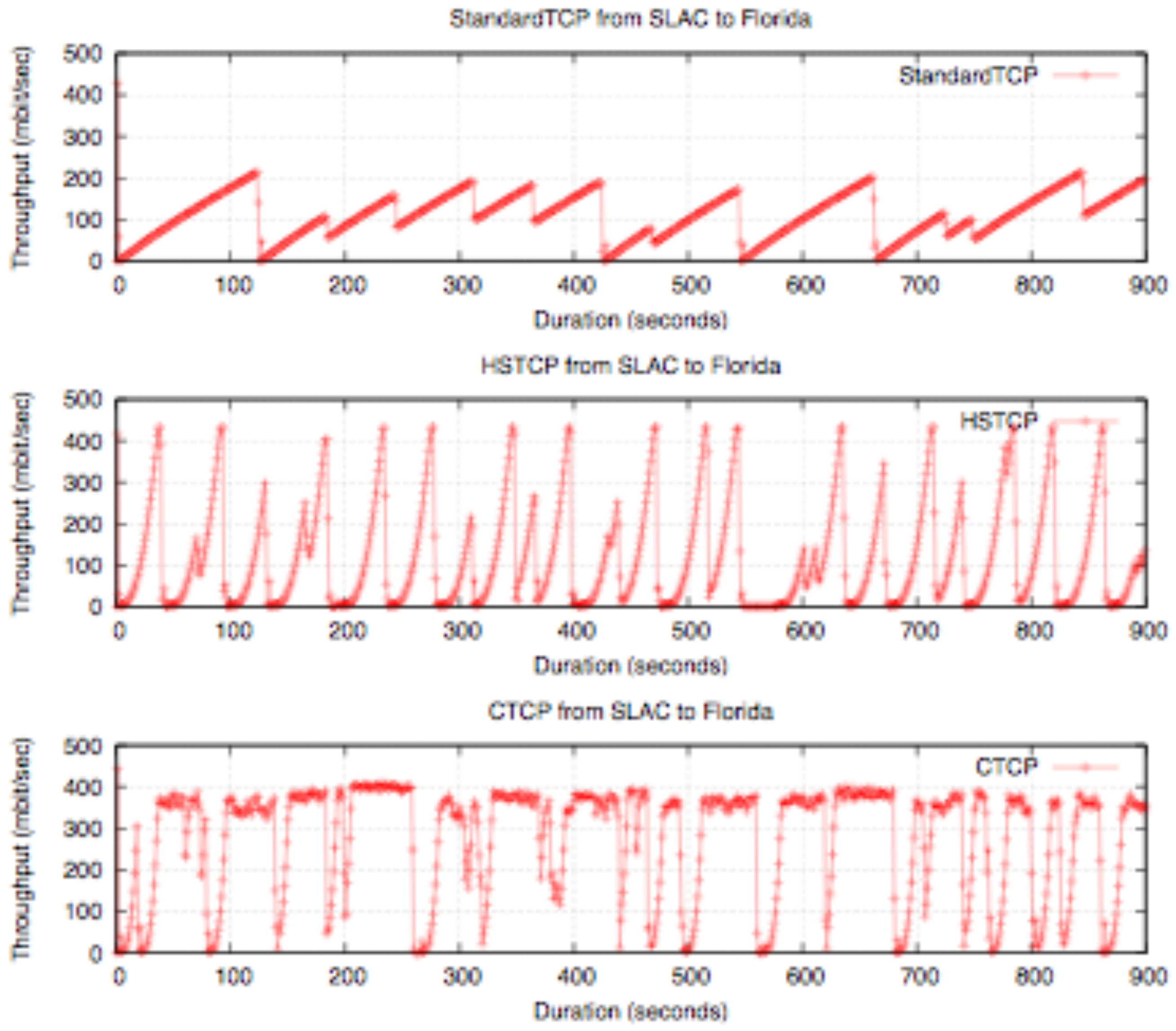
Linux 2.6.12 Results



Note that BIC reaches Max throughput MUCH faster

RTT = 67 ms

Comparison of Various TCP Congestion Control Algorithms



Congestion Algorithms In various Linux Distributions

Linux Distribution	Available Algorithms	Default Algorithm
Centos 5.2 – 5.5	bic, reno	bic
Debian 5.0.3	bic, reno	bic
Debian Unstable (future 6?)	cubic, reno	cubic
Fedora 10 - 12	cubic, reno	cubic
Redhat 5.4	bic, reno	bic
Ubuntu 8.0	reno	reno
Ubuntu 8.10	cubic, reno	cubic
Ubuntu 9.04	cubic, reno	cubic
Ubuntu 9.10	cubic, reno	cubic

Note: CUBIC and BIC both from same group, and they recommend CUBIC because its slightly more fair to competing traffic.

Selecting TCP Congestion Control in Linux

To determine current configuration:

- sysctl -a | grep congestion
- net.ipv4.tcp_congestion_control = cubic
- net.ipv4.tcp_available_congestion_control = cubic reno

Use /etc/sysctl.conf to set to any available congested congestion control.

Supported options (may need to enabled by default in your kernel):

- CUBIC, BIC, HTCP, HSTCP, STCP, LTCP, more..
- E.g.: Centos 5.5 includes these:
  - CUBIC, HSTCP, HTCP, HYBLA, STCP, VEGAS, VENO, Westwood

Use modprobe to add:

- /sbin/modprobe tcp_htcp
- /sbin/modprobe tcp_cubic

Additional Host Tuning for Linux

Linux by default caches ssthresh, so one transfer with lots of congestion will throttle future transfers. To turn that off set:

```
net.ipv4.tcp_no_metrics_save = 1
```

Also should change this for 10GE

```
net.core.netdev_max_backlog = 250000
```

Warning on Large MTUs:

- If you have configured your Linux host to use 9K MTUs, but the MTU discovery reduces this to 1500 byte packets, then you actually need 9/1.5 = 6 times more buffer space in order to fill the pipe.
  - Some device drivers only allocate memory in power of two sizes, so you may even need 16/1.5 = 11 times more buffer space!

NIC Tuning (See: http://fasterdata.es.net)

Defaults are usually fine for 1GE, but 10GE often requires additional tuning

Myricom 10Gig NIC

- The Myricom NIC provides a LOTS of tuning knobs. For more information see links on fasterdata
- In particular, you might benefit from increasing the interrupt coalescing timer: ethtool -C interface rx-usecs 100
- Also consider using the “Throttle” option

Chelsio 10Gig NIC

- TCP Segmentation Offloading (TSO) and TCP Offload Engine (TOE) on the Chelsio NIC radically hurt performance on a WAN (they do help reduce CPU load without affecting throughput on a LAN).
- To turn off TSO do this: ethtool -K interface tso off

Other OS tuning knobs

Mac OSX: http://fasterdata.es.net/TCP-tuning/MacOSX.html

```
sysctl -w net.inet.tcp.win_scale_factor=8
```

Windows Vista/7: http://fasterdata.es.net/TCP-tuning/Windows-Vista.html

- TCP autotuning included, 16M buffers
- Use Compound TCP:

```
netsh interface tcp set global congestionprovider=ctcp"
```

Summary so far

To optimize TCP throughput, do the following:

- Use a newer OS that supports TCP buffer autotuning
- Increase the maximum TCP autotuning buffer size
- Use a few parallel streams if possible
- Use a modern congestion control algorithm

Bulk Data Transfer Tools

Data Transfer Tools

Parallelism is key

- It is much easier to achieve a given performance level with four parallel connections than one connection
- Several tools offer parallel transfers

Latency interaction is critical

- Wide area data transfers have much higher latency than LAN transfers
- Many tools and protocols assume a LAN
- Examples: SCP/SFTP, HPSS mover protocol

Sample Data Transfer Results:
Berkeley CA to Brookhaven, NY, 1 Gbps path

Using the right tool is very important

- scp / sftp : 10 Mbps
  - standard Unix file copy tools
  - fixed 1 MB TCP window in openSSH
    - only 64 KB in openssh versions < 4.7
- ftp : 400-500 Mbps
  - assumes TCP buffer autotuning
- parallel stream FTP: 800-900 Mbps

Why Not Use SCP or SFTP?

Pros:

- Most scientific systems are accessed via OpenSSH
- SCP/SFTP are therefore installed by default
- Modern CPUs encrypt and decrypt well enough for small to medium scale transfers
- Credentials for system access and credentials for data transfer are the same

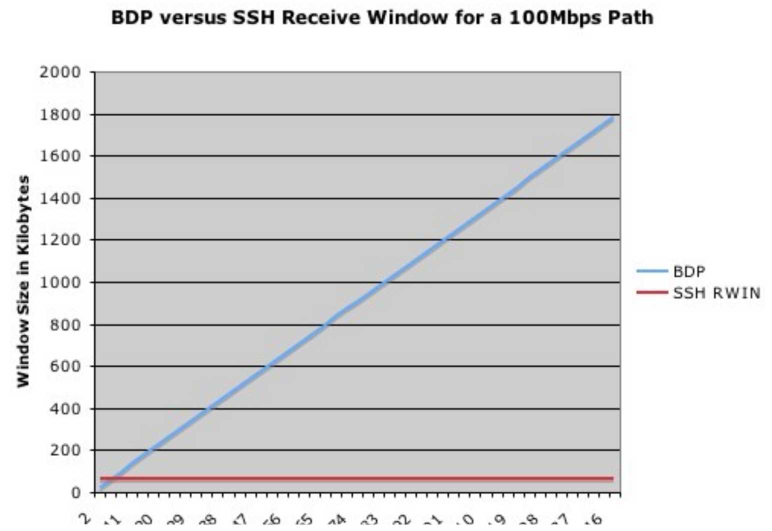
Cons:

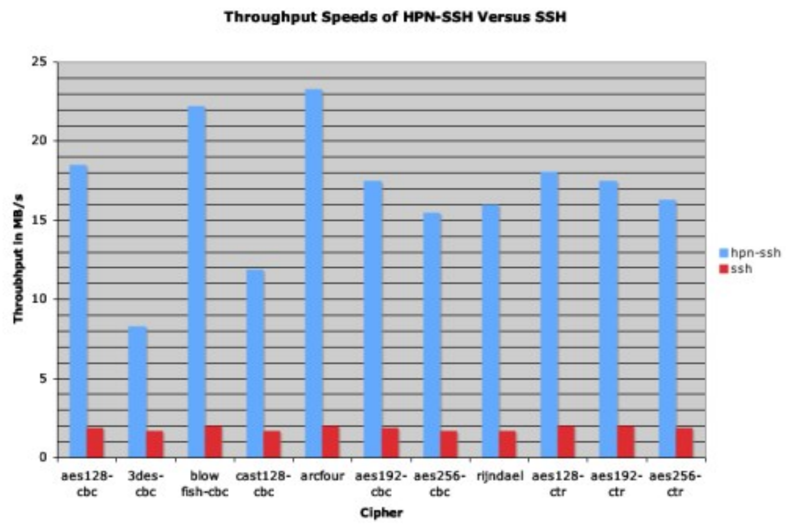
- The protocol used by SCP/SFTP has a fundamental flaw that limits WAN performance
- CPU speed doesn't matter – latency matters
- Fixed-size buffers reduce performance as latency increases
- It doesn't matter how easy it is to use SCP and SFTP – they simply do not perform

Verdict: Do Not Use Without Performance Patches

A Fix For scp/sftp

- PSC has a patch set that fixes problems with SSH
  - http://www.psc.edu/networking/projects/hpn-ssh/
- Significant performance increase
- Advantage – this helps rsync too





sftp

Uses same code as scp, so don't use sftp WAN transfers unless you have installed the HPN patch from PSC

But even with the patch, SFTP has yet another flow control mechanism

- By default, sftp limits the total number of outstanding messages to 16 32KB messages.
- Since each datagram is a distinct message you end up with a 512KB outstanding data limit.
- You can increase both the number of outstanding messages ('-R') and the size of the message ('-B') from the command line though.

Sample command for a 128MB window:

- sftp -R 512 -B 262144 user@host:/path/to/file outfile

GridFTP

GridFTP from ANL has everything needed to fill the network pipe

- Buffer Tuning
- Parallel Streams

Supports multiple authentication options

- anonymous
- ssh (available in starting with Globus Toolkit version 4.2)
- X509

Ability to define a range of data ports

- helpful to get through firewalls

Sample Use:

- globus-url-copy -p 4 sshftp://data.lbl.gov/home/mydata/myfile
  file://home/mydir/myfile

Available from: http://www.globus.org/toolkit/downloads/

newer GridFTP Features

ssh authentication option

- Not all users need or want to deal with X.509 certificates
- Solution: Use SSH for Control Channel
  - Data channel remains as is, so performance is the same
- see http://fasterdata.es.net/gridftp.html for a quick start guide

Optimizations for small files

- Concurrency option (-cc)
  - establishes multiple control channel connections and transfer multiple files simultaneously.
- Pipelining option:
  - Client sends next request before the current completes
- Cached Data channel connections
  - Reuse established data channels (Mode E)
  - No additional TCP or GSI connect overhead

Support for UDT protocol

Globus.org: http://globus.org/service/

The latest iteration of Globus software

- The same Globus vision, but an updated approach

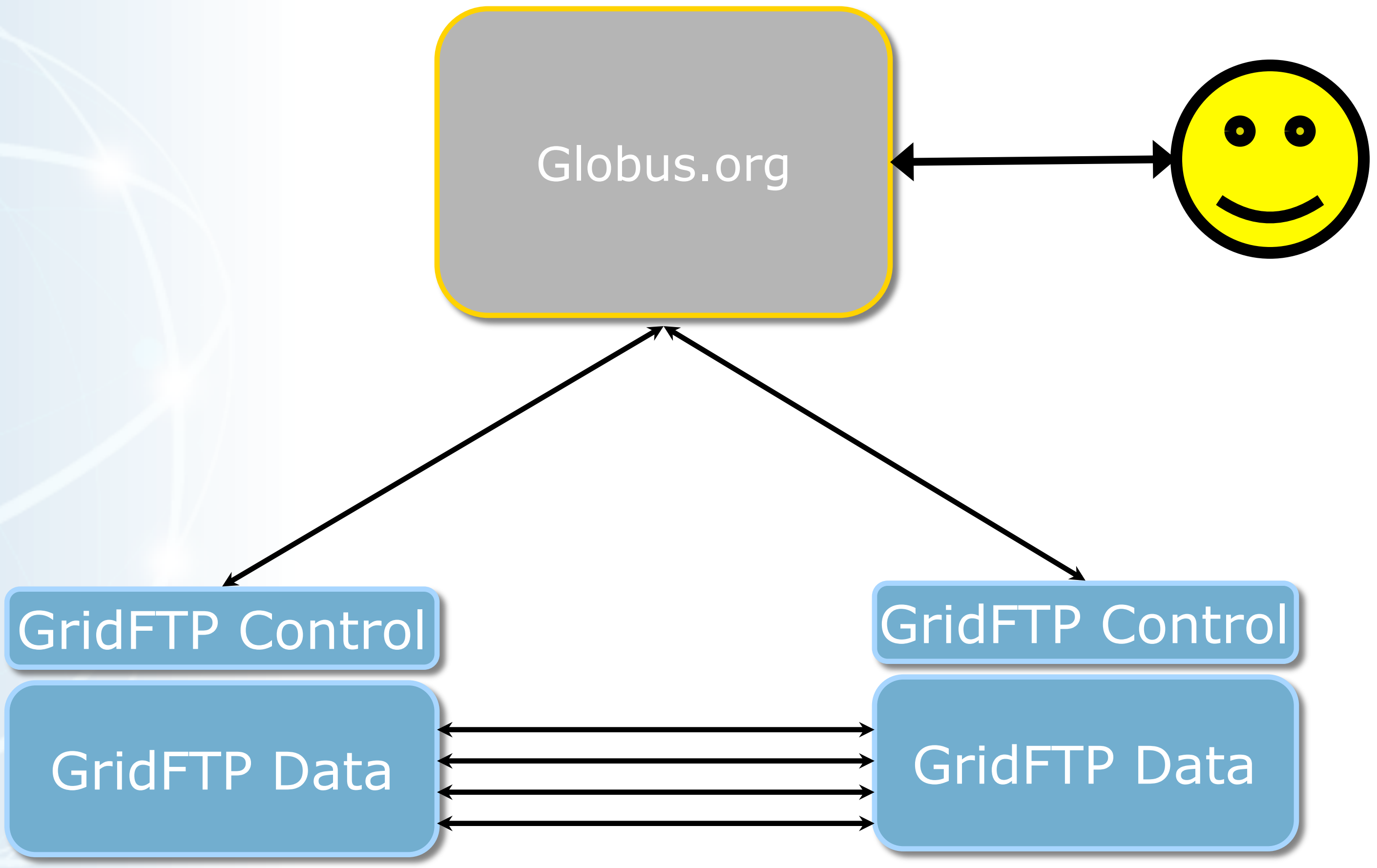
Hosted services

- Data movement initially
- Execution , information, and VO services to follow

Goals:

- Provide scientists with easy access to advanced computing resources
  - Technology interactions that require no special expertise
  - No software to install
- Enable users to focus on domain-specific work
  - Manage technology failures
  - Notifications of interesting events
  - Provide users with enough information to efficiently resolve problems

Globus.org Manages 3rd-Party Transfers



Other Tools

bbcp: http://www.slac.stanford.edu/~abh/bbcp/

- supports parallel transfers and socket tuning
- bbcp -P 4 -v -w 2M myfile remotehost:filename

lftp: http://lftp.yar.ru/

- parallel file transfer, socket tuning, HTTP transfers, and more.
- lftp -e 'set net:socket-buffer 4000000; pget -n 4 [http|ftp]://site/path/file; quit'

axel: http://axel.alioth.debian.org/

- simple parallel accelerator for HTTP and FTP.
- axel -n 4 [http|ftp]://site/file

Download Managers

There are a number of nice browser plugins that can be used to speed up web-initiated data transfers

- all support parallel transfers

Firefox add-on (All OSes):

- DownThemAll : http://www.downthemall.net
- this is my favorite: probably the best/simplest solution

For Linux:

- aria: http://aria-rpm.sourceforge.net

For Windows:

- FDM: http://www.freedownloadmanager.org

For OSX:

- Speed Download: http://www.yazsoft.com/products/ ($25)

FTP Tool: Filezilla

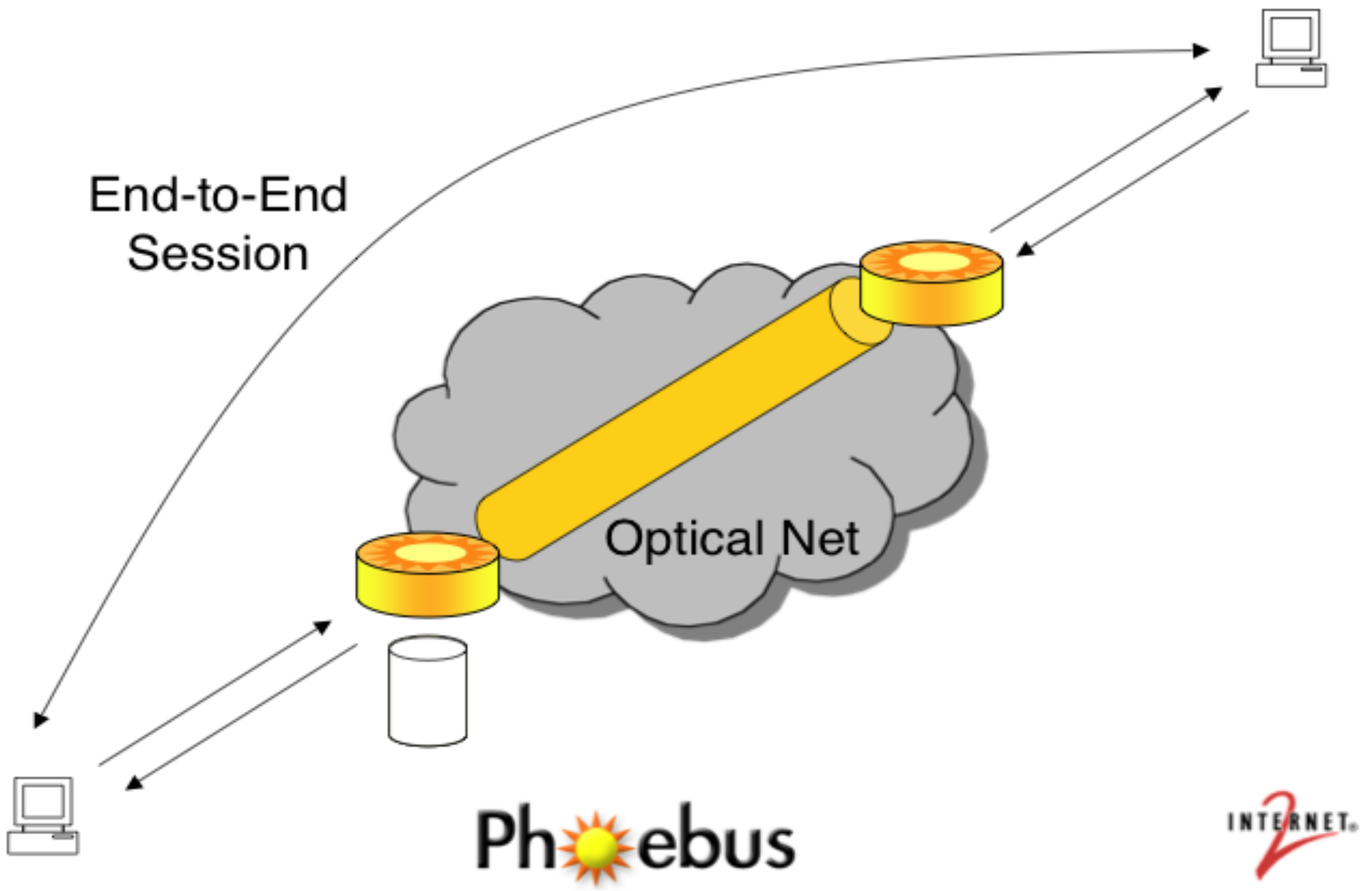
Open Source:

- http://filezilla-project.org/
- includes client and server
- Works on Windows, OSX, Linux

Features:

- ability to transfer multiple files in parallel

Phoebus: http://e2epi.internet2.edu/phoebus.html



Other Bulk Data Transfer Issues

Firewalls

- many firewalls can't handle 1 Gbps flows
  - designed for large number of low bandwidth flow
  - some firewalls even strip out TCP options that allow for TCP buffers > 64 KB

Disk Performance

- In general need a RAID array to get more than around 500 Mbps

Network troubleshooting tools

Iperf

iperf : nice tool for measuring end-to-end TCP/UDP performance

- http://sourceforge.net/projects/iperf
- Can be quite intrusive to the network in UDP mode

Server (receiver): `iperf -s`

```
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 10.0.1.5 port 5001 connected with 10.0.1.10 port 60830
[  4]  0.0-10.0 sec  1.09 GBytes    933 Mbits/sec
[  4] local 10.0.1.5 port 5001 connected with 10.0.1.10 port 60831
[  4]  0.0-10.0 sec  1.08 GBytes    931 Mbits/sec
```

Client (sender): `iperf -c 10.0.1.5`

```
------------------------------------------------------------
Client connecting to 10.0.1.5, TCP port 5001
TCP window size:  129 KByte (default)
------------------------------------------------------------
[  3] local 10.0.1.10 port 60830 connected with 10.0.1.5 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.2 sec  1.09 GBytes   913 Mbits/sec
```

Iperf3 (http://code.google.com/p/iperf/)

Iperf3 is a complete re-write of iperf with the following goals

- Easier to maintain (simple C code instead of C++)
- Built on a library that other programs can call
  - Multiple language binding via SWIG (eventually)
  - Easy to parse output format (name=value pairs)
    - Old format still supported too
- Separate control channel
- Current Status: TCP only, UDP coming soon,

  - Jon Dugan, ESnet, is the lead developer
  - Looking for volunteers to help test (and contribute to the code!)

bwctl (http://e2epi.internet2.edu/bwctl/)

Bwctl is a wrapper for tools like iperf and nuttcp that provides:

- Allows only 1 test at a time
- Controls who can run tests and for how long
- Controls maximum UDP bandwidth

Sample client command:

- bwctl -c hostname –t 20 -f m -i 2 –x

```
bwctl: exec_line: /usr/local/bin/iperf -c 198.124.252.109 -B 198.129.254.62 -f m -m -p 5046 -t 20 -i 2
[  9] local 198.129.254.62 port 5046 connected with 198.124.252.109 port 5046
[ ID] Interval       Transfer     Bandwidth
[  9]  0.0- 2.0 sec    423 MBytes  1773 Mbits/sec
[  9]  2.0- 4.0 sec    589 MBytes  2471 Mbits/sec
…
[  9] 18.0-20.0 sec    755 MBytes  3166 Mbits/sec
[  9]  0.0-20.0 sec  7191 MBytes  3016 Mbits/sec
[  9] MSS size 8192 bytes (MTU 8232 bytes, unknown interface)
```

Using bwctl

Sample results

- bwctl -c hostname –t 20 -f m -i 2 –x

```
bwctl: exec_line: /usr/local/bin/iperf -c 198.124.252.109 -B 198.129.254.62 -f m -m -p
    5046 -t 20 -i 2
[  9] local 198.129.254.62 port 5046 connected with 198.124.252.109 port 5046
[ ID] Interval       Transfer     Bandwidth
[  9]  0.0- 2.0 sec    423 MBytes  1773 Mbits/sec
[  9]  2.0- 4.0 sec    589 MBytes  2471 Mbits/sec
[  9]  4.0- 6.0 sec    709 MBytes  2973 Mbits/sec
…
[  9] 16.0-18.0 sec    754 MBytes  3162 Mbits/sec
[  9] 18.0-20.0 sec    755 MBytes  3166 Mbits/sec
[  9]  0.0-20.0 sec  7191 MBytes  3016 Mbits/sec
[  9] MSS size 8192 bytes (MTU 8232 bytes, unknown interface)
```

NDT/NPAD: web100-based diagnostic tools

Network Diagnistics tool (NDT):

- Good for finding local issues like duplex mis-match, and small TCP buffer.
- Multi-level results allow novice and expert users to view and understand the test results.
- One of the tools used by M-lab (http://www.measurementlab.net/) and http://www.broadband.gov/qualitytest/
- http://e2epi.internet2.edu/ndt/

A similar tool is NPAD, with is more targeted at network experts

- http://www.psc.edu/networking/projects/pathdiag/

Both require server side to be running a web100 kernel

- http://www.web100.org/

Netalyzr: http://netalyzr.icsi.berkeley.edu

Netalyzr is a comprehensive network measurement and debugging tool built as a Java Applet

- A suite of custom servers hosted primarily using Amazon EC2

Multiple properties checked

- NAT properties: Presence. Port Renumbering, DNS proxies
- IP properties: Blacklist membership
- Network properties: Latency, Bandwidth, Buffering, IPv6 capability, Path MTU
- Service properties: Outbound port filtering and proxies on a large list of services
- HTTP properties: Hidden HTTP proxies or caches. Correct operation of proxies and caches
- DNS properties: Latency of lookups, glue policy, DNS wildcarding, name validation, DNS MTU, DNSSEC readiness
- Host properties: Browser identity, Java version, clock accuracy

New ESnet Diagnostic Tool: 10 Gbps IO Tester

16 disk raid array: capable of > 10 Gbps host to host, disk to disk

Runs anonymous read-only GridFTP

Accessible to anyone on any R&E network worldwide

1 deployed on now (west coast, USA)

- 2 more (midwest and east coast) by end of summer

Already used to debug many problems

Will soon be registered in perfSONAR gLS

See: http://fasterdata.es.net/disk_pt.html

Part 2: Joe Metzger

Network troubleshooting

Troubleshooting Assumptions

What can you conclude from the following?

- Tests across campus are good
- Tests to the regional are OK
- Tests to the nearest backbone hub are OK to poor
- Tests to distant backbone nodes are very bad

Is the backbone congested?

Troubleshooting Assumptions

What can you conclude from the following?

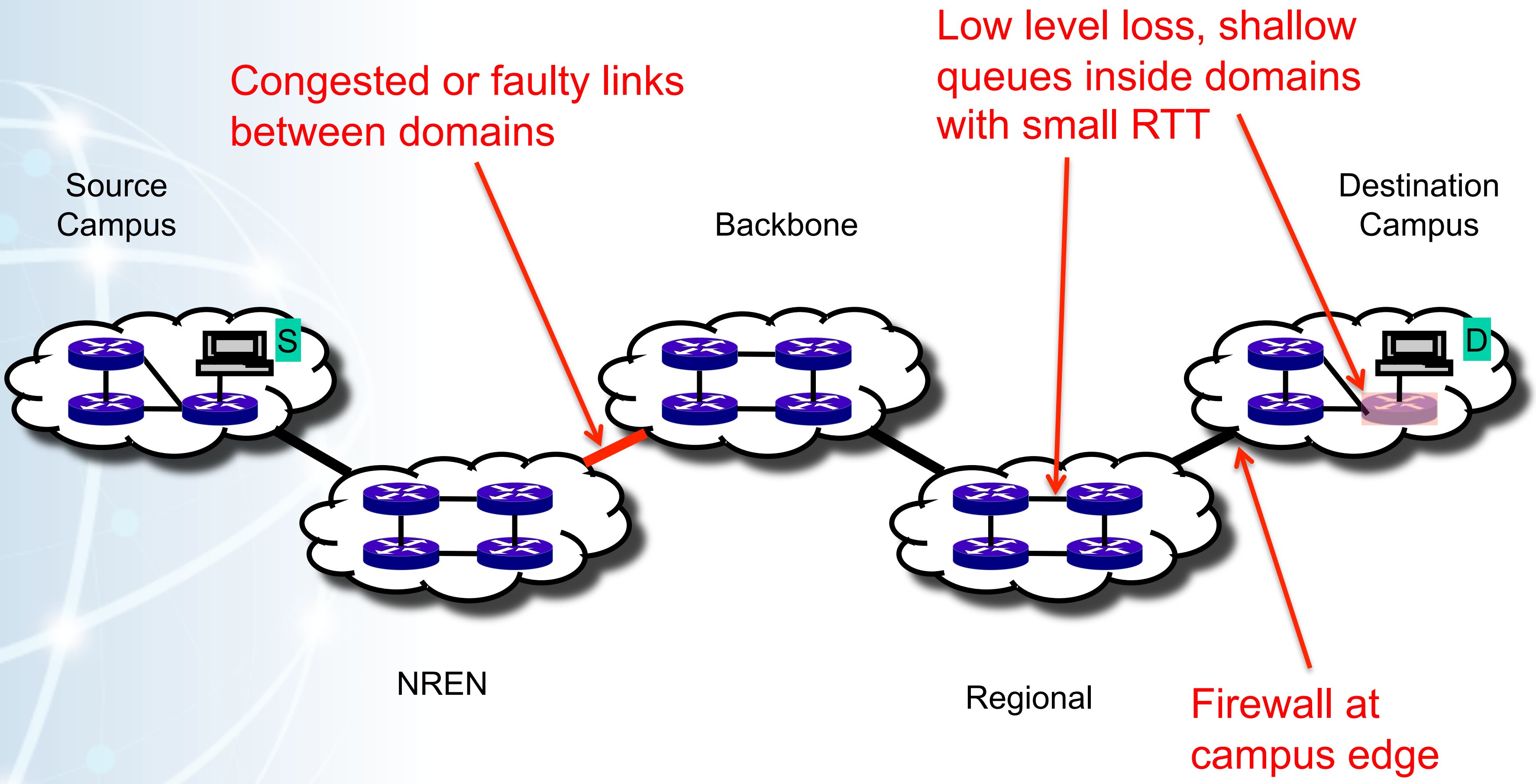
- Tests across campus are good
- Tests to the regional are OK
- Tests to the nearest backbone hub are OK to poor
- Tests to distant backbone nodes are very bad

Throughput is inversely proportional to RTT

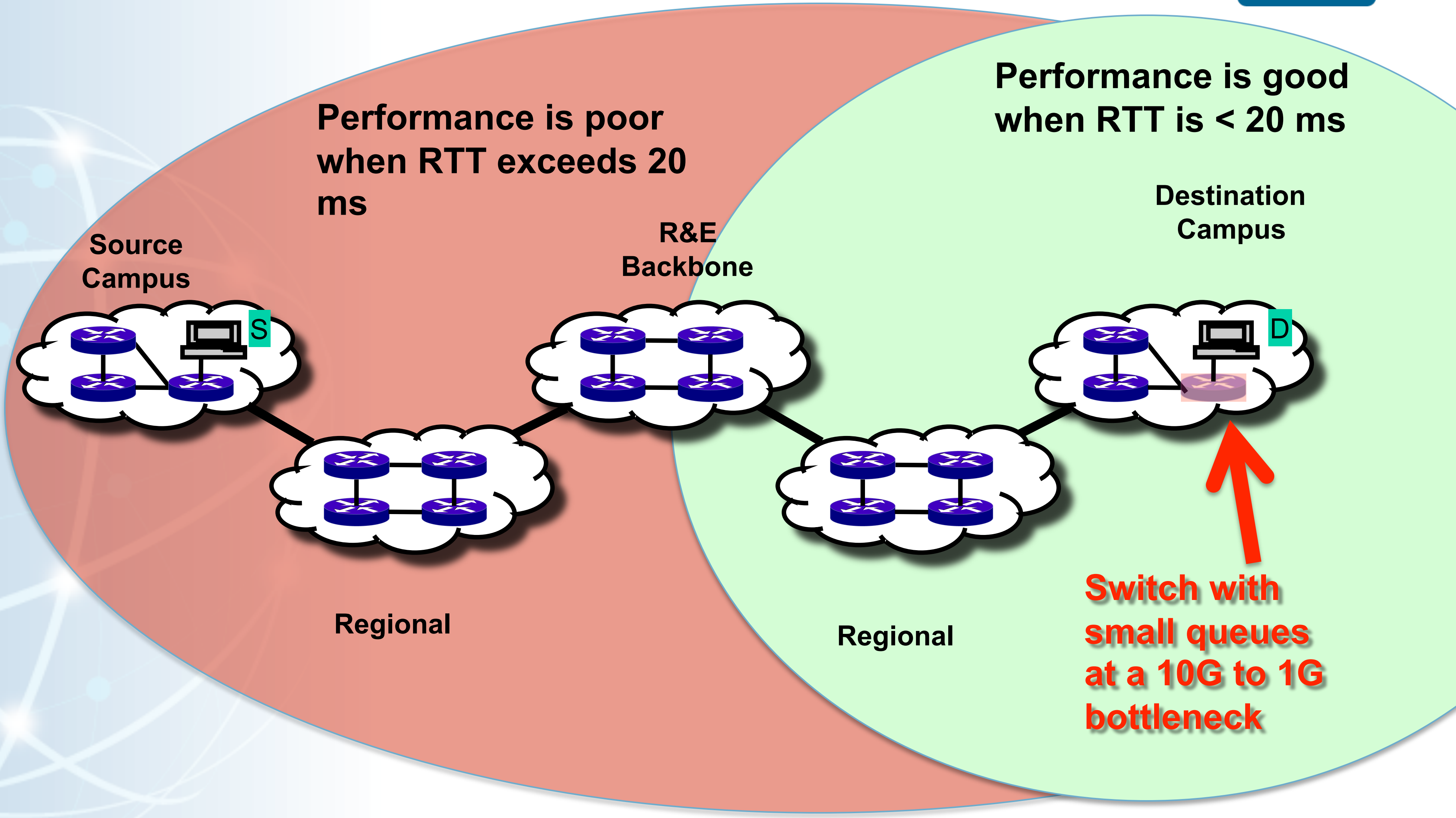
What can cause this?

- Packet loss close to the source due to media errors
- Packet loss close to the source due to overflowing queues
- Un-tuned TCP

Where are common problems?


Congested or faulty links between domains
Low level loss, shallow queues inside domains with small RTT
Source Campus
S
Backbone
Destination Campus
D
NREN
Regional
Firewall at campus edge

Local testing will not find all problems


Performance is poor when RTT exceeds 20 ms
Performance is good when RTT is < 20 ms
Source Campus
S
R&E Backbone
Destination Campus
D
Regional
Regional
Switch with small queues at a 10G to 1G bottleneck

Soft Network Failures

Soft failures are where basic connectivity functions, but high performance is not possible.

TCP was intentionally designed to hide all transmission errors from the user:

- "As long as the TCPs continue to function properly and the internet system does not become completely partitioned, no transmission errors will affect the users." (From IEN 129, RFC 716)

Some soft failures only affect high bandwidth long RTT flows.

Hard failures are easy to detect & fix

- soft failures can lie hidden for years!

One network problem can often mask others

Common Soft Failures

Small Queue Tail Drop

- Switches not able to handle the long packet trains prevalent in long RTT sessions and local cross traffic at the same time

Un-intentional Rate Limiting

- Processor-based switching on routers due to faults, acl’s, or mis-configuration
- Security Devices
  - E.g.: 10X improvement by turning off Cisco Reflexive ACL

Random Packet Loss

- Bad fibers or connectors
- Low light levels due to amps/interfaces failing
- Duplex mismatch

Building a Global Network Diagnostic Framework

Addressing the Problem

Deploy performance testing and monitoring systems throughout the network

Give users access through a standard mechanism: perfSONAR

Sidebar on perfSONAR terminology:

- perfSONAR: standardized schema, protocols, APIs
- perfSONAR-MDM: GÉANT Implementation primarily Java
- perfSONAR-PS: Internet2/ESnet/more Implementation, primarily Perl
- PS-Toolkit: Easy to install Packaging of perfSONAR-PS

Components of a Global Diagnostic Service

Globally accessible measurement services

- Support for both active probes and passive results (SNMP)
- In particular: throughput testing servers
  - Recommended tool for this is bwctl
    - http://www.internet2.edu/performance/bwctl/
    - Includes controls to prevent DDOS attacks

Services must be registered in a globally accessible lookup service

Open to the entire R&E network community

- Ideally light-weight authentication and authorization

Addressing the Problem: perfSONAR

perfSONAR - an open, web-services-based framework for:

- running network tests
- collecting and publishing measurement results

ESnet is:

- Deploying the framework across the science community
- Encouraging people to deploy ‘known good’ measurement points near domain boundaries
  - “known good” = hosts that are well configured, enough memory and CPU to drive the network, proper TCP tuning, clean path, etc.
- Using the framework to find and correct soft network failures.

perfSONAR Services

Lookup Service

- gLS – Global lookup service used to find services
- hLS – Home lookup service for registering local perfSONAR metadata

Measurement Archives (data publication)

- SNMP MA – Interface Data
- pSB MA   -- Scheduled bandwidth and latency data

PS-Toolkit includes these measurement tools:

- BWCTL: network throughput
- OWAMP: network loss, delay, and jitter
- PINGER: network loss and delay

PS-Toolkit includes these Troubleshooting Tools

- NDT (TCP analysis, duplex mismatch, etc.)
- NPAD (TCP analysis, router queuing analysis, etc)

ESNet PerfSONAR Deployment

ESnet Deployment Activities

ESnet has deployed OWAMP and BWCTL servers next to all backbone routers, and at all 10Gb connected sites

- 27 locations deployed, many more planned
- Full list of active services at:
  - http://stats1.es.net/perfSONAR/directorySearch.html
  - Instructions on using these services for network troubleshooting: http://fasterdata.es.net

These services have proven extremely useful to help debug a number of problems

Global PerfSONAR-PS Deployments

Based on “global lookup service” (gLS) registration, May 2010: currently deployed in over 100 locations

- ~ 100 bwctl and owamp servers
- ~ 60 active probe measurement archives
- ~ 25 SNMP measurement archives
- Countries include: USA, Australia, Hong Kong, Argentina, Brazil, Uruguay, Guatemala, Japan, China, Canada, Netherlands, Switzerland

US Atlas Deployment

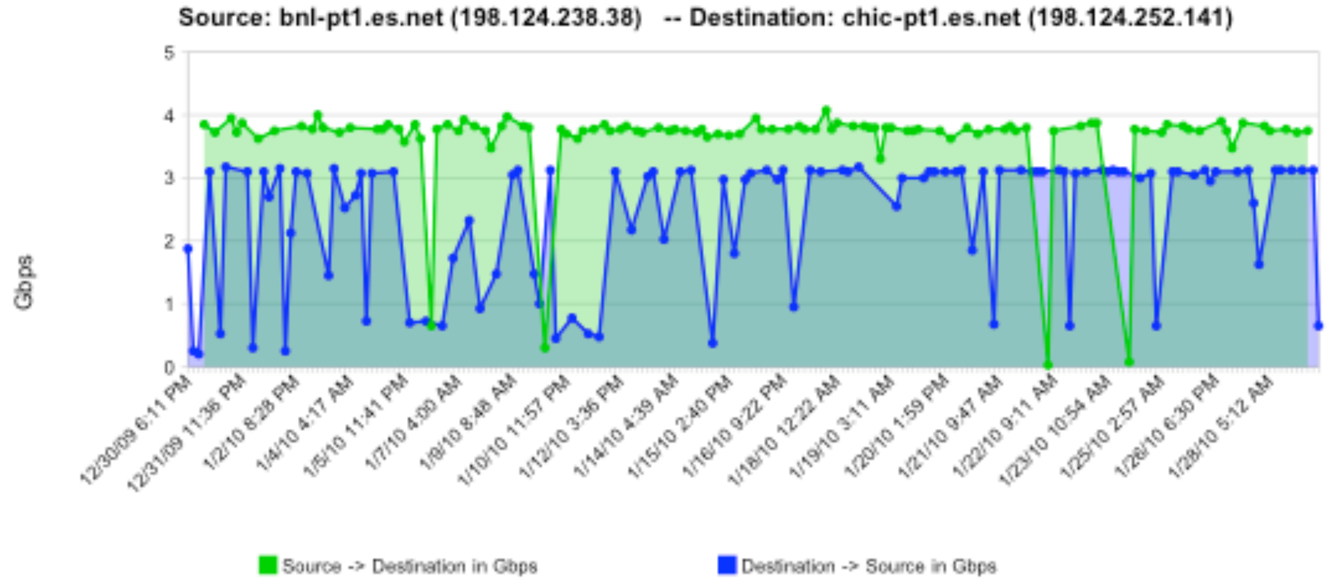
- Monitoring all “Tier 1 to Tier 2” connections

For current list of public services, see:

- http://stats1.es.net/perfSONAR/directorySearch.html

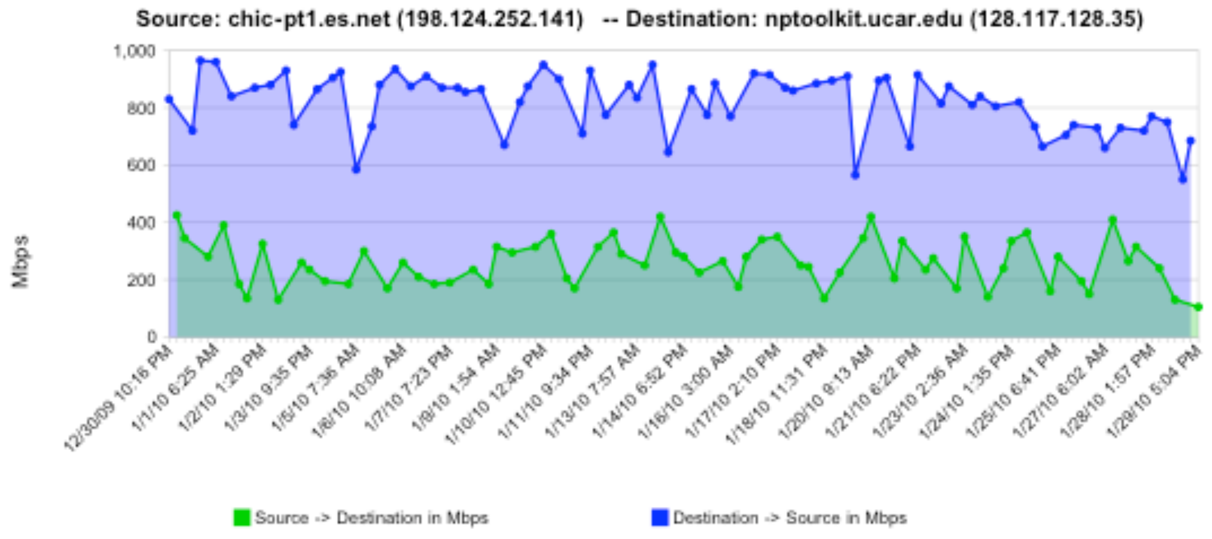
SAMPLE results

Sample Results

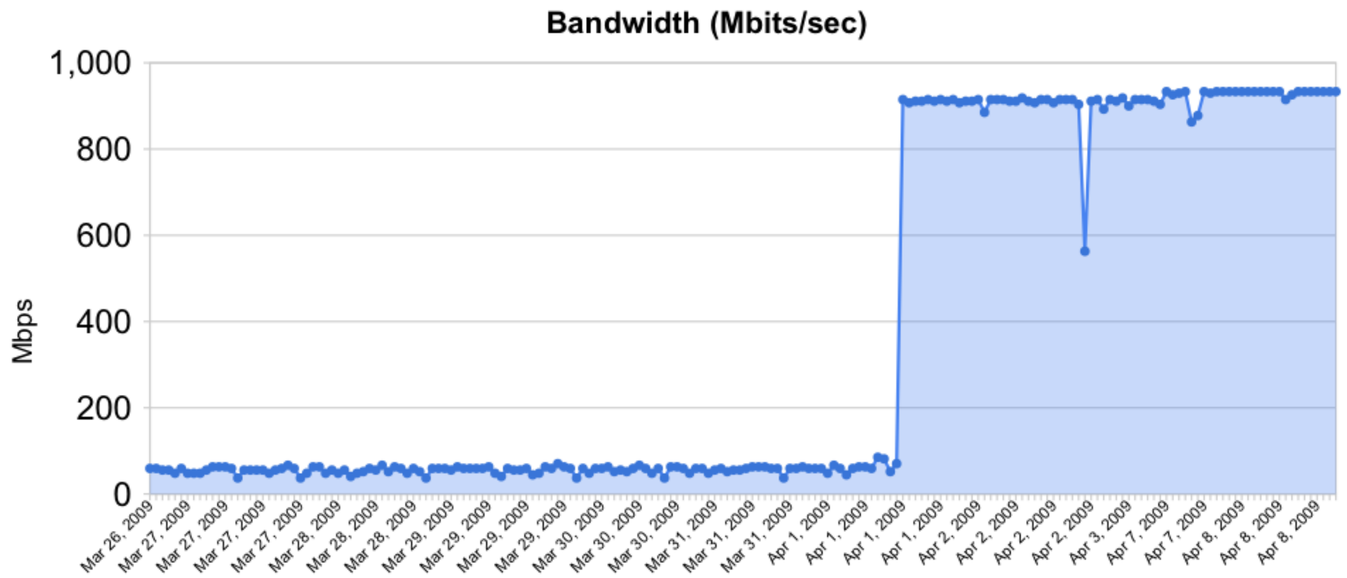


Heavily used path: probe traffic is “scavenger service”

Asymmetric Results: different TCP stacks?

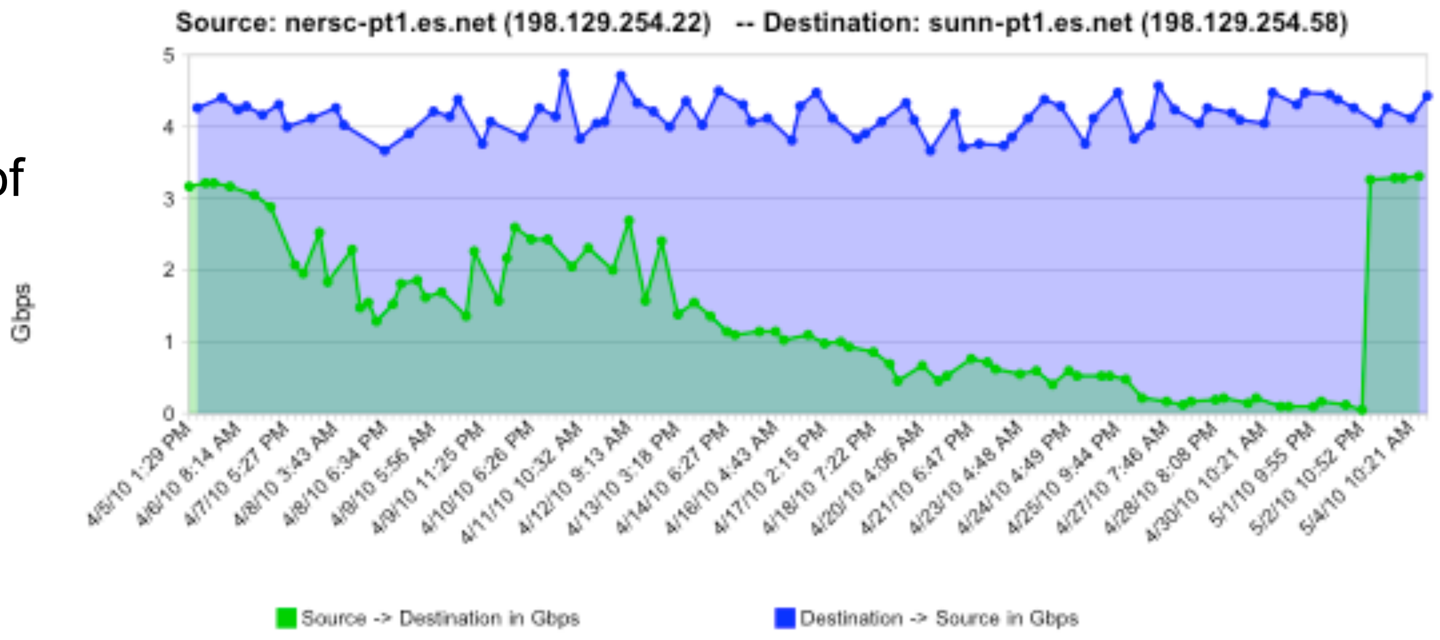


Sample Results: Finding/Fixing soft failures



Rebooted router with full route table

Gradual failure of optical line card



Effective perfsonar Deployment Strategies

Levels of perfSONAR deployment

ESnet classifies perfSONAR deployments into 3 "levels":

Level 1: Run a bwctl server that is registered in the perfSONAR Lookup Service.

- This allows remote sites and ESnet engineers to run tests to your site.

Level 2: Configure "perfSONAR BOUY" to run regularly scheduled tests to/from your host.

- This allows you to establish a performance baseline, and to determine when performance changes.

Level 3: Full set of perfSONAR services deployed (everything on the PS-Toolkit bootable CD)

Deploying perfSONAR-PS Tools In Under 30 Minutes

There are two easy ways to deploy a perfSONAR-PS host

"Level 1" perfSONAR-PS install

- Build a Linux machine as you normally would (configure TCP properly! See: http://fasterdata.es.net/TCP-tuning/)
- Go through the Level 1 HOWTO
- http://fasterdata.es.net/ps_level1_howto.html
- Simple, fewer features, runs on a standard Linux build

Use the perfSONAR-PS Performance Toolkit bootable CD

- Most of the configuration via Web GUI
- http://psps.perfsonar.net/toolkit/
- More features (level 2 or 3), runs from CD

Simple "installation package" coming soon

Measurement Recommendations to DOE Sites

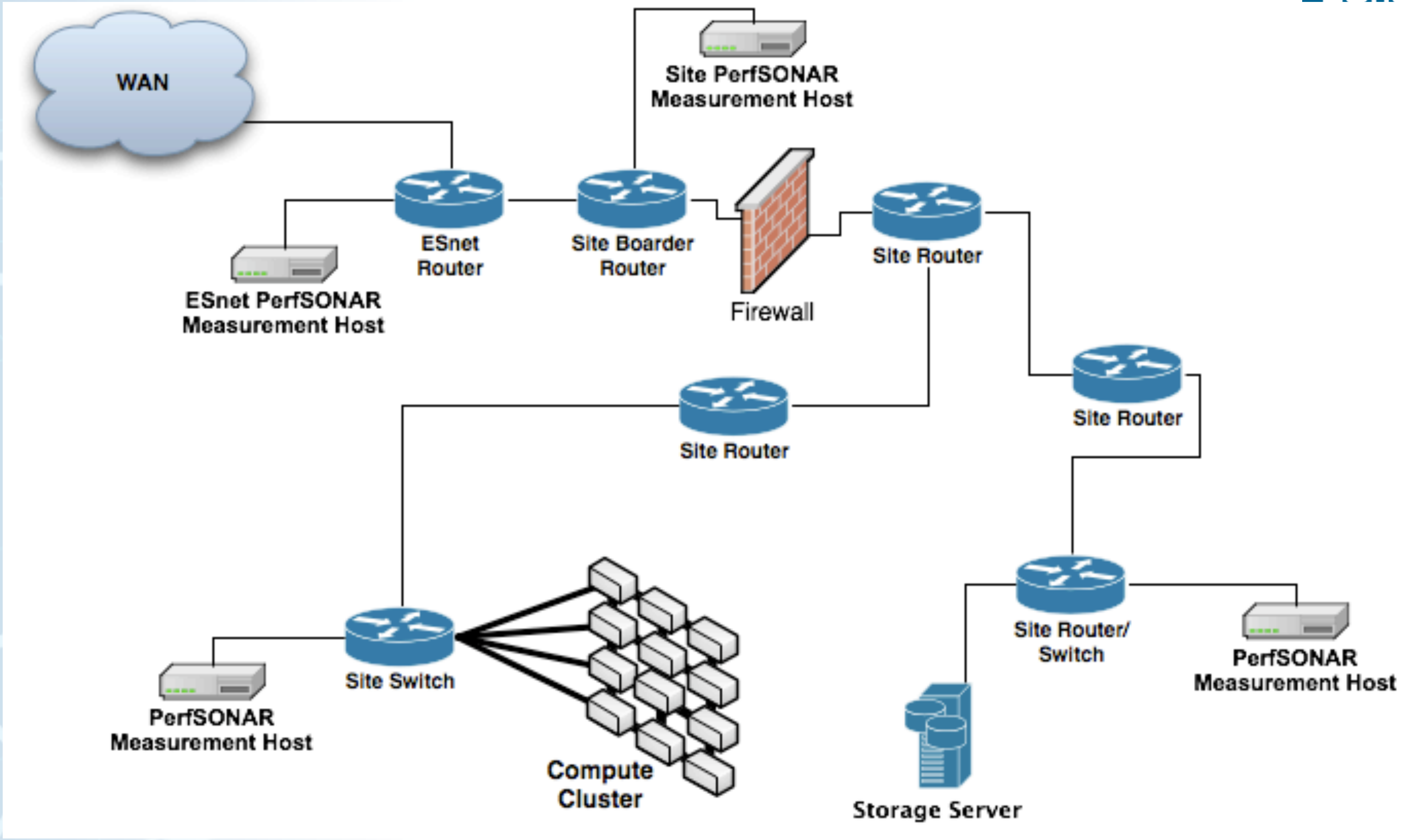
Deploy perfSONAR-PS based test tools

- At Site border
  - Use to rule out WAN issues
- Near important end system
  - Use to rule out LAN issues

Use it to:

- Find & fix current local problems
- Identify when they re-occur
- Set user expectations by quantifying your network services

Sample Site Deployment



Developing a Measurement Plan

What are you going to measure?

- Achievable bandwidth
  - 2-3 regional destinations
  - 4-8 important collaborators
  - 4-12 times per day to each destination
  - 20 second tests within a region, longer across the Atlantic or Pacific
- Loss/Availability/Latency
  - OWAMP: ~10 collaborators over diverse paths
  - PingER: use to monitor paths to collaborators who don't support owamp
- Interface Utilization & Errors

What are you going to do with the results?

- NAGIOS Alerts
- Reports to user community
- Post to Website

Importance of Regular Testing

You can't wait for users to report problems and then fix them (soft failures can go unreported for years!)

Things just break sometimes

- Failing optics
- Somebody messed around in a patch panel and kinked a fiber
- Hardware goes bad

Problems that get fixed have a way of coming back

- System defaults come back after hardware/software upgrades
- New employees may not know why the previous employee set things up a certain way and back out fixes

Important to continually collect, archive, and alert on active throughput test results

Router Tuning

Network Architecture Issues

Most LANs are not purpose-built for science traffic – they carry many types of traffic

- Desktop machines, laptops, wireless
- VOIP
- HVAC control systems
- Financial systems, HR
- Some science data coming from someplace

Bulk data transfer traffic is typically very different than enterprise traffic

IE, Voip vs Bulk Data:

- VOIP: Deep Queuing is bad, drops are OK
- Bulk Data: Deep Queuing is OK, drops are bad.

LAN Design Recommendations

1. Separate desktop/enterprise traffic from bulk data transfer traffic
   - Security requirements & firewall requirements differ.
   - Many enterprise firewalls do not do a good job with bulk data transfer streams
2. Separate local area traffic from wide area traffic
   - Most TCP stacks do not do share bandwidth evenly between short & long RTT time traffic.
   - Short RTT flows cause congestion & then recover quickly.
   - Long RTT flows don't recover as quickly (or at all sometimes)
3. Put perfSONAR measurement points on the boarder & near important data sources & sinks.
   - Support measurement both to the campus border (from both inside & outside)
   - Support measurements all the way to the important resources.

Science vs. Enterprise Networks

Science and Enterprise (change to "commodity" or similar) network requirements are in conflict

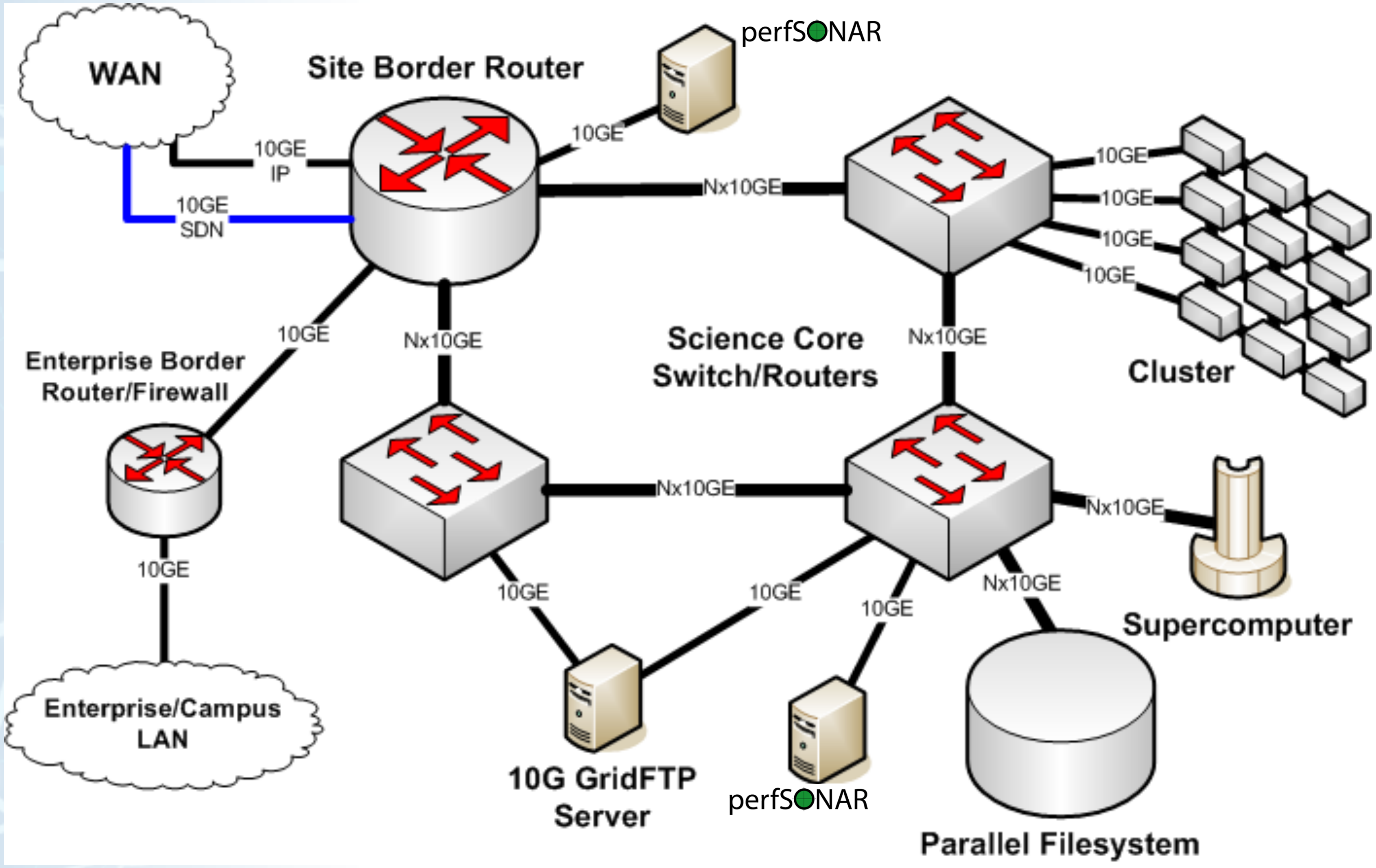
One possible remedy: Split the network at the border into an Enterprise network, and a Science network.

- Put the Enterprise security perimeter at the edge of the enterprise network
- Use security tools compatible with high speed science flows on the science network.

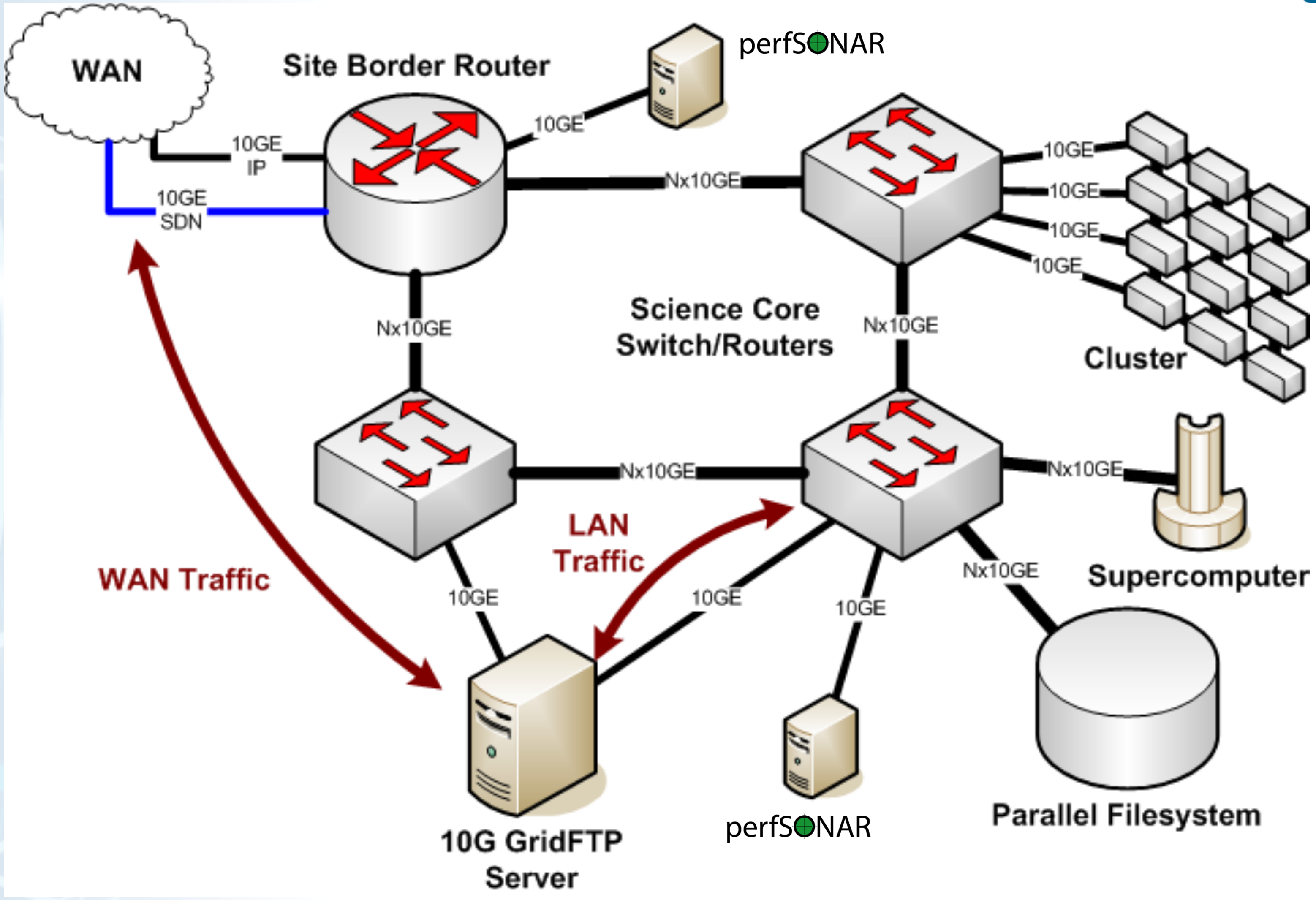
Another Option

- Build a separate enclave at the external border supporting dedicated systems designed & tuned wide-area data transfer

Separate Enterprise and Science Networks



Internal / External Traffic Separation



Router and Switch Configuration

Buffer/queue management

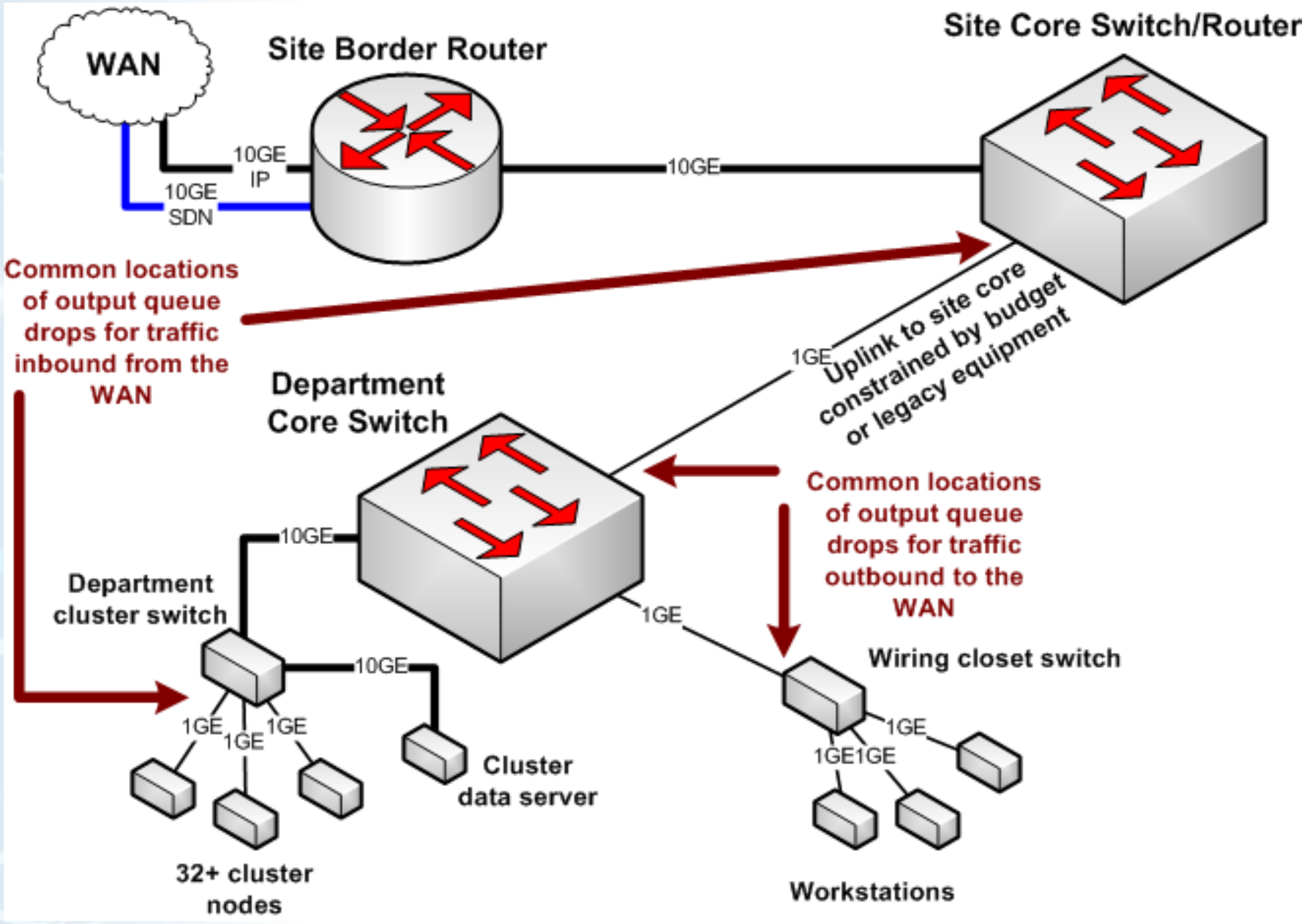
- TCP traffic is bursty
- Coincident bursts destined for a common egress port cause momentary oversubscription of the output queue
- Traffic entering via a 10G interface and leaving via a 1G interface can cause oversubscription of output queue
- Momentary oversubscription of output queues causes packet loss
- Default configuration of many devices is inadequate

Example: Cisco commands

- ‘sho int sum’ – check for output queue drops
- ‘hold-queue 4096 out’ – change from default 40-packet output queue depth

See http://fasterdata.es.net/

Output Queue Oversubscription


WAN
Site Border Router
Site Core Switch/Router
10GE IP
10GE SDN
10GE
Common locations of output queue drops for traffic inbound from the WAN
1GE
Uplink to site core constrained by budget or legacy equipment
Department Core Switch
Common locations of output queue drops for traffic outbound to the WAN
10GE
1GE
Department cluster switch
Wiring closet switch
10GE
1GE
1GE
1GE
1GE
1GE1GE
Cluster data server
32+ cluster nodes
Workstations

Router and Switch Issues

Need to have adequate buffering to handle TCP bursts

- Long fat pipes mean large TCP windows which means large wire-speed bursts
- Changes in speed (e.g. 10Gbps to 1Gbps)

Many devices do not have sufficient output queue resources to handle bulk data flows

- Need to understand device capabilities
- When purchasing new equipment, require adequate buffering

Many defaults assume a different traffic profile (e.g. millions of web browsers)

- Drop traffic at first sign of oversubscription
- Makes TCP back off because of packet loss
- Protects flows such as VOIP and videoconferencing (remember the enterprise traffic?)
- Non-default configuration is necessary

Network Architecture Summary

Build a network to support bulk data transfers with data transfer in mind

- Don't just plug it in anyplace
- Avoid traversing Enterprise infrastructure if you can
- Connect your bulk transfer resources as close to the border router as you can

Configure routers and switches for adequate buffering

- Watch drop counters (e.g. sho int sum or sho int queue)
- Watch error counters

If you have to, collocate your data server near the border router

- On-site transfers to your server in another building will usually be high performance due to low latency
- WAN transfers bypass the Enterprise infrastructure
- Might be better for network administrators as well (no need to upgrade all the switches in the path to accommodate science traffic)

Jumbo Frames

Standard Ethernet packet is 1500 bytes (aka: MTU)

Most gigabit Ethernet hardware now supports “jumbo frames” (jumbo packet) up to 9 KBytes

- Helps performance by reducing the number of host interrupts
- Esnet, Internet2, and GEANT are all 9KB “jumbo-clean”
  - But many sites have not yet enabled Jumbo Frames
  - Some jumbo frame implementations do not interoperate
- Ethernet speeds increased 3000x since the 1500 byte frame was defined
  - Computers now have to work 3000x harder to fill the network

To see if your end-to-end path is jumbo clean:

- ping -M do -s 8972 192.0.2.254 (FreeBSD)
- ping -D -s 8972 192.0.2.254
- header math: 20 bytes IP + 8 bytes ICMP + 8972 bytes payload = 9000

Putting it all Together

Putting It All Together

Build dedicated hosts for WAN data transfers

- Use the right tools: GridFTP, BBCP, HPN-SSH
- Make sure TCP parameters are configured
- Make sure the backend or disk subsystems are properly tuned
- Put it in the right place in your network

Deploy at least a Level 1 perfSONAR host

- Test and measurement are critical
- Even if you're fabulously lucky and don't need it today, you'll need it tomorrow

Don't stop debugging until everything works end-to-end over long RTT paths!

Conclusions

The wizard gap is starting to close (slowly)

- If max TCP autotuning buffers are increased

Tuning TCP is not easy!

- no single solution fits all situations
  - need to be careful to set TCP buffers properly
  - sometimes parallel streams help throughput, sometimes they hurt
- Autotuning helps a lot

Lots of options for bulk data tools

- Choose the one that fills your requirements
- Don’t use unpatched scp!

More Information

http://fasterdata.es.net/

http://psps.perfsonar.net/

email: BLTierney@es.net, metzger@es.net