
Making time for beachcombing
(by improving network performance)

Motivation

- Networks are as good as they are going to get
  - Bandwidth is either cheap or non-existent
  - Hardware-based routers forward packets at line rate with no avoidable jitter
  - Latency remains
- Yet a user still can't fill a 1Gbps ethernet link of useful length
- The reasons for this reside in the host: applications, operating systems, hardware, algorithms

Fundamental TCP

TCP — Transmission control protocol

- User's view
  - a connection between applications: multiplexed, reliable and in order, flow controlled
- Network designer's view
  - cooperative sharing of link bandwidths
  - avoiding the congestion collapse of the Internet
- The genius of TCP is that it uses one mechanism to solve these disparate requirements
  - the windowed Acknnowledgement

TCP window, 1 of 2

- Every transmitted byte has a sequence number*
- Sender
  - track sequence number sent and sequence number acknowleged
  - buffer the sent but un-acknowledged data in case it needs to be retransmitted

* Or with TCP window scaling each 2^n of bytes has a sequence number

TCP window, 2 of 2

- Receiver
  - Buffer incoming segments
  - Ack every second segment or, after a delay, lone segments
  - Implement flow control by lowering the advertised window as receiver buffer is consumed
- Retransmission
  - The amount of data to be re-sent is less than the window, since this caused congestion
  - So, maintain a “congestion window”, the bandwidth the sender thinks it can consume without causing congestion

Slow start mode

- Don't cause congestion collapse with a new connection
  - We have no estimate of the congesting bandwidth
  - Start with one or two segments
  - Double this per round-trip time, ie: exponential
- Congestion occurs, ie: an Ack is late
  - Cwnd was increased too much
    - set the slow-start threshold to half the cwnd
    - Resume slow start from previous cwnd until the ssthresh
    - Now enter congestion avoidance mode, a linear approach to the expected congesting bandwidth

Congestion avoidance mode

- Maintain an existing connection
- Increment the congestion window by one cwnd per round-trip time
  - Gives a linear growth in bandwidth
- If an Ack is late, reduce cwnd by one segment and re-enter slow start
  - An improvement is to drop back only to ssthresh and have ssthresh lag cwnd
- Sensitive to reordered packets
  - so wait for three duplicate Acks if the Ack shows a hole in the transmitted data

Properties of the TCP algorithm

- Slow start is exponential, but still very slow for high-bandwidth connections
- Packet loss during slow start is devastating
- Congestion control leads to a sawtooth "hunting" around the congested bandwidth
  - wasting large absolute amount of bandwidth
- Loss is interpreted as congestion

Host buffer sizing

- Both the sender and receiver need to buffer data
  - the sender's unacknowleged data is more critical
- Size for both is the bandwidth-delay product of the path
- The BDP is easy to compute in general, but difficult for a specific connection
  - requires knowledge of the ISP's networks
  - in general, use the interface bandwidth and a guess at the worst delay, verified with a ping

Operating systems

Buffer sizing in Linux, 1 of 2

- The kernel tries to autotune the buffer size, up to 4MB
  - calculate the BDP, if under 4MB do nothing
- This is fine for ADSL and 802.1g connections in Australia, but too little for gigabit ethernet in Australia
  - it takes 90ms one-way just to cross the Pacific, so the defaults are too low for us

Buffer sizing in Linux, 2 of 2

- Linux has two sysctls
    - net.ipv4.tcp_rmem
    - net.ipv4.tcp_wmem
- These are vectors of <*minimum*, *initial*, *maximum*> memory usage, in bytes
- Set the *maximum* size to the BDP plus a big allowance for kernel data structures
- Keep the *initial* value near the default, as it could be used to DoS your server

Applications and buffer sizing

- Applications can request a TCP buffer size
  - setsockopt(…, SO_SENDBUF, …)
    setsockopt(…, SO_RECVBUF, …)
- These requests are trimmed by
  - net.core.rmem_max
    net.core.wmem_max
- Setting the buffer size explicitly disables autotunung
  - *iperf* always sets the buffer size, so never gives true results for Linuxl. Ouch!

Distributions

- Some distributions detune the TCP stack, undo that
  - net.ipv4.tcp_moderate_rcvbuf = 1
    net.ipv4.tcp_timestamps = 1
    net.ipv4.tcp_window_scaling = 1
    net.ipv4.tcp_sack = 1 *
    net.ipv4.tcp_ecn = 1 *
    net.ipv4.tcp_syncookies = 0
    net.ipv4.tcp_moderate_rcvbuf = 1
    net.ipv4.tcp_adv_win_scale = 7 *

* These parameters trigger bugs in some networking equipment
SACK – Cisco PIX
ECN – Cisco PIX
Window scale > 2 – a number of ADSL gateways

TCP algorithm variations

- The traditional TCP algorithm has reached its limits
  - All operating systems offer an alternative, Linux offers all the alternatives it legally can
- A selection
  - CUBIC. The current default in Linux. Quick slow start, not too much hunting, fairness is poor
  - Westwood+. Tuned for lossy links such as WLANs.
  - Hamilton TCP. Nicely fair.
- It is the sender's choice of algorithm which is important

MTU – Maximum transmission unit

- The largest packet size which can pass down a path
- Why?
  - Larger MTUs reduce the packet-handling overhead of the operating system
  - Above 1Gbps the Mathis, et al formula tells us that MTU > 1500 is needed for a single long-distance connection to be able to fill the pipe
- IP subnets require all hosts on the subnet to have the same MTU

MTU – Ethernet jumbo frame

- Not standard, look for
  - 1Gbps jumbo frame: 9000B
  - 10GE super jumbo frame: 64KB

Networks and larger MTUs

- Use the maximum MTU between network devices
  - Allows 9000 bytes with MPLS and other headers to pass through
  - Aim is to fix the bug with current MTUs visible to hosts and always deliver 9000 bytes to the host adapter
- Worthwhile regardless of customer take-up, as gives outstanding improvement to OSPF and BGP convergence

Low memory fragmentation

- Low memory is used for network and disk buffers
- 880MB on 32-bit processors
- Linux will happily fragment kernel memory, the common case of a network backup server fragments memory in about 2TB and dies in about 6TB with RHEL3 using jumbo frames
- Linux 2.6.24 has anti-fragmentation patches
- 64-bit processors have more low memory

iptables

- Network performance is hampered when a buffer is copied, conntrack modules do this when parsing a packet
- NAT is obviously slow since it has to alter the buffer
- So distros which depend on a iptables firewall for security aren't really suitable for speeds ~1Gbps
  - tcpwrapper is still useful

Virtualisation

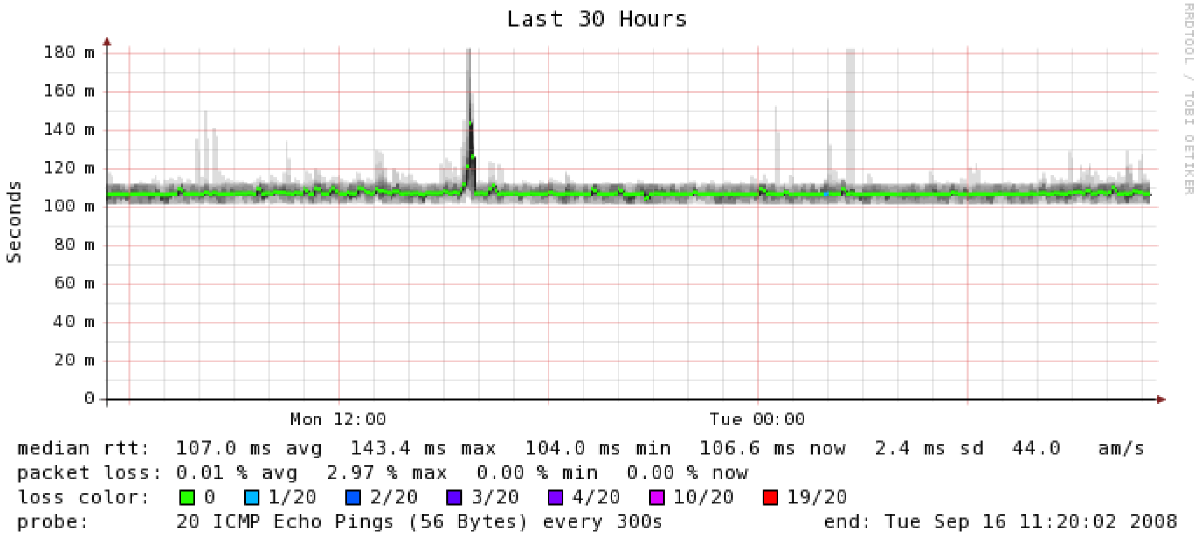
- Don't do this at the moment
- Eventually there will be little effect but at the moment the effect is large
  - Need interfaces to use zero-copy from host to VM
  - Need host interfaces to have a flow cache to cheaply route packets to VMs

Debugging tools

- smokeping
- tcptraceroute
- ttcp
- iperf
- Web100
- wget
- NPAD
- Kernel has a new netlink API for TCP state changes
- Wireshark and passive tap

ping and smokeping

- ping
  - Measures loss and latency
  - Rate limited
- smokeping



traceroute

- Measures path

```
$ traceroute www.unisa.edu.au
   traceroute to www.unisa.edu.au (130.220.79.115), 30 hops max, 40 byte packets
    1  129.96.155.1 (129.96.155.1)  11.061 ms  17.982 ms  18.103 ms
    2  129.96.128.4 (129.96.128.4)  19.397 ms  20.464 ms  28.029 ms
    3  192.168.1.2 (192.168.1.2)  29.718 ms  30.069 ms  39.923 ms
    4  gi1.cpe-flinders-er1.aarnet.net.au (202.158.199.225)  46.175 ms  48.999 ms  50.257 ms
    5  ge-1-0-8.adl-b-bb1.aarnet.net.au (202.158.199.206)  50.738 ms  58.324 ms  58.845 ms
    6  ge-0-0-0.adl-a-bb1.aarnet.net.au (202.158.194.13)  59.823 ms  75.651 ms  70.172 ms
    7  gi0.cpe-unisa-er1.aarnet.net.au (202.158.199.170)  74.040 ms  73.144 ms  72.400 ms
    8  gw1.cpe-unisa-er1.aarnet.net.au (202.158.199.122)  65.091 ms  2.702 ms  3.534 ms
    9  * * *
   10  * * *
   11  * * *
   12  * * *
   13  * * *
   14  *^C
```

tcptraceroute

- Measures path through firewalls

```
# tcptraceroute www.unisa.edu.au
   Selected device ath0, address 129.96.155.200, port 54671 for outgoing packets
   Tracing the path to www.unisa.edu.au (130.220.79.115) on TCP port 80 (http), 30 hops max
    1  129.96.155.1  10.916 ms  6.859 ms  0.948 ms
    2  129.96.128.4  0.974 ms  1.646 ms  7.809 ms
    3  192.168.1.2  3.322 ms  8.016 ms  6.959 ms
    4  gi1.cpe-flinders-er1.aarnet.net.au (202.158.199.225)  2.520 ms  10.013 ms  1.228 ms
    5  ge-1-0-8.abl-b-bb1.aarnet.net.au (202.158.199.206)  8.333 ms  7.476 ms  2.757 ms
    6  ge-0-0-0.adl-a-bb1.aarnet.net.au (202.158.194.13)  9.332 ms  10.467 ms  9.323 ms
    7  gi0.cpe-unisa-er1.aarnet.net.au (202.158.199.170)  10.639 ms  7.002 ms  11.453 ms
    8  gw1.cpe-unisa-er1.aarnet.net.au (202.158.199.122)  2.095 ms  11.477 ms  16.567 ms
    9  * * *
   10  corpweb.city.unisa.edu.au (130.220.79.115) [open]  9.677 ms * 6.502 ms
```

ttcp

- ttcp -r -s -v
  ttcp -t -s -v localhost
- Hidden command on Cisco IOS, useful for following path down routers towards performance hole
- ```
  iss:  627351554  snduna:  627351555  sndnxt:  627351555     sndwnd:  17880
  irs: 1819636890  rcvnxt: 1836414108  rcvwnd:       3520  delrcvwnd:    608

  SRTT: 37 ms, RTTO: 1837 ms, RTV: 1800 ms, KRTT: 0 ms
  minRTT: 0 ms, maxRTT: 300 ms, ACK hold: 200 ms
  Flags: passive open, retransmission timeout, keepalive running
    nagle, gen tcbs, Timestamp option used

  Datagrams (max data segment is 4128 bytes):
  Rcvd: 8136 (out of order: 0), with data: 8134, total data bytes: 16777216
  Sent: 12072 (retransmit: 0), with data: 12072, total data bytes: 772616
  ```
  ```

iperf

- A server and client
  - iperf --format m --nodelay --print_mss --server
  - iperf --time 10 --interval 1 --client localhost
- Buffer sizing
  - explicit, not using Linux's autotuning
  - unrealistic results

Web100

- A big kernel patch, which makes TCP algorithm variables available to user space analysis programs
- Useful user space programs
- Most useful when run on server

NPAD

- Web100 on server
- Java test program on client
- Creates a report of network performance

Passive optical tap

- Running some programs has a heisenberg effect
  - Putting an interface into promiscuous mode disables some performance features
  - Some platforms are too tight to support user space tools without purturbing measurements (eg, ADSL routers)
- An optical tap provides a copy of the data with no effect on the data crossing the link

Debugging technique

- Use a scientific approach
  - Create a hypothesis
  - Design an experiment to test the hypothesis
  - Repeat
- Record results

Debugging – the nightmare

- Solving network performance issues is hard
  - Lots of things to go wrong
  - Don't have access to every configuration item in the path
  - May not even have information about the path and a end-host
  - Cutting edge of computing knowledge
- Made a lot easier if intrumentation of routers and hosts is extensive
  - Conversely, most ISPs can't make graphs public and won't make fault reports public

Applications

Latency

- Speed of light in fibre decreases 5% per decade, diameter of Earth reduces 7mm per decade
- But applications programmers are prolifigate with round-trips
- Example: HTTP
  - Fetch web page, be redirected
  - Fetch web page
  - Fetch CSS
  - Fetch images
- Example: GridFTP

Applications programing

- RPCs often hide unneccessary round-trips
- The database access methods are really slow
- TCP wants to stream data, adding a read/write protocol above this (such as CIFS) slows things terribly
- Application acceptance testing should use tc's NetEm module to add a delay to the test network

OpenSSH

- OpenSSH has its own TCP-like window
  - Which wasn't big enough for transfers from Australia
  - Patch available since 2004, finally integrated in OpenSSH 4.7 in 2007. Shipped in Fedora 8, anticipated in Ubuntu 8.04.
- OpenSSH insists on on-the-fly encryption
  - Network transfers can be CPU bound by the single-threaded OpenSSH encryption process
  - Science sensor data is white noise which requires a supercomputer to make sense of, so the value of encryption is?

NFS and delayed Acks

- NFS sends 8KB blocks using RPC
- Across 1500B TCP connections
- The protocol sends an odd number of packets, which means that the Ack is delayed for each NFS protocol data unit

Networks

Loss

- TCP treats loss as congestion and backs off
- High loss leads to connections never leaving slow start
- The higher the bandwidth the longer recovery from loss takes
- Wireless has a high loss, so use wired links where you care about performance
  - A 802.1n WLAN cannot push a ADSL2+ link to capacity because of loss

Ethernet nway auto-negotiation

- Do me a favour and leave the interface set to autonegotiation. If that doesn't work, throw out the NIC card: this will be cheaper.
- Widely misunderstood
  - Disabling negotiation implies you set the other interface to 10Mbps, half duplex
  - The clocking on your UTP interface brings the link speeds equal
  - But the other interface's duplex is still half and this causes loss
- Either leave autoneg alone or set both the host and the switch to the same manual parameters

Firewalls

- Many firewalls are PCs running Linux, so we're simply moving the performance problem
  - An unexpected result of World Domination
- Many firewalls have TCP bugs
- Firewalls software needs to be kept up to date
  - You would think that they would be, but many firewalls are installed without a plan for non-service effecting software upgrades

Acks need bandwidth too

- The return path can encounter congestion. If this slows Acks down then this will slow the TCP connection, despite the adequate forward path bandwidth
- Actually, just adding jitter to the Acks will effect the RTT variance calculation. A beginner's mistake is to put Acks into a differing QoS class
- ADSL links suffer from this, especially if you congest the uplink by hosting services

Some paths are bad news

- The “European tour” undersea cables (“22 countries in 3 days”) have high loss
- Old cable systems have high loss
- Copper cables between buildings on a campus have high loss unless grounding is sophisticated
- Satelite links have high loss and high latency (about 500ms for geosync up-and-back)
- Ensure an optical loss calculation is done for every optical path longer than 2Km
  - SPF output changes with age and temp

Ethernet switches

- Ethernet switches usually lack adequate buffering, so don't use them for changes in transmission rates
- Ethernet switches are tuned for VoIP, not for TCP throughput and fairness.
  - Ask about nerd knobs

Host hardware

Validation: a myth

- Purchasing hosts for high performance networking has been difficult
  - Motherboards with poor disk controllers
  - Motherboards with near-broken GbE controllers
    - Wouldn't interrupt for received packets until a packet to send is queued
  - Supposedly identical disks which weren't
- Impossible to validate the software
  - Really, really want the latest cutting-edge distro and its kernel
  - Web100 and similar patches unsupported
- Increased risk for projects with fast networking

TCP TOEs

- Only useful at particular stages of hardware development
- Otherwise causes more problems than it solves, since the TCP stack becomes a black box

Linux as a router

Real routers

- Have
  - a forwarding plane
  - a control plane
  - an administrative plane
- Which operate independently
- CPU-based routers combine all these together and have poor isolation
  - excessive forwarding can black-hole routing
  - attacks on the control plane drop the administrative plane
  - no hitless software upgrades

Linux as a toy router

- Linux does as good a job as any CPU-based router if configured correctly
- Buffers
  - Set them to at least 0.25 of the BDP
- QoS
  - Implement the typical DSCPs
  - Implement a good queuing discipline
- Control protocols
  - Set QoS so control protocols not black holed

Services

- There is no good open source routing software
  - Quagga, OpenBGP, xorp are adequate
- NTP
  - Use the vendor service of pool.ntp.org
- Have a online software update strategy
  - Linux is the operating system most responsible for network abuse: its qualities as a network server are as attractive to black hats as to white hats

Take home lessons

Lessons

- Networking bottlenecks are moving from links and routers to the hosts
- Setting buffer memory fixes most performance issues
  - Linux autotuning is getting better all the time
- If you have a host which needs serious network performance
  - Move it outside of the firewall
  - Instrument it and its network to the extreme
  - Run a cutting-edge distro with a cutting-edge kernel
    - Fedora, Ubuntu with your own kernel

Making time for beachcombing (by improving network performance)

Glen Turner

gdt@gdt.id.au

www.gdt.id.au/~gdt/presentations/2008-09-16-linuxsa-tcptune/