

Thinking about the big problem

Matt Mathis

Pittsburgh Supercomputing Center (PSC)

20 May 2009

<http://staff.psc.edu/mathis/unfriendly>

Parable

- You can't build a palace when you live in a hovel
- To build a palace you need an estate
- To build an estate, you need a house
- To build a house, you a shed
- You can build a shed while you live in a hovel

The point

- Making really large changes requires iterating on incremental improvements, some of which might not be part of the final vision.
- Note that the only really precious protocol real-estate is the header fields used by (future) legacy code. Everything else, including standards can be undone or redone.
 - We don't want to regret allocating bits

We must be wary of false tussles

- Question: What conflicts with re-ECN:
 - DSCP/PHB based QoS?
 - Weighted Fair Share Queueing (and variants)?
 - FAST, LEDBAT and other delay sensing TCPs?
 - ECN Nonce?

Answer?

- I claim the only conflict is the ECN-Nonce
 - Because it has conflicting definitions of the bits
- All others should be expected to coexist with RE-ECN in the steady state
 - They address (mostly) orthogonal problems
 - We have to assume that they are optimal somewhere
 - Any attitude “that they are not needed” WEAKENS us

A digression: A potential major change

- Default (as shipped) maximum TCP window size
 - Was 32-64k for most stacks
 - Now 512k-16MB for ALL current products
- This was an explicit goal of the web100 project
 - Out-of-the-box configurations that support
 - Global scale paths at 10 Mb/s
 - (Near) Continental scale paths at 100 Mb/s
 - Metro scale paths above 1 Gb/s
- **Reminder: Standard TCP Congestion control:**
 - Gets its self clock from queued data at a bottleneck
 - Opens the window until it causes losses
 - Normally requires queues

In the past, limited TCP windows permitted,

- At edges (light multiplexing):
 - Any device to have larger buffers than N connections
 - suppresses AIMD sawtooth
 - Not really using congestion control
- In the core (heavy multiplexing):
 - All flows have bottlenecks elsewhere
 - Load grows as population of edges and content
 - Load does not grow with Moore's law at the edges
 - ISPs can out build the load
 - No significant queues in the network
 - Not really using congestion control
 - No need for QoS
- Most flows have bottlenecks outside the network

More than 10x Growth in TCP Window sizes

- Nearly every individual flow has the potential to cause network congestion somewhere
- Don't want network buffers larger than TCP window
 - Can't suppress background CC losses
 - W/O AQM can't avoid symptoms of full queues
 - Synchronization and lockout, etc
 - Must have AQM to prevent full queues
 - Can't set AQM thresholds low enough to protect real-time
- Any drop tail queue anywhere may be problematic
- Can't avoid trying to get AIMD into equilibrium
 - In the past TCP only looked like the text book in a few places
 - In the near future, it will be much more common
 - And it will become obvious to everyone that AIMD doesn't really work

Impact

- Force the deployment of QoS
 - Segregate TCP which requires queues from delay intolerant traffic
- This was a tongue-in-cheek goal of Web100
 - But I said it in public many times
- Relentless TCP (see talk tomorrow)
 - Intended to force the differentiation between:
 - Best Effort (BE)
 - Less than Best Effort (aka BE, LBE or scavenger)
 - Low priority but relatively long queue
 - Optimal for “background” transfers and non-standard CC
 - Optimal to fill otherwise idle gaps

Thoughts about QoS

- It is a mechanism to trade-off:
 - Loss
 - Queueing delay/jitter
 - Relative performance (data rate or “priority”)
 - Price
- I suspect that 3 service classes are important
 - I am doubtful about the rest
- RE-ECN hunch:
 - Nearly completely orthogonal
 - Probably want separate accounting per class

Meta comments about QoS

- The QoS deployment gate is not QoS itself, but other seemingly unrelated technology
- Each advance depends on other technologies to provide the foundation

- The Internet has no brakes
 - A digression
 - Re-ECN is an important part, but NOT sufficient by itself

Returning to the main topic

- Elements of Re-ECN's foundation or context
 - QoS
 - Traffic segregated into a few classes with separate queues
 - Pervasive AQM
 - Eliminate drop tail
 - 1/p Congestion Control (See “Relentless” tomorrow)
 - Eliminating the next TCP lameness
 - Weighted Congestion Control (w/sqrt(p) or w/p)
 - Give the users a knob to optimize their usage
 - Economic/cost models for ALL types of networks
 - RE-ECN supports one particular world view
 - Impact/effect of huge global cost disparities
 - Does RE-ECN have to be “one-size-fits-all”