

Transport Protocols: Bulk File Transfer

Stanislav Shalunov <shalunov@internet2.edu>

Stanford University, 2005-06-01

Bulk Transfers

- The killer application for high-performance networks so far
 - What else do we need fat pipes for?
- Several flavors:
 - straightforward huge file transfer
 - interactive high-throughput applications (e.g., ImmSeg)
 - instrument data transfer (e.g., e-VLBI)

Problem Exists Below Application

- Remains unsolved even in its most simple form (file transfer)
 - best current practice: open n TCP streams, send data
 - typical current practice: $n = 1$ (FTP, HTTP, etc.)
 - worst current practice: $n = 1$, window limited by application (SCP)
- Expected performance (links are not congested): ~ 100 Mb/s
- Typical performance: less than 3Mb/s (Abilene)
- The *wizard gap* gets wider

Top Reasons of Poor Performance (maybe 80% of cases)

- **Bad transport protocols** (layer 4)
- Ethernet duplex mismatch (layer 2)
- Bad last-hop cables (layer 1)

Conventional TCP: Bad Transport

- Fundamental problems:
 - Unstable for high-speed networks
 - Too sensitive to non-congestive packet loss (even after minor fixes)
 - Before a loss happens, buffers need to fill: delay is at least doubled
- Implementation problems
 - Buffers are laughably small:
 - * Normal default buffer sizes: 8kB, 16kB, 32kB, 64kB
 - * Even 64kB over 70ms limits throughput to 7.5Mb/s
 - No provisions for automatic buffer increases

Remedies for TCP's maladies

(In increasing order of invasiveness.)

1. Tuning: buffers, window scaling, timestamps, SACK
2. Use multiple streams
3. **Something else**
4. Replace the kernel and use a different congestion control
5. Replace all routers and kernels

Internet2 Bulk Transport Working Group

- <http://transport.internet2.edu/>
- A group of congestion control researchers and high-end users
- Started in late October 2004
- Goal: do better than conventional TCP
- Most immediate deliverable: a design space survey (almost done)

Transport tool

- High performance
- Completely end-to-end: no router modifications
- Suitable for both bulk file transfer and interactive multimedia
- Portable, easy to install and use (no kernel modifications)
- Advanced congestion control using existing research
- Tolerance for minor non-congestive packet loss
- Security

Design Space for the Tool

- Current version: transport-design-space-10.pdf
- Available from <http://transport.internet2.edu/>
- Specify requirements
- Document independent design questions
- Converge on a design

Design Space Dimensions

- Explicit signaling
- Implicit signaling
- Kernel- vs user-space
- Protocol features
- Window- vs rate-based
- TCP-compatible vs TCP-friendly
- State at sender vs receiver
- Single vs multiple streams
- UI and API

Current ideas about the design

- TCP-friendly, not TCP-compatible
- Security nonces
- Implicit congestion signaling
- Delay-based, with fallback to loss-based
- User-space tool with UDP
- State at receiver where possible
- API and a file transfer/distribution application

Open questions

- Telling a full queue from an empty one
- Noise filtering
 - for current delay
 - * exponentially decaying weighted average?
 - * median of last n ?
 - for minimum
 - * route changes
 - * 1st percentile?
 - * none?
- Controller selection

What's your take?

- Your feedback counts
- Mailing list: transport@internet2.edu
- Teleconferences: about every other Friday, 9AM Pacific Time

More...

- <http://transport.internet2.edu/>
- Join the mailing list: transport@internet2.edu
- Help us with the work

Contributors

Lawrence D. Dunn (Cisco), Yunhong Gu (UIC),
Steven Low (Caltech), Injong Rhee (NCSU),
Steven Senger (UW–La Cross),
Bartek Wydrowski (Caltech), Lisong Xu (UNL)

Questions?