Next Steps for draft-irtf-iccrg-tcpeval-01

David Hayes[1] davidahayes@proton.me Tom Henderson[2] tomh@tomh.org
Mohit P. Tahiliani[3] tahiliani@nitk.edu.in

[1]Refer to https://www.linkedin.com/in/david-hayes-3640782/

[2]Refer to https://tomh.org/

[3]Refer to https://in.linkedin.com/in/mohittahiliani

21 March 2024

Why? What? and Next?

Why did this exist?

- To help facilitate fair comparisons of CCs

What it is not

- Testing standards ready (RFC5033-bis)
- Exhaustive tests of every possible aspect
- Produce graphs for your paper

What is it?

- Small set of standardized tests
- Publicly available implementation in NS2
- Negligible extra work to run
- Small set of summary results
- Suggestions for future revisions

Why is it being presented?

Interest from the ns-3 project

- Old and needs updating (ns-3)
- More than just TCP?
  - QUIC
  - ?
- Revisions to bring it inline with the current Internet
- Revisions to allow speculation about the future Internet

`https://github.com/hayesd/tcp-evaluation-suite-public`

Basic Idea

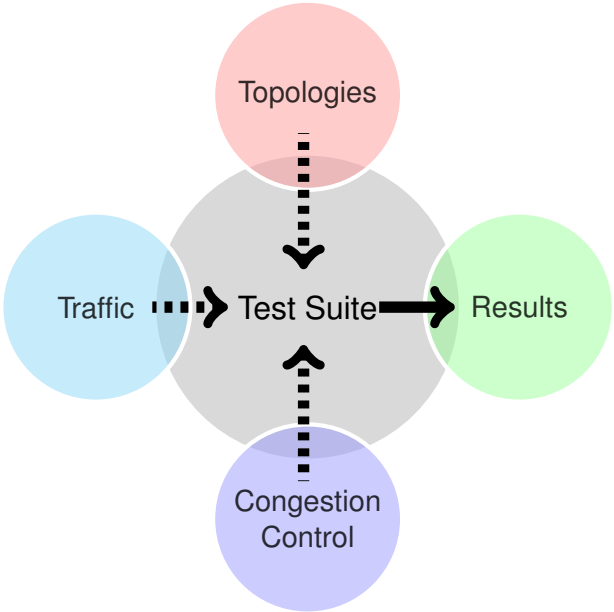
Previous contributors

Current draft authors: David Hayes, David Ros, Lachlan Andrew, and Sally Floyd.

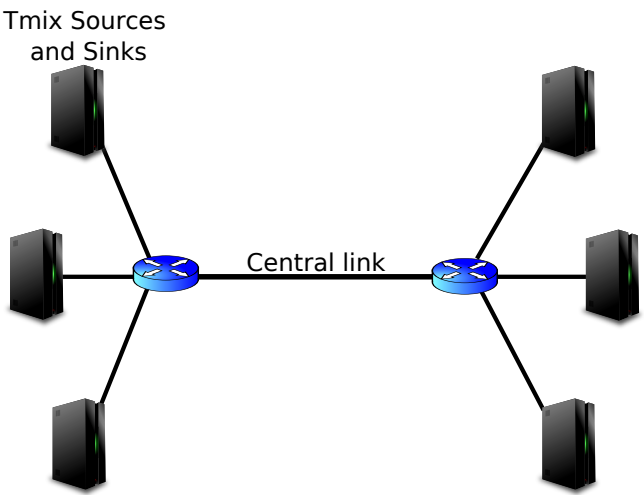
Ideas and tests: Lachlan Andrew, Cesar Marcondes, Sally Floyd, Lawrence Dunn, Romaric Guillier, Wang Gang, Lars Eggert, Sangtae Ha and Injong Rhee.

NS2 Implementation: Gang Wang, Yong Xia, and David Hayes

Feedback: Roman Chertov, Doug Leith, Saverio Mascolo, Ihsan Qazi, Bob Shorten, David Wei and Michele Weigle



Basic bottleneck link tests



Central Link modelled as:

access link, data center, trans-oceanic, geostationary satellite, wifi and dial up

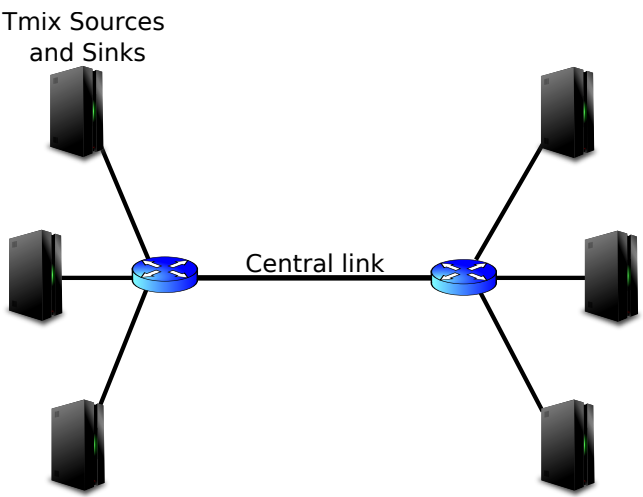
Traffic

- All traffic is TCP and uses the CC being investigated
- Loads of 60%, 85%, and 110%

Metrics

1. Aggregate link utilisation
2. the average packet drop rate
3. the average queueing delay

Latency oriented tests



Central Link modelled as:

access link with buffer sizes of {0.1,0.2,0.5, 1.0, 2.0} BDP

Traffic

CC being investigated and Standard CC in separate simulations

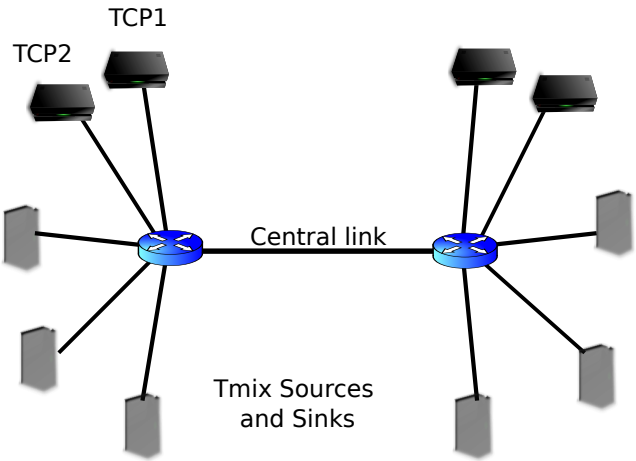
Metrics

1. Average throughput
2. average packet drop rate
3. average queueing delay

AQM

AQM efficacy not included.

Ramp up time



Central Link

10 Mbps and 1 Gbps

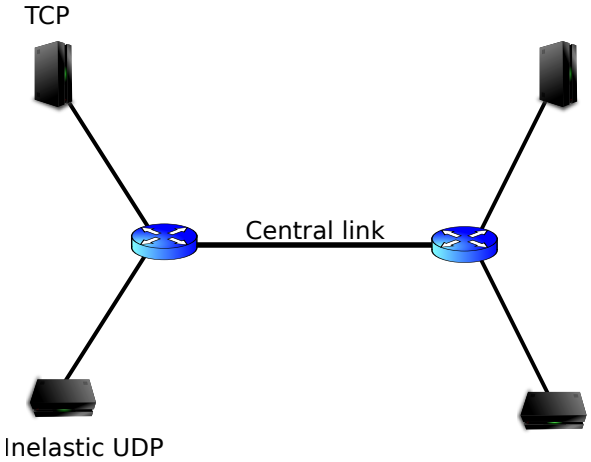
Traffic

- Background traffic load of 50%
- Two long lived test flows starting at different times.
- CC being investigated and Standard CC in separate simulations

Metrics

1. Time until receiver of test flow has received (1500×10^n)

Behaviour with transient traffic



Central Link

100 Mbps, RTT 100 ms, buffer 1024 packets (1.2BDP)

Traffic

- Inelastic UDP with step changes
- CC being investigated and Standard CC in separate simulations

Metrics

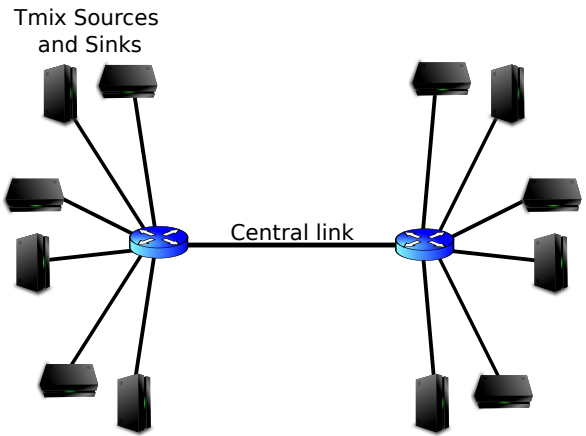
Step decrease:

1. time til {0.6,0.8,0.9}BDP window
2. maximum ΔWindow in an RTT

Step increase:

1. Harm: number of UDP packets dropped in next 100 s

Throughput and Fairness



Central Link

10 Mbps and 1 Gbps

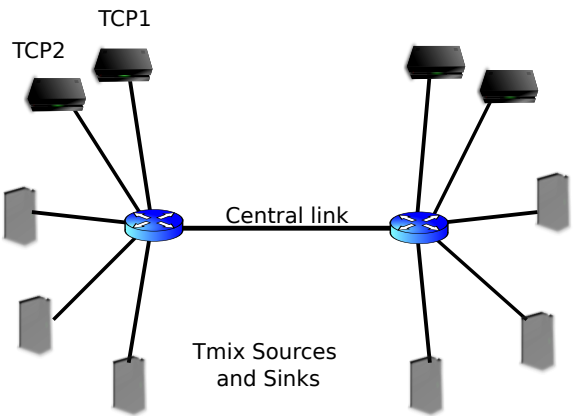
Traffic

- A/B test with identical TMIX spec
- Mix: CC being investigated (B) and Standard CC (A)
- Baseline: Standard TCP (B) and Standard CC (A)

Metrics

$$\text{Gain} = \frac{T^{(B)}_{\text{Mix}}}{T^{(B)}_{\text{Baseline}}} \quad \text{Loss} = \frac{T^{(A)}_{\text{Mix}}}{T^{(A)}_{\text{Baseline}}}$$

Intra-protocol and Inter-RTT fairness



Central Link

- 10 Mbps and 1 Gbps with 50% and 100% load
- RTT:
  - protocol fairness: TCP1=TCP2= {10,20,40,80,160}ms
  - rtt fairness: TCP1=160ms, TCP2={10,20,40,80,160}ms

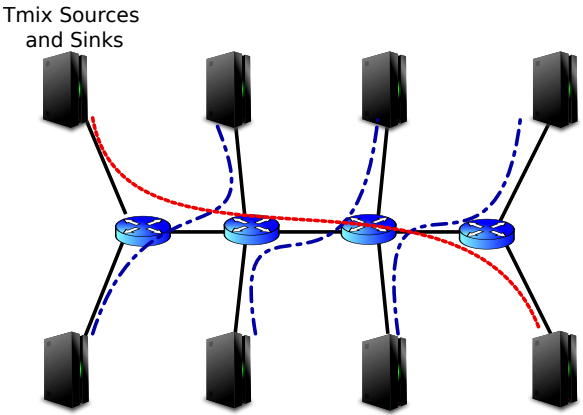
Traffic

CC being investigated

Metrics

Ratio = $\frac{\text{TCP2}}{\text{TCP1}}$

Multiple Bottlenecks



Bottleneck Links

- 100 Mbps, {60%,80%,100%} load
- RTT 60 ms

Traffic

CC being investigated

Metrics

Ratio = $\frac{\text{TCP}_{\text{multi}}}{E[\text{TCP1}]}$

A note on TMIX traffic

Real traffic traces

- From real captured TCP traffic
- **Socket level interaction trace**
  - Converts TCP session to socket layer interactions.
  - Allows TCP to be elastic for its session
  - E.g. interactive session, file download, simultaneous bidirectional transfer
- TMIX trace: time stamped connection vectors
- Originally available for both testbeds and simulations

Processing the TMIX trace

- Non-stationary (load varies over time)
  - This makes it difficult to use in evaluations
- But short term dynamics can be important
- Removing long term variations, keeping short term dynamics, and maintaining socket level interaction integrity
  - Divide TMIX connection vector start times into short blocks of a few seconds
    - maintaining short term dynamics
  - Shuffle the blocks

Trace location: `https://folk.universitetetioslo.no/michawe/research/tools/combined_traces.tbz`

More recent evaluation suites

TEACUP

- http://caia.swin.edu.au/tools/teacup/
- Dumbell testbed
- Dummynet/NetEm
- Iperf traffic
- Manages testbed
- graphs

Limitations

- Buy and build testbed
- Traffic generation
- Topologies
- Only long lived test flows

FLENT

- https://flent.org/
- Testbed with potential for different topologies
- Netperf/D-ITG traffic
- stats and graphs

Limitations

- Buy and build testbed
- Traffic generation
- Only long lived test flows

Others?

Work we do not know about

PANTHEON

- https://pantheon.stanford.edu/
- Publicly available testbed
  - cross Internet
  - (calibrated) emulated network (mahimahi emulator)
- stats and graphs
- continual experimentation

Limitations

- Emulated topology limited
- Only long lived test flows
- No longer seems to be in operation

Some of us are interested in reviving this work

Only with ICCRG support

- Who can help with implementations?
- Who can help with testing?
- Who can help with traffic traces?
- Who can help with discussions?

Suggested approach

1. Full ns-3 implementation of draft
   - some work already done
2. Updating
   1. tcpeval draft
   2. Traffic
      - Tmix traces
      - traffic models?
      - Elastic vs Inelastic traffic proportions
   3. AQM/ECN support
      - not an AQM testbed
   4. Support for non-TCP CC