Network Working Group Matt Mathis
INTERNET-DRAFT Pittsburgh Supercomputing Center
Expiration Date: Aug 1999 Feb 1999

TReno Bulk Transfer Capacity

< draft-ietf-ippm-treno-btc-03.txt >

Status of this Document

Abstract:

TReno is a tools to measure Bulk Transport Capacity (BTC) as defined in [ippm-btc-framework]. This document specifies specific details of the TReno algorithm as require by the BTC framework document.

2. Introduction:

This memo defines a Bulk Transport Capacity (BTC) based on the TReno (``tree-no'') diagnostic [Mathis97a]. It builds on notions introduced in the BTC framework document [ippm-btc-framework] and the IPPM Framework document, RFC 2330 [@@]; the reader is assumed to be familiar with both documents.

The BTC framework document defines pure Congestion Avoidance Capacity (CAC) as the data rate (bits per second) of the Congestion

Avoidance algorithm, subject to the restriction that the Retransmission Timeout and Slow-Start algorithms are not invoked. In principle a CAC metric would be an ideal BTC metric, but there are rather substantial difficulty with using it as such. The Self-Clocking of the Congestion Avoidance algorithm can be very fragile, depending on the specific details of the Fast Retransmit, Fast Recovery or other advanced recovery algorithms. When TCP looses Self-Clock it is reestablished through a retransmission timeout and Slow-Start. These algorithms nearly always take more time than Congestion Avoidance would have taken.

The TReno program implements BTC, CAC and ancillary metrics. The ancillary metrics are designed to instrument all network events that might cause discrepancies between an ideal CAC metric and the TReno BTC, other BTC metrics or real TCP implementations.

We use this multiple metrics approach because the CAC metric is more suitable for analytic modeling while the BTC metrics is more suited to applied measurement. We believe that future research will lead to a strong analytic framework (A-frame) [ippm-btc-framework] that will result in understanding the relationship between CAC metrics and other metrics, including simple metrics (delay, loss) as well as the various different BTC metrics and TCP implementations.

3. The TReno BTC Definition

3.1. Metric Name:

TReno-Type-P-Bulk-Transfer-Capacity

3.2. Metric Parameters:

+ Src, the IP address of a host

+ Dst, the IP address of a host

+ Initial Maximum Segment size

+ a test duration

+ T, a time

3.3. Metric Units:

Bits per second

3.4. Definition:

The average data rate attained by the TReno program over the path under test.

3.5 Congestion Control Algorithms

The BTC framework document [ippm-btc-framework] makes the observation that the standard specifying congestion control algorithms [RFC2001.bis] allows more latitude in their implementation than is appropriate for a metric. Some of the details of the congestion control algorithms that are left to the discretion of the implementor must be fully specified in a metric.

3.5.1 Congestion Avoidance details

TReno computes the window size in bytes. Each acknowledgment opens the congestion window (cwnd) by MSS*MSS/cwnd bytes. The actual number of outstanding bytes in the network is always an integral number of segments such that the total size is less than or equal to cwnd.

@@@ the framework needs to require that delayed Acks emulation be specified.

When a loss is detected the window is reduced using a algorithm that sends one segment per two acknowledgments for exactly one round trip (as determied by sequence numbers). This reduces the window to exactly half of the data that was actually held by the network at the time the first loss was detected. This algorithm, called Rate-Halving, is described in detail in a separate technical note [facknote]. The new cwnd will be (old_cwnd - loss)/2.

The technical not also describes an additional group of algoritms, collectivly called bounding parameters, that assure that rate halving always arrives at a reasonable congestioin window, even under pathological conditions. The bounding parameter algorithms have no effect on TReno under normal conditons. If the bounding parameters are invoked, they are instrumented and an exceptional network event.

The one of the bounding parameters is to set ssthresh to 1/4 of the pre-recovery cwnd. Thus recovery normally ends with cwnd larger than ssthresh, so TReno does not do a one segment slow-start as permitted by RFC2001. However, if more than half a window of data was lost, rate having can arrive at a new cwnd which is smaller than ssthresh, resulting in a slow-start up to ssthresh (which would be 1/4 the prior value of cwnd).

3.5.2 Retransmission Timeouts

The current version of TReno does not include an accurate model for the TCP retransmission timer. Under nearly all normal conditions the timers in TReno are much more conservative than real TCP implementations. TReno takes the view that timeouts indicate a failure to attain a CAC measurement, which an abnormality in the network that should be diagnosed. TReno doem not experience

timeouts unless an entire window of data is lost.

3.5.3 Slow-Start

TReno invokes Slow-start if cwnd is equal to or less than ssthresh. Unlike most TCP implementations this condition is not normally true at the end of recovery.

3.5.4 Advanced Recovery Algorithms

The algorithm used by TReno to emulate the TCP reassembly queue naturally emulates SACK [RFC2018] with the Forward Acknowledgment Algorithm [Mathis96] as updated by [facknote].

3.5.5 Segment Size

TReno can dynamicly discover the correct Maximum Segment Size through path MTU discovery. A smaller MTU can be explicitly selected.

3.6 Ancillary results:

@@@ expand

- Statistics over the entire test
  (data transferred, duration and average rate)
- Statistics over the Congestion Avoidance portion of the test
  (data transferred, duration and average rate)
- Path property statistics (MTU, minimum RTT, maximum congestion window during Congestion Avoidance and during Slow-start)
- Direct measures of the analytic model parameters (Number of congestion signals, average RTT)
- Indications of which TCP algorithms must be present to attain the same performance.
- The estimated load/BW/buffering used on the return path
- Warnings about data transmission abnormalities.
  (e.g. packets out-of-order, events that cause timeouts)
- Warnings about conditions which may affect metric accuracy. (e.g. insufficient tester buffering)
- Alarms about serious data transmission abnormalities.
  (e.g. data duplicated in the network)
- Alarms about internal inconsistencies of the tester and events which might invalidate the results.
- IP address/name of the responding target.
- TReno version.

3.7 Manual calibration checks:

The following discussion assumes that the TReno diagnostic is implemented as a user mode program running under a standard operating system. Other implementations, such as those in dedicated measurement instruments, can have stronger built-in calibration checks.

3.7.1 Tester performance

Verify that the tester and target have sufficient data rates to sustain the test.

The raw performance (data rate) limitations of both the tester and target should be measured by running TReno in a controlled environment (e.g. a bench test). Ideally the observed performance limits should be validated by determining the nature of the bottleneck and verifying that it agrees with other benchmarks of the tester and target (e.g. That TReno performance agrees with direct measures of backplane or memory bandwidth or other bottleneck as appropriate). Currently no routers are reliable targets, although under some conditions they can be used for meaningful measurements. When testing between a pair of modern computer systems at a few megabits per second or less, the tester and target are unlikely to be the bottleneck.

TReno may be less accurate at average rates above half of the known tester or target limits. This is because during the initial Slow-start TReno needs to send bursts which are twice the average data rate.

Likewise, if the link to the first hop is not more than twice as fast as the entire path, some of the path properties such as max congestion window during Slow-start may reflect the testers link interface, and not the path itself.

3.7.2 Tester Buffering

Verify that the tester and target have sufficient buffering to support the window needed by the test.

If they do not have sufficient buffer space, then losses at their own queues may contribute to the apparent losses along the path. There are several difficulties in verifying the tester and target buffer capacity. First, there are no good tests of the targets buffer capacity at all. Second, all validation of the testers buffering depends in some way on the accuracy of reports by the tester's own operating system. Third, there is the confusing result that under many circumstances (particularly when there is much more than sufficient average tester performance) insufficient buffering in the tester does not adversely impact measured performance.

TReno reports (as calibration alarms) any events in which transmit packets were refused due to insufficient buffer space. It reports a warning if the maximum measured congestion window is larger than the reported buffer space. Although these checks are likely to be sufficient in most cases they are probably not sufficient in all cases, and will be the subject of future research.

Note that on a timesharing or multi-tasking system, other activity on the tester introduces burstiness due to operating system scheduler latency. Since some queuing disciplines discriminate against bursty sources, it is important that there be no other system activity during a test. This should be confirmed with other operating system specific tools.

3.7.3 Return Path performance

Verify that the return path is not a bottleneck at the load needed to sustain the test.

In ICMP mode TReno measures the net effect of both the forward and return paths on a single data stream. Bottlenecks and packet losses in the forward and return paths are treated equally.

In traceroute mode, TReno computes and reports the load it contributes to the return path. Unlike real TCP, TReno can not distinguish between losses on the forward and return paths, so ideally we want the return path to introduce as little loss as possible. A good way to test to see if the return path has a large effect on a measurement is to reduce the forward path messages down to ACK size (40 bytes), and verify that the measured packet rate is improved by at least factor of two. [More research is needed.]

3.8 Discussion:

There are many possible reasons why a TReno measurement might not agree with the performance obtained by a TCP-based application. Some key ones include: older TCPs missing key algorithms such as MTU discovery, support for large windows or SACK, or miss-tuning of either the data source or sink. Network conditions which require the newer TCP algorithms are detected by TReno and reported in the ancillary results. Other documents will cover methods to diagnose the difference between TReno and TCP performance.

People using the TReno metric as part of procurement documents should be aware that in many circumstances MTU has an intrinsic and large impact on overall path performance. Under some conditions the difficulty in meeting a given performance specifications is inversely proportional to the square of the path MTU. (e.g. Halving the specified MTU makes meeting the bandwidth specification 4 times harder.)

When used as an end-to-end metric TReno presents exactly the same load to the network as a properly tuned state-of-the-art bulk TCP stream between the same pair of hosts. Although the connection is not transferring useful data, it is no more wasteful than fetching an unwanted web page with the same transfer time.

References

[Jacobson88] Jacobson, V., "Congestion Avoidance and Control", Proceedings of SIGCOMM '88, Stanford, CA., August 1988.

[mathis96] Mathis, M. and Mahdavi, J. "Forward acknowledgment: Refining TCP congestion control", Proceedings of ACM SIGCOMM '96, Stanford, CA., August 1996.

[RFC2018] Mathis, M., Mahdavi, J. Floyd, S., Romanow, A., "TCP Selective Acknowledgment Options", 1996 Obtain via: ftp://ds.internic.net/rfc/rfc2018.txt

[Mathis97a] Mathis, M., TReno source distribution, Obtain via: ftp://ftp.psc.edu/pub/networking/tools/treno.shar

[Mathis97b] Mathis, M., Semke, J., Mahdavi, J., Ott, T., "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", Computer Communications Review, 27(3), July 1997.

[RFC2001] Stevens, W., "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", ftp://ds.internic.net/rfc/rfc2001.txt

[facknote] Mathis, M., Mahdavi, M., TCP Rate-Halving with Bounding Parameters http://www.psc.edu/networking/papers/FACKnotes/current/

Author's Address

Matt Mathis
email: mathis@psc.edu
Pittsburgh Supercomputing Center
4400 Fifth Ave.
Pittsburgh PA 15213