Internet Engineering Task Force (IETF)
Request for Comments: 6349
Category: Informational
ISSN: 2070-1721

B. Constantine
JDSU
G. Forget
Bell Canada (Ext. Consultant)
R. Geib
Deutsche Telekom
R. Schrage
Schrage Consulting

August 2011

Framework for TCP Throughput Testing

Abstract

This framework describes a practical methodology for measuring end-to-end TCP Throughput in a managed IP network. The goal is to provide a better indication in regard to user experience. In this framework, TCP and IP parameters are specified to optimize TCP Throughput.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6349.

Table of Contents

1. Introduction

In the network industry, the SLA (Service Level Agreement) provided to business-class customers is generally based upon Layer 2/3 criteria such as bandwidth, latency, packet loss, and delay variations (jitter). Network providers are coming to the realization that Layer 2/3 testing is not enough to adequately ensure end-users' satisfaction. In addition to Layer 2/3 testing, this framework recommends a methodology for measuring TCP Throughput in order to provide meaningful results with respect to user experience.

Additionally, business-class customers seek to conduct repeatable TCP Throughput tests between locations. Since these organizations rely on the networks of the providers, a common test methodology with predefined metrics would benefit both parties.

Note that the primary focus of this methodology is managed business-class IP networks, e.g., those Ethernet-terminated services for which organizations are provided an SLA from the network provider. Because of the SLA, the expectation is that the TCP Throughput should achieve the guaranteed bandwidth. End-users with "best effort" access could use this methodology, but this framework and its metrics are intended to be used in a predictable managed IP network. No end-to-end performance can be guaranteed when only the access portion is being provisioned to a specific bandwidth capacity.

The intent behind this document is to define a methodology for testing sustained TCP Layer performance. In this document, the achievable TCP Throughput is that amount of data per unit of time that TCP transports when in the TCP Equilibrium state. (See Section 1.3 for the TCP Equilibrium definition). Throughout this document, "maximum achievable throughput" refers to the theoretical achievable throughput when TCP is in the Equilibrium state.

TCP is connection oriented, and at the transmitting side, it uses a congestion window (TCP CWND). At the receiving end, TCP uses a receive window (TCP RWND) to inform the transmitting end on how many Bytes it is capable of accepting at a given time.

Derived from Round-Trip Time (RTT) and network Bottleneck Bandwidth (BB), the Bandwidth-Delay Product (BDP) determines the Send and Received Socket buffer sizes required to achieve the maximum TCP Throughput. Then, with the help of slow start and congestion avoidance algorithms, a TCP CWND is calculated based on the IP network path loss rate. Finally, the minimum value between the calculated TCP CWND and the TCP RWND advertised by the opposite end will determine how many Bytes can actually be sent by the transmitting side at a given time.

Both TCP Window sizes (RWND and CWND) may vary during any given TCP session, although up to bandwidth limits, larger RWND and larger CWND will achieve higher throughputs by permitting more in-flight Bytes.

At both ends of the TCP connection and for each socket, there are default buffer sizes. There are also kernel-enforced maximum buffer sizes. These buffer sizes can be adjusted at both ends (transmitting and receiving). Some TCP/IP stack implementations use Receive Window Auto-Tuning, although, in order to obtain the maximum throughput, it is critical to use large enough TCP Send and Receive Socket Buffer sizes. In fact, they SHOULD be equal to or greater than BDP.

Many variables are involved in TCP Throughput performance, but this methodology focuses on the following:

- BB (Bottleneck Bandwidth)

- RTT (Round-Trip Time)

- Send and Receive Socket Buffers

- Minimum TCP RWND

- Path MTU (Maximum Transmission Unit)

This methodology proposes TCP testing that SHOULD be performed in addition to traditional tests of the Layer 2/3 type. In fact, Layer 2/3 tests are REQUIRED to verify the integrity of the network before conducting TCP tests. Examples include "iperf" (UDP mode) and manual packet-layer test techniques where packet throughput, loss, and delay measurements are conducted. When available, standardized testing similar to [RFC2544], but adapted for use in operational networks, MAY be used.

Note: [RFC2544] was never meant to be used outside a lab environment.

Sections 2 and 3 of this document provide a general overview of the proposed methodology. Section 4 defines the metrics, while Section 5 explains how to conduct the tests and interpret the results.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. Terminology

The common definitions used in this methodology are as follows:

- TCP Throughput Test Device (TCP TTD) refers to a compliant TCP host that generates traffic and measures metrics as defined in this methodology, i.e., a dedicated communications test instrument.

- Customer Provided Equipment (CPE) refers to customer-owned equipment (routers, switches, computers, etc.).

- Customer Edge (CE) refers to a provider-owned demarcation device.

- Provider Edge (PE) refers to a provider's distribution equipment.

- Bottleneck Bandwidth (BB) refers to the lowest bandwidth along the complete path. "Bottleneck Bandwidth" and "Bandwidth" are used synonymously in this document. Most of the time, the Bottleneck Bandwidth is in the access portion of the wide-area network (CE - PE).

- Provider (P) refers to provider core network equipment.

- Network Under Test (NUT) refers to the tested IP network path.

- Round-Trip Time (RTT) is the elapsed time between the clocking in of the first bit of a TCP segment sent and the receipt of the last bit of the corresponding TCP Acknowledgment.

- Bandwidth-Delay Product (BDP) refers to the product of a data link's capacity (in bits per second) and its end-to-end delay (in seconds).

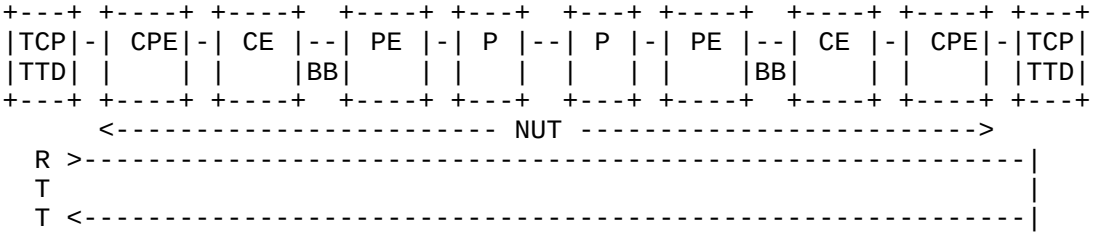


Figure 1.2. Devices, Links, and Paths

Note that the NUT may be built with a variety of devices including, but not limited to, load balancers, proxy servers, or WAN acceleration appliances. The detailed topology of the NUT SHOULD be well-known when conducting the TCP Throughput tests, although this methodology makes no attempt to characterize specific network architectures.

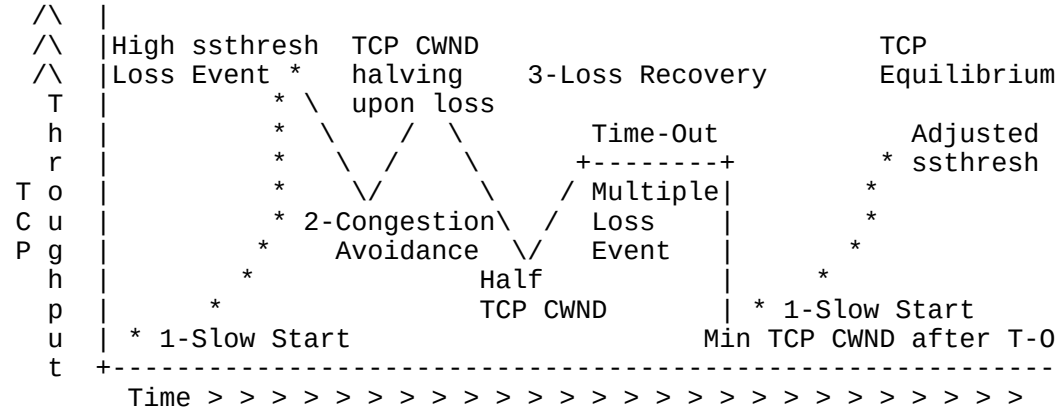
1.3. TCP Equilibrium

TCP connections have three (3) fundamental congestion window phases, which are depicted in Figure 1.3.

1. The Slow Start phase, which occurs at the beginning of a TCP transmission or after a retransmission Time-Out.

2. The Congestion Avoidance phase, during which TCP ramps up to establish the maximum achievable throughput. It is important to note that retransmissions are a natural by-product of the TCP congestion avoidance algorithm as it seeks to achieve maximum throughput.

3. The Loss Recovery phase, which could include Fast Retransmit (Tahoe) or Fast Recovery (Reno and New Reno). When packet loss occurs, the Congestion Avoidance phase transitions either to Fast Retransmission or Fast Recovery, depending upon the TCP implementation. If a Time-Out occurs, TCP transitions back to the Slow Start phase.



Note: ssthresh = Slow Start threshold.

Figure 1.3. TCP CWND Phases

A well-tuned and well-managed IP network with appropriate TCP adjustments in the IP hosts and applications should perform very close to the BB when TCP is in the Equilibrium state.

This TCP methodology provides guidelines to measure the maximum achievable TCP Throughput when TCP is in the Equilibrium state. All maximum achievable TCP Throughputs specified in Section 3.3 are with respect to this condition.

It is important to clarify the interaction between the sender's Send Socket Buffer and the receiver's advertised TCP RWND size. TCP test programs such as "iperf", "ttcp", etc. allow the sender to control the quantity of TCP Bytes transmitted and unacknowledged (in-flight), commonly referred to as the Send Socket Buffer. This is done independently of the TCP RWND size advertised by the receiver.

2. Scope and Goals

Before defining the goals, it is important to clearly define the areas that are out of scope.

- This methodology is not intended to predict the TCP Throughput during the transient stages of a TCP connection, such as during the Slow Start phase.

- This methodology is not intended to definitively benchmark TCP implementations of one OS to another, although some users may find value in conducting qualitative experiments.

- This methodology is not intended to provide detailed diagnosis of problems within endpoints or within the network itself as related to non-optimal TCP performance, although results interpretation for each test step may provide insights to potential issues.

- This methodology does not propose to operate permanently with high measurement loads. TCP performance and optimization within operational networks MAY be captured and evaluated by using data from the "TCP Extended Statistics MIB" [RFC4898].

In contrast to the above exclusions, the primary goal is to define a method to conduct a practical end-to-end assessment of sustained TCP performance within a managed business-class IP network. Another key goal is to establish a set of "best practices" that a non-TCP expert SHOULD apply when validating the ability of a managed IP network to carry end-user TCP applications.

Specific goals are to:

- Provide a practical test approach that specifies tunable parameters (such as MTU (Maximum Transmission Unit) and Socket Buffer sizes) and how these affect the outcome of TCP performance over an IP network.

- Provide specific test conditions such as link speed, RTT, MTU, Socket Buffer sizes, and achievable TCP Throughput when TCP is in the Equilibrium state. For guideline purposes, provide examples of test conditions and their maximum achievable TCP Throughput. Section 1.3 provides specific details concerning the definition of TCP Equilibrium within this methodology, while Section 3 provides specific test conditions with examples.

- Define three (3) basic metrics to compare the performance of TCP connections under various network conditions. See Section 4.

- Provide some areas within the end host or the network that SHOULD be considered for investigation in test situations where the recommended procedure does not yield the maximum achievable TCP Throughput. However, this methodology is not intended to provide detailed diagnosis on these issues. See Section 5.2.

3. Methodology

This methodology is intended for operational and managed IP networks. A multitude of network architectures and topologies can be tested. The diagram in Figure 1.2 is very general and is only provided to illustrate typical segmentation within end-user and network provider domains.

Also, as stated in Section 1, it is considered best practice to verify the integrity of the network by conducting Layer 2/3 tests such as [RFC2544] or other methods of network stress tests; although it is important to mention here that [RFC2544] was never meant to be used outside a lab environment.

It is not possible to make an accurate TCP Throughput measurement when the network is dysfunctional. In particular, if the network is exhibiting high packet loss and/or high jitter, then TCP Layer Throughput testing will not be meaningful. As a guideline, 5% packet loss and/or 150 ms of jitter may be considered too high for an accurate measurement.

TCP Throughput testing may require cooperation between the end-user customer and the network provider. As an example, in an MPLS (Multiprotocol Label Switching) network architecture, the testing SHOULD be conducted either on the CPE or on the CE device and not on the PE (Provider Edge) router.

The following represents the sequential order of steps for this testing methodology:

1. Identify the Path MTU. Packetization Layer Path MTU Discovery (PLPMTUD) [RFC4821] SHOULD be conducted. It is important to identify the path MTU so that the TCP TTD is configured properly to avoid fragmentation.

2. Baseline Round-Trip Time and Bandwidth. This step establishes the inherent, non-congested Round-Trip Time (RTT) and the Bottleneck Bandwidth (BB) of the end-to-end network path. These measurements are used to provide estimates of the TCP RWND and Send Socket Buffer sizes that SHOULD be used during subsequent test steps.

3. TCP Connection Throughput Tests. With baseline measurements of Round-Trip Time and Bottleneck Bandwidth, single- and multiple-TCP-connection throughput tests SHOULD be conducted to baseline network performance.

These three (3) steps are detailed in Sections 3.1 to 3.3.

Important to note are some of the key characteristics and considerations for the TCP test instrument. The test host MAY be a standard computer or a dedicated communications test instrument. In both cases, it MUST be capable of emulating both a client and a server.

The following criteria SHOULD be considered when selecting whether the TCP test host can be a standard computer or has to be a dedicated communications test instrument:

- TCP implementation used by the test host, OS version (e.g., LINUX OS kernel using TCP New Reno), TCP options supported, etc. will obviously be more important when using dedicated communications test instruments where the TCP implementation may be customized or tuned to run in higher-performance hardware. When a compliant TCP TTD is used, the TCP implementation SHOULD be identified in the test results. The compliant TCP TTD SHOULD be usable for complete end-to-end testing through network security elements and SHOULD also be usable for testing network sections.

- More importantly, the TCP test host MUST be capable of generating and receiving stateful TCP test traffic at the full BB of the NUT. Stateful TCP test traffic means that the test host MUST fully implement a TCP/IP stack; this is generally a comment aimed at dedicated communications test equipment that sometimes "blasts" packets with TCP headers. At the time of this publication, testing TCP Throughput at rates greater than 100 Mbps may require high-performance server hardware or dedicated hardware-based test tools.

- A compliant TCP Throughput Test Device MUST allow adjusting both Send and Receive Socket Buffer sizes. The Socket Buffers MUST be large enough to fill the BDP.

- Measuring RTT and retransmissions per connection will generally require a dedicated communications test instrument. In the absence of dedicated hardware-based test tools, these measurements may need to be conducted with packet capture tools, i.e., conduct TCP Throughput tests and analyze RTT and retransmissions in packet captures. Another option MAY be to use the "TCP Extended Statistics MIB" [RFC4898].

- The [RFC4821] PLPMTUD test SHOULD be conducted with a dedicated tester that exposes the ability to run the PLPMTUD algorithm independently from the OS stack.

3.1. Path MTU

TCP implementations should use Path MTU Discovery techniques (PMTUD). PMTUD relies on ICMP 'need to frag' messages to learn the path MTU. When a device has a packet to send that has the Don't Fragment (DF) bit in the IP header set and the packet is larger than the MTU of the next hop, the packet is dropped, and the device sends an ICMP 'need to frag' message back to the host that originated the packet. The ICMP 'need to frag' message includes the next-hop MTU, which PMTUD uses to adjust itself. Unfortunately, because many network managers completely disable ICMP, this technique does not always prove reliable.

Packetization Layer Path MTU Discovery (PLPMTUD) [RFC4821] MUST then be conducted to verify the network path MTU. PLPMTUD can be used with or without ICMP. [RFC4821] specifies search_high and search_low parameters for the MTU, and we recommend using those parameters. The goal is to avoid fragmentation during all subsequent tests.

3.2. Round-Trip Time (RTT) and Bottleneck Bandwidth (BB)

Before stateful TCP testing can begin, it is important to determine the baseline RTT (i.e., non-congested inherent delay) and BB of the end-to-end network to be tested. These measurements are used to calculate the BDP and to provide estimates of the TCP RWND and Send Socket Buffer sizes that SHOULD be used in subsequent test steps.

3.2.1. Measuring RTT

As previously defined in Section 1.2, RTT is the elapsed time between the clocking in of the first bit of a TCP segment sent and the receipt of the last bit of the corresponding TCP Acknowledgment.

The RTT SHOULD be baselined during off-peak hours in order to obtain a reliable figure of the inherent network latency. Otherwise, additional delay caused by network buffering can occur. Also, when sampling RTT values over a given test interval, the minimum measured value SHOULD be used as the baseline RTT. This will most closely estimate the real inherent RTT. This value is also used to determine the Buffer Delay Percentage metric defined in Section 4.3.

The following list is not meant to be exhaustive, although it summarizes some of the most common ways to determine Round-Trip Time. The desired measurement precision (i.e., ms versus us) may dictate whether the RTT measurement can be achieved with ICMP pings or by a dedicated communications test instrument with precision timers. The objective of this section is to list several techniques in order of decreasing accuracy.

- Use test equipment on each end of the network, "looping" the far-end tester so that a packet stream can be measured back and forth from end to end. This RTT measurement may be compatible with delay measurement protocols specified in [RFC5357].

- Conduct packet captures of TCP test sessions using "iperf" or FTP, or other TCP test applications. By running multiple experiments, packet captures can then be analyzed to estimate RTT. It is important to note that results based upon the SYN -> SYN-ACK at the beginning of TCP sessions SHOULD be avoided, since Firewalls might slow down 3-way handshakes. Also, at the sender’s side, Ostermann’s LINUX TCPTRACE utility with -l -r arguments can be used to extract the RTT results directly from the packet captures.

- Obtain RTT statistics available from MIBs defined in [RFC4898].

- ICMP pings may also be adequate to provide Round-Trip Time estimates, provided that the packet size is factored into the estimates (i.e., pings with different packet sizes might be required). Some limitations with ICMP ping may include ms resolution and whether or not the network elements are responding to pings. Also, ICMP is often rate-limited or segregated into different buffer queues. ICMP might not work if QoS (Quality of Service) reclassification is done at any hop. ICMP is not as reliable and accurate as in-band measurements.

3.2.2. Measuring BB

Before any TCP Throughput test can be conducted, bandwidth measurement tests SHOULD be run with stateless IP streams (i.e., not stateful TCP) in order to determine the BB of the NUT. These measurements SHOULD be conducted in both directions, especially in asymmetrical access networks (e.g., Asymmetric Bit-Rate DSL (ADSL) access). These tests SHOULD be performed at various intervals throughout a business day or even across a week.

Testing at various time intervals would provide a better characterization of TCP Throughput and better diagnosis insight (for cases where there are TCP performance issues). The bandwidth tests SHOULD produce logged outputs of the achieved bandwidths across the complete test duration.

There are many well-established techniques available to provide estimated measures of bandwidth over a network. It is a common practice for network providers to conduct Layer 2/3 bandwidth capacity tests using [RFC2544], although it is understood that [RFC2544] was never meant to be used outside a lab environment. These bandwidth measurements SHOULD use network capacity techniques as defined in [RFC5136].

3.3. Measuring TCP Throughput

This methodology specifically defines TCP Throughput measurement techniques to verify maximum achievable TCP performance in a managed business-class IP network.

With baseline measurements of RTT and BB from Section 3.2, a series of single- and/or multiple-TCP-connection throughput tests SHOULD be conducted.

The number of trials and the choice between single or multiple TCP connections will be based on the intention of the test. A single-TCP-connection test might be enough to measure the achievable throughput of Metro Ethernet connectivity. However, it is important

to note that various traffic management techniques can be used in an IP network and that some of those techniques can only be tested with multiple connections. As an example, multiple TCP sessions might be required to detect traffic shaping versus policing. Multiple sessions might also be needed to measure Active Queue Management performance. However, traffic management testing is not within the scope of this test methodology.

In all circumstances, it is RECOMMENDED to run the tests in each direction independently first and then to run them in both directions simultaneously. It is also RECOMMENDED to run the tests at different times of the day.

In each case, the TCP Transfer Time Ratio, the TCP Efficiency Percentage, and the Buffer Delay Percentage MUST be measured in each direction. These 3 metrics are defined in Section 4.

3.3.1. Minimum TCP RWND

The TCP TTD MUST allow the Send Socket Buffer and Receive Window sizes to be set higher than the BDP; otherwise, TCP performance will be limited. In the business customer environment, these settings are not generally adjustable by the average user. These settings are either hard-coded in the application or configured within the OS as part of a corporate image. In many cases, the user’s host Send Socket Buffer and Receive Window size settings are not optimal.

This section provides derivations of BDPs under various network conditions. It also provides examples of achievable TCP Throughput with various TCP RWND sizes. This provides important guidelines showing what can be achieved with settings higher than the BDP, versus what would be achieved in a variety of real-world conditions.

The minimum required TCP RWND size can be calculated from the Bandwidth-Delay Product (BDP), which is as follows:

```
BDP (bits) = RTT (sec) X BB (bps)
```

Note that the RTT is being used as the "Delay" variable for the BDP. Then, by dividing the BDP by 8, we obtain the minimum required TCP RWND size in Bytes. For optimal results, the Send Socket Buffer MUST be adjusted to the same value at each end of the network.

```
Minimum required TCP RWND = BDP / 8
```

As an example, on a T3 link with 25-ms RTT, the BDP would equal ~1,105,000 bits, and the minimum required TCP RWND would be ~138 KB.

Note that separate calculations are REQUIRED on asymmetrical paths. An asymmetrical-path example would be a 90-ms RTT ADSL line with 5 Mbps downstream and 640 Kbps upstream. The downstream BDP would equal ~450,000 bits, while the upstream one would be only ~57,600 bits.

The following table provides some representative network link speeds, RTT, BDP, and their associated minimum required TCP RWND sizes.

Link Speed* (Mbps)	RTT (ms)	BDP (bits)	Minimum Required TCP RWND (KBytes)
1.536	20.00	30,720	3.84
1.536	50.00	76,800	9.60
1.536	100.00	153,600	19.20
44.210	10.00	442,100	55.26
44.210	15.00	663,150	82.89
44.210	25.00	1,105,250	138.16
100.000	1.00	100,000	12.50
100.000	2.00	200,000	25.00
100.000	5.00	500,000	62.50
1,000.000	0.10	100,000	12.50
1,000.000	0.50	500,000	62.50
1,000.000	1.00	1,000,000	125.00
10,000.000	0.05	500,000	62.50
10,000.000	0.30	3,000,000	375.00

* Note that link speed is the BB for the NUT

Table 3.3.1. Link Speed, RTT, Calculated BDP, and Minimum TCP RWND

In the above table, the following serial link speeds are used:

- T1 = 1.536 Mbps (for a B8ZS line encoding facility)
- T3 = 44.21 Mbps (for a C-Bit framing facility)

The previous table illustrates the minimum required TCP RWND. If a smaller TCP RWND size is used, then the TCP Throughput cannot be optimal. To calculate the TCP Throughput, the following formula is used:

TCP Throughput = TCP RWND X 8 / RTT

An example could be a 100-Mbps IP path with 5-ms RTT and a TCP RWND of 16 KB; then:

```
TCP Throughput = 16 KBytes X 8 bits / 5 ms
TCP Throughput = 128,000 bits / 0.005 sec
TCP Throughput = 25.6 Mbps
```

Another example, for a T3 using the same calculation formula, is illustrated in Figure 3.3.1a:

```
TCP Throughput = 16 KBytes X 8 bits / 10 ms
TCP Throughput = 128,000 bits / 0.01 sec
TCP Throughput = 12.8 Mbps*
```

When the TCP RWND size exceeds the BDP (T3 link and 64-KByte TCP RWND on a 10-ms RTT path), the maximum Frames Per Second (FPS) limit of 3664 is reached, and then the formula is:

```
TCP Throughput = max FPS X (MTU - 40) X 8
TCP Throughput = 3664 FPS X 1460 Bytes X 8 bits
TCP Throughput = 42.8 Mbps**
```

The following diagram compares achievable TCP Throughputs on a T3 with Send Socket Buffer and TCP RWND sizes of 16 KB versus 64 KB.

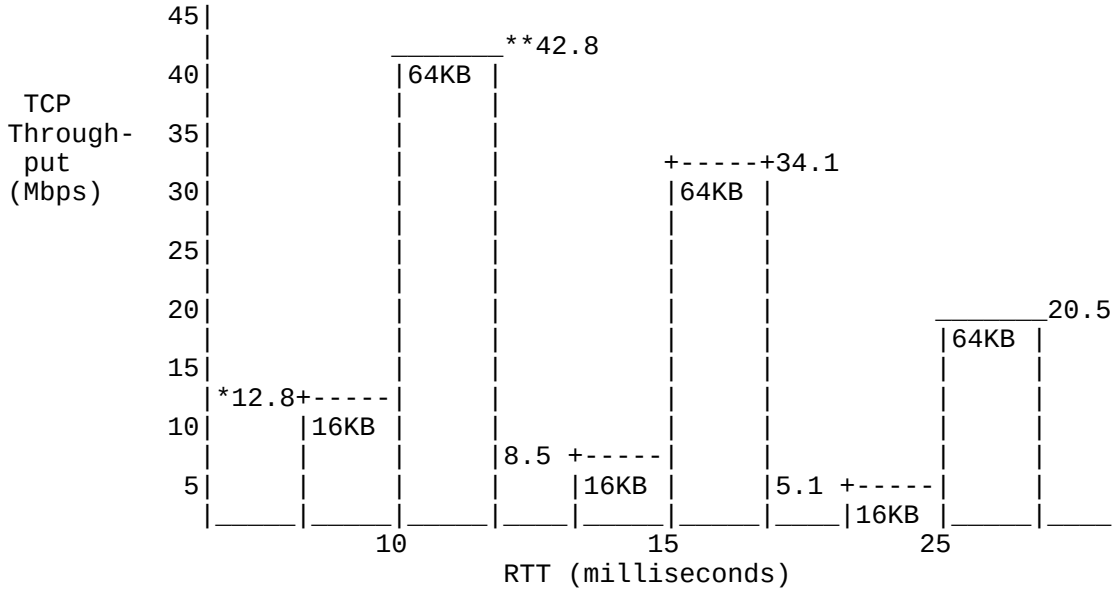
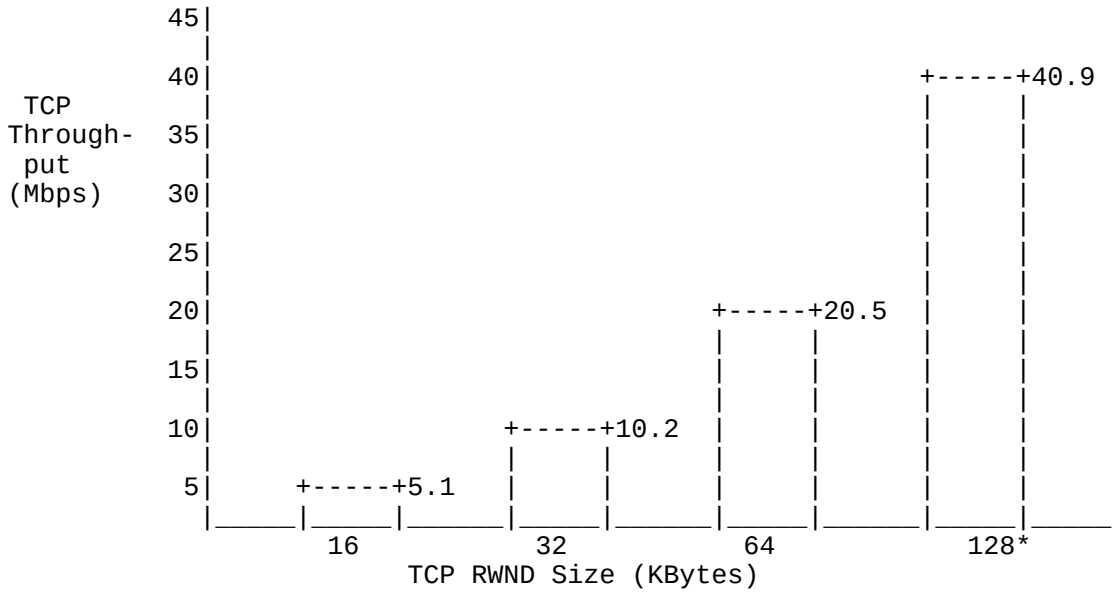


Figure 3.3.1a. TCP Throughputs on a T3 at Different RTTs

The following diagram shows the achievable TCP Throughput on a 25-ms T3 when Send Socket Buffer and TCP RWND sizes are increased.



* Note that 128 KB requires the [RFC1323] TCP Window Scale option.

Figure 3.3.1b. TCP Throughputs on a T3 with Different TCP RWND

4. TCP Metrics

This methodology focuses on a TCP Throughput and provides 3 basic metrics that can be used for better understanding of the results. It is recognized that the complexity and unpredictability of TCP makes it very difficult to develop a complete set of metrics that accounts for the myriad of variables (i.e., RTT variations, loss conditions, TCP implementations, etc.). However, these 3 metrics facilitate TCP Throughput comparisons under varying network conditions and host buffer size/RWND settings.

4.1. Transfer Time Ratio

The first metric is the TCP Transfer Time Ratio, which is simply the ratio between the Actual TCP Transfer Time versus the Ideal TCP Transfer Time.

The Actual TCP Transfer Time is simply the time it takes to transfer a block of data across TCP connection(s).

The Ideal TCP Transfer Time is the predicted time for which a block of data SHOULD transfer across TCP connection(s), considering the BB of the NUT.

$$\text{TCP Transfer Time Ratio} = \frac{\text{Actual TCP Transfer Time}}{\text{Ideal TCP Transfer Time}}$$

The Ideal TCP Transfer Time is derived from the Maximum Achievable TCP Throughput, which is related to the BB and Layer 1/2/3/4 overheads associated with the network path. The following sections provide derivations for the Maximum Achievable TCP Throughput and example calculations for the TCP Transfer Time Ratio.

4.1.1. Maximum Achievable TCP Throughput Calculation

This section provides formulas to calculate the Maximum Achievable TCP Throughput, with examples for T3 (44.21 Mbps) and Ethernet.

All calculations are based on IP version 4 with TCP/IP headers of 20 Bytes each (20 for TCP + 20 for IP) within an MTU of 1500 Bytes.

First, the maximum achievable Layer 2 throughput of a T3 interface is limited by the maximum quantity of Frames Per Second (FPS) permitted by the actual physical layer (Layer 1) speed.

The calculation formula is:

```
FPS = T3 Physical Speed / ((MTU + PPP + Flags + CRC16) X 8)

FPS = (44.21 Mbps /
          ((1500 Bytes + 4 Bytes + 2 Bytes + 2 Bytes) X 8 )))
FPS = (44.21 Mbps / (1508 Bytes X 8))
FPS = 44.21 Mbps / 12064 bits
FPS = 3664
```

Then, to obtain the Maximum Achievable TCP Throughput (Layer 4), we simply use:

```
(MTU - 40) in Bytes X 8 bits X max FPS
```

For a T3, the maximum TCP Throughput =

```
1460 Bytes X 8 bits X 3664 FPS

Maximum TCP Throughput = 11680 bits X 3664 FPS
Maximum TCP Throughput = 42.8 Mbps
```

On Ethernet, the maximum achievable Layer 2 throughput is limited by the maximum Frames Per Second permitted by the IEEE802.3 standard.

The maximum FPS for 100-Mbps Ethernet is 8127, and the calculation formula is:

FPS = (100 Mbps / (1538 Bytes X 8 bits))

The maximum FPS for GigE is 81274, and the calculation formula is:

FPS = (1 Gbps / (1538 Bytes X 8 bits))

The maximum FPS for 10GigE is 812743, and the calculation formula is:

FPS = (10 Gbps / (1538 Bytes X 8 bits))

The 1538 Bytes equates to:

MTU + Ethernet + CRC32 + IFG + Preamble + SFD
(IFG = Inter-Frame Gap and SFD = Start of Frame Delimiter)

where MTU is 1500 Bytes, Ethernet is 14 Bytes, CRC32 is 4 Bytes, IFG is 12 Bytes, Preamble is 7 Bytes, and SFD is 1 Byte.

Then, to obtain the Maximum Achievable TCP Throughput (Layer 4), we simply use:

(MTU - 40) in Bytes X 8 bits X max FPS

For 100-Mbps Ethernet, the maximum TCP Throughput =

1460 Bytes X 8 bits X 8127 FPS

Maximum TCP Throughput = 11680 bits X 8127 FPS
Maximum TCP Throughput = 94.9 Mbps

It is important to note that better results could be obtained with jumbo frames on Gigabit and 10-Gigabit Ethernet interfaces.

4.1.2. TCP Transfer Time and Transfer Time Ratio Calculation

The following table illustrates the Ideal TCP Transfer Time of a single TCP connection when its TCP RWND and Send Socket Buffer sizes equal or exceed the BDP.

Link Speed (Mbps)	RTT (ms)	BDP (KBytes)	Maximum Achievable TCP Throughput(Mbps)	Ideal TCP Transfer Time (seconds)*
1.536	50.00	9.6	1.4	571.0
44.210	25.00	138.2	42.8	18.0
100.000	2.00	25.0	94.9	9.0
1,000.000	1.00	125.0	949.2	1.0
10,000.000	0.05	62.5	9,492.0	0.1

* Transfer times are rounded for simplicity.

Table 4.1.2. Link Speed, RTT, BDP, TCP Throughput, and Ideal TCP Transfer Time for a 100-MB File

For a 100-MB file (100 X 8 = 800 Mbits), the Ideal TCP Transfer Time is derived as follows:

$$\text{Ideal TCP Transfer Time} = \frac{800 \text{ Mbits}}{\text{Maximum Achievable TCP Throughput}}$$

To illustrate the TCP Transfer Time Ratio, an example would be the bulk transfer of 100 MB over 5 simultaneous TCP connections (each connection transferring 100 MB). In this example, the Ethernet service provides a Committed Access Rate (CAR) of 500 Mbps. Each connection may achieve different throughputs during a test, and the overall throughput rate is not always easy to determine (especially as the number of connections increases).

The Ideal TCP Transfer Time would be ~8 seconds, but in this example, the Actual TCP Transfer Time was 12 seconds. The TCP Transfer Time Ratio would then be 12/8 = 1.5, which indicates that the transfer across all connections took 1.5 times longer than the ideal.

4.2. TCP Efficiency

The second metric represents the percentage of Bytes that were not retransmitted.

$$\text{TCP Efficiency \%} = \frac{\text{Transmitted Bytes} - \text{Retransmitted Bytes}}{\text{Transmitted Bytes}} \times 100$$

Transmitted Bytes are the total number of TCP Bytes to be transmitted, including the original and the retransmitted Bytes.

4.2.1. TCP Efficiency Percentage Calculation

As an example, if 100,000 Bytes were sent and 2,000 had to be retransmitted, the TCP Efficiency Percentage would be calculated as:

$$\text{TCP Efficiency \%} = \frac{102{,}000 - 2{,}000}{102{,}000} \times 100 = 98.03\%$$

Note that the Retransmitted Bytes may have occurred more than once; if so, then these multiple retransmissions are added to the Retransmitted Bytes and to the Transmitted Bytes counts.

4.3. Buffer Delay

The third metric is the Buffer Delay Percentage, which represents the increase in RTT during a TCP Throughput test versus the inherent or baseline RTT. The baseline RTT is the Round-Trip Time inherent to the network path under non-congested conditions as defined in Section 3.2.1. The average RTT is derived from the total of all measured RTTs during the actual test at every second divided by the test duration in seconds.

$$\text{Average RTT during transfer} = \frac{\text{Total RTTs during transfer}}{\text{Transfer duration in seconds}}$$

$$\text{Buffer Delay \%} = \frac{\text{Average RTT during transfer} - \text{Baseline RTT}}{\text{Baseline RTT}} \times 100$$

4.3.1. Buffer Delay Percentage Calculation

As an example, consider a network path with a baseline RTT of 25 ms. During the course of a TCP transfer, the average RTT across the entire transfer increases to 32 ms. Then, the Buffer Delay Percentage would be calculated as:

$$\text{Buffer Delay \%} = \frac{32 - 25}{25} \times 100 = 28\%$$

Note that the TCP Transfer Time Ratio, TCP Efficiency Percentage, and the Buffer Delay Percentage MUST all be measured during each throughput test. A poor TCP Transfer Time Ratio (i.e., Actual TCP Transfer Time greater than the Ideal TCP Transfer Time) may be diagnosed by correlating with sub-optimal TCP Efficiency Percentage and/or Buffer Delay Percentage metrics.

5. Conducting TCP Throughput Tests

Several TCP tools are currently used in the network world, and one of the most common is "iperf". With this tool, hosts are installed at each end of the network path; one acts as a client and the other as a server. The Send Socket Buffer and the TCP RWND sizes of both client and server can be manually set. The achieved throughput can then be measured, either uni-directionally or bi-directionally. For higher-BDP situations in lossy networks (Long Fat Networks (LFNs) or satellite links, etc.), TCP options such as Selective Acknowledgment SHOULD become part of the window size/throughput characterization.

Host hardware performance must be well understood before conducting the tests described in the following sections. A dedicated communications test instrument will generally be REQUIRED, especially for line rates of GigE and 10 GigE. A compliant TCP TTD SHOULD provide a warning message when the expected test throughput will exceed the subscribed customer SLA. If the throughput test is expected to exceed the subscribed customer SLA, then the test SHOULD be coordinated with the network provider.

The TCP Throughput test SHOULD be run over a long enough duration to properly exercise network buffers (i.e., greater than 30 seconds) and SHOULD also characterize performance at different times of the day.

5.1. Single versus Multiple TCP Connections

The decision whether to conduct single- or multiple-TCP-connection tests depends upon the size of the BDP in relation to the TCP RWND configured in the end-user environment. For example, if the BDP for

a Long Fat Network (LFN) turns out to be 2 MB, then it is probably more realistic to test this network path with multiple connections. Assuming typical host TCP RWND sizes of 64 KB (e.g., Windows XP), using 32 TCP connections would emulate a small-office scenario.

The following table is provided to illustrate the relationship between the TCP RWND and the number of TCP connections required to fill the available capacity of a given BDP. For this example, the network bandwidth is 500 Mbps and the RTT is 5 ms; then, the BDP equates to 312.5 KBytes.

TCP RWND	Number of TCP Connections to fill available bandwidth
16 KB	20
32 KB	10
64 KB	5
128 KB	3

Table 5.1. Number of TCP Connections versus TCP RWND

The TCP Transfer Time Ratio metric is useful when conducting multiple-connection tests. Each connection SHOULD be configured to transfer payloads of the same size (e.g., 100 MB); then, the TCP Transfer Time Ratio provides a simple metric to verify the actual versus expected results.

Note that the TCP transfer time is the time required for each connection to complete the transfer of the predetermined payload size. From the previous table, the 64-KB window is considered. Each of the 5 TCP connections would be configured to transfer 100 MB, and each one should obtain a maximum of 100 Mbps. So for this example, the 100-MB payload should be transferred across the connections in approximately 8 seconds (which would be the Ideal TCP Transfer Time under these conditions).

Additionally, the TCP Efficiency Percentage metric MUST be computed for each connection as defined in Section 4.2.

5.2. Results Interpretation

At the end, a TCP Throughput Test Device (TCP TTD) SHOULD generate a report with the calculated BDP and a set of Window size experiments. Window size refers to the minimum of the Send Socket Buffer and TCP RWND. The report SHOULD include TCP Throughput results for each TCP Window size tested. The goal is to provide achievable versus actual TCP Throughput results with respect to the TCP Window size when no fragmentation occurs. The report SHOULD also include the results for

the 3 metrics defined in Section 4. The goal is to provide a clear relationship between these 3 metrics and user experience. As an example, for the same results in regard to Transfer Time Ratio, a better TCP Efficiency could be obtained at the cost of higher Buffer Delays.

For cases where the test results are not equal to the ideal values, some possible causes are as follows:

- Network congestion causing packet loss, which may be inferred from a poor TCP Efficiency % (i.e., higher TCP Efficiency % = less packet loss).

- Network congestion causing an increase in RTT, which may be inferred from the Buffer Delay Percentage (i.e., 0% = no increase in RTT over baseline).

- Intermediate network devices that actively regenerate the TCP connection and can alter TCP RWND size, MTU, etc.

- Rate limiting by policing instead of shaping.

- Maximum TCP Buffer Space. All operating systems have a global mechanism to limit the quantity of system memory to be used by TCP connections. On some systems, each connection is subject to a memory limit that is applied to the total memory used for input data, output data, and controls. On other systems, there are separate limits for input and output buffer spaces per connection. Client/server IP hosts might be configured with Maximum TCP Buffer Space limits that are far too small for high-performance networks.

- Socket Buffer sizes. Most operating systems support separate per-connection send and receive buffer limits that can be adjusted as long as they stay within the maximum memory limits. These socket buffers MUST be large enough to hold a full BDP of TCP Bytes plus some overhead. There are several methods that can be used to adjust Socket Buffer sizes, but TCP Auto-Tuning automatically adjusts these as needed to optimally balance TCP performance and memory usage.

  It is important to note that Auto-Tuning is enabled by default in LINUX since kernel release 2.6.6 and in UNIX since FreeBSD 7.0. It is also enabled by default in Windows since Vista and in Mac since OS X version 10.5 (Leopard). Over-buffering can cause some applications to behave poorly, typically causing sluggish interactive response and introducing the risk of running the system out of memory. Large default socket buffers have to be considered carefully on multi-user systems.

- TCP Window Scale option [RFC1323]. This option enables TCP to support large BDP paths. It provides a scale factor that is required for TCP to support window sizes larger than 64 KB. Most systems automatically request WSCALE under some conditions, such as when the Receive Socket Buffer is larger than 64 KB or when the other end of the TCP connection requests it first. WSCALE can only be negotiated during the 3-way handshake. If either end fails to request WSCALE or requests an insufficient value, it cannot be renegotiated. Different systems use different algorithms to select WSCALE, but it is very important to have large enough buffer sizes. Note that under these constraints, a client application wishing to send data at high rates may need to set its own receive buffer to something larger than 64 KBytes before it opens the connection, to ensure that the server properly negotiates WSCALE. A system administrator might have to explicitly enable [RFC1323] extensions. Otherwise, the client/server IP host would not support TCP Window sizes (BDP) larger than 64 KB. Most of the time, performance gains will be obtained by enabling this option in LFNs.

- TCP Timestamps option [RFC1323]. This feature provides better measurements of the Round-Trip Time and protects TCP from data corruption that might occur if packets are delivered so late that the sequence numbers wrap before they are delivered. Wrapped sequence numbers do not pose a serious risk below 100 Mbps, but the risk increases at higher data rates. Most of the time, performance gains will be obtained by enabling this option in Gigabit-bandwidth networks.

- TCP Selective Acknowledgments (SACK) option [RFC2018]. This allows a TCP receiver to inform the sender about exactly which data segment is missing and needs to be retransmitted. Without SACK, TCP has to estimate which data segment is missing, which works just fine if all losses are isolated (i.e., only one loss in any given round trip). Without SACK, TCP takes a very long time to recover after multiple and consecutive losses. SACK is now supported by most operating systems, but it may have to be explicitly enabled by the system administrator. In networks with unknown load and error patterns, TCP SACK will improve throughput performance. On the other hand, security appliance vendors might have implemented TCP randomization without considering TCP SACK, and under such circumstances, SACK might need to be disabled in the client/server IP hosts until the vendor corrects the issue. Also, poorly implemented SACK algorithms might cause extreme CPU loads and might need to be disabled.

- Path MTU. The client/server IP host system SHOULD use the largest possible MTU for the path. This may require enabling Path MTU Discovery [RFC1191] and [RFC4821]. Since [RFC1191] is flawed, Path MTU Discovery is sometimes not enabled by default and may need to be explicitly enabled by the system administrator. [RFC4821] describes a new, more robust algorithm for MTU discovery and ICMP black hole recovery.

- TOE (TCP Offload Engine). Some recent Network Interface Cards (NICs) are equipped with drivers that can do part or all of the TCP/IP protocol processing. TOE implementations require additional work (i.e., hardware-specific socket manipulation) to set up and tear down connections. Because TOE NIC configuration parameters are vendor-specific and not necessarily RFC-compliant, they are poorly integrated with UNIX and LINUX. Occasionally, TOE might need to be disabled in a server because its NIC does not have enough memory resources to buffer thousands of connections.

Note that both ends of a TCP connection MUST be properly tuned.

6. Security Considerations

Measuring TCP network performance raises security concerns. Metrics produced within this framework may create security issues.

6.1. Denial-of-Service Attacks

TCP network performance metrics, as defined in this document, attempt to fill the NUT with a stateful connection. However, since the test MAY use stateless IP streams as specified in Section 3.2.2, it might appear to network operators to be a denial-of-service attack. Thus, as mentioned at the beginning of Section 3, TCP Throughput testing may require cooperation between the end-user customer and the network provider.

6.2. User Data Confidentiality

Metrics within this framework generate packets from a sample, rather than taking samples based on user data. Thus, our framework does not threaten user data confidentiality.

6.3. Interference with Metrics

The security considerations that apply to any active measurement of live networks are relevant here as well. See [RFC4656] and [RFC5357].

7. Acknowledgments

Thanks to Lars Eggert, Al Morton, Matt Mathis, Matt Zekauskas, Yaakov Stein, and Loki Jorgenson for many good comments and for pointing us to great sources of information pertaining to past works in the TCP capacity area.

8. Normative References

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.

[RFC1323] Jacobson, V., Braden, R., and D. Borman, "TCP Extensions for High Performance", RFC 1323, May 1992.

[RFC2018] Mathis, M., Mahdavi, J., Floyd, S., and A. Romanow, "TCP Selective Acknowledgment Options", RFC 2018, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, March 1999.

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, September 2006.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

[RFC4898] Mathis, M., Heffner, J., and R. Raghunarayan, "TCP Extended Statistics MIB", RFC 4898, May 2007.

[RFC5136] Chimento, P. and J. Ishac, "Defining Network Capacity", RFC 5136, February 2008.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, October 2008.

Authors' Addresses

Barry Constantine
JDSU, Test and Measurement Division
One Milesone Center Court
Germantown, MD 20876-7100
USA

Phone: +1 240 404 2227
EMail: barry.constantine@jdsu.com

Gilles Forget
Independent Consultant to Bell Canada
308, rue de Monaco, St-Eustache
Qc. J7P-4T5 CANADA

Phone: (514) 895-8212
EMail: gilles.forget@sympatico.ca

Ruediger Geib
Heinrich-Hertz-Strasse 3-7
Darmstadt, 64295 Germany

Phone: +49 6151 5812747
EMail: Ruediger.Geib@telekom.de

Reinhard Schrage
Osterende 7
Seelze, 30926
Germany
Schrage Consulting

Phone: +49 (0) 5137 909540
EMail: reinhard@schrageconsult.com