Network Working Group
Internet Draft
Intended status: Informational
Expires: February 2014
September 4, 2013

B. Constantine
JDSU
T. Copley
Level-3
R. Krishnan
Brocade Communications

Traffic Management Benchmarking
draft-constantine-bmwg-traffic-management-02.txt

Status of this Memo

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on July 5, 2013.

Abstract

This framework describes a practical methodology for benchmarking the traffic management capabilities of networking devices (i.e. policing, shaping, etc.). The goal is to provide a repeatable test method that objectively compare performance of the device's traffic management capabilities and to specify the means to benchmark traffic management with representative application traffic.

Table of Contents

1. Introduction

Traffic management (i.e. policing, shaping, etc.) is an increasingly important component when engineering network Quality of Service (QoS) today. There is currently no framework to benchmark these features although some standards address specific areas. This draft provides a framework to conduct repeatable traffic management benchmarks for devices and systems in a lab environment.

Specifically, this framework defines the methods to characterize the capacity of traffic management features in network devices, such as classification, policing, shaping, and active queue management.

This benchmarking framework can also be used as a test procedure to assist in the tuning of traffic management parameters before field deployment. In addition to Layer 2/3 benchmarking, Layer 4 test patterns are proposed by this draft in order to benchmark as close as possible to real end-user traffic.

1.1. Traffic Management Overview

In general, a device with traffic management capabilities performs the following functions:

- Traffic classification: identifies traffic according to various configuration rules (i.e. VLAN, DSCP, etc.) and marks this traffic internally to the network device. Multiple external priorities (DSCP, 802.1p, etc.) can map to the same priority in the device.
- Traffic policing: limits the rate of traffic that enters a network device according to the traffic classification. If the traffic exceeds the contracted limits, the traffic is either dropped or remarked and sent onto to the next network device
- Traffic Scheduling: provides traffic classification within the network device by directing packets to various types of queues and applies a dispatching algorithm to assign the forwarding sequence of packets
- Traffic shaping: a traffic control measure of actively buffering and metering the output rate in an attempt to adapt bursty traffic to the configured limits
- Active Queue Management (AQM): monitors the status of internal queues and actively drops (or re-marks) packets, which causes hosts using congestion-aware protocols to back-off and in turn can alleviate queue congestion.

The following diagram is a generic model of the traffic management capabilities within a network device. It is not intended to represent all variations of manufacturer traffic management capabilities, but provide context to this test framework.

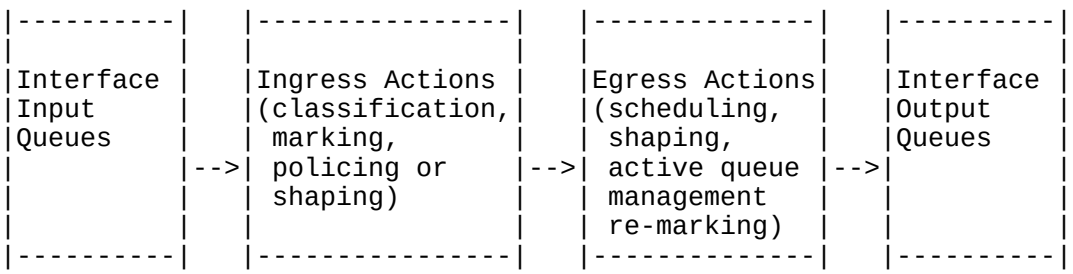


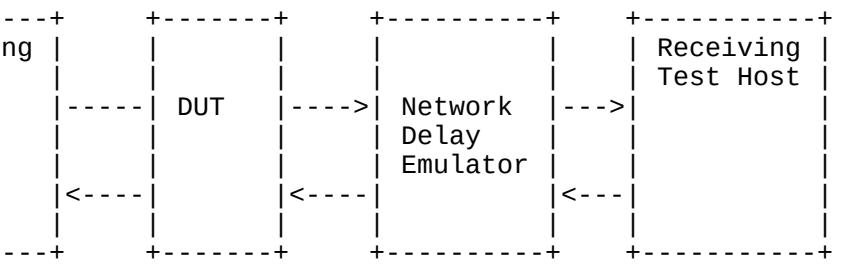
Figure 1: Generic Traffic Management capabilities of a Network Device

Ingress actions such classification are defined in RFC 4689 and include IP addresses, port numbers, DSCP, etc. In terms of marking, RFC 2697 and RFC 2698 define a single rate and dual rate, three color marker, respectively.

The MEF specifies policing and shaping in terms of Ingress and Egress Subscriber/Provider Conditioning Functions in MEF12.1; Ingress and Bandwidth Profile attributes in MEF 10.2 and MEF 26.

1.2 DUT Lab Configuration and Testing Overview

The following is the description of the lab set-up for the traffic management tests:



As shown the test diagram, the framework supports uni-directional and bi-directional traffic management tests.

This testing framework describes the individual tests and metrics for each of the following traffic management functions:
- Policing Tests
- Shaping Tests
- Queue / Scheduling Tests
- Congestion Management Tests

The tests are divided into individual tests and rated capacity tests. The individual tests are intended to verify the traffic management function according to the specifications. As an example, suppose a traffic shaper is to be tested at a CIR of 20 Mbps. The intent of the individual test is to test one instance of the shaper and it’s ability to properly shape according to the metrics defined in section 4.

The capacity tests verify traffic management functions under full load. This involves concurrent testing of multiple interfaces with the specific traffic management function enabled, and doing so to the capacity limit of each interface.

For an example: a device is specified to be capable of shaping on all of it’s egress ports. The individual test would first be conducted to benchmark the advertised shaping function against the metrics defined in section 4. Then the capacity test would be executed to test the shaping function concurrently on all interfaces and with maximum traffic load.

Also note that the Network Delay Emulator (NDE) should be passive in nature such as a fiber spool. This is recommended to eliminate the potential effects that an active delay element (i.e. test impairment generator) may have on the test flows. In the case that a fiber spool is not practical due to the desired latency, an active NDE must be independently verified to be capable of adding the configured delay without loss. In other words, the DUT would be removed and the NDE performance benchmarked independently.

Note the NDE should be used in "full pipe" delay mode. Most NDEs allow for per flow delay actions, emulating QoS prioritization. For this framework, the NDE’s sole purpose is simply to add delay to all packets (emulate network latency). So to benchmark the performance of the NDE, maximum offered load should be tested against the following frame sizes: 128, 256, 512, 768, 1024, 1500,and 9600 bytes. The delay acuracy at each of these packet sizes can then be used to callibrate the range of expected BDPs for the TCP stateful tests.

2. Conventions used in this document

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].

The following acronyms are used:

BB: Bottleneck Bandwidth

BDP: Bandwidth Delay Product

BSA: Burst Size Achieved

CBS: Committed Burst Size

CIR: Committed Information Rate

DUT: Device Under Test

EBS: Excess Burst Size

EIR: Excess Information Rate

NDE: Network Delay Emulator

SP: Strict Priority Queuing

QL: Queue Length

QoS: Quality of Service

RED: Random Early Discard

RTT: Round Trip Time

SBS: Shaper Burst Size

SR: Shaper Rate

SSB: Send Socket Buffer

Tc: CBS Time Interval

Te: EBS Time Interval

Ti Transmission Interval

TTP: TCP Test Pattern

TTPET: TCP Test Pattern Execution Time

WRED: Weighted Random Early Discard

3. Scope and Goals

The scope of this work is to develop a framework for benchmarking and testing the traffic management capabilities of network devices in the lab environment. These network devices may include but are not limited to:

- Switches (including Layer 2/3 devices)
- Routers
- Firewalls
- General Layer 4-7 appliances (Proxies, WAN Accelerators, etc.)

Essentially, any network device that performs traffic management as defined in section 1.1 can be benchmarked or tested with this framework.

The primary goal is to assess the maximum forwarding performance that a network device can sustain without dropping or impairing packets, or compromising the accuracy of multiple instances of traffic management functions. This is the benchmark for comparison between devices.

Within this framework, the metrics are defined for each traffic management test but do not include pass / fail criterion, which is not within the charter of BMWG. This framework provides the test methods and metrics to conduct repeatable testing, which will provide the means to compare measured performance between DUTs.

As mentioned in section 1.2, this framework describes the individual tests and metrics for several management functions. It is also within scope that this framework will benchmark each function in terms of overall rated capacity. This involves concurrent testing of multiple interfaces with the specific traffic management function enabled, up to the capacity limit of each interface.

It is not within scope of this framework to specify the procedure for testing multiple traffic management functions concurrently. The multitudes of possible combinations is almost unbounded and the ability to identify functional "break points" would be most times impossible.

However, section 6.5 provides suggestions for some usual profiles of concurrent functions that would be useful to benchmark. The key requirement for any concurrent test function is that tests must produce reliable and repeatable results.

Also, it is not within scope to perform conformance testing. Tests defined in this framework benchmark the traffic management functions according to the metrics defined in section 4 and do not address any conformance to standards related to traffic management. Traffic management specifications largely do not exist and this is a prime driver for this framework; to provide an objective means to compare vendor traffic management functions.

Another goal is to devise methods that utilize flows with congestion-aware transport (TCP) as part of the traffic load and still produce repeatable results in the isolated test environment. This framework will derive stateful test patterns (TCP or application layer) that can also be used to further benchmark the performance of applicable traffic management techniques such as shaping and congestion management techniques such as RED/WRED. In cases where the network device is stateful in nature (i.e. firewall, etc.), stateful test pattern traffic is important to test along with stateless, UDP traffic in specific test scenarios (i.e. applications using TCP transport and UDP VoIP, etc.)

And finally, this framework will provide references to open source tools that can be used to provide stateless and/or stateful traffic generation emulation.

4. Traffic Benchmarking Metrics

The metrics to be measured during the benchmarks are divided into two (2) sections: packet layer metrics used for the stateless traffic testing and segment layer metrics used for the stateful traffic testing.

4.1. Metrics for Stateless Traffic Tests

For the stateless traffic tests, the metrics are defined at the layer 3 packet level versus layer 2 packet level for consistency.

Stateless traffic measurements require that sequence number and time-stamp be inserted into the payload for lost packet analysis. Delay analysis may be achieved by insertion of timestamps directly into the packets or timestamps stored elsewhere (packet captures). This framework does not specify the packet format to carry sequence number or timing information. However, RFC 4689 provides recommendations for sequence tracking along with defintions of in-sequence and out-of-order packets.

The following are the metrics to be used during the stateless traffic benchmarking components of the tests:

- Burst Size Achieved (BSA): for the traffic policing and network queue tests, the tester will be configured to send bursts to test either the Committed Burst Size (CBS) or Excess Burst Size (EBS) of a policer or the queue / buffer size configured in the DUT. The Burst Size Achieved metric is a measure of the actual burst size received at the egress port of the DUT with no lost packets. As an example,the configured CBS of a DUT is 64KB and after the burst test, only a 63 KB can be achieved without packet loss. Then 63KB is the BSA. Also, the average Packet Delay Variation (PDV see below) is experienced by the packets sent at the BSA burst size should be recorded.

- Lost Packets (LP): For all traffic management tests, the tester will transmit the test packets into the DUT ingress port and the number of packets received at the egress port will be measured. The difference between packets transmitted into the ingress port and received at the egress port is the number of lost packets as measured at the egress port. These packets must have unique identifiers such that only the test packets are measured. RFC 4737 and RFC 2680 describe the need to to establish the threshold to designate when a packet as lost, and the threshold MUST be reported with the results.

- Out of Sequence (OOS): in additions to LP metric, the test packets must be monitored for sequence and the out-of-sequence (OOS) packets. RFC 4689 defines the general function of sequence tracking, as well as definitions for in-sequence and out-of-order packets. Out-of-order packets will be counted per RFC 4737 and RFC 2680.

- Packet Delay (PD): the Packet Delay metric is the difference between the timestamp of the received egress port packets and the packets transmitted into the ingress port and specified in RFC 2285.

- Packet Delay Variation (PDV): the Packet Delay Variation metric is the variation between the timestamp of the received egress port packets and specified in RFC 5481.

4.2. Metrics for Stateful Traffic Tests

The stateful metrics will be based on RFC 6349 TCP metrics and will include:

- TCP Test Pattern Execution Time (TTPET): RFC 6349 defined the TCP Transfer Time for bulk transfers, which is simply the measured time to transfer bytes across single or concurrent TCP connections. The TCP test patterns used in traffic management tests will include bulk transfer and interactive applications. The interactive patterns include application models such as HTTP business applications, database applications, etc. The TTPET will be the measure of the time for a single execution of a TCP Test Pattern (TTP). Average, minimum, and maximum times will be measured or calculated.

An example would be an interactive HTTP TTP session which should take 5 seconds on a GigE network with 0.5 msec latency. During ten (10) executions of this TTP, the TTPER results might be: average of 6.5 seconds, minimum of 5.0 seconds, and maximum of 7.9 seconds.

- TCP Efficiency: after the execution of the TCP Test Pattern, TCP Efficiency 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. These retransmitted bytes should be recorded from the sender’s TCP/IP stack perspective, to avoid any misinterpretation that a reordered packet is a retransmitted packet (as may be the case with packet decode interpretation).

- Buffer Delay: represents the increase in RTT during a TCP test versus the baseline DUT RTT (non congested, inherent latency). RTT and the technique to measure RTT (average versus baseline) are defined in RFC 6349. Referencing RFC 6349, the average RTT is derived from the total of all measured RTTs during the actual test sampled at every second divided by the test duration in seconds.

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

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

Note that even though this was not explicitly stated in RFC 6349, retransmitted packets should not be used in RTT measurements.

Also, the test results should record the average RTT in msec across the entire test duration and number of samples.

5. Tester Capabilities

The testing capabilities of the traffic management test environment are divided into two (2) sections: stateless traffic testing and stateful traffic testing

5.1. Stateless Test Traffic Generation

The test set must be capable of generating traffic at up to the link speed of the DUT. The test set must be calibrated to verify that it will not drop any packets. The test set's inherent PD and PDV must also be calibrated and subtracted from the PD and PDV metrics. The test set must support the encapsulation to be tested such as VLAN, Q-in-Q, MPLS, etc. Also, the test ser must allow control of the classification techniques defined in RFC 4689 (i.e. IP address, DSCP, TOS, etc classification).

The open source tool "iperf" can be used to generate stateless UDP traffic and is discussed in Appendix A. Since iperf is a software based tool, there will be performance limitations at higher link speeds (e.g. GigE, 10 GigE, etc.). Careful calibration of any test environment using iperf is important. At higher link speeds, it is recommended to select commercial hardware based packet test equipment.

5.2. Stateful Test Pattern Generation

The TCP test host will have many of the same attributes as the TCP test host defined in RFC 6349. The TCP test device 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.

For any test using stateful TCP test traffic, the Network Delay Emulator (NDE) function from the lab set-up must be used in order to provide a meaningful BDP. As referenced in section 2, the target traffic rate and configured RTT must be verified independently using just the NDE for all stateful tests (to ensure the NDE can delay without loss).

The TCP test host must be capable to generate and receive stateful TCP test traffic at the full link speed of the DUT. As a general rule of thumb, testing TCP Throughput at rates greater than 100 Mbps may require high performance server hardware or dedicated hardware based test tools.

(TC comment: You mention that a device to do rates greater than 100Mbit may require a high performance server. We also need to discuss how window Sizes or flows impact that.)

The TCP test host must allow adjusting both Send and Receive Socket Buffer sizes. The Socket Buffers must be large enough to fill the BDP for bulk transfer TCP test application traffic.

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.

The TCP implementation used by the test host must be specified in the test results (i.e. OS version, i.e. LINUX OS kernel using TCP New Reno, TCP options supported, etc). In some cases, scaled down TCP implementations can also be used as is sometimes the case for high performance, hardware-based commercial implementations.

While RFC 6349 defined the means to conduct throughput tests of TCP bulk transfers, the traffic management framework will extend TCP test execution into interactive TCP application traffic. Examples include email, HTTP, business applications, etc. This interactive traffic is bi-directional and can be chatty.

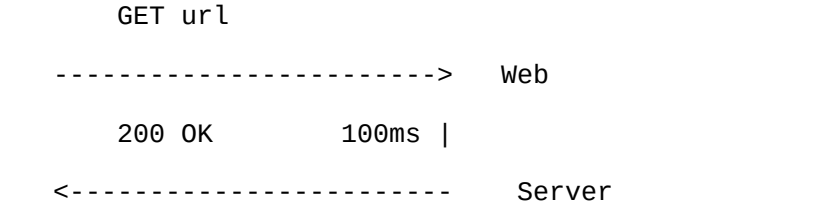
The test device must not only support bulk TCP transfer application traffic but also chatty traffic. A valid stress test SHOULD include both traffic types. This is due to the non-uniform, bursty nature of chatty applications versus the relatively uniform nature of bulk transfers (the bulk transfer smoothly stabilizes to equilibrium state under lossless conditions).

While iperf is an excellent choice for TCP bulk transfer testing, the open source tool "Flowgrind" (referenced in Appendix A). Flowgrind is client server based and emulates interactive applications at the TCP layer. As with any software based tool, the performance must be qualified to the link speed to be tested. Commercial test equipment should be considered for reliable results at higher links speeds (e.g Gig, 10 GigE).

5.2.1. TCP Test Pattern Definitions

As mentioned in the goals of this framework, techniques to define Layer 4 traffic test patterns will be defined to benchmark the traffic management technique(s) under realistic conditions. Some network devices such as firewalls, will not process stateless test traffic which is another reason why stateful TCP test traffic must be used.

An application could be fully emulated up to Layer 7, however this framework proposes that stateful TCP test patterns be used in order to provide granular and repeatable control for the benchmarks. The following diagram illustrates a simple Web Browsing application (HTTP).



In this example, the Client Web Browser (Client) requests a URL and then the Web Server delivers the web page content to the Client (after a Server delay of 100 msec). This asynchronous, "request / response" behavior is intrinsic to most TCP based applications such as Email (SMTP), File Transfers (FTP and SMB), Database (SQL), Web Applications (SOAP), etc. The impact to the network elements is due to the multitudes of Clients and the variety of bursty traffic, which stresses network resources such as buffers, shapers, and other QoS management techniques. The actual emulation of the specific application protocols is not required and TCP test patterns can be defined to mimic the application behavior.

This framework does not specify a fixed set of TCP test patterns, but does provide examples in Appendix B. These examples reflect those specificed in "draft-ietf-bmwg-ca-bench-meth-04" which suggests traffic mixes for a variety of representative application profiles.

There are two (2) techniques recommended by this framework to develop standard TCP test patterns for traffic management benchmarking.

The first technique involves modeling, which have been described in "3GPP2 C.R1002-0 v1.0" and describe the behavior of HTTP, FTP, and WAP applications at the TCP layer. The models have been defined with various mathematical distributions for the Request/Response bytes and inter-request gap times. The Flowgrind tool (Appendix A) supports many of the distributions and is a good choice as long as the processing limits of the server platform are taken into consideration.

The second technique is to conduct packet captures of the applications to test and then to statefully play the application back at the TCP layer. The TCP playback includes the request byte size, response byte size, and inter-message gaps at both the client and the server. The advantage of this method is that very realistic test patterns can be defined based on real world application traffic.

Appendix B provides an overview of the modeling technique with Flowgrind, capture technique with TCP playback, and some representative application traffic that can be used with either techniques.

(TC comment: In addition to application test patterns, I'd also like to see some of the standard ways mentioned like 2544 all 1's all F's all 0's and the Alternating)

6. Traffic Benchmarking Methodology

The traffic benchmarking methodology uses the test set-up from section 2 and metrics defined in section 4. Each test should be run for a minimum test time of 5 minutes.

Each test should compare the network device's internal statistics (available via command line management interface, SNMP, etc.) to the measured metrics defined in section 4. This evaluates the accuracy of the internal traffic management counters under verification test conditions and capacity test conditions that are defined in each subsection.

6.1. Policing Tests

The intent of the policing tests is to verify the policer performance (i.e. CIR-CBS and EIR-EBS parameters). The tests will verify that the network device can handle the CIR with CBS and the EIR with EBS and will use back-back packet testing concepts from RFC 2544 (but adapted to burst size algorithms and terminology). Also MEF-14,19,37 provide some basis for specific components of this test.

The tests are divided into two (2) sections; individual policer function verification tests and then full capacity policing tests. It is important to verify the basic functionality of the individual policer then proceed into the fully rated capacity of the device. This capacity may include the number of policing policies per device and the number of policers simultaneously active across all ports.

6.1.1 Policer Individual Tests

Policing tests should use stateless traffic. Stateful TCP test traffic will generally be adversely affected by a policer in the absence of traffic shaping. So while TCP traffic could be used, it is more accurate to benchmark a policer with stateless traffic.

The policer test shall test a policer as defined by RFC 4115 or MEF 10.2, depending upon the equipment's specification. As an example for RFC 4115, consider a CBS and EBS of 64KB and CIR and EIR of 100 Mbps on a 1GigE physical link (in color-blind mode). A stateless traffic burst of 64KB would be sent into the policer at the GigE rate. This equates to approximately a 0.512 msec burst time and the burst-to-burst spacing would be 5.12 msec.

The metrics defined in section 4.1 shall be measured at the egress port and recorded; the primary result is to verify the BSA and that no packets are dropped.

In addition to verifying that the policer allows the specified CBS and EBS bursts to pass, the policer test must verify that the policer will police at the specified CBS/EBS values.

For this portion of the test, the CBS/EBS value should be incremented by 1000 bytes higher than the configured CBS and that the egress port measurements must show that the majority of packets are dropped.

Additional tests beyond the simple color-blind example might include: color-aware mode, configurations where EIR is greater than CIR, etc.

6.1.2 Policer Capacity Tests

The intent of the capacity tests is to verify the policer performance in a scaled environment with multiple ingress customer policers on multiple physical ports. This test will benchmark the maximum number of active policers as specified by the device manufacturer.

As an example, a Layer 2 switching device may specify that each of the 32 physical ports can be policed using a pool of policing service policies. The device may carry a single customer’s traffic on each physical port and a single policer is instantiated per physical port. Another possibility is that a single physical port may carry multiple customers, in which case many customer flows would be policed concurrently on an individual physical port.

The specified policing function capacity is generally expressed in terms of the number of policers active on each individual physical port as well as the number of unique policer rates that are utilized. For all of the capacity tests, the benchmarking methodology described in Section 6.1.1 for a single policer should be applied to each of the physical port policers.

6.1.2.1 Maximum Policers on Single Physical Port

The first policer capacity test will benchmark a single physical port, maximum policers on that physical port.

Assume multiple categories of ingress policers at rates r1, r2,...rn. There are multiple customers on a single physical port. Each customer could be represented by a single tagged vlan, double tagged vlan, VPLS instance etc. Each customer is mapped to a different policer. Each of the policers can be of rates r1, r2,..., rn. Policer granularity guideline (do we need ??)

An example configuration would be
- Y1 customers, policer rate r1
- Y2 customers, policer rate r2
- Y3 customers, policer rate r3
...
- Yn customers, policer rate rn

Some bandwidth on the physical port is dedicated for other traffic (non customer traffic); this includes network control protocol traffic. There is a separate policer for the other traffic. Typical deployments have 3 categories of policers; there may be some deployments with more or less than 3 categories of ingress policers.

6.1.2.2 Single Policer on All Physical Ports

The second policer capacity test involves a single Policer function per physical port with all physical ports active. In this test, there is a single policer per physical port. The policer can have one of the rates r1, r2,.., rn. All the physical ports in the networking device are active.

6.1.2.3 Maximum Policers on All Physical Ports

Finally the third policer capacity test involves a combination of the first and second capacity test, namely maximum policers active per physical port and all physical ports are active .

6.2. Queue and Scheduler Tests

Queues and traffic Scheduling are closely related in that a queue's priority dictates the manner in which the traffic scheduler's transmits packets out of the egress port.

Since device queues / buffers are generally an egress function, this test framework will discuss testing at the egress (although the technique can be applied to ingress side queues).

Similar to the policing tests, the tests are divided into two sections; individual queue/scheduler function verification tests and then full capacity tests.

6.2.1 Queue/Scheduler Individual Tests

The various types of scheduling techniques include FIFO, Strict Priority (SP), Weighted Fair Queueing (WFQ) along with other variations. This test framework recommends to test at a minimum these three techniques although it is the discretion of the tester to benchmark other device scheduling algorithms.

6.2.1.1 Testing Queue/Scheduler with Stateless Traffic

A network device queue is memory based unlike a policing function, which is token or credit based. However, the same concepts from section 6.1 can be applied to testing network device queue.

The device's network queue should be configured to the desired size in KB (queue length, QL) and then stateless traffic should be transmitted to test this QL.

The transmission interval (Ti) can be defined for the traffic bursts and is based off of the QL and Bottleneck Bandwidth (BB) of the egress interface. The equation is similar to the Tc / Te time interval discussed in the policer section 6.1 and is as follows:

Ti = QL * 8 / BB

Important to note that the assumption is that the aggregate ingress throughput is higher than the BB or the queue test is not relevant since there will not be any over subscription.

The stateless traffic shall be transmitted at the link speed within the Ti time interval. The metrics defined in section 4.1 shall be measured at the egress port and recorded; the primary result is to verify the BSA and that no packets are dropped.

The scheduling function must also be characterize during the test to benchmark the device's ability to schedule the queues according to the priority. An example would be 2 levels of priority including SP and FIFI queueing. Under flow load greater the the egress port speed, the higher priority packets should be transmitted without drops (and also maintain low latency), while the lower priority (or best effort) queue may be dropped.

6.2.1.2 Testing Queue/Scheduler with Stateful Traffic

To provide a more realistic benchmark and to test queues in layer 4 devices such as firewalls, stateful traffic testing is recommended for the queue tests. Stateful traffic tests will also utilize the Network Delay Emulator (NDE) from the network set-up configuration in section 2.

The BDP of the TCP test traffic must be calibrated to the QL of the device queue. Referencing RFC6349, the BDP is equal to:

BB * RTT / 8 (in bytes)

The NDE must be configured to an RTT value which is great enough to allow the BDP to be greater than QL. An example test scenario is defined below:

- Ingress link = GigE

- Egress link = 100 Mbps (BB)

- QL = 32KB

RTT(min) = QL * 8 / BB and would equal 2.56 msec and the BDP = 32KB

In this example, one (1) TCP connection with window size / SSB of 32KB would be required to test the QL of 32KB. This Bulk Transfer Test can be accomplished using iperf as described in Appendix A.

The test metrics will be recorded per the stateful metrics defined in 4.2, primarily the TCP Test Pattern Execution Time (TTPET), TCP Efficiency, and Buffer Delay.

In addition to a Bulk Transfer Test, it is recommended to run the Bursty Test Pattern from appendix B at a minimum. Other tests from include: Small Web Site, Email, Citrix, etc.

The traffic is bi-directional - the same queue size is assumed for both directions.

6.2.2 Queue / Scheduler Capacity Tests

The intent of these capacity tests is to verify queue/scheduler performance in a scaled environment with multiple queues/schedulers active on multiple egress physical ports. This test will benchmark the maximum number of queues and schedulers as specified by the device manufacturer. Each priority in the system will map to a separate queue.

6.2.2.1 Multiple Queues / Single Port Active

For the first scheduler / queue capacity test, multiple queues per port will be tested on a single physical port. In this case, all the queues (typically 8) are active on a single physical port. Traffic from multiple ingress physical ports are directed to the same egress physical port which will cause oversubscription on the egress physical port.

There are many types of priority schemes and combinations of priorities that are managed by the scheduler. The following sections specify the priority schemes that should be tested.

6.2.2.1.1 Strict Priority on Egress Port

For this test, Strict Priority (SP) scheduling on the egress physical port should be tested and the benchmarking methodology specified in section 6.2.1 should be applied here. For a given priority, each ingress physical port should get a fair share of the egress physical port bandwidth.

6.2.2.1.2 Strict Priority + Weighted Fair Queue (WFQ) on Egress Port

For this test, Strict Priority (SP) and Weighted Fair Queue (WFQ) should be enabled simultaneously in the scheduler but on a single egress port. The benchmarking methodology specified in Section 6.2.1 should be applied here. Additionally, the egress port bandwidth sharing among weighted queues should be proportional to the assigned weights. For a given priority, each ingress physical port should get a fair share of the egress physical port bandwidth.

6.2.2.2 Single Queue per Port / All Ports Active

Traffic from multiple ingress physical ports are directed to the same egress physical port, which will cause oversubscription on the egress physical port. Also, the same amount of traffic is directed to each egress physical port.

The benchmarking methodology specified in Section 6.2.1 should be applied here. Each ingress physical port should get a fair share of the egress physical port bandwidth. Additionally, each egress physical port should receive the same amount of traffic.

6.2.2.3 Multiple Queues per Port, All Ports Active

Traffic from multiple ingress physical ports are directed to all queues of each egress physical port, which will cause oversubscription on the egress physical ports. Also, the same amount of traffic is directed to each egress physical port.

The benchmarking methodology specified in Section 6.2.1 should be applied here. For a given priority, each ingress physical port should get a fair share of the egress physical port bandwidth. Additionally, each egress physical port should receive the same amount of traffic.

6.3. Shaper tests

The intent of the shaper tests is to verify the shaper performance parameters of shape rate (SR) and shape burst size (SBS). The tests will verify that the device can handle the CIR rate with CBS and smooth the traffic bursts to the shaper rate.

Since device queues / buffers are generally an egress function, this framework will discuss testing at the egress (although the technique can be applied to ingress and internal queues).

Similar to the policing tests, the tests are divided into two sections; individual shaper function verification tests and then full capacity shaper tests.

6.3.1 Shaper Individual Tests

A network device’s traffic shaper will generally either shape to an average rate or provide settings similar to a policer (e.g. CIR and CBS. In the context of a shaper, the CBS indicates the size of the burst that the shaper can accept within the shaping time interval.

The shaping time interval depends upon whether the average method or CIR/CBS method is supported by the network device. If only the average method is supported, then the shaping time interval (period at which bursts will be shaped) must be determined through manufacturer product specifications.

For shapers that utilize the CIR/CBS method, the shaper time interval is the same as Tc for the policer which is indicated in section 6.1.

(TC comment: We need to be able to measure PD over a shaper. That should be the ms of queue depth.)

6.3.1.1 Testing Shaper with Stateless Traffic

A traffic shaper is memory based like a queue, but with the added intelligence of an active shaping element. The same concepts from section 6.2 (Queue testing) can be applied to testing network device shaper.

The device's traffic shaping function should be configured to the desired SR and SBS (for devices supporting this parameter) and then stateless traffic should be transmitted to test the SBS.

The same example from section 6.1 is used with SBS of 64KB and CIR of 100 Mbps; both ingress and egress ports are GigE. The Tc equates to 5.12 msec and the 64KB burst should be transmitted into the ingress port at full GigE rate, then wait for 5.12 msec for the next burst, etc.

While the ingress traffic will burst up to GigE link speed for the duration of the SBS burst, the egress traffic should be smoothed or averaged to the CIR rate on the egress interface.

In addition to the egress metrics to be measured per section 4.1, the stateless shaper test shall record:

- Average shaper rate on the egress port
- Variation (min, max) around the shaper rate

6.3.1.2 Testing Shaper with Stateful Traffic

To provide a more realistic benchmark and to test queues in layer 4 devices such as firewalls, stateful traffic testing is also recommended for the shaper tests. Stateful traffic tests will also utilize the Network Delay Emulator (NDE) from the network set-up configuration in section 2.

The BDP of the TCP test traffic must be calculated as described in section 6.2.2. To properly stress network buffers and the traffic shaping function, the cumulative TCP window should exceed the BDP which will stress the shaper. BDP factors of 1.1 to 1.5 are recommended, but the values are the discretion of the tester and should be documented.

The cumulative TCP Window Sizes* (RWND at the receiving end & CWND at the transmitting end) equates to:

TCP window size* for each connection x number of connections

* as described in section 3 of RFC6349, the SSB MUST be large enough to fill the BDP

Example, if the BDP is equal to 256 Kbytes and a connection size of 64Kbytes is used for each connection, then it would require four (4) connections to fill the BDP and 5-6 connections (over subscribe the BDP) to stress test the traffic shaping function.

Two types of tests are recommended: Bulk Transfer test and Bursty Test Pattern as documented in Appendix B at a minimum. Other tests types may include: Small Web Site, Email, Citrix, etc.

The test results will be recorded per the stateful metrics defined in section 4.2, primarily the TCP Test Pattern Execution Time (TTPET), TCP Efficiency, and Buffer Delay.

The traffic is bi-directional involving multiple egress ports.

In addition to the egress metrics to be measured per section 4.2, the stateful shaper test shall record:

- Average shaper rate on each egress interface
- Variation (min, max) around the shaper rate

6.3.2 Shaper Capacity Tests

The intent of these scalability tests is to verify shaper performance in a scaled environment with shapers active on multiple queues on multiple egress physical ports. This test will benchmark the maximum number of shapers as specified by the device manufacturer.

For all of the capacity tests, the benchmarking methodology described in Section 6.3.1 for a single shaper should be applied to each of the physical port and/or queue shapers.

6.3.2.1 Single Queue Shaped, All Physical Ports Active

The first shaper capacity test involves per port shaping, all physical ports active. Traffic from multiple ingress physical ports are directed to the same egress physical port and this will cause oversubscription on the egress physical port. Also, the same amount of traffic is directed to each egress physical port.

The benchmarking methodology described in Section 6.3.1 should be applied to each of the physical ports. Each ingress physical port should get a fair share of the egress physical port bandwidth.

6.3.2.2 All Queues Shaped, Single Port Active

The second shaper capacity test is conducted with all queues actively shaping on a single physical port. The benchmarking methodology described in per port shaping test (previous section) serves as the foundation for this. Additionally, each of the SP queues on the egress physical port is configured with a shaper. For the highest priority queue, the maximum amount of bandwidth available is limited by the bandwidth of the shaper. For the lower priority queues, the maximum amount of bandwidth available is limited by the bandwidth of the shaper and traffic in higher priority queues.

6.3.2.3 All Queues Shaped, All Ports Active

And for the third shaper capacity test (which is a combination of the tests in the previous two sections),all queues will be actively shaping and all physical ports active.

6.4. Congestion Management tests

The intent of the congestion management tests is to benchmark the performance of various active queue management (AQM) discard techniques such as RED, WRED, etc. AQM techniques vary, but the main goal is to discard traffic before the queue overflows as is the case for a FIFO queue. This discard in effect sends implicit congestion notification warning to protocols such as TCP, which causes TCP to back-off and ideally improves aggregate throughput by preventing global TCP session loss(tail drop).

Similar to the policing tests, the tests are divided into two (2) sections; individual AQM function verification tests and then full capacity AQM tests.

6.4.1 Congestion Management Verification Tests

The key parameter for AQM techniques is the discard threshold of the queue. (RK comment: The discard is also probabilistic http://en.wikipedia.org/wiki/Random_early_detection). In some network devices, this discard threshold is discretely configurable (e.g. percent of queue depth) and in others the discard threshold is intrinsic to the AQM technique itself.

As such AQM benchmark testing may involve a certain level of characterization experiments in which the burst size transmitted may increase as a portion of the queue depth.

6.4.1.1. Testing Congestion Management with Stateless Traffic

If the queue discard threshold is discreetly configurable, then the stateless burst techniques described in sections 6.2.1 (queuing tests) can be applied directly to the AQM tests. In other words, the queue will be over-subscribed and burst transmitted into the device within the Ti interval as defined in 6.2.1

For AQM techniques where the discard threshold is not discreetly configurable, then a stair case ramp is recommended to characterize and compare the AQM technique between devices. For example if the QL = 32KB, then it would be reasonable to test with burst sizes in increments of 25% to include 8KB, 16KB, 32KB and record the results per section 4.2. (RK comment: We should send a burst and examine if there are discontinuous drops - in the case of tail drop, the drops will be continuous)

6.4.1.2 Testing Congestion Management with Stateful Traffic

Similar to the Queue tests (section 6.2) and Shaper tests (section 6.3), stateful traffic tests will utilize the Network Delay Emulator (NDE) to add RTT. The RTT should be configured such that BDP would equal at least 64KB.

The key metric to be measured for the stateful tests is the TCP Test Pattern Execution Time (TTPET). AQM is intended to improve TCP performance by preventing tail-drop and it is the TTPET that provides the appropriate metric to compare the AQM techniques between vendors.

An example is as follows: transmit n TCP flows using the AQM Test Pattern (reference Appendix B) and measure the TTPET with and without AQM enabled. The number of flows should be configured to exceed the BDP with recommended oversubscription within the 1.1 - 1.5 range.

The test results will be recorded per the stateful metrics defined in 4.2, primarily the TCP Test Pattern Execution Time (TTPET), TCP Efficiency, and Buffer Delay.

6.4.2 Congestion Management Capacity Tests (TBD)

Only for the data queues (bursty traffic), different AQM techniques = RED, WRED, etc.

6.4.2.1 All Data Queues with AQM, Single Physical Port

TBD

6.4.2.1 All Data Queues with AQM, Multiple Physical Ports

TBD

6.5 Concurrent Capacity Load Tests

As mentioned in the scope of this document, it is impossible to specify the various permuations of concurrent traffic management functions that should be tested in a device for capacity testing. However, some profiles are listed below which may be useful to test under capacity as well:

- Policers on ingress and queuing on egress
- Policers on ingress and shapers on egress (not intended for a flow to be policed then shaped, these would be two different flows tested at the same time)
- etc

Appendix A: Open Source Tools for Traffic Management Testing

This framework specifies that stateless and stateful behaviors should both be tested. Two (2) open source tools that can be used are iperf and Flowgrind to accomplish many of the tests proposed in this framework.

Iperf can generate UDP or TCP based traffic; a client and server must both run the iperf software in the same traffic mode. The server is set up to listen and then the test traffic is controlled from the client. Both uni-directional and bi-directional concurrent testing are supported.

The UDP mode can be used for the stateless traffic testing. The target bandwidth, packet size, UDP port, and test duration can be controlled. A report of bytes transmitted, packets lost, and delay variation are provided by the iperf receiver.

The TCP mode can be used for stateful traffic testing to test bulk transfer traffic. The TCP Window size (which is actually the SSB), the number of connections, the packet size, TCP port and the test duration can be controlled. A report of bytes transmitted and throughput achieved are provided by the iperf sender.

Flowgrind is a distributed network performance measurement tool. Using the flowgrind controller, tests can be setup between hosts running flowgrind. For the purposes of this traffic management testing framework, the key benefit of Flowgrind is that it can emulate non-bulk transfer applications such as HTTP, Email, etc. This is due to fact that Flowgrind supports the concept of request and response behavior while iperf does not.

Traffic generation options include the request size, response size, inter-request gap, and response time gap. Additionally, various distribution types are supported including constant, normal, exponential, pareto, etc. These powerful traffic generation parameters facilitate the modeling of complex application test patterns at the TCP layer which are discussed in Appendix B.

Since these tools are software based, the host hardware must be qualified to be capable of generating the target traffic loads without packet loss and within the packet delay variation threshold.

Appendix B: Stateful TCP Test Patterns

This framework does not specify a fixed set of TCP test patterns, but proposes two (2) techniques to specify repeatable TCP test patterns for traffic management benchmarking and provides examples of the following test patterns:

- Bulk: generate concurrent TCP connections whose aggregate number of in-flight data bytes would fill the BDP. Guidelines from RFC 6349 are used to create this traffic model.

- Bursty: generate precise burst patterns within a single or multiple TCP session(s). The idea is for TCP to establish equilibrium and then burst application bytes at defined sizes.

- AQM: generate various burst sizes within a TCP session, spacing the bursts apart such that burst size achieved (BSA) can be easily determined. In a sense, this could be considered a TCP stair case or ramp test.

- Small Web Site: mimic the request and response (chatty) and bulk transfer (page download) behavior of a less complex web site. This example uses the modeling technique from Flowgrind to generate this TCP test pattern.

- Cirix: mimic chatty behavior of Citrix. This example uses the packet capture technique to model the behavior and discusses the requirements for test tools to playback the packet capture statefully.

TBD: Detailed definitions for each of the test patterns listed above.

7. Security Considerations

8. IANA Considerations

9. Conclusions

10. References

10.1. Normative References

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

[2] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

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

[RFC2234] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

10.2. Informative References

11. Acknowledgments

Authors' Addresses

Barry Constantine

JDSU, Test and Measurement Division

Germantown, MD 20876-7100, USA

Phone: +1 240 404 2227

Email: barry.constantine@jdsu.com

Timothy Copley

Level 3 Communications

14605 S 50th Street

Phoenix, AZ 85044

Email: Timothy.copley@level3.com

Ram Krishnan

Brocade Communications

San Jose, 95134, USA

Phone: +001-408-406-7890

Email: ramk@brocade.com