Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: September 15, 2011

M. Hamilton
BreakingPoint Systems
S. Banks
Cisco Systems
March 14, 2011

Benchmarking Methodology for Content-Aware Network Devices
draft-hamilton-bmwg-ca-bench-meth-06

Abstract

The purpose of this document is to define a set of test scenarios which may be used to create a series of statistics that will help to better understand the performance of network devices that operate at nework layers above IP. More specifically, these scenarios are designed to most accurately predict performance of these devices when subjected to modern traffic patterns. This document will operate within the constraints of the Benchmarking Working Group charter, namely black box characterization in a laboratory environment.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 15, 2011.

Copyright Notice

Table of Contents

1. Introduction

Content-aware and deep packet inspection (DPI) device penetration has grown significantly over the last decade. No longer are devices simply using Ethernet headers and IP headers to make forwarding decisions. Devices that could historically be classified as 'stateless' or raw forwarding devices are now seeing more DPI functionality. Devices such as core and edge routers are now being developed with DPI functionality to make more intelligent routing and forwarding decisions.

The Benchmarking Working Group (BMWG) has historically produced Internet Drafts and Requests for Comment that are focused specifically on creating output metrics that are derived from a very specific and well-defined set of input parameters that are completely and unequivocally reproducible from testbed to testbed. The end goal of such methodologies is to, in the words of the BMWG charter "reduce specmanship" from network equipment manufacturers(NEM's). Existing BMWG work has certainly met this stated goal.

Today, device sophistication has expanded beyond existing methodologies, allowing vendors to reengage in specmanship. In order to achieve the stated BMWG goals, the methodologies designed to hold vendors accountable must evolve with the enhanced device functionality.

The BMWG has historically avoided the use of the term "realistic" throughout all of its drafts and RFCs. While this document will not explicitly use this term, the end goal of the terminology and methodology is to generate performance metrics that will be as close as possible to equivalent metrics in a production environment. It should be further noted than any metrics acquired from a production network MUST be captured according to the policies and procedures of the IPPM or PMOL working groups.

An explicit non-goal of this document is to replace existing methodology/terminology pairs such as RFC 2544 [1]/RFC 1242 [2] or RFC 3511 [3]/RFC 2647 [4]. The explicit goal of this document is to create a methodology and terminology pair that is more suited for modern devices while complementing the data acquired using existing BMWG methodologies. Existing BMWG work generally revolves around completely repeatable input stimulus, expecting fully repeatable output. This document departs from this mantra due to the nature of modern traffic and is more focused on output repeatability than on static input stimulus.

Some of the terms used throughout this draft have previously been defined in "Benchmarking Terminology for Firewall Performance" RFC

2647 [4]. This document SHOULD be consulted prior to using this document.

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 [5].

2. Scope

Content-aware devices take many forms, shapes and architectures. These devices are advanced network interconnect devices that inspect deep into the application payload of network data packets to do classification. They may be as simple as a firewall that uses application data inspection for rule set enforcement, or they may have advanced functionality such as performing protocol decoding and validation, anti-virus, anti-spam and even application exploit filtering.

It shall be explicitly stated that this methodology does not imply the use of traffic captured from live networks and replayed.

This document is strictly focused on examining performance and robustness across a focused set of metrics that may be used to more accurately predict device performance when deployed in modern networks. These metrics will be implementation independent.

It should also be noted that the purpose of this document is not to perform functional testing of the potential features in the Device/ System Under Test (DUT/SUT)[4] nor specify the configurations that should be tested. Various definitions of proper operation and configuration may be appropriate within different contexts. While the definition of these parameters are outside the scope of this document, the specific configuration of both the DUT and tester SHOULD be published with the test results for repeatability and comparison purposes.

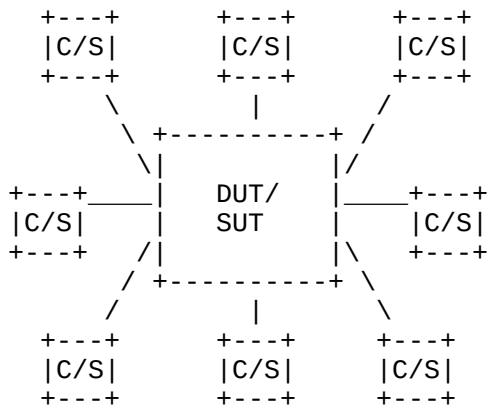
While a list of devices that fall under this category will quickly become obsolete, an initial list of devices that would be well served by utilizing this type of methodology should prove useful. Devices such as firewalls, intrusion detection and prevention devices, application delivery controllers, deep packet inspection devices, and unified threat management systems generally fall into the content-aware category.

3. Test Setup

This document will be applicable to most test configurations and will not be confined to a discussion on specific test configurations. Since each DUT/SUT will have their own unique configuration, users MUST configure their device with the same parameters that would be used in the actual deployment of the device. The DUT configuration MUST be published with the final benchmarking results. If available, command-line scripts used to configured the DUT SHOULD be published with the final results.

The lines between network boundaries are rapidly blurring. No longer are there just single and dual-homed devices; this methodology will be based on a fully meshed network topology. Organizations deploying content-aware devices are doing so throughout their network infrastructure. These devices inspect deep into the application flow to perform quality of service monitoring, filtering, metering, threat mitigation and more.

Figure 1 illustrates a network topology that is fully meshed.



Fully Meshed Device

Figure 1: Fully Meshed Device

3.1. Test Considerations

3.2. Clients and Servers

Content-aware device testing SHOULD involve multiple clients and multiple servers. As with RFC 3511 [3], this methodology will use the terms virtual clients/servers throughout. Similarly defined in RFC 3511 [3], a data source may emulate multiple clients and/or

servers within the context of the same test scenario. The test report MUST indicate the number of virtual clients/servers used during the test. In Appendix C of RFC 2544 [1], the range of IP addresses assigned to the BMWG by the IANA are listed. This address range SHOULD be adhered to in accordance with RFC 2544 [1]. Additionally, section 5.2 of RFC 5180 [6] SHOULD be consulted for the appropriate address ranges when testing IPv6-enabled configurations.

3.3. Traffic Generation Requirements

The explicit purposes of content-aware devices vary widely, but these devices use information deeper inside the application flow to make decisions and classify traffic. This methodology will not utilize traffic flows representing application traffic, but will use the shells of these application flows for benchmarking purposes. The term "Application Flow" is defined in RFC 2722 [7]. Using the shell simply means sending arbitrary payload over the established session rather than actual application payload.

The test tool MUST be able to open TCP connections on multiple destination ports and MUST be able to direct UDP traffic to multiple destination ports. The transport layer payload SHOULD be alternating zeros and ones, but MAY be random.

This document will illustrate an example mix of what traffic may look like on a sample modern network, though the authors understand that no two networks look alike. If a user of this methodology understands the traffic patterns in their modern network, that user MAY use the framework for traffic specification to evaluate their DUT.

3.4. Framework for Traffic Specification

The following table MUST be specified for each application. In cases where there are multiple destination ports, they should be evenly distributed across.

- Percentage of Total Bandwidth: 25%
- Client Originated Flow Bandwidth: 15%
- Server Originated Flow Bandwidth: 85%
- Transport Protocol: TCP
- Destination Port: 80

- o Average Layer 4 Flow Size: 256 kB

3.5. Multiple Client/Server Testing

In actual network deployments, connections are being established between multiple clients and multiple servers simultaneously. Device vendors have been known to optimize the operation of their devices for easily defined patterns. The connection sequence ordering scenarios a device will see on a network will likely be much less deterministic. Thus, users SHOULD setup the test equipment to issue requests at random to the virtual servers rather than in a predictable round-robin fashion. This method will help to appropriately reflect network deployment behavior in the test setup.

3.6. Network Address Translation

Many content-aware devices are capable of performing Network Address Translation (NAT)[4]. If the final deployment of the DUT will have this functionality enabled, then the DUT MUST also have it enabled during the execution of this methodology. It MAY be beneficial to perform the test series in both modes in order to determine the performance differential when using NAT. The test report MUST indicate whether NAT was enabled during the testing process.

3.7. TCP Stack Considerations

As with RFC 3511 [3], TCP options SHOULD remain constant across all devices under test in order to ensure truly comparable results. This document does not attempt to specify which TCP options should be used, but all devices tested SHOULD be subject to the same configuration options.

3.8. Other Considerations

Various content-aware devices will have widely varying feature sets. In the interest of representative test results, the DUT features that will likely be enabled in the final deployment SHOULD be used. This methodology is not intended to advise on which features should be enabled, but to suggest using actual deployment configurations.

4. Benchmarking Tests

4.1. Maximum Application Connection Establishment Rate

4.1.1. Objective

To determine the maximum rate through which a device is able to establish application-specific sessions as defined by RFC 2647 [4].

4.1.2. Setup Parameters

The following parameters MUST be defined for all tests:

4.1.2.1. Transport-Layer Parameters

- Aging Time: The time, expressed in seconds that the DUT will keep a connection in its state table after receiving a TCP FIN or RST packet.

- Maximum Segment Size: The size in bytes of the largest segment which may be sent over a TCP connection.

4.1.2.2. Application-Layer Parameters

For each application protocol in use during the test run, the table provided in Section 3.4 must be published.

4.1.3. Procedure

The test SHOULD generate application network traffic that meets the conditions of Section 3.3. The traffic pattern SHOULD begin with an application session establishment rate of 10% of expected maximum. The test SHOULD be configured to increase the attempt rate in units of 10 up through 110% of expected maximum. The duration of each loading phase SHOULD be at least 30 seconds. This test MAY be repeated, each subsequent iteration beginning at 5% of expected maximum and increasing session establishment rate to 10% more than the maximum observed from the previous test run.

This procedure MAY be repeated any number of times with the results being averaged together.

4.1.4. Measurement

The following metrics MAY be determined from this test, and SHOULD be observed for each application protocol within the traffic mix:

4.1.4.1. Maximum Application Connection Establishment Rate

The test tool SHOULD report the maximum rate at which application connections were established, as defined by RFC 2647 [4], Section 3.7. This rate SHOULD be reported individually for each application

protocol present within the traffic mix.

4.1.4.2. Application Connection Setup Time

The test tool SHOULD report the minimum, maximum and average application setup time, as defined by RFC 2647 [4], Section 3.9. This rate SHOULD be reported individually for each application protocol present within the traffic mix.

4.1.4.3. Application Connection Response Time

The test tool SHOULD report the minimum, maximum and average application session response times. This metric is defined as the time between when the first SYN was sent and the arrival of the corresponding SYN-ACK. This metric does not apply for non connection-based protocols.

4.1.4.4. Application Connection Time To Close

The test tool SHOULD report the minimum, maximum and average application session time to close, as defined by RFC 2647 [4], Section 3.13. This rate SHOULD be reported individually for each application protocol present within the traffic mix.

4.1.4.5. Packet Loss

The test tool SHOULD report the number of network packets lost or dropped from source to destination.

4.1.4.6. Application Latency

The test tool SHOULD report the minimum, maximum and average amount of time an application packet takes to traverse the DUT, as defined by RFC 1242 [2], Section 3.13. This rate SHOULD be reported individually for each application protocol present within the traffic mix.

4.2. Application Throughput

4.2.1. Objective

To determine the maximum rate through which a device is able to forward bits when using stateful applications.

4.2.2. Setup Parameters

The following parameters MUST be defined and reported for all tests:

4.2.2.1. Parameters

The same transport and application parameters as described in Section 4.1.2 MUST be used.

4.2.3. Procedure

This test will attempt to send application data through the device at a session rate of 30% of the maximum established as observed in Section 4.1. This procedure MAY be repeated with the results from each iteration averaged together.

4.2.4. Measurement

The following metrics MAY be determined from this test, and SHOULD be observed for each application protocol within the traffic mix:

4.2.4.1. Maximum Throughput

The test tool SHOULD report the minimum, maximum and average application throughput.

4.2.4.2. Packet Loss

The test tool SHOULD report the number of network packets lost or dropped from source to destination.

4.2.4.3. Application Connection Setup Time

The test tool SHOULD report the minimum, maximum and average application setup time, as defined by RFC 2647 [4], Section 3.9. This rate SHOULD be reported individually for each application protocol present within the traffic mix.

4.2.4.4. Application Connection Response Time

The test tool SHOULD report the minimum, maximum and average application session response times. This metric is defined as the time between when the first SYN was sent and the arrival of the corresponding SYN-ACK. This metric does not apply for non-connection oriented protocols.

4.2.4.5. Application Connection Time To Close

The test tool SHOULD report the minimum, maximum and average application session time to close, as defined by RFC 2647 [4], Section 3.13. This rate SHOULD be reported individually for each application protocol present within the traffic mix.

4.2.4.6. Application Latency

The test tool SHOULD report the minimum, maximum and average amount of time an application packet takes to traverse the DUT, as defined by RFC 1242 [2], Section 3.13. This rate SHOULD be reported individually for each application protocol present within the traffic mix.

4.3. Malicious Traffic Handling

4.3.1. Objective

To determine the effects on performance that malicious traffic may have on the DUT. While this test is not designed to characterize accuracy of detection or classification, it MAY be useful to record these measurements as specified below.

4.3.2. Setup Parameters

The same parameters must be used for Transport-Layer and Application Layer Parameters previously specified in Section 4.1.2 and Section 4.2.2, respectively. Additionally, the following parameters MUST be defined and reported for all tests:

- Attack List: A listing of the malicious traffic that was generated by the test.

4.3.3. Procedure

This test will utilize the procedures specified previously in Section 4.1.3 and Section 4.2.3. When performing the procedures listed previously, during the steady-state time, the tester should generate malicious traffic representative of the final network deployment. The mix of attacks MAY include software vulnerability exploits, network worms, back-door access attempts, network probes and other malicious traffic.

If a DUT may be run with and without the attack mitigation, both procedures SHOULD be run with and without the feature enabled on the DUT to determine the affects of the malicious traffic on the baseline metrics previously derived. If a DUT does not have active attack mitigation capabilities, this procedure SHOULD be run regardless. Certain malicious traffic could affect device performance even if the DUT does not actively inspect packet data for malicious traffic.

4.3.4. Measurement

The metrics specified by Section 4.1.4 and Section 4.2.4 SHOULD be determined from this test.

4.4. Malformed Traffic Handling

4.4.1. Objective

To determine the effects on performance and stability that malformed traffic may have on the DUT.

4.4.2. Setup Parameters

The same parameters must be used for Transport-Layer and Application Layer Parameters previously specified in Section 4.1.2 and Section 4.2.2.

4.4.3. Procedure

This test will utilize the procedures specified previously in Section 4.1.3 and Section 4.2.3. When performing the procedures listed previously, during the steady-state time, the tester should generate malformed traffic at all protocol layers. This is commonly known as fuzzed traffic. Fuzzing techniques generally modify portions of packets, including checksum errors, invalid protocol options, and improper protocol conformance. This test SHOULD be run on a DUT regardless of whether it has built-in mitigation capabilities.

4.4.4. Measurement

For each protocol present in the traffic mix, the metrics specified by Section 4.1.4 and Section 4.2.4 MAY be determined. This data may be used to ascertain the effects of fuzzed traffic on the DUT.

5. Appendix A: Example Test Case

This appendix shows an example case of a protocol mix that may be used with this methodology.

Protocol	Label	Value
Web		
	Total BW	50%
	Client BW	15%
	Server BW	85%
	Transport Protocol	TCP
	Destination Port(s)	80
	Flow Size	256 kB
BitTorrent		
	Total BW	25%
	Client BW	2%
	Server BW	98%
	Transport Protocol	TCP
	Destination Port(s)	6881-6889
	Flow Size	150 MB
SMTP Email		
	Total BW	10%
	Client BW	90%
	Server BW	10%
	Transport Protocol	TCP
	Destination Port(s)	25
	Flow Size	40 kB
IMAP Email		
	Total BW	5%
	Client BW	20%
	Server BW	80%
	Transport Protocol	TCP
	Destination Port(s)	143
	Flow Size	30 kB
DNS		
	Total BW	5%
	Client BW	50%
	Server BW	50%
	Transport Protocol	UDP
	Destination Port(s)	53
	Flow Size	2 kB
RTP		
	Total BW	5%
	Client BW	1%
	Server BW	99%
	Transport Protocol	UDP
	Destination Port(s)	20000-65000
	Flow Size	100 MB

Table 1: Sample Traffic Pattern

6. IANA Considerations

This memo includes no request to IANA.

All drafts are required to have an IANA considerations section (see the update of RFC 2434 [8] for a guide). If the draft does not require IANA to do anything, the section contains an explicit statement that this is the case (as above). If there are no requirements for IANA, the section will be removed during conversion into an RFC by the RFC Editor.

7. Security Considerations

Benchmarking activities as described in this memo are limited to technology characterization using controlled stimuli in a laboratory environment, with dedicated address space and the other constraints RFC 2544 [1].

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network, or misroute traffic to the test management network

8. References

8.1. Normative References

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

[2] Bradner, S., "Benchmarking terminology for network interconnection devices", RFC 1242, July 1991.

[3] Hickman, B., Newman, D., Tadjudin, S., and T. Martin, "Benchmarking Methodology for Firewall Performance", RFC 3511, April 2003.

[4] Newman, D., "Benchmarking Terminology for Firewall Performance", RFC 2647, August 1999.

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

[6] Popoviciu, C., Hamza, A., Van de Velde, G., and D. Dugatkin, "IPv6 Benchmarking Methodology for Network Interconnect Devices", RFC 5180, May 2008.

[7] Brownlee, N., Mills, C., and G. Ruth, "Traffic Flow Measurement: Architecture", RFC 2722, October 1999.

8.2. Informative References

[8] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

Authors' Addresses

Mike Hamilton
BreakingPoint Systems
Austin, TX 78717
US

Phone: +1 512 636 2303
Email: mhamilton@breakingpoint.com

Sarah Banks
Cisco Systems
San Jose, CA 95134
US

Email: sabanks@cisco.com