SOFTWARE-DEFINED WIDE AREA NETWORKING TEST REPORT

Fortinet FortiGate 61E v6.0.1 GA Build 5068

AUGUST 8, 2018

Author –Julian Owusu-Abrokwa

Overview

NSS Labs performed an independent test of the Fortinet FortiGate 61E v6.0.1 GA Build 5068 (which included three Fortigate 61E SD-WAN products). The SD-WAN products were subjected to thorough testing at the NSS facility in Austin, Texas, based on the Software-Defined Wide Area Network (SD-WAN) Test Methodology v1.2, which is available at www.nsslabs.com. This test was conducted free of charge and NSS did not receive any compensation in return for Fortinet's participation.

While the companion Comparative Reports on performance and total cost of ownership (TCO) will provide information about all tested products, this Test Report provides detailed information not available elsewhere.

NSS research indicates that the marriage of software-defined networking (SDN) benefits to wide area network (WAN) technology yields the software-defined wide area network (SD-WAN), which allows consumer-grade links (or links without assured performance) to be leveraged for business-class services. Through the use of common VPN capabilities and the separation of data and control planes within SDN, software-managed connections can be established and managed between multiple sites over any number of link types (e.g., fixed circuit, DSL, cable, mobile, MPLS, etc.) without the operational challenges of having to manage multiple links simultaneously.

The SD-WAN products were configured with Fortinet's pre-defined or recommended (i.e., "out-of-the-box") settings in order to provide readers with relevant quality of experience (QoE) and performance based on their expected usage.

	NSS-Tested VPN Throughput[1]	VoIP QoE Score	Video QoE Score	3-Year TCO (US$)
Fortinet FortiGate 61E v6.0.1 GA Build 5068	749 Mbps	4.38	4.26	$3,522

Figure 1 – Overall Test Results

Using the recommended policy, the FortiGate 61E achieved a VoIP QoE score of 4.38 and a Video QoE score of 4.26 out of maximum achievable scores of 4.41 and 4.53, respectively.

The FortiGate 61E is rated by NSS at 749 Mbps VPN throughput, out of a maximum achievable of 1,092 Mbps per the SD-WAN Test Methodology. *NSS-Tested Throughput* is calculated as a weighted average of the traffic that NSS expects an SD-WAN to experience in an enterprise environment. For more details, please see Appendix A: Scorecard.

.

[1] Performance testing was conducted between Branch A and the headquarters site over two established tunnels and was limited to 1,092 Mbps, as described in the NSS Labs Software-Defined Wide Area Network (SD-WAN) Test Methodology.

Table of Contents

Table of Figures

SD-WAN Test Architecture

The SD-WAN test architecture simulates an enterprise infrastructure with two branches connected to a data center. Each branch location has two links; one is an MPLS link, and the other is a standard broadband connection. The WAN environment is provisioned with behavioral characteristics similar to those typically encountered over normal WAN link states, and the test harness baseline is recorded to ensure consistent behavior. The SD-WAN is deployed and each test case is measured against the baseline. All tests are performed across the VPN links established according to the use case topology.

The traffic flows used in this test were a mix of real-time, interactive, and bulk traffic. The MPLS Link is set to 100 Mbps and the ISP Link is set to 1 Gbps (maximum achievable: 1,092 Mbps). The metrics used to measure the health of the network are VoIP (real-time protocol [RTP]) MOS and video (relative) MOS. A score of 4.41/4.53 represents an excellent voice call/video stream, and any score below 3.5 represents a significantly degraded voice call/video stream. Figure 2 depicts how the SD-WAN products under test were configured and priced.

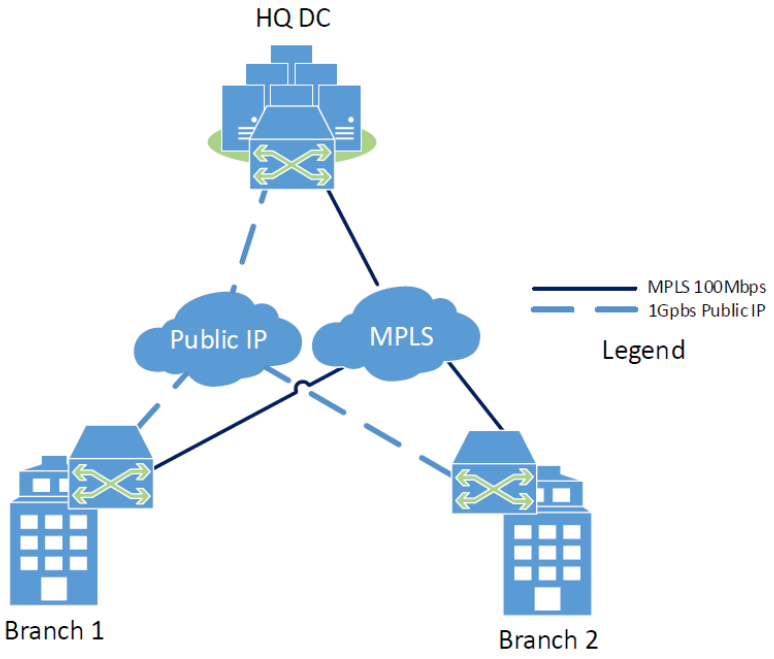

Figure 2 – Testing Architecture

Remote Initial Configuration

Zero-touch provisioning simplifies SD-WAN deployment. This is achieved either by connecting to a cloud service and receiving a base configuration or by connecting to a central management system locally. The FortiGate 61E passed this test.

WAN Impairment and Link Failover

A critical function of any SD-WAN product is the identification and correct routing of traffic based on policy prioritization (autonomous or configured), which is influenced by network performance characteristics (e.g., variability, latency, jitter, etc.). Link impairment tests subject connected links to testing that represents real-world conditions encountered by enterprises today. The solution adapts the capabilities of the WAN for bandwidth, congestion, loss, latency, and jitter in real time.

In each test case, background traffic was introduced to populate links with sufficient activity as to represent typical enterprise network communications. Additionally, traffic-specific flows were introduced in order to capture accurate measurements, including RTP MOS for VoIP, relative MOS for video, and one-way delay for RTP. These measurements provide guidance as to how sensitive applications behave across a tested SD-WAN configuration.

Dynamic Path Selection with SLA Measurements

The goal of this test was to determine how long it took for traffic to move to an available link when preconfigured impairments were applied. To limit any visible user impact, the SD-WAN should support path decisions based on the conditions that exist on those links. The time to select a new path was measured, as was any impact to applications.

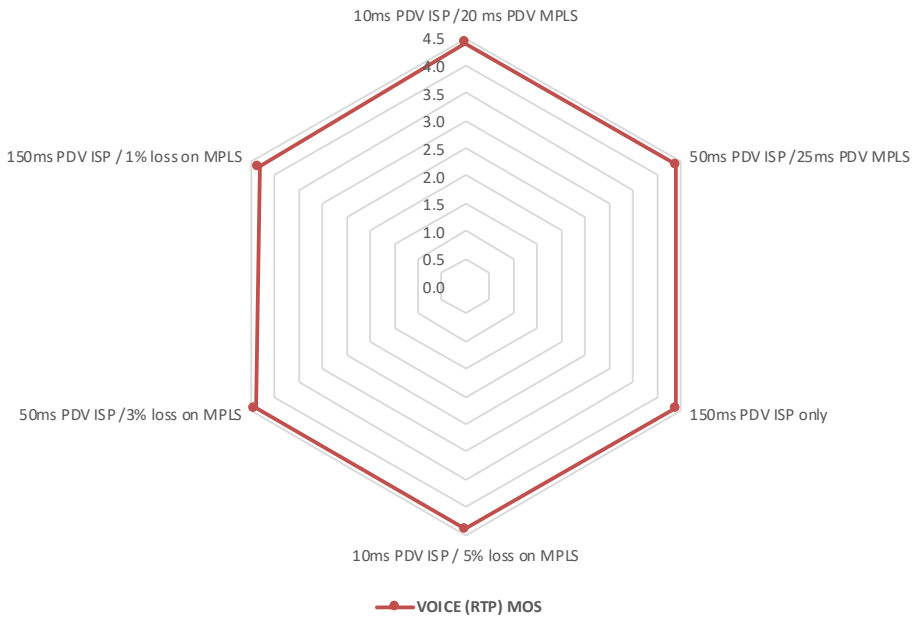

Figure 3 – Packet Delay Variation and Packet Loss (Voice)

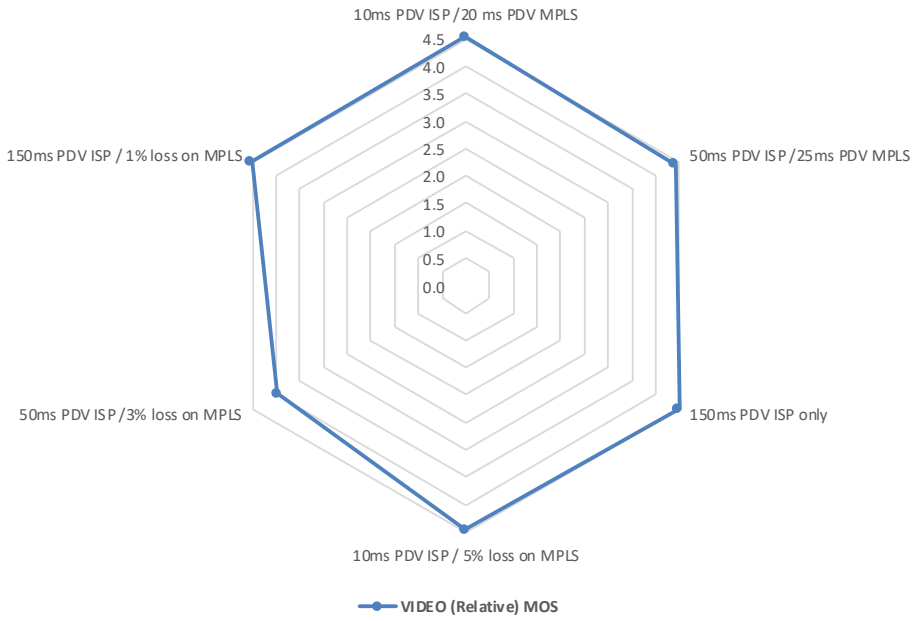

Figure 4– Packet Delay Variation and Packet Loss (Video)

Path Conditioning and Application-Aware Steering

SD-WANs employ various techniques to condition WAN links in order to ensure reliability of data transmission. Some SD-WANs employ packet duplication, forward error correction, bonding, or load balancing. An SD-WAN product should identify the best path and guarantee priority policies (application, protocol, or other configured guidance) over known good links with other traffic transmitted as best effort.

QoS is important for business-critical applications such as voice and video. These applications must be prioritized if a link has bad performance indicators. This test measures QoS using voice traffic and video stream. The test includes MOS scores for video and call measurements for VoIP (one-way delay for RTP). The SD-WAN should manage traffic according to configured QoS classification settings.

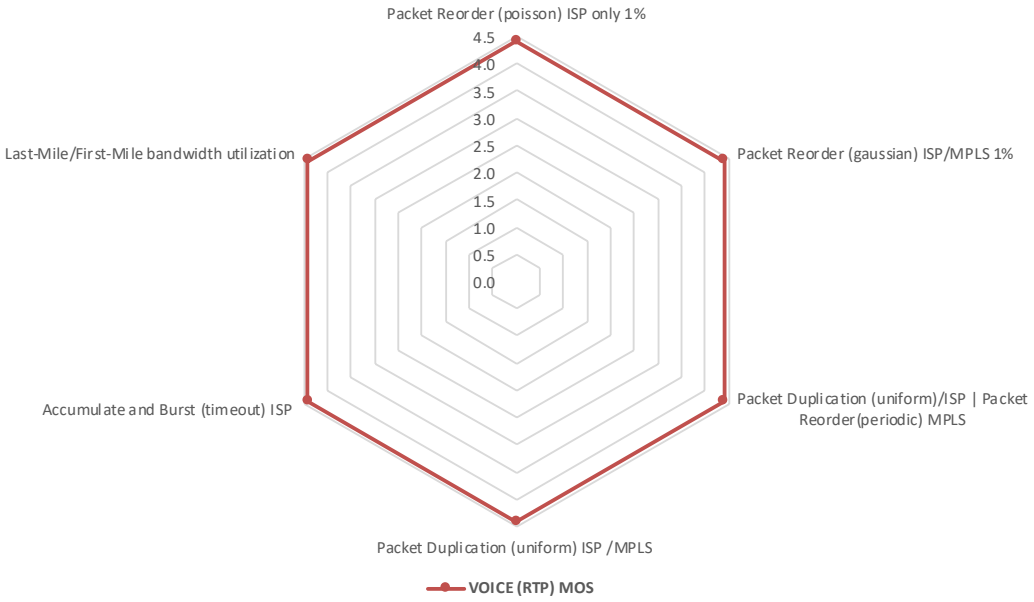

Figure 5 – Path Conditioning and Application-Aware Steering (Voice)

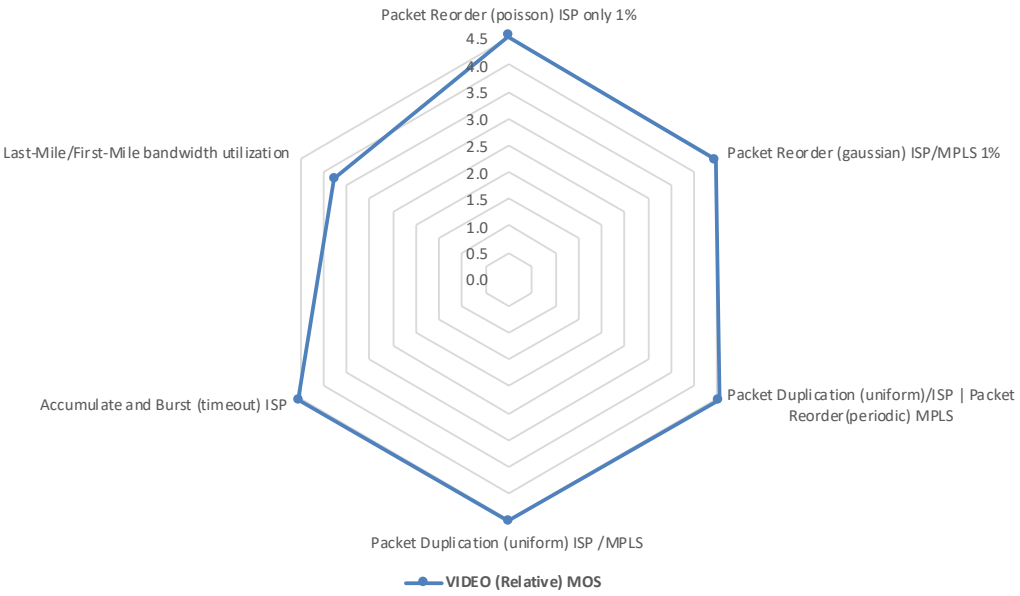

Figure 6 – Path Conditioning and Application-Aware Steering (Video)

Link Saturation and Congestion

As global QoS awareness can prevent congestion during the last mile of data delivery, the goal of this test is to ensure reliable use of bandwidth by the controller in the SD-WAN.

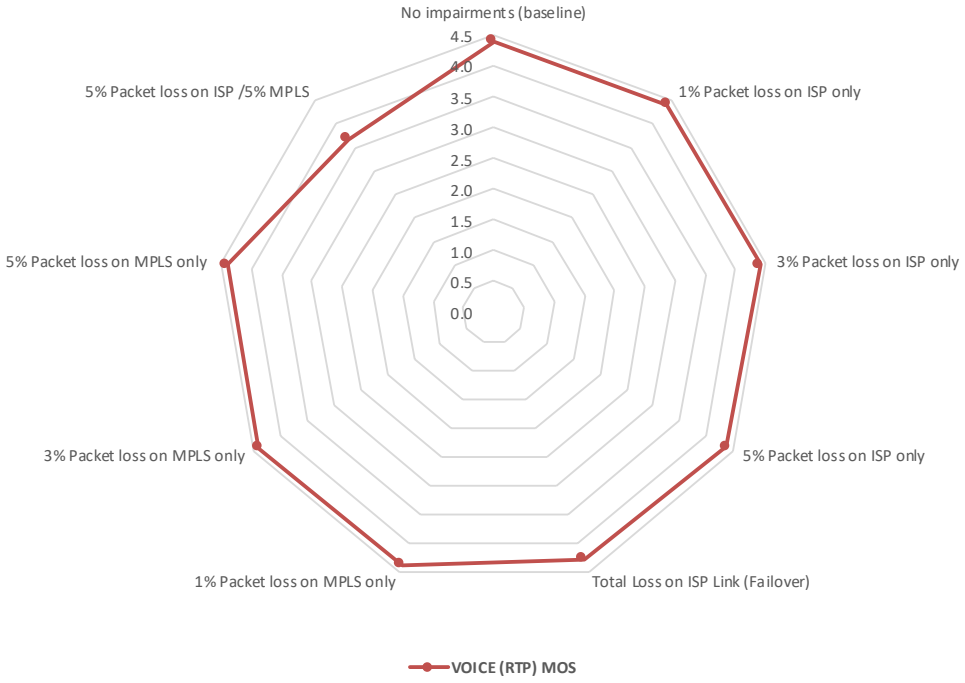

Figure 7 – Congestion and Saturation Impairments (Voice)

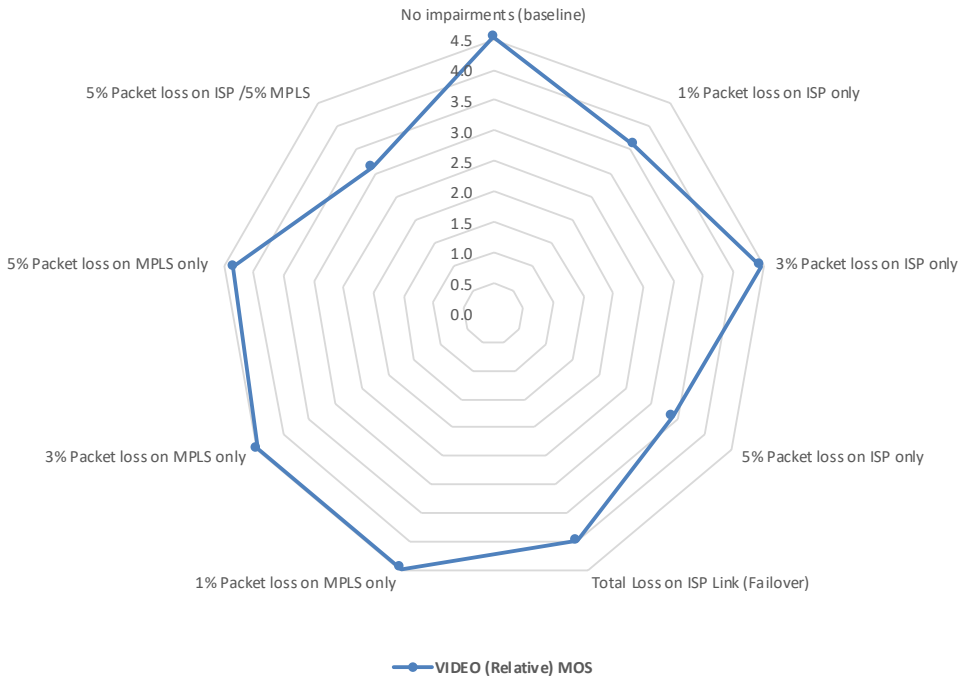

Figure 8 – Congestion and Saturation Impairments (Video)

Application-Aware Traffic Steering

This test verifies how the SD-WAN directs various application traffic flows for applications besides video and VoIP. Behavior was observed and recorded to establish whether voice/video and data were sent over the same link once impairments were applied and to establish which application took precedence. Figure 9 captures the SD-WAN's FTP Connection Latency in milliseconds.

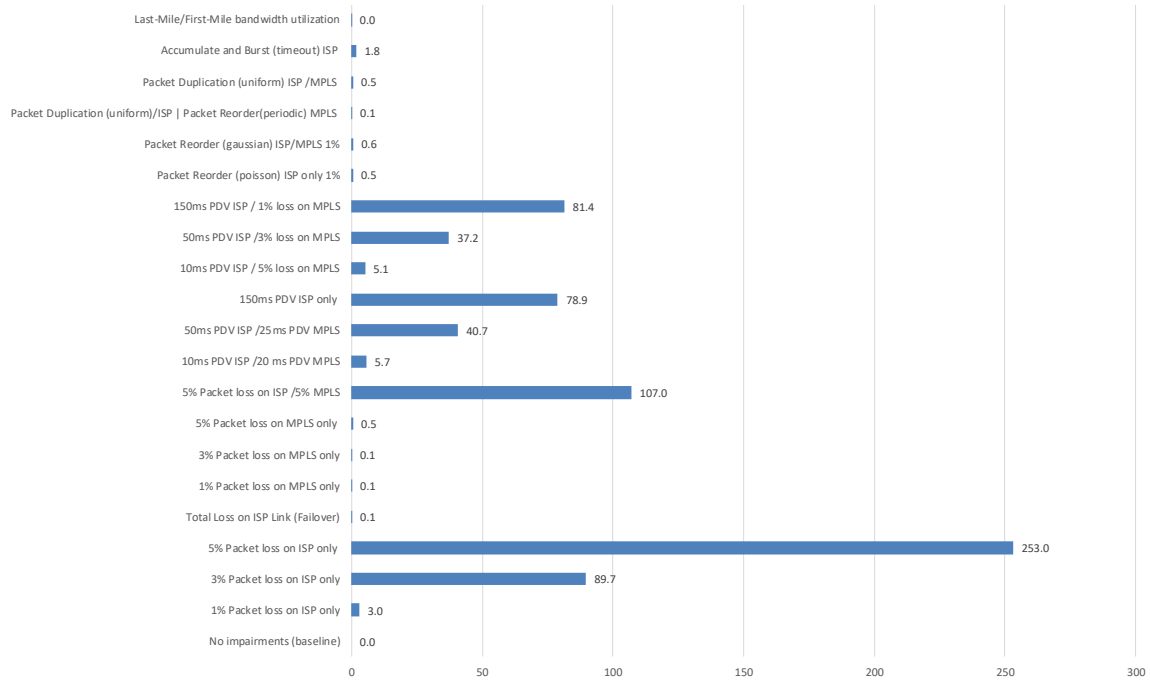

Figure 9 – Application-Aware Traffic Steering with All Impairments (milliseconds)

Single Application Flows

This test measures the performance of the SD WAN using single application flows. These application flows are what NSS expects an SD-WAN product will see in an enterprise environment. Using a frame size distribution ranging from 64 to 1024 bytes, performance testing was conducted between Branch 1 and the headquarters site over two established tunnels and was limited to 1,092 Mbps, as described in the SD-WAN Test Methodology.

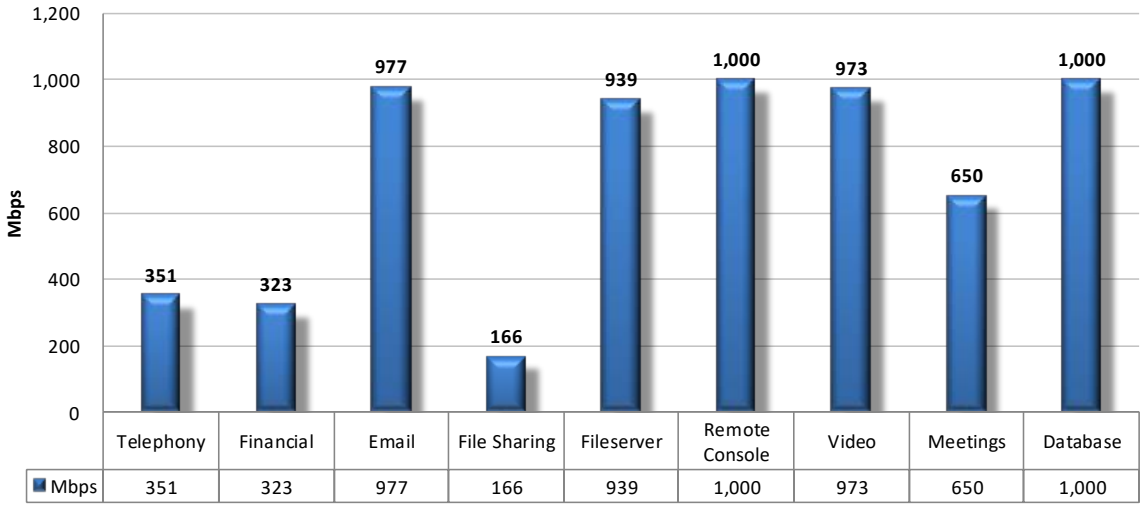

Figure 10 – Single Application Throughput over VPN

Raw Packet Processing Performance (UDP Throughput)

This test uses UDP packets of varying sizes generated by test equipment. A constant stream of the appropriate packet size along with variable source and destination IP addresses is transmitted bidirectionally across the WAN links.

The percentage load and frames per second (fps) figures across the WAN links are verified by network monitoring tools before each test begins. Multiple tests are run and averages are taken where necessary.

The aim of the test is to determine the raw packet processing capability of the SD-WAN as well as its effectiveness at forwarding packets quickly, in order to provide the highest level of network performance with the least amount of latency.

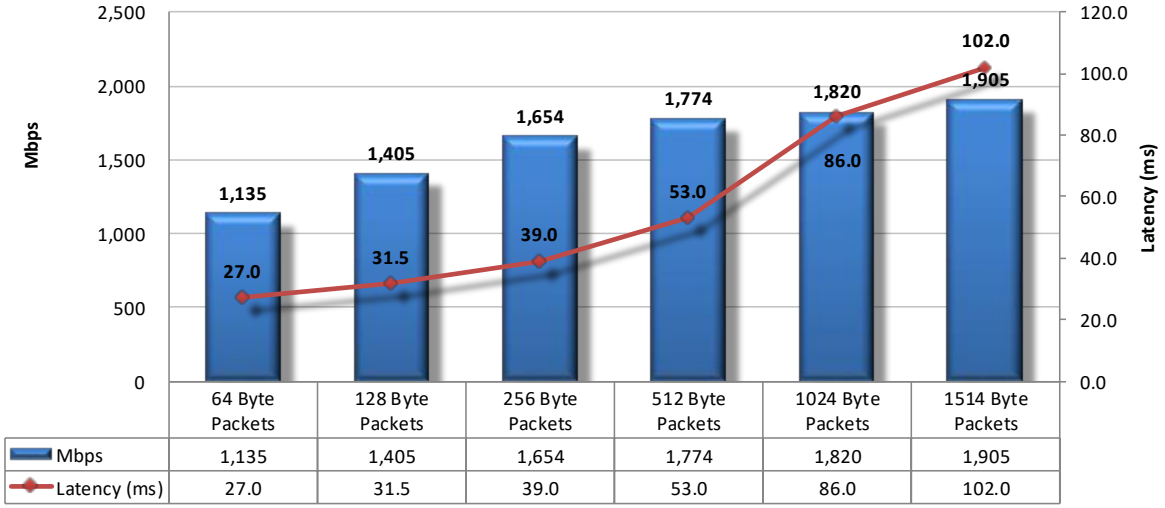

Figure 11 – Raw Packet Processing Performance (UDP Traffic)

Raw Packet Processing Performance (UDP Latency)

SD-WANs that introduce high levels of latency lead to unacceptable response times for users, especially where multiple security devices are placed in the data path. Figure 12 depicts UDP latency (in milliseconds) as recorded during the UDP throughput tests at 90% of maximum load.

Latency – UDP	Milliseconds
64-Byte Packets	27.0
128-Byte Packets	31.5
256-Byte Packets	39.0
512-Byte Packets	53.0
1024-Byte Packets	86.0
1514-Byte Packets	102.0

Figure 12 – UDP Latency in Microseconds

Maximum Capacity

The use of traffic generation appliances allows NSS engineers to create “real-world” traffic at multi-Gigabit speeds as a background load for the tests. Where applicable, the aim of these tests is to stress the inspection engine and determine how it copes with high volumes of TCP connections per second, application-layer transactions per second, and concurrent open connections. All packets contain valid payload and address data, and these tests provide an excellent representation of a live network at various connection/transaction rates.

Note that in all tests the following critical "breaking points"—where the final measurements are taken—are used:

- **Excessive concurrent TCP connections** – Latency within the SD-WAN is causing an unacceptable increase in open connections.
- **Excessive concurrent HTTP connections** – Latency within the SD-WAN is causing excessive delays and increased response time.
- **Unsuccessful HTTP transactions** – Normally, there should be zero unsuccessful transactions. Once these appear, it is an indication that excessive latency within the SD-WAN is causing connections to time out.

Maximum Capacity	CPS
Max Concurrent TCP Connections	1,218,776
Max TCP Connections per Second	33,000
Max HTTP Connections per Second	21,390
Max HTTP Transactions per Second	45,990

Figure 13 – Concurrency and Connection Rates

HTTP Capacity

The aim of the HTTP capacity tests is to stress the HTTP detection engine and determine how the SD-WAN copes with network loads of varying average packet size and varying connections per second. By creating multiple tests using genuine session-based traffic with varying session lengths, the SD-WAN is forced to track valid HTTP sessions, thus ensuring a higher workload than for simple packet-based background traffic.

Each transaction consists of a single HTTP GET request. All packets contain valid payload (a mix of binary and ASCII objects) and address data. This test provides an excellent representation of a live network (albeit one biased toward HTTP traffic) at various network loads.

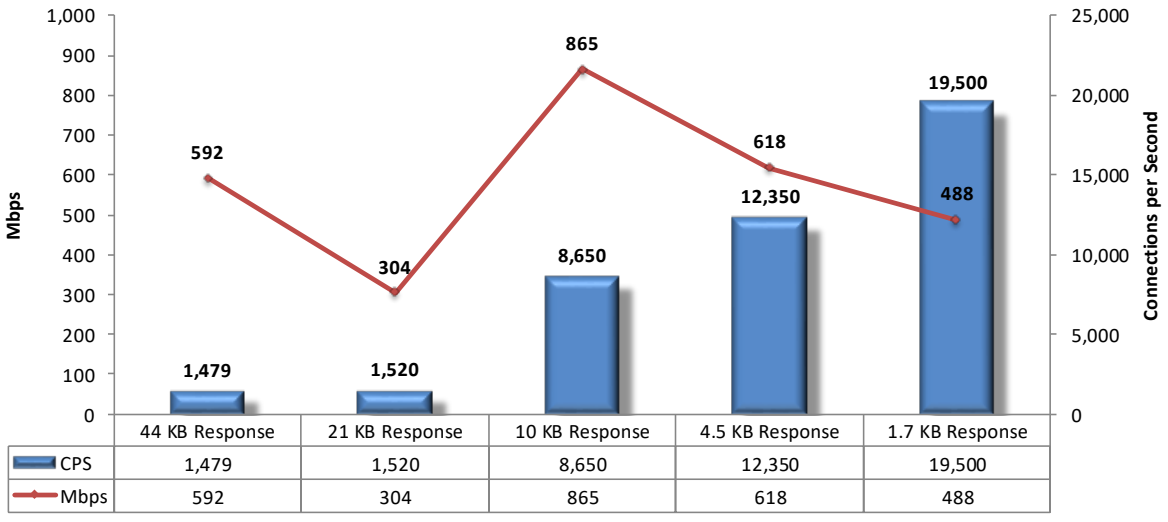

Figure 14 – HTTP Capacity

Application Average Response Time – HTTP

Application Average Response Time – HTTP (at 90% Maximum Load)	Milliseconds
2,500 Connections per Second – 44 KB Response	2.2
5,000 Connections per Second – 21 KB Response	1.1
10,000 Connections per Second – 10 KB Response	2.1
20,000 Connections per Second – 4.5 KB Response	1.0
40,000 Connections per Second – 1.7 KB Response	0.1

Figure 15 – Average Application Response Time (Milliseconds)

HTTP Capacity with HTTP Persistent Connections

This test will use HTTP persistent connections, with each TCP connection containing 10 HTTP GETs and associated responses. All packets contain valid payload (a mix of binary and ASCII objects) and address data, and this test provides an excellent representation of a live network at various network loads. The stated response size is the total of all HTTP responses within a single TCP session.

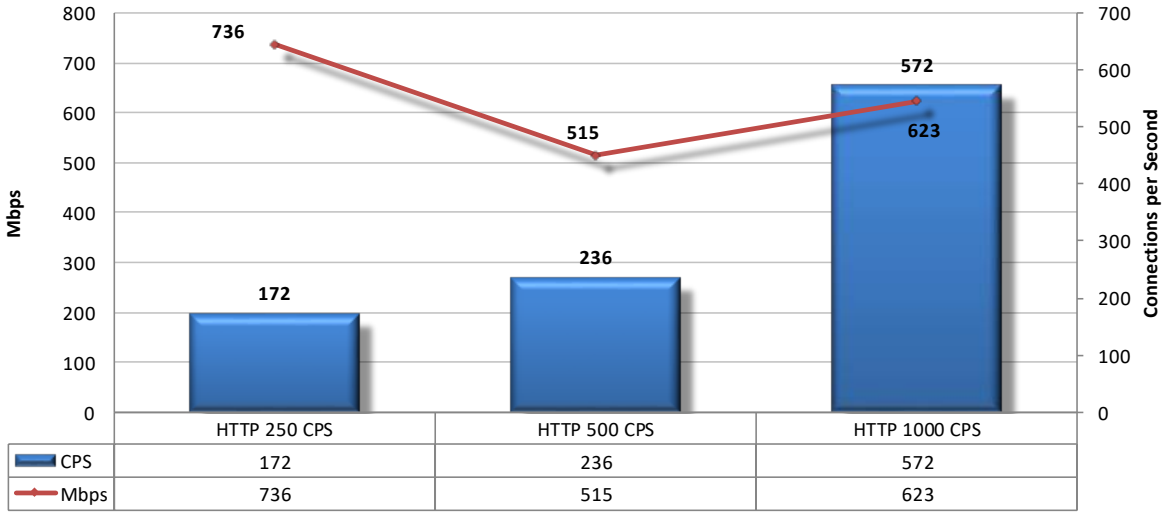

Figure 16 – HTTP Capacity HTTP with Persistent Connections

Total Cost of Ownership (TCO)

Implementation of infrastructure and security solutions can be complex, with several factors affecting the overall cost of deployment, maintenance, and upkeep. All of the following should be considered over the course of the useful life of the SD-WAN:

- **Product Purchase** – The cost of acquisition
- **Operational Benefits –** The "zero-touch" provisioning concept for SD-WAN cites considerably reduced deployment requirements, specifically regarding configuration and tuning; for example, time to add a new site is measured in hours rather than days or weeks. These reduced configuration requirements contribute to operational savings for the enterprise.
- **ROI Assessment –** There are savings associated with moving from high-cost, service-assured links (e.g., MPLS) to commercial broadband. There is value both in aggregating multiple low-cost links to support demand as well as in the ease of deployment and recurring service cost reductions that are associated with moving from expensive, service-assured links to less expensive options.
- **Product Maintenance** – The fees paid to the vendor, including software and hardware support, maintenance, and other updates
- **Installation** – The time required to take the product out of the box, configure it, put it into the network, apply updates and patches, and set up desired logging and reporting
- **Upkeep** – The time required to apply periodic updates and patches from the vendor, including hardware, software, and other updates

For the purposes of this report, capital expenditure (capex) items are included for three SD-WANs only (the cost of acquisition and installation).

Installation Hours

Figure 17 depicts the number of hours of labor required to install each SD-WAN using either local device management or cloud deployment options. The table accurately reflects the amount of time that NSS engineers, with the help of Fortinet engineers, needed to install and configure the SD-WAN to the point where it operated successfully in the test harness, passed legitimate traffic, and blocked and detected prohibited or malicious traffic. This closely mimics a typical enterprise deployment scenario for the tested SD-WAN configuration.

The installation cost is based on the time that an experienced network security engineer would require to perform the installation tasks described above. This approach allows NSS to hold constant the talent cost and measure only the difference in time required for installation. Readers should substitute their own costs to obtain accurate TCO figures.

	Installation (Hours)
Fortinet FortiGate 61E v6.0.1 GA Build 5068	8

Figure 17 – Branch Deployment Time (Hours)

Total Cost of Ownership

Calculations are based on vendor-provided pricing information. Where possible, the 24/7 maintenance and support option with 24-hour replacement is utilized, since this is the option typically selected by enterprise customers. Prices are for SD-WAN management and maintenance only; costs for central management solutions (CMS) may be extra.

	Year 1 Cost	Year 2 Cost	Year 3 Cost	3-Year TCO
Fortinet FortiGate 61E v6.0.1 GA Build 5068	$2,850	$336	$336	$3,522

Figure 18 –3-Year TCO (US$)

- **Year 1 Cost** is calculated by adding installation costs (US$75 per hour fully loaded labor x installation time) + purchase price + first-year maintenance/support fees.
- **Year 2 Cost** consists only of maintenance/support fees.
- **Year 3 Cost** consists only of maintenance/support fees.

For additional TCO analysis, including for the CMS, refer to the TCO Comparative Report.

Appendix A: Scorecard

Description		Result	
WAN Impairment		VoIP	Video
Failover		4.29	3.98
Dynamic Path Selection		4.40	4.40
Path Conditioning		4.41	4.51
Application Aware Steering		4.41	4.14
Performance			
Raw Packet Processing Performance (UDP Traffic)	Weighting for NSS-Rated Throughput	Mbps	
64-Byte Packets	0%	1,135	
128-Byte Packets	1%	1,405	
256-Byte Packets	1%	1,654	
512-Byte Packets	1%	1,774	
1024-Byte Packets	3%	1,820	
1514-Byte Packets	3%	1,905	
Latency – UDP		Microseconds	
64-Byte Packets		27.0	
128-Byte Packets		31.5	
256-Byte Packets		39.0	
512-Byte Packets		53.0	
1024-Byte Packets		86.0	
1514-Byte Packets		102.0	
Maximum Capacity		CPS	
Theoretical Max. Concurrent TCP Connections		1,218,776	
Maximum TCP Connections per Second		33,000	
Maximum HTTP Connections per Second		21,390	
Maximum HTTP Transactions per Second		45,990	
HTTP Capacity	Weighting for NSS-Rated Throughput	CPS	
2,500 Connections per Second – 44 KB Response	8%	1,479	
5,000 Connections per Second – 21 KB Response	8%	1,520	
10,000 Connections per Second – 10 KB Response	7%	8,650	
20,000 Connections per Second – 4.5 KB Response	7%	12,350	
40,000 Connections per Second – 1.7 KB Response	4%	19,500	
Application Average Response Time – HTTP (at 90% Max Load)		Milliseconds	
2.500 Connections per Second – 44 KB Response		2.2	
5,000 Connections per Second – 21 KB Response		1.1	
10,000 Connections per Second – 10 KB Response		2.1	
20,000 Connections per Second – 4.5 KB Response		1.0	
40,000 Connections per Second – 1.7 KB Response		0.1	
HTTP Capacity with HTTP Persistent Connections		CPS	
250 Connections per Second		172	
500 Connections per Second		236	

1000 Connections per Second		572
Single Application Flows	Weighting for NSS-Rated Throughput	Mbps
Telephony	17%	351
Financial	0%	323
Email	12%	977
File Sharing	7%	166
Fileserver	0%	939
Remote Console	1%	1,000
Video	16%	973
Meetings	1%	650
Database	3%	1,000

Impairment					
Impairment Test	WAN Impairment	VIDEO (Relative) MOS	VOICE (RTP) MOS	One-way delay (ms)	FTP Connection Latency (ms)
Failover	No impairments (baseline)	4.53	4.41	0.60	0.0
	1% Packet loss on ISP only	3.59	4.40	0.60	3.0
	3% Packet loss on ISP only	4.46	4.41	0.14	89.7
	5% Packet loss on ISP only	3.39	4.38	0.51	253.0
	Total Loss on ISP Link (Failover)	3.99	4.28	0.20	0.1
	1% Packet loss on MPLS only	4.47	4.37	0.19	0.1
	3% Packet loss on MPLS only	4.48	4.40	0.13	0.1
	5% Packet loss on MPLS only	4.34	4.40	0.58	0.5
	5% Packet loss on ISP /5% MPLS	3.11	3.68	0.43	107.0
Dynamic path selection	10ms PDV ISP /20 ms PDV MPLS	4.53	4.41	9.01	5.7
	50ms PDV ISP /25ms PDV MPLS	4.42	4.41	10.56	40.7
	150ms PDV ISP only	4.50	4.41	0.14	78.9
	10ms PDV ISP / 5% loss on MPLS	4.45	4.41	4.29	5.1
	50ms PDV ISP /3% loss on MPLS	3.96	4.41	23.79	37.2
	150ms PDV ISP / 1% loss on MPLS	4.52	4.32	0.14	81.4
Path Conditioning	Packet Reorder (poisson) ISP only 1%	4.53	4.41	0.13	0.5
	Packet Reorder (gaussian) ISP/MPLS 1%	4.46	4.41	0.32	0.6
	Packet Duplication (uniform)/ISP | Packet Reorder(periodic) MPLS	4.53	4.41	0.14	0.1
	Packet Duplication (uniform) ISP /MPLS	4.52	4.41	2.05	0.5
Application-aware steering	Accumulate and Burst (timeout) ISP	4.53	4.41	4.00	1.8
Application-aware steering	Last-Mile/First-Mile bandwidth utilization	3.75	4.41	0.13	0.0

Security Effectiveness	
Exploit Library	
Exploit Library Block Rate	99.9%
Coverage by Attack Vector	
Attacker Initiated	99.9%
Target Initiated	99.9%
Evasions	
Resistance to Evasion	**PASS**
IP Packet Fragmentation/ TCP Segmentation	PASS
(overlapping small IP fragments favoring new data)	PASS
(overlapping small IP fragments favoring new data in reverse order)	PASS
(overlapping small IP fragments favoring new data in random order)	PASS
(overlapping small IP fragments favoring new data; delay first fragment)	PASS
(overlapping small IP fragments favoring new data in reverse order; delay last fragment)	PASS
(overlapping small IP fragments favoring new data; interleave chaff (invalid IP options))	PASS
(overlapping small IP fragments favoring new data in random order; interleave chaff (invalid IP options))	PASS
(overlapping small IP fragments favoring new data in random order; interleave chaff (invalid IP options); delay random fragment)	PASS
(overlapping small IP fragments favoring new data; interleave chaff (invalid IP options); DSCP value 16)	PASS
(overlapping small IP fragments favoring new data in random order; interleave chaff (invalid IP options); delay random fragment; DSCP value 34	PASS
(small IP fragments)	PASS
(small IP fragments in reverse order)	PASS
(small IP fragments in random order)	PASS
(small IP fragments; delay first fragment)	PASS
(small IP fragments in reverse order; delay last fragment)	PASS
(small IP fragments; interleave chaff (invalid IP options))	PASS
(small IP fragments in random order; interleave chaff (invalid IP options))	PASS
(small IP fragments in random order; interleave chaff (invalid IP options); delay random fragment)	PASS
(small IP fragments; interleave chaff (invalid IP options); DSCP value 16)	PASS
(small IP fragments in random order; interleave chaff (invalid IP options); delay random fragment; DSCP value 34)	PASS
(overlapping small TCP segments favoring new data)	PASS
(overlapping small TCP segments favoring new data in reverse order)	PASS
(overlapping small TCP segments favoring new data in random order)	PASS
(overlapping small TCP segments favoring new data; delay first segment)	PASS
(overlapping small TCP segments favoring new data in reverse order; delay last segment)	PASS
(overlapping small TCP segments favoring new data; interleave chaff (invalid TCP checksums); delay first segment)	PASS
(overlapping small TCP segments favoring new data in random order; interleave chaff (older PAWS timestamps); delay last segment)	PASS

(overlapping small TCP segments favoring new data in random order; interleave chaff (out-of-window sequence numbers); TCP MSS option)	PASS
(overlapping small TCP segments favoring new data in random order; interleave chaff (requests to resynch sequence numbers mid-stream); TCP window scale option)	PASS
(overlapping small TCP segments favoring new data in random order; interleave chaff (requests to resynch sequence numbers mid-stream); TCP window scale option; delay first segment)	PASS
(small TCP segments)	PASS
(small TCP segments in reverse order)	PASS
(small TCP segments in random order)	PASS
(small TCP segments; delay first segment)	PASS
(small TCP segments in reverse order; delay last segment)	PASS
(small TCP segments; interleave chaff (invalid TCP checksums); delay first segment)	PASS
(small TCP segments in random order; interleave chaff (older PAWS timestamps); delay last segment)	PASS
(small TCP segments in random order; interleave chaff (out-of-window sequence numbers); TCP MSS option)	PASS
(small TCP segments in random order; interleave chaff (requests to resynch sequence numbers mid-stream); TCP window scale option)	PASS
(small TCP segments in random order; interleave chaff (requests to resynch sequence numbers mid-stream); TCP window scale option; delay first segment)	PASS
(overlapping small TCP segments favoring new data; small IP fragments)	PASS
(small TCP segments; overlapping small IP fragments favoring new data)	PASS
(overlapping small TCP segments favoring new data; overlapping small IP fragments favoring new data)	PASS
(overlapping small TCP segments favoring new data in random order; small IP fragments in random order)	PASS
(small TCP segments in random order; overlapping small IP fragments favoring new data in random order)	PASS
(overlapping small TCP segments favoring new data in random order; overlapping small IP fragments favoring new data in random order)	PASS
(overlapping small TCP segments favoring new data in random order; overlapping small IP fragments favoring new data in random order; interleave chaff (invalid IP options))	PASS
(overlapping small TCP segments favoring new data; interleave chaff (invalid TCP checksums); small IP fragments; interleave chaff (invalid IP options))	PASS
(small TCP segments; interleave chaff (invalid TCP checksums); overlapping small IP fragments favoring new data; interleave chaff (invalid IP options))	PASS
(small TCP segments; interleave chaff (invalid TCP checksums); delay last segment; overlapping small IP fragments favoring new data; interleave chaff (invalid IP options))	PASS
(small TCP segments; small IP fragments)	PASS
(small TCP segments; small IP fragments in reverse order)	PASS
(small TCP segments in random order; small IP fragments)	PASS
(small TCP segments; small IP fragments in random order)	PASS
(small TCP segments in random order; small IP fragments in reverse order)	PASS
(small TCP segments in random order; interleave chaff (invalid TCP checksums); small IP fragments in reverse order; interleave chaff (invalid IP options))	PASS

(small TCP segments; interleave chaff (invalid TCP checksums); delay last segment; small IP fragments; interleave chaff (invalid IP options))	PASS
(small TCP segments; interleave chaff (invalid TCP checksums); small IP fragments; interleave chaff (invalid IP options); delay last fragment)	PASS
(small TCP segments in random order; interleave chaff (out-of-window sequence numbers); TCP MSS option; small IP fragments in random order; interleave chaff (invalid IP options); delay random fragment)	PASS
(small TCP segments in random order; interleave chaff (requests to resynch sequence numbers mid-stream); TCP window scale option; delay first segment; small IP fragments)	PASS
HTTP Evasions	PASS
(HTTP/0.9 response (no response headers))	PASS
(Declared HTTP/0.9 response; but includes response headers; space (hex '20') after server header)	PASS
(HTTP/1.1 chunked response with chunk sizes followed by a space (hex '20'))	PASS
(HTTP/1.1 chunked response with chunk sizes followed by a tab (hex '09'))	PASS
(HTTP/1.1 chunked response with chunk sizes followed by an 'x' (hex '78'))	PASS
(HTTP/1.1 chunked response with chunk sizes followed by a comma (hex '2c'))	PASS
(HTTP/1.1 chunked response with chunk sizes followed by null character (hex '00'))	PASS
(HTTP/1.1 chunked response with Server header before Status-Line; with chunk sizes followed by a vertical tab (hex '0b'))	PASS
(HTTP/1.1 chunked response with chunk sizes followed by form feed (hex '0c'))	PASS
(HTTP/1.1 chunked response with final chunk size of '00' (hex '20 20' rather than hex '20'))	PASS
(HTTP/1.1 chunked response with final chunk size of '0000000000000000000' (rather than '0'))	PASS
(HTTP/1.1 chunked response with chunk sizes followed by a space (hex '20') then an 'x' (hex '78'))	PASS
(HTTP/1.1 response with line folded transfer-encoding header declaring chunking ('Transfer-Encoding: ' followed by CRLF (hex '0d 0a') followed by space (hex '20') followed by 'chunked' followed by CRLF (hex '0d 0a')); served without chunking)	PASS
(HTTP/1.1 response with transfer-encoding header declaring chunking with lots of whitespace ('Transfer-Encoding: ' followed by 500 spaces (hex '20' * 500) followed by 'chunked' followed by CRLF (hex '0d 0a')); served chunked)	PASS
(HTTP/1.0 response declaring chunking; served without chunking)	PASS
(HTTP/1.0 response declaring chunking with content-length header; served without chunking)	PASS
(<tab>Transfer-Encoding: chunked as first header line; served chunked)	PASS
(<tab>Transfer-Encoding: chunked as continuation of some header line; served chunked)	PASS
(line with empty field name (single colon on line); followed by TE chunked; served chunked)	PASS
(TE chunked prefixed with <CR><CR>;served chunked)	PASS
(HTTP/1.1\nTransfer-Encoding:chunked; served chunked)	PASS
(HTTP/1.1 200 OK\r\nTransfer-Encoding:chunked; served chunked)	PASS
(single \n instead of \r\n and chunked)	PASS
(HTTP/1.1\rTransfer-Encoding: chunked; served chunked)	PASS
(double <LF> before header; chunked)	PASS
(double <CR><LF> before header; chunked)	PASS
(junk followed by single <CR><LF> before header; chunked)	PASS
(SIP/2.0 200 ok followed by single <CR><LF> before header; chunked)	PASS
(space+junk followed by single <CR><LF> before header; chunked)	PASS

(space+"SIP/2.0 200 ok" followed by single <CR><LF> before header; chunked)	PASS
(single <LF> before header; chunked)	PASS
(H before header; chunked)	PASS
(HT before header; chunked)	PASS
(HTT before header; chunked)	PASS
(HTTX before header; chunked)	PASS
(HTTXY before header; chunked)	PASS
(HTTP/1.1 response with content-encoding header for gzip; followed by content-encoding header for deflate; no space between ':' and declaration of encoding types; served with no compression)	PASS
(HTTP/1.1 response with content-encoding declaration of "gzip x"; served uncompressed)	PASS
(header end \n\r\n; gzip)	PASS
(header end \n\r\n; gzip with content-length)	PASS
(header end \n\013\n\n and gzip)	PASS
(header end \n\013\n\n and gzip with content length)	PASS
(header end \r\n\013\r\n\r\n and gzip)	PASS
(header end \r\n\013\r\n\r\n and gzip with content-length)	PASS
(header end \n\r\r\n; gzip)	PASS
(header end \n\r\r\n; gzip with content-length)	PASS
(header end "\n\x20 \n" and gzip)	PASS
(header end "\n\x20\n" and gzip with content-length)	PASS
(header end \n\011\n and gzip)	PASS
(header end \n\011\n and gzip with content-length)	PASS
(header end \n\n; gzip)	PASS
(HTTP/1.0 response with status code 100 followed by message-body; no content-length header)	PASS
(HTTP/1.0 response with status code 206 followed by message-body; no content-length header)	PASS
(HTTP/1.0 response with status code 304 followed by message-body; no content-length header)	PASS
(HTTP/1.0 response with status code 404 followed by message-body; no content-length header)	PASS
(HTTP/1.0 response with status code 500 followed by message-body; no content-length header)	PASS
(HTTP/1.1 response with status code 600 followed by a space; followed by message-body)	PASS
(HTTP/1.1 response with status code 900 followed by a space; followed by message-body)	PASS
(status code 101 with body)	PASS
(status code 102 with body)	PASS
(HTTP/1.1 response with content-length header size declaration followed by space and letter A (hex '20 41'))	PASS
(Chunked Header and HTTP/1.01. Served chunked)	PASS
(Chunked Header and HTTP/1.10. Served chunked)	PASS
(Chunked Header and HTTP/01.1. Served chunked and with gzip)	PASS
(Chunked Header and HTTP/11.01. Served chunked and with gzip)	PASS
(Chunked Header and HTTP/11.10. Served chunked and with gzip)	PASS
(version HTTP/1.010 instead of HTTP/1.1 and chunked)	PASS
(version HTTP/2.B instead of HTTP/1.1 and chunked)	PASS
(version HTTP/9.-1 instead of HTTP/1.1 and chunked)	PASS

(double Transfer-Encoding: first empty; last chunked. Served with content-length and gzipped; not chunked)	PASS
HTML Evasions	PASS
(aaencode)	PASS
(babel-minify)	PASS
(closure)	PASS
(code-protect)	PASS
(confusion)	PASS
(jfogs)	PASS
(jfogs-reverse)	PASS
(jjencode)	PASS
(jsbeautifier)	PASS
(jsmin)	PASS
(js-obfuscator)	PASS
(qzx-obfuscator)	PASS
(scripts_encryptor)	PASS
(stunnix)	PASS
(uglifyjs-es)	PASS
(chunked aaencode)	PASS
(chunked babel-minify)	PASS
(chunked closure)	PASS
(chunked code-protect)	PASS
(chunked confusin)	PASS
(chunked jfogs)	PASS
(chunked jfogs-reverse)	PASS
(chunked jjencode)	PASS
(chunked jsbeautifier)	PASS
(chunked jsmin)	PASS
(chunked js-obfuscator)	PASS
(chunked qzx-obfuscator)	PASS
(chunked scripts_encryptor)	PASS
(chunked stunnix)	PASS
(chunked uglifyjs-es)	PASS
(gzip compressed aaencode)	PASS
(gzip compressed babel-minify)	PASS
(gzip compressed closure)	PASS
(gzip compressed code-protect)	PASS
(gzip compressed confusion)	PASS
(gzip compressed jfogs)	PASS
(gzip compressed jfogs-reverse)	PASS
(gzip compressed jjencode)	PASS
(gzip compressed jsbeautifier)	PASS
(gzip compressed jsmin)	PASS

(gzip compressed js-obfuscator)	PASS
(gzip compressed qzx-obfuscator)	PASS
(gzip compressed scripts_encryptor)	PASS
(gzip compressed stunnix)	PASS
(gzip compressed uglifyjs-es)	PASS
(deflate compressed aaencode)	PASS
(deflate compressed babel-minify)	PASS
(deflate compressed closure)	PASS
(deflate compressed code-protect)	PASS
(deflate compressed confusion)	PASS
(deflate compressed jfogs)	PASS
(deflate compressed jfogs-reverse)	PASS
(deflate compressed jjencode)	PASS
(deflate compressed jsbeautifier)	PASS
(deflate compressed jsmin)	PASS
(deflate compressed js-obfuscator)	PASS
(deflate compressed qzx-obfuscator)	PASS
(deflate compressed scripts_encryptor)	PASS
(deflate compressed stunnix)	PASS
(deflate compressed uglifyjs-es)	PASS
(UTF-8 encoding with BOM)	PASS
(UTF-8 encoding)	PASS
(UTF-16 encoding with BOM)	PASS
(UTF-8 encoding; no http or html declarations)	PASS
(UTF-8 encoding with BOM; no http or html declarations)	PASS
(UTF-16 encoding with BOM; no http or html declarations)	PASS
(padded with 1MB)	PASS
(padded with 15MB)	PASS
(padded with 30MB)	PASS
(padded with 1MB and chunked)	PASS
(padded with 15MB and chunked)	PASS
(padded with 30MB and chunked)	PASS
(padded with 1MB and compressed with gzip)	PASS
(padded with 15MB and compressed with gzip)	PASS
(padded with 30MB and compressed with gzip)	PASS
(padded with 1MB and compressed with deflate)	PASS
(padded with 15MB and compressed with deflate)	PASS
(padded with 30MB and compressed with deflate)	PASS
Total Cost of Ownership	
Ease of Use	
Initial Setup (Hours)	8
Time Required for Upkeep (Hours per Year)	See Comparative

Time Required to Tune (Hours per Year)	See Comparative
Expected Costs	
Initial Purchase (hardware as tested)	$1,914
Installation Labor Cost (@$75/hr)	$600
Annual Cost of Maintenance & Support (hardware/software)	$336
Annual Cost of Updates (IPS/AV/etc.)	$0
Initial Purchase (enterprise management system)	See Comparative
Annual Cost of Maintenance & Support (enterprise management system)	See Comparative
Total Cost of Ownership	
Year 1	$2,850
Year 2	$336
Year 3	$336
3 Year Total Cost of Ownership	$3,522

Figure 19 – Detailed Scorecard

Test Methodology

NSS Labs Software-Defined Wide Area Networking (SD-WAN) Test Methodology v1.2

A copy of the test methodology is available on the NSS Labs website at www.nsslabs.com.

Contact Information

3711 South Mopac Expressway
Building 1, Suite 400
Austin, TX 78746
info@nsslabs.com
www.nsslabs.com