TEST METHODOLOGY

Next Generation Firewall (NGFW)

DECEMBER 7, 2017

V8.0

Table of Contents

1 Introduction

1.1 The Need for Next Generation Firewalls (NGFWs)

Firewall technology is one of the largest and most mature security markets. Firewalls have undergone several stages of development, from early packet filtering and circuit relay firewalls to application layer (proxy-based) and dynamic packet filtering firewalls. Throughout their history, however, the goal has been to enforce an access control policy between two networks, and they should therefore be viewed as an implementation of policy.

A firewall is a mechanism used to protect a trusted network from an untrusted network, while allowing authorized communications to pass from one side to the other, thus facilitating secure business use of the Internet. With the emergence of new web applications and security threats, however, firewalls are evolving further. Next generation firewalls (NGFWs) traditionally have been deployed to defend the network on the edge, but enterprises have expanded deployment options to include internal segmentation.

As Web 3.0 trends push critical business applications through firewall ports that previously were reserved for a single function, such as HTTP, legacy firewall technology is effectively blinded. It is unable to differentiate between actual HTTP traffic and non-HTTP services tunneling over port 80, such as VoIP or instant messaging. Today, application-level monitoring must be performed in addition to analysis of port and destination. Firewalls are evolving to address this increased complexity.

It is no longer possible to rely on port and protocol combinations alone to define network applications. The next generation firewall (NGFW) must be capable of performing deep packet inspection on all packets, on all ports, and over all protocols in order to determine which applications are running over which ports and thus secure them effectively. In addition, with the expanded use of SSL/TLS in much of the traffic traversing the modern network, inspection of encrypted content is required.

1.2 About This Test Methodology

NSS Labs' test reports are designed to address the challenges faced by enterprise security and IT professionals in selecting and managing security products. The scope of this particular methodology includes:

- Security effectiveness
- Performance
- Stability and reliability
- Total cost of ownership (TCO)

As NGFWs are deployed at critical choke points in the network, their stability and reliability is imperative. Therefore, regardless of any new deep inspection capabilities, the main requirement of any NGFW is that it must be as stable, as reliable, as fast, and as flexible as the firewall that it is replacing.

The following capabilities are considered essential in an NGFW:

- Traditional "first generation firewall" features, including:
    - Basic packet filtering
    - Stateful multi-layer inspection
    - Network Address Translation (NAT)

  - Virtual private network (VPN)
- "Next generation firewall" features, including:
  - Application awareness/control
  - User/group control
  - Integrated intrusion prevention system (IPS)
  - Ability to operate at OSI Layer 3 ("traditional")
  - External intelligence to enhance blocking decisions (i.e., "reputation services")
  - Anti-malware
  - SSL inspection ability

Tuning: Security engineers typically will tune an IPS to ensure its protection coverage matches the needs of the environment it is being placed in. Though this strategy works well for data centers and demilitarized zones (DMZs), protecting desktops is a different matter. In surveying enterprises, NSS researchers discovered that many enterprises do not strictly control the desktop, and in larger enterprises, there can be a wide range of applications running on the typical endpoint. As such, enterprises expect IPS and NGFW vendors to provide maximum security for desktop client applications with their out-of-the-box *recommended* or *default* policies. In addition, research indicates that enterprises are not ready to replace their dedicated IPS solutions in the data center with NGFWs.

This leads us to conclude that the intrusion prevention functionality within an NGFW typically will be used to protect desktop clients, with optimal protection predefined via vendor-provided settings, and this is how the devices will be tested. Anti-malware technology is allowed in NGFW 8.0 testing, as many vendors have integrated core security engine functions into their anti-malware functionality.

1.3 Inclusion Criteria

In order to encourage the greatest participation, and to allay any potential concerns of bias, NSS invites all security vendors claiming NGFW capabilities to submit their products at no cost. Vendors with major market share, as well as challengers with new technology, will be included.

The NGFW should be supplied as a single appliance, where possible (cluster controller solutions are also acceptable), with the appropriate number of physical interfaces capable of achieving the required level of connectivity and performance (minimum of one inline port pair per Gigabit of throughput, or one inline 10 Gbps port pair per 10 Gbps of throughput).

Firewall products in a traditional edge deployment must be implemented as Layer 3 (routing) devices. 1 Gbps (copper) or 10 Gbps (fiber) connections will be made from the external to internal switches via the device under test, subject to a minimum of one inline port pair per Gigabit (copper) or per 10 Gigabits (fiber) of throughput. Thus, an 8 Gbps device with only four 1Gbps port pairs will be limited to 4 Gbps. The test will be limited to a 10 G maximum throughput test. The management interface must be 1 Gb copper, or 10 Gb fiber.

Once installed in the test lab, the device will be configured for the use case appropriate to the target deployments (corporate network perimeter and internal segmentation). The device must also be configured to block all traffic when resources are exhausted or when traffic cannot be analyzed for any reason.

2 Product Guidance

NSS issues summary product guidance based on evaluation criteria that is important to information security professionals. The evaluation criteria are as follows:

- **Security effectiveness** – The purpose of an NGFW is to separate internal trusted networks from external untrusted networks through policy and routing, and to identify and block attacks against assets while allowing select controlled traffic to flow between trusted and untrusted networks.
- **Resistance to evasion** – Failure in any evasion class permits attackers to circumvent protection.
- **Stability and reliability** – Long-term stability is particularly important for an inline device, where failure can produce network outages.
- **Performance** – Correctly sizing an NGFW is essential.
- **Value** – Customers should seek low TCO and high effectiveness and performance rankings.

Products are listed in rank order according to their guidance rating.

2.1 Recommended

A *Recommended* rating from NSS indicates that a product has performed well and deserves strong consideration. Only the top technical products earn a *Recommended* rating from NSS, regardless of market share, company size, or brand recognition.

2.2 Neutral

A *Neutral* rating from NSS indicates that a product has performed reasonably well and should continue to be used if it is the incumbent within an organization. Products that earn a *Neutral* rating from NSS deserve consideration during the purchasing process.

2.3 Caution

A *Caution* rating from NSS indicates that a product has performed poorly. Organizations using one of these products should review their security posture and other threat mitigation factors, including possible alternative configurations and replacement. Products that earn a *Caution* rating from NSS should not be short-listed or renewed.

3 Security Effectiveness

This section verifies that the device is capable of enforcing a specified security policy effectively. NSS firewall analysis is conducted by incrementally building upon a baseline configuration (simple routing with no policy restrictions) to a complex, real-world, multiple-zone configuration supporting many addressing modes, policies, applications, and inspection engines.

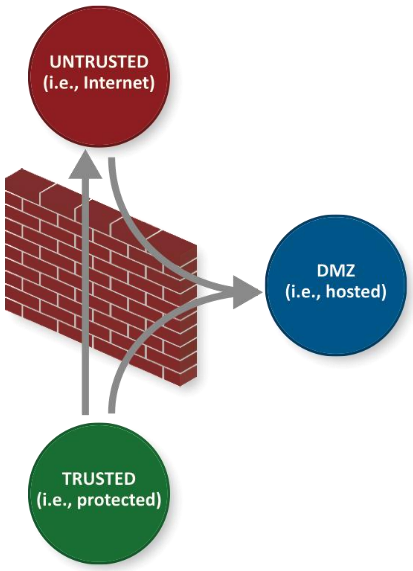
At each level of complexity, test traffic is passed across the firewall to ensure that only specified traffic is allowed, and the rest denied, and that the appropriate log entries are recorded. Administrative visibility is critical; to facilitate debugging, NSS recommends logging any function that results in dropped traffic.

The NGFW must support stateful firewalling either by managing state tables to prevent "traffic leakage," or as a stateful proxy. The NGFW must be able to manage firewall policy across multiple interfaces/zones. NSS also requires that a single security policy be applied to all interfaces under test. At a minimum, the firewall must provide a "trusted" internal interface, an "untrusted" external/Internet interface, and (optionally) one or more DMZ interfaces. In addition, a dedicated management interface (virtual or otherwise) is preferred.

3.1 Firewall Policy Enforcement

Policies are rules that are configured on a firewall to permit or deny access from one network resource to another, based on identifying criteria such as source, destination, and service. A term typically used to define the demarcation point of a network where policy is applied is a *demilitarized zone* (DMZ). Policies are typically written to permit or deny network traffic from one or more of the following zones:

- **Untrusted –** This is typically an external network and is considered to be unknown and not secure. An example of an untrusted network would be the Internet.
- **DMZ –** This is a network that is being isolated by the firewall restricting network traffic to and from hosts contained within the isolated network.
- **Trusted –** This is typically an internal network; a network that is considered secure and protected.



NSS' firewall tests verify performance and the ability to enforce policy between the following:

- Trusted to Untrusted
- Untrusted to DMZ
- Trusted to DMZ

Note: Firewalls must provide at a minimum one DMZ interface in order to provide a DMZ or "transition point" between untrusted and trusted networks.

3.1.1 Baseline Policy

This test uses a routed configuration with an “allow all” policy.

3.1.2 TCP Split Handshake Spoof

This test attempts to confuse the firewall into allowing traffic to pass from one network segment to another. The TCP split handshake blends features of both the three-way handshake and the simultaneous-open connection.

The result is a TCP spoof attack that allows an attacker to bypass the firewall by instructing the target to “initiate” the session back to the attacker. Popular TCP/IP networking stacks respect this handshaking method with no modification, including Microsoft, Apple, and Linux stacks.[1]

The device is expected to protect against TCP split handshake spoofing.

3.2 Intrusion Prevention

These are policies consisting of threat protection signatures, which verify that the device is capable of correctly blocking malicious traffic based on a comparison of packet/session contents against signatures/filters/protocol decoders.

The latest signature pack is acquired from the vendor’s support site, and the device is deployed using vendor-provided settings. The vendor may not tune the product. Once deployed, the device's inspection capabilities are governed solely through firmware and signature updates. All signatures used must be available to the general public at the time of testing; no custom signatures are permitted.NSS research has found that this approach reflects a typical enterprise deployment and will align results from testing with product performance in the field. The NGFW is required to block and log exploit attempts and hostile traffic.

NSS considers it unacceptable for a product of this nature to be sold without at least one vendor-provided profile. If multiple preconfigured profiles exist, vendors may use the highest-security profile available as their recommended profile, as long as that profile meets the following criteria:

- The security profile must be available to all of the vendor’s customers.
- With the security profile installed, the device must experience zero false positive events during testing.

If a device does experience false positive events for the selected security profile, the vendor will be **required** to select a lower-security profile, and the test process will be repeated until a profile is found that produces zero false positives.

Note: NSS will make a distinction between “false positive” and “true positive” events, where a device legitimately identifies a problem with the test traffic. The device will not be required to be reconfigured as a result of “true positive” events.

[1] *“The TCP Split Handshake: Practical Effects on Modern Network Equipment,”* Tod Alien Beardsley & Jin Qian, http://www.macrothink.org/journal/index.php/npa/article/view/285

3.2.1 False Positive Testing

The ability of the device to identify and allow legitimate traffic while maintaining protection against threats and exploits is just as important as its abilility to protect against malicious content. This test will include a varied sample of legitimate application traffic, which should be identified and allowed, or blocked, based on policy rules.

3.2.2 Exploit Library

NSS' *Security Effectiveness* testing leverages the deep expertise of our engineers who utilize multiple commercial, open-source, and proprietary tools, including NSS' network live stack test environment[2] as appropriate. With thousands of exploits, this is the industry's most comprehensive test to date. Most notably, all of the live exploits and payloads in the NSS exploit test have been validated in our lab such that one or more of the following is true:

- A reverse shell is returned
- A bind shell is opened on the target allowing the attacker to execute arbitrary commands
- Arbitrary code is executed
- A malicious payload is installed
- A system is rendered unresponsive
- Etc.

This test goes far beyond replaying packet captures or pressing the button on a test tool. In short, NSS engineers trigger vulnerabilities for the purpose of validating that an exploit was able to pass through the device.

3.2.3 Coverage by Attack Vector

Threats and exploits can be initiated by either the target or the attacker targeting local or remote vulnerabilities. Common examples of attacker-based threats and target-based threats for both local and network vulnerabilities are presented below:

	Network	Local
Attacker	RPC Exploit	Root Kit
Target	Browser Exploit	Trojan

*Example exploits included above for reference purposes.

3.2.3.1 Attacker Initiated

Also referred to as "server-side" exploits, the threat/exploit is executed remotely by the attacker against a vulnerable application and/or operating system.

3.2.3.2 Target Initiated

The threat/exploit is initiated by the vulnerable target. The attacker has little or no control as to when the target user or application will execute the threat. Since NGFW devices are typically deployed to protect end users, this class of exploit is the main focus of the NGFW *Security Effectiveness* test.

[2] For more information on the NSS "Live Testing™" harness and methodology, please refer to the latest *Security Stack (IPS): Test Methodology* located at https://www.nsslabs.com/research-advisory/library/search/category/library/methodologies/

3.2.3.3 Network

Threats/exploits that are initiated as a result of network communication.

3.2.3.4 Local

Local execution that requires existing access to the target (not applicable to the NGFW Test Methodology).

Protective ratings are reported in raw percentages of mitigated attacks and their resulting impact: system, service, fault, reconnaissance. Although a system or service exploit may be partially mitigated by the device, the service may have crashed because of residual communications resulting in a fault impact on the service or operating system.

3.2.4 Coverage by Impact Type

The *NSS Exploit Library* contains thousands of publicly available exploits (including multiple variants of each exploit), from which groups of exploits are carefully selected to test based on appropriate usage. Each exploit has been validated to impact the target vulnerable host(s).

3.2.4.1 System Exposure

These attacks result in remote system compromise, which provide the attacker with the ability to execute arbitrary system-level commands. Most of the exploits in this class are weaponized and offer the attacker a fully interactive remote shell on the target client or server.

3.2.4.2 Service Exposure

Such attacks result in an individual service compromise, but they do not provide the attacker with the ability to execute arbitrary, system-level commands, nor do they immediately result in full system-level access to the operating system and all services. However, by using additional localized system attacks, it may be possible for the attacker to move from the service level to the system level.Typical attacks in this category include service-specific attacks, such as SQL injection, that enable the attacker to execute arbitrary SQL commands within the database service.

3.2.4.3 System or Service Fault

These attacks result in a system or service-level fault that crashes the targeted service or application and requires administrative action to restart the service or reboot the system. These attacks do not enable the attacker to execute arbitrary commands. However, the resulting impact to the business could be severe given that the attacker could crash the protected system or service.

3.2.5 Coverage by Date

The typical enterprise will run a mix of both old and new applications, and NSS research shows that crimeware kits will frequently include exploits that date back several years. Therefore, NSS security effectiveness testing will include exploits current at the time of the test, as well as target vulnerabilities covering multiple years dating backwards from the time of the test. Results will be reported by year, beginning in 2005, extending to the year of the most current NGFW test. Where applicable, results prior to that time period, will be aggregated into the oldest "bucket."

Exploits are added to the NSS Strike Packs according to the year the strike was added to the NSS exploit harness, rather than according to the year that the CVE was discovered and documented. For example, NSS may add an exploit with a CVE indicating a date of 2013 to the 2016 NSS Strike Pack.

3.2.6 Coverage by Vendor

NSS' exploit test contains many vendors, including but not limited to the following.

- 3Com
- Adobe
- Alt-N
- Apache
- Apple
- Atrium
- Avast
- BEA
- BitDefender
- Borland
- CA
- Cisco
- Citrix
- ClamAV
- EMC
- Facebook
- GNU
- Google
- HP
- IBM
- IPSwitch
- ISC
- Kaspersky
- LanDesk
- Lighttpd
- Linux
- Macromedia
- MacroVision
- Mailenable
- McAfee
- Mercury
- Microsoft
- MIT
- Mozilla
- Mplayer
- Multiple vendors
- MySQL
- NOD32
- Novell
- Nullsoft
- OpenLDAP
- OpenOffice
- OPenSSH
- OPenSSL
- Oracle
- Other misc.
- Panda
- RealNetworks
- Samba
- SAP
- Snort
- Sophos
- SpamAssassin
- Squid
- Sun Microsystems
- Symantec
- Trend Micro
- Trillian
- UltraVNC
- Veritas
- VideoLAN
- VMWare
- Winamp
- WinFTP
- Winzip
- Yahoo

3.2.7 Coverage by Result

The following results of exploitation are represented in NSS' exploit test.

3.2.7.1 Arbitrary Code Execution

A software bug that allows an attacker to execute any commands of the attacker's choice on a target machine or in a target process.

3.2.7.2 Buffer Overflow

The exploitation of a software bug due to improperly established memory bounds allows an attacker to overwrite adjacent memory and execute a command.

3.2.7.3 Code Injection

The exploitation of a software bug that allows the processing of invalid data within a program. Code injection can be used by an attacker to introduce code into a computer program and change the course of execution.

3.2.7.4 Cross-Site Script

The exploitation of a web application that enables attackers to insert malicious script into web pages, which can then be executed by other users.

3.2.7.5 Directory Traversal

The exploitation of a lack of security in an application (as opposed to exploiting a bug in the code), which allows user-supplied input with characters representing “traverse to parent directory” to be passed to the file APIs.

The goal of this attack is to order an application to access a file or executable that would not normally be accessible.

3.2.7.6 Privilege Escalation

This exploit type allows an attacker to gain access to resources that would not normally be available.

3.2.7.7 Target Type

The following web target types are represented in NSS’ exploit test:

Web server
Web browser
ActiveX
JavaScript
Browser plug-ins/add-ons

3.3 Evasions

Please refer to the NSS Labs Evasions Test Methodology v1.0.

4 Performance

This section measures the performance of the device using various traffic conditions that provide metrics for real-world performance. Individual implementations will vary based on usage; however, these quantitative metrics provide a gauge as to whether a particular device is appropriate for a given environment.

4.1 Raw Packet Processing Performance (UDP Throughput)

This test uses UDP packets of varying sizes generated by traffic generation appliances. A constant stream of the appropriate packet size—with variable source and destination IP addresses transmitting from a fixed source port to a fixed destination port—is transmitted bi-directionally through each port pair of the device. Each packet contains dummy data, and is targeted at a valid port on a valid IP address on the target subnet. The percentage load and frames per second (fps) figures across each inline port pair are verified by network monitoring tools before each test begins. Multiple tests are run and averages are taken where necessary.

This traffic does not attempt to simulate any form of real-world network condition. No TCP sessions are created during this test, and there is very little for the detection engine to do. However, each vendor will be required to write a signature to detect the test packets to ensure that they are being passed through the detection engine and are not being "fast-tracked" from the inbound port to the outbound port.

The goal of this test is to determine the raw packet processing capability of each inline port pair of the device, as well as its effectiveness at forwarding packets quickly in order to provide the highest level of network performance and with the lowest latency.

4.1.1 64 Byte Packets

Maximum 1,488,000 frames per second per Gigabit of traffic. This test determines the ability of a device to process packets from the wire under the most challenging packet processing conditions.

4.1.2 128 Byte Packets

Maximum 844,000 frames per second per Gigabit of traffic

4.1.3 256 Byte Packets

Maximum 452,000 frames per second per Gigabit of traffic.

4.1.4 512 Byte Packets

Maximum 234,000 frames per second per Gigabit of traffic. This test provides a reasonable indication of the ability of a device to process packets from the wire on an "average" network.

4.1.5 1024 Byte Packets

Maximum 119,000 frames per second per Gigabit of traffic

4.1.6 1514 Byte Packets

Maximum 81,000 frames per second per Gigabit of traffic. This test has been included to demonstrate how easy it is to achieve good results using large packets. Readers should use caution when taking into consideration those test results that quote only performance figures using similar packet sizes.

4.2 Latency

The goal of the latency and user response time tests is to determine the effect the device has on traffic passing through it under various load conditions. Test traffic is passed across the infrastructure switches and through all inline port pairs of the device simultaneously (the latency of the basic infrastructure is known and is constant throughout the tests).

Packet loss and average latency (µs) are recorded for each packet size (64, 128, 256, 512, 1024, and 1514 bytes) at a load level of 90% of the maximum throughput with zero packet loss as previously determined in section 4.1.

4.2.1 64 Byte Frames

Maximum 1,488,000 frames per second per Gigabit of traffic

4.2.2 128 Byte Frames

Maximum 844,000 frames per second per Gigabit of traffic

4.2.3 256 Byte Packets

Maximum 452,000 frames per second per Gigabit of traffic.

4.2.4 512 Byte Packets

Maximum 234,000 frames per second per Gigabit of traffic.

4.2.5 1024 Byte Packets

Maximum 119,000 frames per second per Gigabit of traffic.

4.2.6 1514 Byte Packets

Maximum 81,000 frames per second per Gigabit of traffic.

4.3 Maximum Capacity

The use of traffic generation appliances allows NSS engineers to create true “real-world” traffic at multi-Gigabit speeds as a background load for the tests.

The goal of these tests is to stress the inspection engine and determine how it handles 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 NGFW is causing an unacceptable increase in open connections.
- **Excessive concurrent HTTP connections** – Latency within the NGFW 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 NGFW is causing connections to time out.

4.3.1 Theoretical Maximum Concurrent TCP Connections

This test is designed to determine the maximum concurrent TCP connections of the device with no data passing across the connections. This type of traffic would not typically be found on a normal network, but it provides the means to determine the maximum possible concurrent connections figure.

An increasing number of Layer 4 TCP sessions are opened through the device. Each session is opened normally and then held open for the duration of the test as additional sessions are added up to the maximum possible. Load is increased until no more connections can be established, and this number is recorded.

4.3.2 Maximum TCP Connections per Second

This test is designed to determine the maximum TCP connection rate of the device with one byte of data passing across the connections. This type of traffic would not typically be found on a normal network, but it provides the means to determine the maximum possible TCP connection rate.

An increasing number of new sessions are established through the device, ramped slowly to determine the exact point of failure. Each session is opened normally, one byte of data is passed to the host, and then the session is closed immediately. Load is increased until one or more of the breaking points defined earlier is reached.

4.3.3 Maximum HTTP Connections per Second

This test is designed to determine the maximum TCP connection rate of the device with a 1-byte HTTP response size. The response size defines the number of bytes contained in the body, excluding any bytes associated with the HTTP header. A 1-byte response size is designed to provide a theoretical maximum HTTP connections per second rate.

Client and server are using HTTP 1.0 without keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. This ensures that all TCP connections are closed immediately upon the request being satisfied; and thus any concurrent TCP connections will be caused purely as a result of latency the device introduces on the network. Load is increased until one or more of the breaking points defined earlier is reached.

4.3.4 Maximum HTTP Transactions per Second

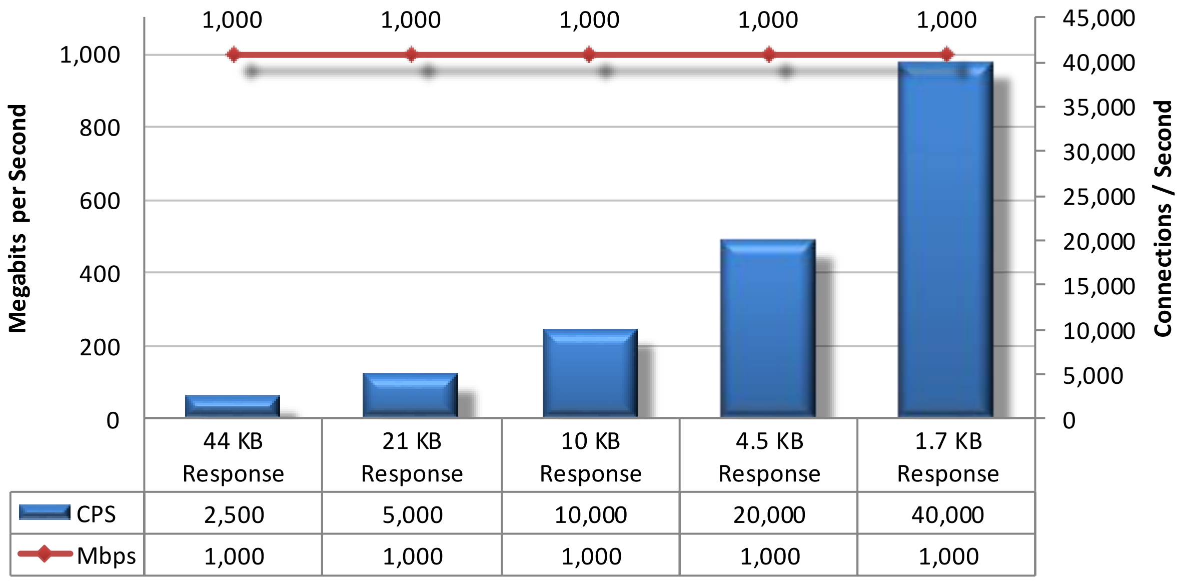
This test is designed to determine the maximum HTTP transaction rate of the device with a 1-byte HTTP response size. The object size defines the number of bytes contained in the body, excluding any bytes associated with the HTTP header. A 1-byte response size is designed to provide a theoretical maximum connections per second rate.

Client and server are using HTTP 1.1 with persistence, and the client will open a TCP connection, send 10 HTTP requests, and close the connection. This ensures that TCP connections remain open until all 10 HTTP transactions are complete, thus eliminating the maximum connection per second rate as a bottleneck (one TCP connection = 10 HTTP transactions). Load is increased until one or more of the breaking points defined earlier is reached.

4.4 HTTP Capacity

The aim of these tests is to stress the HTTP detection engine and determine how the device copes with network loads of varying average packet size and varying connections per second. By creating genuine session-based traffic with varying session lengths, the device is forced to track valid TCP sessions, thus ensuring a higher workload than for simple packet-based background traffic. This provides a test environment that is as close to real-world conditions as it is possible to achieve in a lab environment, while ensuring absolute accuracy and repeatability.

Each transaction consists of a single HTTP GET request ,and there are no transaction delays (i.e., the web server responds immediately to all requests). 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 (albeit one biased towards HTTP traffic) at various network loads.



	44 KB Response	21 KB Response	10 KB Response	4.5 KB Response	1.7 KB Response
CPS	2,500	5,000	10,000	20,000	40,000
Mbps	1,000	1,000	1,000	1,000	1,000

4.4.1 44 KB HTTP Response Size – 2,500 Connections per Second

Maximum 2,500 new connections per second per Gigabit of traffic with a 44 KB HTTP response size—maximum 140,000 packets per second per Gigabit of traffic. With relatively low connection rates and large packet sizes, all hosts should be capable of performing well throughout this test.

4.4.2 21 KB HTTP Response Size – 5,000 Connections per Second

Maximum 5,000 new connections per second per Gigabit of traffic with a 21 KB HTTP response size—maximum 185,000 packets per second per Gigabit of traffic. With average connection rates and average packet sizes, this is a good approximation of a real-world production network, and all hosts should be capable of performing well throughout this test.

4.4.3 10 KB HTTP Response Size – 10,000 Connections per Second

Maximum 10,000 new connections per second per Gigabit of traffic with a 10 KB HTTP response size—maximum 225,000 packets per second per Gigabit of traffic. With smaller packet sizes coupled with high connection rates, this represents a very heavily used production network.

4.4.4 4.5 KB HTTP Response Size – 20,000 Connections per Second

Maximum 20,000 new connections per second per Gigabit of traffic with a 4.5 KB HTTP response size—maximum 300,000 packets per second per Gigabit of traffic. With small packet sizes and extremely high connection rates, this is an extreme test for any host.

4.4.5 1.7 KB HTTP Response Size – 40,000 Connections per Second

Maximum 40,000 new connections per second per Gigabit of traffic with a 1.7 KB HTTP response size—maximum 445,000 packets per second per Gigabit of traffic. With small packet sizes and extremely high connection rates, this is an extreme test for any host.

4.5 Application Average Response Time: HTTP

Test traffic is passed across the infrastructure switches and through all inline port pairs of the device simultaneously (the latency of the basic infrastructure is known and is constant throughout the tests). The results are recorded at each response size (44 KB, 21 KB, 10 KB, 4.5 KB, and 1.7 KB HTTP responses) load level of 90% of the maximum throughput with zero packet loss as previously determined in section 4.4, section 4.6, and section 4.8.

4.6 HTTP Capacity with HTTP Persistent Connections

The aim of these tests is to determine how the device copes with network loads of varying average packet size, varying connections per second while inspecting all traffic. By creating genuine session-based traffic with varying session lengths, the device is forced to track valid TCP sessions, thus ensuring a higher workload than for simple packet-based background traffic. This provides a test environment that is as close to real-world conditions as it is possible to achieve in a lab environment, while ensuring absolute accuracy and repeatability.

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.

4.6.1 250 Connections per Second

This test will simulate HTTP persistent connections, each containing a total of 10 HTTP GET/responses of various sizes. The total HTTP response size for each persistent connection will be equal to four megabits, transmitted over a maximum of 250 connections per second for each gigabit of traffic.

4.6.2 500 Connections per Second

This test will simulate HTTP persistent connections, each containing a total of HTTP 10 GET/responses of various sizes. The total HTTP response size for each persistent connection will be equal to two megabits, transmitted over a maximum of 500 connections per second for each gigabit of traffic.

4.6.3 1000 Connections per Second

This test will simulate HTTP persistent connections, each containing a total of 10 HTTP GETs/responses of various sizes. The total HTTP response size for each persistent connection will be equal to one megabit, transmitted over a maximum of 1000 connections per second, for each gigabit of traffic.

4.7 SSL/TLS Performance

See the NSS Labs SSL/TLS Performance Test Methodology v1.2.

4.8 "Real-World" Single Application Flows

Where previous tests provide a pure HTTP environment with varying connection rates and average packet sizes, the goal of this test is to simulate real-world single application traffic.

4.8.1 Single Application SIP Flow

4.8.2 Single Application FIX Flow

4.8.3 Single Application SMTP Flow

4.8.4 Single Application FTP Flow

4.8.5 Single Application SMB Flow

4.8.6 Single Application RDP Flow

4.8.7 Single Application YouTube Flow

4.8.8 Single Application WebEx Flow

4.8.9 Single Application MSSQL Flow

5 Stability and Reliability

Long-term stability is particularly important for an inline device, where failure can produce network outages. These tests verify the stability of the device along with its ability to maintain security effectiveness while under normal load and while passing malicious traffic. Products that are not able to sustain legitimate traffic (or that crash) while under hostile attack will not pass.

The device is required to remain operational and stable throughout these tests, and to block 100% of previously blocked traffic, raising an alert for each. If any prohibited traffic passes successfully, caused by either the volume of traffic or by the device failing open for any reason, this will result in a FAIL.

5.1 Blocking Under Extended Attack

The device is exposed to a constant stream of security policy violations over an extended period of time. The device is configured to block and alert, and thus this test provides an indication of the effectiveness of both the blocking and alert handling mechanisms.

A continuous stream of security policy violations mixed with legitimate traffic is transmitted through the device for eight hours at a maximum of 100 Mbps, with no additional background traffic. This is not intended as a stress test in terms of traffic load (covered in the previous section); it is merely a reliability test in terms of consistency of blocking performance.

The device is expected to remain operational and stable throughout this test and to block 100% of recognizable violations, raising an alert for each. If any recognizable policy violations are passed, caused by either the volume of traffic or the device failing open for any reason, this will result in a FAIL.

5.2 Passing Legitimate Traffic under Extended Attack

This test is identical to test 5.1 where the external interface of the device is exposed to a constant stream of exploits over an extended period of time.

The device is expected to remain operational and stable throughout this test, and to pass most/all of the legitimate traffic. If an excessive amount of legitimate traffic is blocked throughout this test, caused by either the volume of traffic or the DUT failing for any reason, this will result in a FAIL.

5.3 Behavior of the State Engine Under Load

This test determines whether the device is capable of preserving state across a large number of open connections over an extended time period.

At various points throughout the test (including after the maximum has been reached), it is confirmed that the device is still capable of inspecting and blocking traffic that is in violation of the currently applied security policy, whilst confirming that legitimate traffic is not blocked (perhaps as a result of exhaustion of the resources allocated to state tables). The device must be able to apply policy decisions effectively based on inspected traffic at all load levels.

5.3.1 Attack Detection/Blocking – Normal Load

This test determines wheher the device is able to detect and block policy violations as the number of open sessions reaches 75% of the maximum determined in Test 4.3.1.

5.3.2 State Preservation – Normal Load

This test determines if the sensor maintains the state of pre-existing sessions as the number of open sessions reaches 75% of the maximum determined in Test 4.3.1.

A legitimate HTTP session is opened, and the first packet of a two-packet exploit is transmitted. As the number of open connections approaches the maximum, the initial HTTP session is then completed with the second half of the exploit, and the session is closed. If the sensor is still maintaining state on the original session, the exploit will be recorded. If the state tables have been exhausted, the exploit string will be seen as a non-stateful attack and will thus be ignored. Both halves of the exploit are required to trigger an alert. A product will fail the test if it fails to generate an alert after the second packet is transmitted, or if it raises an alert on either half of the exploit on its own.

5.3.3 Pass Legitimate Traffic – Normal Load

This test ensures that the device continues to pass legitimate traffic as the number of open sessions reaches 75% of the maximum determined in Test 4.3.1.

5.3.4 State Preservation – Maximum Exceeded

This test determines whether the device maintains the state of pre-existing sessions as the number of open sessions exceeds the maximum determined in Test 4.3.1. The method of execution is identical to Test 5.3.2.

5.3.5 Drop Legitimate Traffic – Maximum Exceeded

This test ensures that the device continues to drop all traffic as the number of open sessions exceeds the maximum determined in Test 4.3.1.

Note: If a device allows traffic to “leak” due to the way it expires old connections, this will result in an automatic fail for the entire test.

5.4 Power Fail

Power to the device is cut whilst passing a mixture of legitimate and disallowed traffic. Firewalls should always be configured to fail closed—no traffic should be passed once power has been cut.

5.5 Backup/Restore

Backing up and restoring a device’s configuration is a critical part of deploying any managed device within a live network. It should be possible to export configurations and store them offline for backup purposes. Additionally, it should be possible to completely reconfigure the device using the offline configuration file(s). This includes restoring all policies and interface information in order to deploy a device.

5.6 Persistence of Data

The device should retain all configuration data, policy data, and locally logged data once it has been restored to operation following power failure.

6 Total Cost of Ownership and Value

Implementation of 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 product:

- **Product Purchase** – The cost of acquisition
- **Product Maintenance** – The fees paid to the vendor, including software and hardware support, maintenance, and other updates
- **Installation** – The time required to take the device 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 vendors, including hardware, software, and other updates

Appendix A: Change Log

Version 8.0 – 13 September, 2017

- Section 1.3: Inclusion Criteria: Added wording regarding 10 Gbps max limit throughput
- Removed Section 3.1.6: Syn Flood Protection
- Removed Section 3.1.7: IP Address Spoofing
- Removed references to SSL/TLS testing and inserted Section 4.7: SSL/TLS Performance
- Added Section 3.3: Evasions
- Removed Section 4.6: HTTP Capacity with Transaction Delays
- Removed Section 3.1.3: Complex Policies
- Removed Sections 3.1.4 and 3.1.5 (NAT testing)
- Removed Section 3.4.8: FTP/Telnet Evasion
- Changed Section 4.9 "Real-World Traffic Mixes to read as "Real-World" Single Application Flows
- Added Section 5.5: Backup/Restore

Version 7.0 – 15 July, 2016

- Added requirements for false positive testing to section 3.3.
- Added the Veil Evasion Framework to section 3.4.6
- Added sections 4.7, "HTTP Capacity with HTTP Persistent Connections," and 4.8 "HTTPS Capacity with HTTP Persistent Connections."
- In section 4.7, noted that SSL/TLS inspection will only be required for this particular section in NGFW v7.0. For the next methodology release, inspection will be required for all NSS NGFW testing.
- Changed all occurrences of "Anti-X" to "Anti-malware."
- Added application protocol percentages to all real-world tests in section 4.9
- Removed Datacenter and Education traffic mixes from the real-world testing in section 4.9. Added Internal Segmentation traffic mix.
- Added section 4.10 for IPSec functionality testing, which includes language around IKEv1 or IKEv2.

Version 6.0 – 24 February, 2015

- Removed IPv6 section
- Removed sections referencing centralized management
- Removed sections referencing Active Directory integration and NGFW DB
- Clarified section 3.3
- Clarified section 3.4
- Moved information regarding Live Exploit Testing from section 1.2 to section 3.4
- Added Arbitrary Code Execution to list of fail criteria in section 3.4.1
- Removed sections referencing SMB & NetBIOS evasions
- Clarified section 4.3
- Removed references to average packet size in sub-sections of 4.4 and 4.6
- Removed section referencing Redundancy
- Removed section referencing High Availability
- Adjusted contact information and legal information

Version 5.4 – 30 September 2013

- Capitalized SYN in subsection title
- Removed SSL section
- Clarified section 3.4.4
- Removed references to specific test tools

Version 5.3 – 04 December 2012

No Change Log available. Change Log Appendix added with version 5.4.

Contact Information

NSS Labs, Inc.
3711 South MoPac Expressway
Building 1, Suite 400
Austin, TX 78746-8022
USA
info@nsslabs.com
www.nsslabs.com