TEST METHODOLOGY

Next Generation Intrusion Prevention System (NGIPS)

FEBRUARY 2, 2018

V4.0

Table of Contents

1 Introduction

1.1 The Need for Next Generation Intrusion Prevention Systems

Introduced over a decade ago, the first network intrusion prevention systems (IPS) were built on generic Intel servers with the purpose of blocking exploits that target vulnerable servers. Soon after, attacks against desktop clients emerged and the first generation of intrusion prevention struggled to maintain performance and security. This led to a new hardware-accelerated generation of IPS that could inspect much more traffic and at higher speeds than software-only solutions.

Over the past few years, several trends have emerged, each posing challenges for intrusion prevention: social media, remote workers, wireless, bring your own device (BYOD), and the explosion of business/personal web applications all have led to the near-disintegration of the network perimeter. Simultaneously, cybercriminals have grown more aggressive, increasingly targeting corporate assets including clients, browsers, and plug-ins. The growing number of vulnerability disclosures in widely deployed operating systems and applications is a multi-faceted problem. Therefore, a new generation of intrusion prevention is required to meet the challenges of organizations without clearly defined perimeters.

These next generation intrusion prevention systems (NGIPS) must provide organizations with the ability to identify both the applications and the users on their internal networks. As with their predecessors, NGIPS must protect the enterprise user against threats/exploits. Designed to identify and block attacks against internal computing assets, a good NGIPS can provide temporary protection and relief from the immediate need to patch affected systems. The NGIPS must catch sophisticated attacks while producing as few false positives as possible.

Since the role of an NGIPS is to protect users and provide granular visibility into network traffic, the NGIPS should be considered a perimeter device and should be deployed behind a firewall to supplement the overall security of the enterprise.

1.2 About This Test Methodology

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

- Stateful, multi-layer inspection
- Detection of protocol violations
- Application and user controls
- Performance
- Stability and reliability
- Total cost of ownership (TCO)

NGIPS products are deployed at critical points in the network, and their stability and reliability is imperative. Regardless of any technical features, an NGIPS must be as stable, reliable, fast, and flexible as the infrastructure it protects, and it must improve the security posture of an enterprise. It should also be possible to incorporate an NGIPS into an existing security architecture without requiring a network redesign.

The following capabilities are considered essential in an NGIPS:

- Intrusion prevention
- Application identification

- Resistance to known evasion techniques
- Reputation awareness
- Highly resilient and stable
- Operation at Layer 2 (network transparency)

NSS Labs test methodologies are continually evolving in response to feedback. If you would like to provide input, please contact advisors@nsslabs.com. For a list of changes, please reference the Change Log in the Appendix.

1.3 Inclusion Criteria

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

The NGIPS should be implemented as an inline device operating at Layer 2; that is, it should be transparent so that it can be added to an existing network infrastructure without altering the network. The NGIPS should be supplied as a single appliance where possible. Cluster controller solutions are also acceptable.

The NGIPS device should be equipped with an appropriate number of physical interfaces, with a minimum of one inline Gigabit Ethernet (GbE) port pair per gigabit of throughput or one inline 10GbE port pair per 10 Gbps of throughput. For example, an 8-Gbps device with only four gigabit port pairs will be limited to 4 Gbps. The minimum number of port pairs will be connected to support the claimed maximum bandwidth of the device. For example, an 8-Gbps device with ten port pairs will be tested with eight 1-GbE connections.

Once installed in the test lab, the device will be configured for its most common use case: a corporate network segment. The device will 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 NGIPS 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 NGIPS 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

Vendors are permitted—and encouraged—to assist NSS engineers in tuning their devices to eliminate false positives. Some vendors provide customers with factory-tuned policies instead of manually tuning their products. These policies usually contain all the signatures that can be safely deployed without triggering false positives. This approach allows customers to quickly deploy a more aggressive security policy with relatively little effort or risk and will be reflected positively in the time and cost sections of the test.

Once testing begins, the product version and configuration will be frozen to preserve the integrity of the test.

3.1 Exploits 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 as appropriate.[1] 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 a device.

3.2 False Positive Testing

The ability of the device to identify and allow legitimate traffic is of equal importance to providing protection against malicious content. This test will include a varied sample of legitimate application traffic that should properly be identified and allowed.

After completion of the false positive testing, and prior to the mitigation testing, signatures that were deemed to cause the false positive alerts will be disabled within the security policy.

3.3 Coverage by Attack Vector

Threats and exploits can be initiated either by the target or by the attacker, targeting either local or remote vulnerabilities.

[1] For more information on the NSS "Live Testing™" harness and methodology, please refer to the latest *Security Stack Test Methodology* on the NSS Labs website.

3.3.1 Attacker-Initiated

The threat/exploit is remotely executed by the attacker against a vulnerable application and/or operating system.

3.3.2 Target-Initiated

The threat/exploit is initiated by the vulnerable target. The attacker has little or no control over when the target user or application will execute the threat.

3.3.3 Network

The threat/exploit is initiated as a result of network communication.

3.3.4 Local

The threat/exploit permits local execution that requires access to the target host.

3.3.5 Coverage by Impact Type

The NSS threat and attack suite contains thousands of publically 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). Based on the impact of the threat against the target, the following metrics are reported:

3.3.5.1 System Exposure

These are attacks resulting in an individual service compromise but not arbitrary system-level command execution. Typical attacks in this category include service-specific attacks, such as SQL injection, which enable the attacker to execute arbitrary SQL commands within the database service. These attacks are somewhat isolated to the service and do not 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 go from the service level to the system level*.*

3.3.5.2 System or Service Fault

These are attacks resulting in system-level or service-level faults that crash targeted services or applications and require administrative action to restart the services or reboot the systems. These attacks do not enable attackers to execute arbitrary commands; however, the resulting impact to businesses could be severe given that the attackers could crash protected systems or services.

3.3.6 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, and extending to the year of the current NGIPS 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, and not 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.

3.3.7 Coverage by Vendor

Exploits will target both client and server-side applications and services. NSS' exploit test contains many vendors, including but not limited to:

- 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

3.3.8 Coverage by Result

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

3.3.8.1 Arbitrary Code Execution

This describes the exploitation of 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.

3.3.8.2 Buffer Overflow

This describes the exploitation of a software bug due to improperly established memory bounds, which allows an attacker to overwrite adjacent memory and execute a command.

3.3.8.3 Code Injection

This describes the exploitation of a software bug that allows for invalid data to be processed within a program. An attacker can use code injection to introduce code into a computer program to change the course of execution.

3.3.8.4 Cross-Site Script

This describes the exploitation of a web application that enables attackers to inject a malicious script into web pages, which can then be executed by other users.

3.3.8.5 Directory Traversal

This describes the exploitation of a lack of security in an application (as opposed to exploiting a bug in the code) that allows user-supplied input with characters representing “traverse to parent directory” to be passed through to the file APIs. The goal of this attack is to order an application to access a file or executable that is not intended to be accessible.

3.3.8.6 Privilege Escalation

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

3.3.9 Target Type

The following web target types are represented in the NSS live exploit test. This list is not intended to be all-inclusive, and additional target types may be added or removed.

- Browser plug-ins/add-ons
- ActiveX
- .NET
- Web browser
- JavaScript

3.4 Evasions

Please refer to the most current version of the NSS Labs Evasions Test Methodology.

4 Performance

This section measures the performance of a 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 device is appropriate for a given environment.

Security devices often exhibit an inverse correlation between security effectiveness and performance. An increased level of deep packet inspection results in an increase in latency, which is the amount of time a device requires to forward packets.

Furthermore, it is important to consider the real-world mix of traffic that a device will encounter. NSS utilizes a range of traffic types and mixes.

Performance and security effectiveness testing are always measured with both the vendor-recommended policy and the tuned policy. Once testing begins, the product version and configuration will be frozen to preserve the integrity of the test.

4.1 Raw Packet Processing Performance (UDP Traffic)

This test uses UDP packets of specific sizes, generated by a traffic generation tool, to validate the packet performance of a device. A constant stream of packets of a specified size, with varying source and destination IP addresses, transmitting from varying source ports to a fixed destination port, is transmitted bi-directionally through each tested 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. Vendors will be expected to create an ad hoc signature to detect random payload data prior to running the first test to demonstrate that deep packet inspection is occurring on this traffic. Multiple tests are run and averages taken where necessary.

This traffic does not attempt to simulate any form of real-world network use case. No TCP sessions are created during this test, and there is very little for the detection engine to do. The goal of this test is to determine the raw packet processing capability of each inline port pair of the device and its effectiveness at forwarding packets quickly, in order to provide the highest level of network performance and lowest latency.

The raw packet performance for each packet size is recorded at 95% of the maximum throughput with zero packet loss.

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 volume 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 assesses a device’s maximum achievable bit rate, with a minimum of influence by its packet forwarding capacity.

4.1.7 4096 Byte Packets (Jumbo Frames)

This demonstrates the ability of the device to forward and process “jumbo” frames.

4.1.8 9000 Byte Packets (Jumbo Frames)

This demonstrates the ability of the device to forward and process “jumbo” frames.

4.2 Latency

The purpose of this test is to determine the amount of time it takes for network traffic to pass through a device under a range of load conditions.

The average latency (s) is recorded for each specified packet size at a load level of 95% 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

4.2.7 4096 Byte Packets (Jumbo Frames)

Maximum 3,100 frames per second per gigabit of traffic

4.2.8 9000 Byte Packets (Jumbo Frames)

Maximum 1,388 frames per second per gigabit of traffic

4.3 Maximum Capacity

The purpose 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.

The following behaviors are used as failure criteria for each of these tests:

- Excessive concurrent TCP connections
- Excessive response time for HTTP transactions
- Unsuccessful HTTP transactions

4.3.1 Theoretical Maximum Concurrent TCP Connections

This test is designed to determine the maximum concurrent TCP connection capacity of the device, with no data passing across the connections. This type of traffic would not be typical of a normal network, but it provides a means to assess a device's connection capacity, unencumbered by traffic payloads.

An increasing number of TCP sessions are opened through the device. Each session is established and then held open for the duration of the test as additional sessions are added, up to the maximum possible. The test traffic load is increased until no further connections can be established. The maximum number of established connections is recorded.

4.3.2 Theoretical Maximum Concurrent TCP Connections with Data

This test is identical to the test in section 4.3.1, but with the addition of 21 KB of data, which is transmitted in 1 KB segments during the session. This ensures that the device is capable of passing data between the connections once they have been established.

4.3.3 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 is atypical of a normal network, but it provides a means to determine the maximum possible TCP connection rate of the device.

An increasing number of new sessions are established through the device and 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. The test traffic load is increased until one or more of the points of failure defined in section 4.3 is reached.

4.3.4 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.

The test uses HTTP 1.0 without persistent connections (HTTP keep-alive). The client opens a TCP connection, sends one HTTP request, and once the request is fulfilled, promptly closes the connection. This ensures that any concurrent TCP connections that occur are a result of the latency induced by the device. The test traffic load is increased until one or more of the points of failure defined in section 4.3 is reached.

4.3.5 Maximum HTTP Transactions per Second

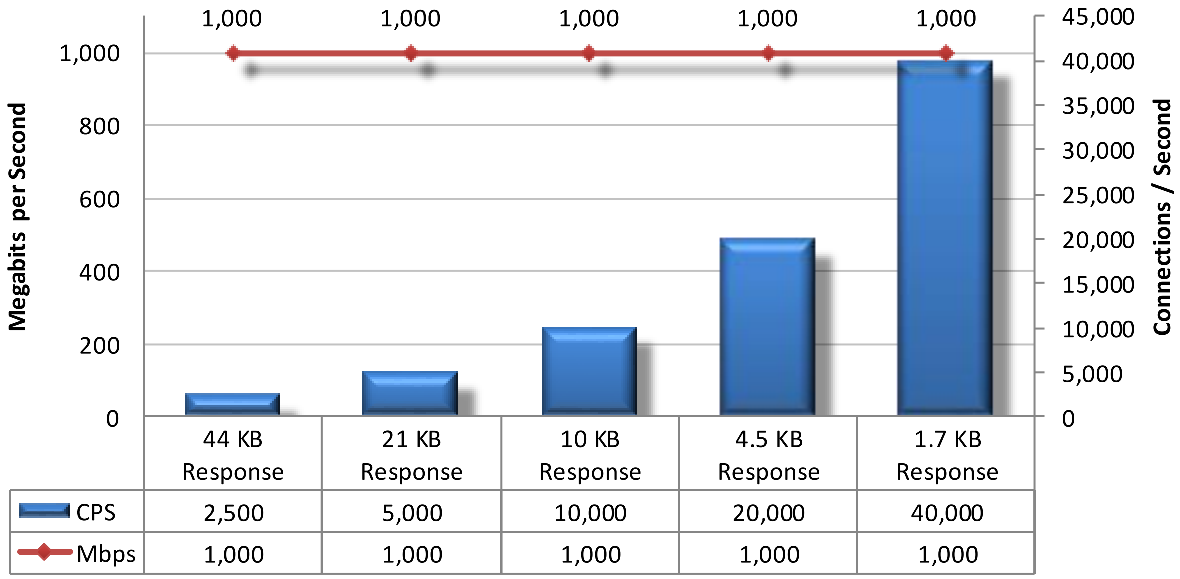
This test is designed to determine the maximum HTTP transaction rate of the device with a 1-byte HTTP response size. The object size defines the number of bytes contained in the body, excluding any bytes associated with the HTTP header. A 1-byte response size is designed to provide a theoretical maximum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

Client and server are using HTTP 1.1 with keep-alive, 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 (1 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 resulting in a higher workload than for simple packet-based background traffic. This provides a test environment that simulates real-world HTTP transactions in the lab 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.



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 – average packet size 900 bytes – maximum 140,000 packets per second per gigabit of traffic. With relatively low connection rates and large packet sizes, all devices 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 – average packet size 670 bytes – 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 devices 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 – average packet size 550 bytes – 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 – average packet size 420 bytes – 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 device.

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 – average packet size 270 bytes – 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 device.

4.5 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.

4.5.1 250 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.

4.5.2 500 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.

4.5.3 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.

4.6 "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 a real-world single application traffic.

4.6.1 Single Application SIP flow

4.6.2 Single Application FIX flow

4.6.3 Single Application SMTP flow

4.6.4 Single Application FTP flow

4.6.5 Single Application SMB flow

4.6.6 Single Application RDP flow

4.6.7 Single Application YouTube flow

4.6.8 Single Application WebEx flow

4.6.9 Single Application MSSQL flow

5 Stability and Reliability

Long-term stability is particularly important for an inline device, where failures can result in network outages. In addition, a device that cannot be managed or monitored can be a security risk. These tests assess the ability of a device to maintain security effectiveness while under normal and excessive utilization and while managing malicious traffic.

The failure criteria for this category of tests are as follows:

- Device is unable to pass legitimate traffic while under attack
- Device allows prohibited, malicious traffic to pass through
- Device becomes unreachable or unusable during or after the test interval

The device is required to remain operational, manageable, and capable of passing non-malicious traffic throughout these tests, and it is required to continue managing all previously identified malicious traffic.

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 security event management and alert handling mechanisms.

A continuous stream of security policy violations mixed with legitimate traffic is transmitted through the device at a rate not to exceed 80% of the device's stated capacity for eight hours with a steady stream of legitimate traffic mixed in. This test is not intended as a stress test in terms of traffic load. It is merely a reliability test in terms of consistency of its blocking performance. Any leak of exploit traffic will be considered a fail for this test.

5.2 Passing Legitimate Traffic under Extended Attack

A continuous stream of legitimate traffic is transmitted through the device at a rate not to exceed 80% of the device's stated capacity for eight hours with a steady stream of exploits mixed in. This test is not intended as a stress test in terms of traffic load. It is merely a reliability test in terms of consistency of passing legitimate traffic.

The device is expected to remain operational and stable throughout this test and have no failure to pass legitimate traffic. The connection rate at the point of failure will be recorded.

5.3 Power Fail

Devices will be tested for both fail open and fail closed cases. In all power-fail scenarios, the device will return to a fully functional operating condition without need for any manual intervention.

Power to the device is removed while passing a mixture of legitimate and disallowed traffic. The device may fail closed by default (link is down, no traffic passes). The device will not begin forwarding traffic again until it is again fully operational and is inspecting traffic.

The device may optionally contain additional hardware that will allow traffic to continue to pass uninspected during a power failure. Where this feature is available, it will be customer-configurable. In this scenario, the device will continue to forward traffic but without inspecting it. Inspection will resume once the device is fully operational.

5.4 Power Redundancy

The device will have at least two independent power supplies and will operate normally when only one of the power supplies is energized.

5.5 Persistence of Data

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

6 Total Cost of Ownership and Value

Organizations should be concerned with the ongoing, amortized cost of operating security products. This section evaluates the costs associated with the purchase, installation, and ongoing management of the device, including:

- **Product Purchase** – The cost of acquisition
- **Product Maintenance** – The fees paid to the vendor (including software and hardware support, maintenance, and updates)
- **Installation** – The time required to take the device out of the box, configure it, install it in the network, apply updates and patches, perform initial tuning, and set up desired logging and reporting
- **Upkeep** – The time required to apply periodic updates and patches from vendors, including hardware, software, and firmware updates

Appendix A: Change Log

Version 3.9 DRAFT – 29 January, 2018

- Removed Behavior of the State Engine Under Load
- Removed section Application ID
- Removed Live Drive-by
- Removed section HTTP Transactions with Delay
- Renamed section to HTTP Transactions (removed no delay)
- Removed Protocol Fuzzing
- Removed Drive by Exploits
- Removed Application ID
- Added Section 2 – Product Guidance
- Expanded Coverage Attack by Vendor (3.3.7)
- Added additional Coverage types (3.3.x)
- Removed Evasions and referenced external document (3.6)
- Performance – Added UDP Jumbo frame tests (4.1.7, 4.1.8)
- Removed “Theoretical Maximum Throughput on single Connection”
- Removed “Real World Protocol Mix” tests
- Added “HTTP Capacity with HTTP Persistent Connections” (4.7)
- Added “Real-World" Single Application Flows” (4.8)
- Expanded “Behavior of the State Engine under Load” (5.3)
    - Attack Detection/Blocking – Normal Load
    - State Preservation – Normal Load
    - Pass Legitimate Traffic – Normal Load
    - State Preservation – Maximum Exceeded
    - Drop Legitimate Traffic – Maximum Exceeded
- Clarified details of “Power Fail” (5.5)
- Added “Power Redundancy” (5.6)
- Clarified “Persistence of Data” (5.7)

Version 3.1 – 29 July, 2017

- Added Section 2.4.11: HTTP Evasions

Version 3.0 – March 2017

- Minor edits for clarity
- Updated Connections/Second table
- Section 1.2 – Added “Application and User Controls”
- Section 1.3 – Most common use case: a corporate network segment.
- Section 4.5 – Changed to “fail open”

Version 2.0 – September 2015

Version 1.0 – March 7, 2014

Contact Information

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

This and other related documents available at: **www.nsslabs.com**. To receive a licensed copy or report misuse, please contact NSS Labs.

Please read the disclaimer in this box because it contains important information that binds you. If you do not agree to these conditions, you should not read the rest of this report but should instead return the report immediately to us. "You" or "your" means the person who accesses this report and any entity on whose behalf he/she has obtained this report.

1. The information in this report is subject to change by us without notice, and we disclaim any obligation to update it.

2. The information in this report is believed by us to be accurate and reliable at the time of publication, but is not guaranteed. All use of and reliance on this report are at your sole risk. We are not liable or responsible for any damages, losses, or expenses of any nature whatsoever arising from any error or omission in this report.

3. NO WARRANTIES, EXPRESS OR IMPLIED ARE GIVEN BY US. ALL IMPLIED WARRANTIES, INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT, ARE HEREBY DISCLAIMED AND EXCLUDED BY US. IN NO EVENT SHALL WE BE LIABLE FOR ANY DIRECT, CONSEQUENTIAL, INCIDENTAL, PUNITIVE, EXEMPLARY, OR INDIRECT DAMAGES, OR FOR ANY LOSS OF PROFIT, REVENUE, DATA, COMPUTER PROGRAMS, OR OTHER ASSETS, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

4. This report does not constitute an endorsement, recommendation, or guarantee of any of the products (hardware or software) tested or the hardware and/or software used in testing the products. The testing does not guarantee that there are no errors or defects in the products or that the products will meet your expectations, requirements, needs, or specifications, or that they will operate without interruption.

5. This report does not imply any endorsement, sponsorship, affiliation, or verification by or with any organizations mentioned in this report.

6. All trademarks, service marks, and trade names used in this report are the trademarks, service marks, and trade names of their respective owners.