

Security Content Appliance Testing Procedure (V1.0)

The aim of this procedure is to provide a thorough test of all the main components of an in-line *Security Content Appliance* (SCA) device in a controlled and repeatable manner and in the most “real world” environment that can be simulated in a test lab.

For the purposes of this test, a SCA device is defined as a single appliance combining the following possible functions:

1. *Anti Malware*
2. *Anti Spam*
3. *Content Filtering*
4. *Web Filtering*

The tests are designed to determine the suitability of a particular SCA product for use as a basic, Secure Content Management gateway security device and will focus on the effects of combining multiple security technologies (as listed above) in a single appliance. Thus, the overall focus of the tests will be on the manageability, performance and capabilities of the appliance as transparent bridge, and how the performance is affected by enabling/disabling the additional security functions.

Note that, whereas we will confirm the basic capabilities of each individual function - i.e. that an AV engine is capable of detecting a blocking virus-infected Web and e-mail traffic - we do not intend as part of this test to perform tests for complete virus signature coverage, complete spam filter coverage, and so on.

We will, of course, be using a selection of AV infected traffic, spam traffic, etc. to ensure that the device is capable of processing such traffic effectively, especially when under load. However, we do not intend to ensure, for example, that the AV engine detects every single virus on the WildList. There are already third-party certification schemes used by all AV vendors which cover this type of coverage testing.

The Test Environment

The network is 100/1000Mbit Ethernet with CAT 5e cabling and multiple Cisco Catalyst 6500-Series switches (these have a mix of fibre and copper Gigabit interfaces). All devices are expected to be provided as appliances. The Device Under Test (DUT) is normally configured as a perimeter device during testing and will be installed in routing mode (i.e. functioning as the main Internet gateway/firewall). Alternatively, it may be configured as a transparent bridge if this is the **only** mode supported by the device.

Machines generating exploits and Spirent Smartbits *transmit* port are connected to the “external” network, whilst the “target” hosts for the malicious traffic and the Spirent Smartbits *receive* port are connected to the internal network - this allows simulation of malicious inbound traffic.

The Spirent Avalanche clients are connected to the internal network, whilst the Reflector servers are connected to the external network - this allows simulation of large numbers of internal clients performing outbound browsing operations. The DUT is connected between two switches - one at the edge of the external network connected to the gateway router, and one connected directly to the internal network.

All “normal” network traffic, background load traffic and exploit traffic will therefore be transmitted **through** the DUT. The same traffic is mirrored to a single SPAN port of the external gateway switch, to which an Adtech network monitoring device is connected. The Adtech AX/4000 monitors the same mirrored traffic to monitor and confirm the composition and amount of both background and malicious traffic.

The management interface is used to connect the DUT to the management console on a private subnet. This ensures that the DUT and console can communicate even when the target subnet is subjected to heavy loads, in addition to preventing attacks on the console itself.

The Tests

The aim of this section is to determine the baseline performance of the device as a transparent security gateway with no modules enabled (if possible).

Once this has been achieved, the remainder of the security modules will be activated one by one. The performance comparison tests will be re-run as each module is activated in order to determine the effect of enabling additional modules on the baseline performance. Additional module-specific tests will be run for each new security module.

Each module is disabled once tests have been run, ensuring that we are measuring the effects of each individual module. A cumulative performance test with **all** modules enabled is run as the final test.

Where a vendor does not offer a security module for the required test suite (i.e. not everyone provides anti-spam or anti-virus) those results will be marked as **N/A** in the report or will be omitted altogether.

No judgement will be offered by NSS as to which modules are essential or not - products will not be “marked down”, for example, if they do not provide AV or anti-spam. It will be up to the individual reader to determine which of the features offered by each vendor are essential or superfluous to their requirements.

Section 1 - Baseline

Where possible, all modules will be disabled in order to determine the “baseline” performance of the product as a transparent packet forwarding device with no security in place.

Test 1.1 - Performance Comparison - Baseline

All security modules are disabled - all traffic is passed uninspected. *Note that in all tests, the following critical “breaking points” - where the final measurements are taken - are used:*

- **Concurrent TCP connections exceeds 200** - latency within the DUT is causing unacceptable increase in open connections on the server-side
- **Current response time for HTTP transactions/SMTP sessions exceeds 100 ms** - latency within the DUT is causing excessive delays and increased response time to client
- **Unsuccessful HTTP transactions/SMTP sessions** - normally there should be zero unsuccessful transactions. Once these appear, it is an indication that excessive latency within the DUT is causing connections to time out

Test ID	Test Description
1.1.1	<p>Maximum TCP Connections Per Second</p> <p><i>This test is designed to determine the maximum TCP connection rate of the DUT with a 1024 byte object size (Avalanche load specification is Connections Per Second). The object size defines the number of bytes contained in the body, excluding any bytes associated with the HTTP header.</i></p> <p><i>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 the request is satisfied, thus any concurrent TCP connections will be caused purely as a result of latency within the DUT. Load is increased until one or more of the defined breaking points is reached.</i></p>
1.1.2	<p>Maximum HTTP Transactions Per Second</p> <p><i>This test is designed to determine the maximum HTTP transaction rate of the DUT with a 1024 byte object size (Avalanche load specification is Transactions Per Second).</i></p> <p><i>Client and server are using HTTP 1.1 with persistence, and the client will open a TCP connection, send ten HTTP requests, and close the connection. This ensures that TCP connections remain open until all ten 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 defined breaking points is reached.</i></p>
1.1.3	<p>Maximum Concurrent TCP Connections</p> <p><i>This test is designed to determine the maximum concurrent TCP connections of the DUT with a 1024 byte object size (Avalanche load specification is Connections).</i></p> <p><i>Client and server are using HTTP 1.0 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 ten HTTP transactions are complete, and a “user think time” of 30 seconds between every HTTP Transaction ensures a high number of concurrent connections. Load is increased until one or more of the defined breaking points is reached (the concurrent TCP connections breaking point does not apply to this test)</i></p>
1.1.4	<p>Maximum Bandwidth for HTTP traffic</p> <p><i>This test is designed to determine the maximum bandwidth of the DUT (Avalanche load specification is Transactions Per Second). A 100Kbyte object size is used and the server breaks the response into 650 byte packets, giving an average packet size of around 550 bytes throughout the test.</i></p> <p><i>Client and server are using HTTP 1.1 with persistence, and the client will open a TCP connection, send ten HTTP requests, and close the connection. This ensures that TCP connections remain open until all ten HTTP transactions are complete, and the large response size ensures that the bandwidth limit of the DUT is reached before connections per second or transactions per second become an issue. Load is increased until the first unsuccessful HTTP transaction is recorded.</i></p>
1.1.5	<p>Maximum SMTP Sessions Per Second</p> <p><i>This test is designed to determine the maximum SMTP session rate of the DUT with a 1024 byte mail object size (Avalanche load specification is Connections Per Second). The object size defines the number of bytes contained in the mail body, excluding any bytes associated with the SMTP header.</i></p> <p><i>The client will initiate an SMTP session, transmit a single message, and close the connection. This ensures that all TCP connections are closed immediately the request is satisfied, thus any concurrent TCP connections will be caused purely as a result of latency within the DUT. Load is increased until one or more of the defined breaking points is reached.</i></p> <p><i>Note that due to the way Avalanche reports results in these particular tests, these SMTP performance figures are only intended to be used as a baseline/comparison between different security modules, and do not provide an accurate indication of the number of messages per second which could be achieved in real-world deployments.</i></p>

Test 1.2 - Baseline Throughput

Spirent SmartFlow is typically used to test network devices such as switches and firewalls, and is designed to measure throughput and latency using UDP packet flows.

These tests determine the maximum throughput of the DUT with zero packet loss using a range of packet sizes.

Test ID	Test Description
---------	------------------

1.2.1	Maximum Throughput With Security Modules Disabled
-------	--

SmartFlow is used to measure maximum UDP throughput with zero packet loss with packet sizes of 1024, 512, 256, 128 and 64 bytes in a single session. The DUT is configured with all security modules disabled.

Test 1.3 - Baseline Latency

Spirent SmartFlow is used to measure latency of the DUT under various load conditions and with various packet sizes. Latency measurements are run with 25%, 50%, 75% and 100% of maximum load as determined in Test 1.2, where the entire load is UDP latency-measurement traffic.

Test ID	Test Description
---------	------------------

1.3.1	Latency With Security Modules Disabled
-------	---

SmartFlow is used to measure UDP latency with packet sizes of 1024, 512, 256, 128 and 64 bytes in a single session. The DUT is configured with all security modules disabled.

Section 2 - Anti Malware

Test 2.1 - Performance Comparison - Anti Malware

The Anti Malware module is evaluated with both HTTP and SMTP traffic where available. The Anti Malware engine will be configured to replace all infected content with a standard message informing the recipient of the action. Although disinfection and quarantine capabilities will be verified where available, they will not form part of the performance testing.

The same performance breaking points are used as in section 1.1. Note that HTTP responses and e-mail body/content remain consistent across all tests, whether the content is "good" or "bad".

Test ID	Test Description
---------	------------------

2.1.1	Maximum TCP Connections Per Second
-------	---

Test 1.1.1 is repeated with Anti Malware module enabled only.

2.1.2	Maximum HTTP Transactions Per Second
-------	---

Test 1.1.2 is repeated with Anti Malware module enabled only.

2.1.3	Maximum Concurrent TCP Connections
-------	---

Test 1.1.3 is repeated with Anti Malware module enabled only.

2.1.4	Maximum Bandwidth for HTTP traffic
-------	---

Test 1.1.4 is repeated with Anti Malware module enabled only.

2.1.5	Maximum SMTP Sessions Per Second
-------	---

Test 1.1.5 is repeated with Anti Malware modules enabled only.

Test 2.2 - Anti Malware Performance With Varying File Sizes

The performance of the Anti Malware module is further evaluated using Test 2.1.1, but this time with a range of file sizes, in order to determine the performance variance when scanning larger and smaller files.

The same performance breaking points are used as in section 1.1.

Test ID	Test Description
---------	------------------

2.2.1	Maximum TCP Connections Per Second - 512 Byte File Size
-------	--

*This test is designed to determine the maximum TCP connection rate of the DUT with a 512 byte object size (Avalanche load specification is **Connections Per Second**) and with only the Anti Malware module enabled. The object size defines the number of bytes contained in the body, excluding any bytes associated with the HTTP header.*

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 the request is satisfied, thus any concurrent TCP connections will be caused purely as a result of latency within the DUT. Load is increased until one or more of the defined breaking points is reached.

2.2.2	Maximum TCP Connections Per Second - 1024 Byte File Size
-------	---

Test 2.2.1 is repeated with 1Kbyte object size.

2.2.3	Maximum TCP Connections Per Second - 5120 Byte File Size
-------	---

Test 2.2.1 is repeated with 5Kbyte object size.

2.2.4	Maximum TCP Connections Per Second - 10240 Byte File Size
-------	--

Test 2.2.1 is repeated with 10Kbyte object size.

2.2.5	Maximum TCP Connections Per Second - 51200 Byte File Size
-------	--

Test 2.2.1 is repeated with 50Kbyte object size.

2.2.6	Maximum TCP Connections Per Second - 102400 Byte File Size
-------	---

Test 2.2.1 is repeated with 100Kbyte object size.

Test 2.3 - HTTP Anti Malware Filter

These tests use a corpus of live virus files which NSS has created from its current library of over 10,000. A subset of 100 viruses was selected for these tests, all of which are readily available from multiple virus archive sites on the Internet.

Eighty of these viruses are on the current WildList (www.wildlist.org) which is widely accepted as the most accurate and current research of its kind. The remaining twenty are common “zoo” viruses which The NSS Group receives on a regular basis in its own AV quarantine. All of the files selected have been scanned and verified by three separate leading desktop AV products as being virus infected.

The following tests are run to determine HTTP Anti Malware performance:

Test ID	Test Description
---------	------------------

2.3.1	Maximum TCP Connections Per Second (Clean)
-------	---

*This test is designed to determine the maximum TCP connection rate of the DUT with a 4Kbyte object size and 100% clean (non-infected) traffic (Avalanche load specification is **Connections Per Second**).*

Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is allowed.

2.3.2 **Maximum TCP connections Per Second (Virus Infected)**

*This test is designed to determine the maximum TCP connection rate of the DUT with a 4Kbyte virus-infected object and 100% infected traffic (Avalanche load specification is **Connections Per Second**). The Anti Malware engine will be expected to detect and alert on this traffic at a minimum, and should preferably replace infected content with a notification to the recipient.*

Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is denied.

2.3.3 **Maximum TCP Connections Per Second (Mix)**

*This test is designed to determine the maximum TCP connection rate of the DUT with a 4K byte object size and a mix of 90% clean and 10% infected traffic (Avalanche load specification is **Connections Per Second**). The Anti Malware engine will be expected to pass 100% of the normal traffic whilst detecting and alerting on 100% of the infected traffic, and should preferably replace infected content with a notification to the recipient.*

Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the "clean" traffic is allowed, and 100% of the "bad" traffic is denied.

Test 2.4 - SMTP Anti Malware Filter

These tests use the same virus corpus subset of 100 viruses as for Test 2.2.

The following tests are run to determine SMTP Anti Malware performance:

Test ID	Test Description
----------------	-------------------------

2.4.1	Maximum SMTP Sessions Per Second (Clean)
--------------	---

*This test is designed to determine the maximum SMTP session rate of the DUT with each e-mail comprising a 1Kbyte body and 4Kbyte attachment, and 100% clean (non-infected) traffic (Avalanche load specification is **Connections Per Second**).*

Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is allowed.

2.4.2	Maximum SMTP Sessions Per Second (Virus Infected)
--------------	--

*This test is designed to determine the maximum SMTP session rate of the DUT with each e-mail comprising a 1Kbyte body and 4Kbyte attachment, and 100% infected traffic (Avalanche load specification is **Connections Per Second**). The Anti Malware engine will be expected to detect and alert on this traffic at a minimum, and should preferably replace infected content with a notification to the recipient.*

Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is denied.

2.4.3	Maximum SMTP Sessions Per Second (Mix)
--------------	---

*This test is designed to determine the maximum SMTP session rate of the DUT with each e-mail comprising a 1Kbyte body and 4Kbyte attachment, and a mix of 90% clean and 10% infected traffic (Avalanche load specification is **Connections Per Second**). The Anti Malware engine will be expected to pass 100% of the normal traffic whilst detecting and alerting on 100% of the infected traffic, and should preferably replace infected content with a notification to the recipient.*

Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the "clean" traffic is allowed, and 100% of the "bad" traffic is denied.

Test 2.5 - Anti Malware Capabilities

It is important that the Anti Malware module blocks all known virus-infected traffic at all times via one or more of the following options: *block session*, *remove/replace* inappropriate content, *quarantine* inappropriate content. The Anti Malware module of a SCA device should also be easy to configure and manage, and alerts should be raised in accordance with the applied policy.

Test ID	Test Description
2.5.1	Anti Malware Policy (Detect & Alert - "In The Wild" Viruses) <i>A subset of 80 common live Windows viruses and cross-platform worms/Trojans is taken from our current corpus of over 10,000 viruses. Every virus file in the test suite has been verified as infected by three separate leading desktop AV products, and every one is currently on the WildList. The DUT will be expected to detect and alert on 100% of these.</i>
2.5.2	Anti Malware Policy (Detect & Alert - "Zoo" Viruses) <i>A subset of 20 common live Windows viruses and cross-platform worms/Trojans is taken from our current corpus of over 10,000 viruses. Every virus file in the test suite has been verified as infected by three separate leading desktop AV products, but these viruses are not necessarily to be found on the current WildList. There is no minimum requirement for detecting these test cases - this test is designed to show which products retain signatures for older viruses or which are capable of detecting virus-like content via heuristic scanning.</i>
2.5.3	Anti Malware Policy (Disinfect) <i>Test 2.5.1 will be repeated with the Anti Malware filter configured to disinfect instead of detect and alert. Disinfection effectiveness will be noted (if available).</i>
2.5.4	Anti Malware Policy (Quarantine) <i>Test 2.5.1 will be repeated with the Anti Malware filter configured to quarantine instead of detect and alert. Quarantine effectiveness will be noted (if available).</i>
2.5.5	Anti Malware Policy (Detection Of Malware Within Compressed Files) <i>Test 2.5.1 will be repeated with the infected files contained within a ZIP archive. Effectiveness of the detection mechanism under these conditions will be noted.</i>
2.5.6	Anti Malware Policy (Detection Of Malware Within Nested Compressed Files) <i>Test 2.5.1 will be repeated with the infected files contained within a nested ZIP archive (ZIPped multiple times). Effectiveness of the detection mechanism under these conditions will be noted.</i>
2.5.6	Anti Malware Management <i>The Anti Malware module of the DUT should be managed from a central console - two-tier or three-tier management architectures are acceptable. Anti Malware configuration should be restricted to enabling/disabling the engine, perhaps with the ability to define those file types which should/should not be scanned. New Anti Malware signature updates should be available at regular intervals, and it should be possible to have these downloaded and applied automatically.</i>
2.5.7	Anti Virus Alerting/Logging <i>The Anti Malware module of the DUT should be capable of raising appropriate alerts and log entries in accordance with the applied filtering policy. Alerts should be available on a central console, whilst log entries should be stored in a secure manner. Additional alert methods (SMTP, SNMP, syslog, etc.) may be configurable. Analysis/reporting capabilities should be provided for log entries, and all pertinent data (date/time, source, destination, description of objectionable content, action, etc.) should be available.</i>

Section 3 - Anti Spam

Test 3.1 - Performance Comparison - Anti Spam

The Anti Spam module is evaluated with SMTP traffic only, and the module will be configured to scan all SMTP content and reject spam messages **or** add a flag to the subject line or e-mail header to indicate suspicious content.

Although quarantine capabilities will be verified where available, they will not form part of the performance testing.

The same performance breaking points are used as in section 1.1. Note that e-mail body/content remain consistent across all tests, whether the content is “good” or “bad”.

Test ID	Test Description
3.1.1	Maximum TCP Connections Per Second <i>Test 1.1.1 is repeated with Anti Spam module enabled only.</i>
3.1.2	Maximum HTTP Transactions Per Second <i>Test 1.1.2 is repeated with Anti Spam module enabled only.</i>
3.1.3	Maximum Concurrent TCP Connections <i>Test 1.1.3 is repeated with Anti Spam module enabled only.</i>
3.1.4	Maximum Bandwidth for HTTP traffic <i>Test 1.1.4 is repeated with Anti Spam module enabled only.</i>
3.1.5	Maximum SMTP Sessions Per Second <i>Test 1.1.5 is repeated with Anti Spam module enabled only.</i>

Test 3.2 - SMTP Anti Spam Filter

These tests use a corpus of current genuine spam e-mails which NSS has created from its current library of over 100,000. A subset of 50 spam mails was selected for these tests, all of which are readily available from archive sites on the Internet.

Most of the spam mails were collected from the most recent archives on SpamArchive (www.spamarchive.org), one of the most complete and current collections of spam e-mails available. Others are common spam mails which The NSS Group receives on a regular basis in its own spam quarantine.

Note that no use will be made of real-time blacklists - the DUT will be expected to identify spam purely from the e-mail content via techniques such as lexical analysis of body and subject line, analysis of graphical content, URL links, etc.

Each spam e-mail consists of a multi-part body with clearly-identifiable spam elements, plus a subject line which comprises subjects identified by current research as being amongst the top 10 most common spam subject lines. The following tests are run to determine SMTP Anti Spam performance:

Test ID	Test Description
3.2.1	Maximum SMTP Sessions Per Second (Clean) <i>This test is designed to determine the maximum SMTP session rate of the DUT with each e-mail comprising a 5Kbyte multi-part body, and 100% clean (non-spam) traffic (Avalanche load specification is Connections Per Second). <i>Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is allowed.</i></i>
3.2.2	Maximum SMTP Sessions Per Second (Spam) <i>This test is designed to determine the maximum SMTP session rate of the DUT with each e-mail comprising a 5Kbyte multi-part body with typical spam-related subject line, and 100% spam traffic (Avalanche load specification is Connections Per Second). The spam engine will be expected to detect this traffic and reject the message or modify the subject line or e-mail header to insert a spam notification for further handling by desktop spam filters. <i>Load is increased until one or more of the defined breaking points is reached and it is verified that at maximum load, 100% of the traffic is denied/modified.</i></i>

3.2.3 **Maximum SMTP Sessions Per Second (Mix)**

*This test is designed to determine the maximum SMTP session rate of the DUT with each e-mail comprising a 5Kbyte multi-part body with typical spam-related subject line, and a mix of 90% clean and 10% spam traffic (Avalanche load specification is **Connections Per Second**). The spam engine will be expected to pass 100% of the normal traffic whilst detecting 100% of the spam traffic and rejecting the messages or modifying the subject line or e-mail header to insert a spam notification for further handling by desktop spam filters.*

Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the "clean" traffic is allowed, and 100% of the "bad" traffic is denied/modified.

Test 3.3 - Anti Spam Capabilities

It is important that the Anti Spam module detects and handles inappropriate traffic in accordance with the applied policy at all times via one or more of the following options:

- *Reject message*
- *Accept and discard message*
- *Insert spam flag into subject line or mail header*
- *Quarantine message.*

The Anti Spam module of a SCA device should also be easy to configure and manage, and alerts should be raised in accordance with the applied policy.

Test ID	Test Description
----------------	-------------------------

3.3.1	<i>Anti Spam Policy (Detect & Flag)</i>
--------------	--

A subset of common spam e-mails is taken from our current corpus of over 100,000 entries taken from www.spamarchive.org and The NSS Group's own spam quarantine. These will usually contain easily identifiable spam content, but many will employ various obfuscation techniques such as graphical representations of text, obfuscated/encoded URL links, hidden background text containing "normal" text designed to fool lexical analysis engines, and so on. The DUT will be expected to detect >50% of these and mark them as spam (or apply a spam rating) in the subject line or mail header for further processing by third party spam filters.

3.3.2	<i>Anti Spam Policy (Quarantine)</i>
--------------	---

Test 3.3.1 will be repeated with the spam filter configured to quarantine instead of detect and flag. Quarantine effectiveness will be noted (if available). Quarantine facilities should be fully manageable by the administrator, but with the ability to delegate release/deletion of suspected spam to individual users.

3.3.3	<i>Anti Spam Policy (Reject Message)</i>
--------------	---

Test 3.3.1 will be repeated with the spam filter configured to reject message instead of detect and flag. Rejection effectiveness will be noted (if available).

3.3.4	<i>Anti Spam Policy (Accept & Discard Message)</i>
--------------	---

Test 3.3.1 will be repeated with the spam filter configured to accept and discard message instead of detect and flag. Accept/Discard effectiveness will be noted (if available).

3.3.5	<i>Anti Spam Management</i>
--------------	------------------------------------

The Anti Spam module of the DUT should be managed from a central console - two-tier or three-tier management architectures are acceptable. Spam policy definition should be straightforward, allowing the administrator to simply enable/disable the spam engine, or select which techniques should be applied (where multiple techniques are offered, such as lexical analysis, RBL, etc.). The administrator should also be able to override configured settings via the use of white/black lists where required.

3.3.6	<i>Anti Spam Alerting/Logging</i>
--------------	--

The Anti Spam module of the DUT will not normally raise alerts. However, some data should be retained for analysis/reporting purposes.

Section 4 - Content/Web Filtering

Test 4.1 - Performance Comparison - Content/Web Filtering

The Content/Web Filtering module is evaluated in three main areas: *URL filtering* (ability to block access to unacceptable Web sites based on the URL), *HTTP/SMTP Content Filtering* (ability to block unacceptable Web/e-mail content as the data is transmitted from server to client), and *File Blocking* (ability to block HTTP/SMTP traffic based on “bad” file extensions).

Devices may not offer all these features, but, where they are available, they will be enabled with all options/categories activated. Note also that some devices will implement some of these features as part of other modules (i.e. file blocking may be part of the AV module).

The same performance breaking points are used as in section 1.1. Note that HTTP responses and e-mail body/content remain consistent across all tests, whether the content is “good” or “bad”.

Test ID	Test Description
4.1.1	Maximum TCP Connections Per Second <i>Test 1.1.1 is repeated with firewall and Content/Web Filtering modules enabled only.</i>
4.1.2	Maximum HTTP Transactions Per Second <i>Test 1.1.2 is repeated with firewall and Content/Web Filtering modules enabled only.</i>
4.1.3	Maximum Concurrent TCP Connections <i>Test 1.1.3 is repeated with firewall and Content/Web Filtering modules enabled only.</i>
4.1.4	Maximum Bandwidth for HTTP traffic <i>Test 1.1.4 is repeated with firewall and Content/Web Filtering modules enabled only.</i>
4.1.5	Maximum SMTP Sessions Per Second <i>Test 1.1.5 is repeated with firewall and Content/Web Filtering modules enabled only.</i>

Test 4.2 - HTTP URL Filter

This test uses lists of URLs which are known will be allowed or denied by the HTTP URL filtering in the DUT. “Clean” URLs are all valid business-oriented sites.

To create the corpus of “bad” URLs, NSS performed extensive analysis of five of the market leading URL filtering databases to determine the most popular categories in terms of percentage of URLs. The following results were obtained:

1. *News (31%)*
2. *On-line shopping/Auctions (24%)*
3. *Adult oriented (19%)*
4. *Travel (16%)*
5. *Sports (10%)*

NSS created a list of 100 URLs taken from those databases and in the above proportions to use in the following URL filtering tests (both good and bad URLs are cycled, ensuring different client IPs request different URLs during the test):

Test ID	Test Description
4.2.1	<p>Maximum TCP Connections Per Second (Allow)</p> <p><i>This test is designed to determine the maximum TCP connection rate of the DUT with a 64 byte object size and 100% “clean” URLs (Avalanche load specification is Simulated Users Per Second).</i></p> <p><i>Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is allowed.</i></p>
4.2.2	<p>Maximum TCP connections Per Second (Deny)</p> <p><i>This test is designed to determine the maximum TCP connection rate of the DUT with a 64 byte object size and 100% “bad” URLs (Avalanche load specification is Simulated Users Per Second).</i></p> <p><i>Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is denied.</i></p>
4.2.3	<p>Maximum TCP Connections Per Second (Mix)</p> <p><i>This test is designed to determine the maximum TCP connection rate of the DUT with a 64 byte object size and a mix of 90% “clean” URLs and 10% “bad” URLs (Avalanche load specification is Simulated Users Per Second).</i></p> <p><i>Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the “clean” traffic is allowed, and 100% of the “bad” traffic is denied.</i></p>

Test 4.3 - HTTP Content Filter

This test uses Web content which is known will be allowed or denied by the HTTP content filtering in the DUT. “Clean” Web pages contain 4KB of innocuous content, whilst “bad” Web pages contain the same content interspersed with “trigger” words such as:

- *SEX!!!*
- *Viagra*
- *V1@gr@*
- *V I A G R A*
- *Xanax*
- *X@n@x*

If the DUT relies on the administrator to create lists of unacceptable words and phrases rather than providing built-in lists of bad content, the above words will be entered as custom filters. The following tests are then run:

Test ID	Test Description
4.3.1	<p>Maximum TCP Connections Per Second (Allow)</p> <p><i>This test is designed to determine the maximum TCP connection rate of the DUT with a 4096 byte object size and 100% “clean” Web content (Avalanche load specification is Connections Per Second).</i></p> <p><i>Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is allowed.</i></p>
4.3.2	<p>Maximum TCP connections Per Second (Deny)</p> <p><i>This test is designed to determine the maximum TCP connection rate of the DUT with a 4096 byte object size and 100% “bad” Web content (Avalanche load specification is Connections Per Second).</i></p>

Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is denied.

4.3.3 **Maximum TCP Connections Per Second (Mix)**

This test is designed to determine the maximum TCP connection rate of the DUT with a 4096 byte object size and a mix of 90% "clean" Web content and 10% "bad" Web content (Avalanche load specification is **Connections Per Second**).

Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the "clean" traffic is allowed, and 100% of the "bad" traffic is denied.

Test 4.4 - SMTP Content Filter

This test uses e-mail content which is known will be allowed or denied by the SMTP content filtering in the DUT.

"Clean" e-mails consist of 1KB of innocuous body content with a 4KB plain text attachment, whilst "bad" e-mails consist of the same attachment, and the same body content interspersed with "trigger" words such as:

- *SEX!!!*
- *Viagra*
- *V1@gr@*
- *V I A G R A*
- *Xanax*
- *X@n@x*

If the DUT relies on the administrator to create lists of unacceptable words and phrases rather than providing built-in lists of bad content, the above words will be entered as custom filters.

The following tests are then run:

Test ID	Test Description
---------	------------------

4.4.1	Maximum SMTP Sessions Per Second (Allow)
-------	---

This test is designed to determine the maximum SMTP session rate of the DUT with a 1Kbyte mail body, a 4Kbyte plain text attachment, and 100% "clean" mail content (Avalanche load specification is **Simulated Users Per Second**).

Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is allowed.

4.4.2	Maximum SMTP Sessions Per Second (Deny)
-------	--

This test is designed to determine the maximum SMTP session rate of the DUT with a 1Kbyte mail body, a 4Kbyte plain text attachment, and 100% "bad" mail content (Avalanche load specification is **Simulated Users Per Second**).

Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is denied.

4.4.3	Maximum SMTP Sessions Per Second (Mix)
-------	---

This test is designed to determine the maximum SMTP session rate of the DUT with a 1Kbyte mail body, a 4Kbyte plain text attachment, and a mix of 90% "clean" mail content and 10% "bad" mail content (Avalanche load specification is **Simulated Users Per Second**).

Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the "clean" traffic is allowed, and 100% of the "bad" traffic is denied.

Test 4.5 - HTTP File Blocking

This test requests files from the Web server which are known will be allowed or denied by the file blocking capability of the DUT. "Clean" Web content consists of 4Kbyte files with extensions such as:

- *HTM*
- *HTML*
- *JPEG*
- *JPG*
- *MPEG*
- *MPG*
- *GIF*

"Bad" Web content consists of 4Kbyte files with extensions such as:

- *BAS*
- *BAT*
- *CMD*
- *COM*
- *DLL*
- *EXE*
- *INF*
- *PIF*
- *SCR*
- *VB*
- *VBE/VBS*

If the DUT relies on the administrator to create lists of unacceptable file extensions rather than providing built-in lists, the above extensions will be entered as custom filters. The following tests are then run:

Test ID	Test Description
4.5.1	<p>Maximum TCP Connections Per Second (Allow)</p> <p><i>This test is designed to determine the maximum TCP connection rate of the DUT with a 4Kbyte object size and 100% "clean" Web content (Avalanche load specification is Connections Per Second).</i></p> <p><i>Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is allowed.</i></p>
4.5.2	<p>Maximum TCP connections Per Second (Deny)</p> <p><i>This test is designed to determine the maximum TCP connection rate of the DUT with a 4Kbyte object size and 100% "bad" Web content (Avalanche load specification is Connections Per Second).</i></p> <p><i>Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is denied.</i></p>
4.5.3	<p>Maximum TCP Connections Per Second (Mix)</p> <p><i>This test is designed to determine the maximum TCP connection rate of the DUT with a 4Kbyte object size and a mix of 90% "clean" Web content and 10% "bad" Web content (Avalanche load specification is Connections Per Second).</i></p> <p><i>Client and server are using HTTP 1.0 with no keep-alive, and the client will open a TCP connection, send one HTTP request, and close the connection. Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the "clean" traffic is allowed, and 100% of the "bad" traffic is denied.</i></p>

Test 4.6 - SMTP File Blocking

This test uses e-mail content which is known will be allowed or denied by the file blocking capability of the DUT.

“Clean” e-mails consist of 1Kbyte of innocuous body content with a 4Kbyte attachment with one of the following extensions:

- *HTM*
- *HTML*
- *JPEG*
- *JPG*
- *MPEG*
- *MPG*
- *GIF*

“Bad” e-mail content consists of the same 1Kbyte body, and the same 4Kbyte attachment, but this time with extensions such as:

- *BAS*
- *BAT*
- *CMD*
- *COM*
- *DLL*
- *EXE*
- *INF*
- *PIF*
- *SCR*
- *VB*
- *VBE/VBS*

If the DUT relies on the administrator to create lists of unacceptable file extensions rather than providing built-in lists, the above extensions will be entered as custom filters. The following tests are then run:

Test ID	Test Description
4.6.1	<p>Maximum SMTP Sessions Per Second (Allow)</p> <p><i>This test is designed to determine the maximum SMTP session rate of the DUT with 100% of e-mails consisting of a 1Kbyte mail body and a 4Kbyte attachment with “permitted” extension (Avalanche load specification is Connections Per Second).</i></p> <p><i>Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is allowed.</i></p>
4.6.2	<p>Maximum SMTP Sessions Per Second (Deny)</p> <p><i>This test is designed to determine the maximum SMTP session rate of the DUT with 100% of e-mails consisting of a 1Kbyte mail body and a 4Kbyte attachment with “denied” extension (Avalanche load specification is Connections Per Second).</i></p> <p><i>Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the traffic is denied.</i></p>
4.6.3	<p>Maximum SMTP Sessions Per Second (Mix)</p> <p><i>This test is designed to determine the maximum SMTP session rate of the DUT with a 1Kbyte mail body, a 4Kbyte attachment, and a mix of 90% “permitted” and 10% “denied” file extensions (Avalanche load specification is Connections Per Second).</i></p> <p><i>Load is increased until one or more of the defined breaking points is reached, and it is verified that at maximum load, 100% of the “clean” traffic is allowed, and 100% of the “bad” traffic is denied.</i></p>

Test 4.7 - Content Filtering Capabilities

It is important that the Content Filtering module denies inappropriate traffic in accordance with the applied filtering policy at all times via one or more of the following options:

- *Block session*
- *Remove/replace inappropriate content*
- *Quarantine inappropriate content*

The Content Filtering module of a UTM device should also be easy to configure and manage, and alerts should be raised in accordance with the applied filtering policy.

Test ID	Test Description
4.7.1	Content Filtering Policy (URL-Deny) <i>A URL list consisting of known "bad" URLs (in the proportions/categories mentioned in section 4.2) will be used to determine that the content filter is configured to detect and block unwanted URLs. The device will be expected to detect >75% of these.</i>
4.7.2	Content Filtering Policy (URL-Allow) <i>A URL list consisting of known "good" URLs will be used to determine that the content filter is configured to allow these with no false positives. The device will be expected to allow 100% of these.</i>
4.7.3	Content Filtering Policy (HTTP Response-Deny) <i>A corpus of Web pages where the body contains unacceptable trigger words/phrases such as those mentioned in section 4.3 will be used to determine that the content filter is configured to detect and block inappropriate Web traffic. The device will be expected to detect 100% of these.</i>
4.7.4	Content Filtering Policy (HTTP Response-Allow) <i>A corpus of Web pages containing no inappropriate content (but where the HTTP response size is identical to test 4.7.3) will be used to determine that the content filter is configured to allow these with no false positives. The device will be expected to allow 100% of these.</i>
4.7.5	Content Filtering Policy (E-Mail-Deny) <i>A corpus of e-mails where the body contains unacceptable trigger words/phrases such as those mentioned in section 4.4 will be used to determine that the content filter is configured to detect and block inappropriate e-mail traffic. The device will be expected to detect 100% of these.</i>
4.7.6	Content Filtering Policy (E-Mail-Allow) <i>A corpus of e-mails containing no inappropriate content (but where the body and attachment sizes are identical to test 4.7.5) will be used to determine that the content filter is configured to allow these with no false positives. The device will be expected to allow 100% of these.</i>
4.7.7	Content Filtering Policy (HTTP Response File Extension-Deny) <i>A corpus of files with known "bad" extensions (such as those mentioned in section 4.5) are requested via HTTP to determine that the content filter is configured to detect and block unwanted files. The device will be expected to detect 100% of these.</i>
4.7.8	Content Filtering Policy (HTTP Response File Extension-Allow) <i>A corpus of files with known "good" extensions (such as those mentioned in section 4.5) are requested via HTTP to determine that the content filter is configured to allow these with no false positives. The device will be expected to allow 100% of these.</i>
4.7.9	Content Filtering Policy (E-Mail Attachment File Extension-Deny) <i>A corpus of files with known "bad" extensions (such as those mentioned in section 4.6) are transmitted as e-mail attachments to determine that the content filter is configured to detect and block unwanted files. The device will be expected to detect 100% of these.</i>

4.7.10 **Content Filtering Policy (E-Mail Attachment File Extension-Allow)**

A corpus of files with known "good" extensions (such as those mentioned in section 4.6) are transmitted as e-mail attachments to determine that the content filter is configured to allow these with no false positives. The device will be expected to allow 100% of these.

4.7.11 **Content Filter Management**

The Content Filter module of the DUT should be managed from a central console - two-tier or three-tier management architectures are acceptable. Filter policy definition should be straightforward, allowing the administrator to select URL categories/sub-categories, lists of words/phrases designated as inappropriate content, and file extensions to block, or to simply select vendor recommended settings.

The administrator should also be able to override configured settings via the use of white/black lists, and custom lists of inappropriate words/phrases/file extensions.

Where multiple types of Content Filtering are available (URL filtering, Web, E-mail, file extension blocking) it should be possible to enable/disable these individually (as well as select HTTP/SMTP as required).

New Content Filter updates should be available at regular intervals, and it should be possible to have these downloaded and applied automatically.

4.7.12 **Content Filter Alerting/Logging**

The Content Filter module of the DUT should be capable of raising appropriate alerts and log entries in accordance with the applied filtering policy. Alerts should be available on a central console, whilst log entries should be stored in a secure manner. Additional alert methods (SMTP, SNMP, syslog, etc.) may be configurable. Analysis/reporting capabilities should be provided for log entries, and all pertinent data (date/time, source, destination, description of objectionable content, action, etc.) should be available.

Section 5 - All Modules

Most companies deploying these devices will wish to enable all security modules to provide the maximum protection. A SCA device should therefore allow the administrator to enable all modules without seriously impacting the throughput and latency of the DUT. Unrealistic performance claims will be identified in this test, which we believe represents the most common deployment scenario.

Test 5.1 - Performance Comparison - All Modules

For these tests, **all** available security modules are enabled simultaneously to determine the cumulative effect on performance.

The same performance breaking points are used as in section 1.1.

Test ID	Test Description
5.1.1	Maximum TCP Connections Per Second <i>Test 1.1.1 is repeated with all security modules enabled.</i>
5.1.2	Maximum HTTP Transactions Per Second <i>Test 1.1.2 is repeated with all security modules enabled.</i>
5.1.3	Maximum Concurrent TCP Connections <i>Test 1.1.3 is repeated with all security modules enabled.</i>
5.1.4	Maximum Bandwidth for HTTP traffic <i>Test 1.1.4 is repeated with all security modules enabled.</i>
5.1.5	Maximum SMTP Sessions Per Second <i>Test 1.1.5 is repeated with all security modules enabled.</i>

Section 6 – Management and Configuration

Naturally, the management and reporting capabilities will be evaluated to the same high standard we have applied to previous tests.

For SCA devices, we will be paying particular attention to ensuring that management, alerting and reporting is as seamless as possible across all the various security modules.

Test 6.1 - Management & Configuration Features

The aim of this section is to fully evaluate each product in terms of ease of use, management and configuration, and alerting and reporting capabilities. Clearly it is not possible to “benchmark” a product in these areas, and so this section comprises an in-depth technical evaluation covering all the main features and benefits of the product.

The following areas will be examined in detail:

- *Architecture - two/three-tier management - capabilities and scalability*
- *High availability capabilities (where offered) - multiple redundant components, multiple mirrored sensors, stateful failover, etc.*
- *Ease of installation - appliance, management server (if applicable) and console*
- *Ease of management – once installed, remote sensors should be manageable from a central console*
- *Authentication and encryption between sensor and management server, and between management server and console, for secure communication*
- *Policy definition - ease of use and flexibility*
- *Ability to share policies, global settings/variables across security components (i.e. only define internal network addresses once)*
- *Policy deployment - Can policies be defined centrally and rolled out to multiple sensors (globally, or in logical groups)?*
- *Policy changes - Can the console determine which sensors are using which policies and deploy automatically following amendments?*
- *Ability to define custom filtering/blocking rules - if available, how flexible is this process?*
- *How attack/virus/spam signatures are obtained/deployed, and frequency of updates*
- *Reporting from sensor/management server to console - range of action/alert options*
- *Console interface - notifications and alerts. How easy is it to filter and extract individual events?*
- *Accuracy and readability of alerts*
- *Log file maintenance – automatic rotation, archiving, reporting from archived logs, etc.*
- *Management reporting – range of reports/custom reports/the ease with which detail can be extracted and reported*
- *What are the limitations and restrictions on enterprise-wide alerting and reporting? Can reports consolidate output from every (i) management server, (ii) sensor?*
- *Documentation – What documentation is included? On-line or hard copy? Supplemental information available on-line?*
- *Licensing and pricing model, including maintenance/update costs*