Fortinet FortiGate-800 V2.80

Technical Evaluation

An NSS Group Report

First published January 2005 (Version 1.0)

Published by The NSS Group
Mas la Carrière, Route de Ganges
30440 Sumène, France

Tel : +33 (0)4 67 81 49 11
E-mail : info@nss.co.uk
Internet : http://www.nss.co.uk

TABLE OF CONTENTS

TABLE OF FIGURES

The NSS Group

The NSS Group is the world's foremost independent security testing facility.

With British headquarters, and security and network infrastructure testing facilities in the South of France, The NSS Group offers a range of specialist IT, networking and security-related services to vendors and end-user organisations world-wide.

The NSS Group's Security Testing Laboratories are available to vendors and end-users for fully independent testing of networking, communications and security hardware and software.

The NSS Group also operates certification schemes for vendors and certification bodies, and currently provides evaluation and certification of a wide range of security products, including IDS/IPS appliances, firewalls, VPN's, Web Application firewalls, multi-function security appliances, cryptographic devices and PKI products.

Output from the labs, including detailed research reports, articles and white papers on the latest network and security technologies, are made available on the NSS web site at http://www.nss.co.uk.

The NSS Group awards are recognised world-wide as being the most desirable and essential when it comes to security products. Vendors consider the awards to be a crucial step in any security-related marketing campaign, whilst feedback from readers of the reports indicates that participation in an NSS Group test and/or one of the **NSS Approved** awards is a prerequisite for any security product in order to be considered for purchase.

Foreword

Following the huge success the first comprehensive *Intrusion Prevention System* (IPS) test of its kind, The NSS Group is pleased to present the results of its second IPS Group Test which includes a number of new products not included in the first report.

As with Edition 1, this exhaustive review will give readers a complete perspective of the capabilities, maturity and suitability for immediate deployment of each of the products tested. The NSS Group established this test as IPS products are being actively deployed as a new layer in defence-in-depth security architectures.

It is interesting to note that between publishing Edition 1 and Edition 2 the analyst groups who were previously so sure that IDS was dead and IPS stillborn have now come around to our way of thinking - while the so-called "*deep inspection firewalls*" are not ready for prime-time deployments, security administrators need to make the best use of the technology that **is** available, and for now that means a combination of firewalls, in-line intrusion prevention devices, and intrusion detection systems. They are likely to be in use for quite some time to come, too!

The NSS IPS Group Test evaluates the performance, reliability, security effectiveness, and usability of Network IPS products. The test consists of seven sections within three primary areas: *performance and reliability, security accuracy,* and *usability*.

Overall, the brand new test suite contains over **800 individual tests**, many of which are run multiple times, to provide the most thorough and complete evaluation of IPS products available anywhere today. This edition also sees the introduction of a new *Rate-Based IPS* methodology to complement our exiting *Content-Based IPS* methodology used in Edition 1. This has allowed us to more accurately test Rate-Based/Attack Mitigation products, and two devices were tested against this new methodology in the latest report (one of them actually tested against **both** methodologies – a first).

It is worth pointing out that not every product submitted for testing receives an *NSS Approved* award. Standards are very high, and out of nine products signed up for this group test initially, only the five included in the final Edition 2 report have received ***NSS Approved*** awards.

We believe that our IPS test methodologies - which have been updated for this test - will become the *de facto* standard for testing in-line Intrusion Prevention/Attack Mitigation devices, and the *NSS Approved* logo an essential item on the list of requirements when purchasing these products.

We also believe that this report is essential reading for anyone considering deploying Intrusion Prevention Systems in their networks, either in a test or live situation, and we hope that you find it both informative and useful in making your purchasing decisions. The **IPS Group Test (Edition 2)** report can be viewed on-line at *www.nss.co.uk/ips*.

Bob Walder

INTRODUCTION

In a survey commissioned by VanDyke Software, some 66 per cent of the companies who responded said that they perceive system penetration to be the largest threat to their enterprises.

The survey revealed that the top eight threats experienced by those surveyed were *viruses* (78 per cent of respondents), *system penetration* (50 per cent), *DoS* (40 per cent), *insider abuse* (29 per cent), *spoofing* (28 per cent), *data/network sabotage* (20 per cent), and *unauthorised insider access* (16 per cent).

Although 86 per cent of respondents use firewalls (a disturbingly **low** figure in this day and age, to be honest!), it is apparent that firewalls are not always effective against many intrusion attempts. The average firewall is designed to deny clearly suspicious traffic - such as an attempt to telnet to a device when corporate security policy forbids telnet access completely - but is also designed to allow some traffic through - Web traffic to an internal Web server, for example.

The problem is, that many exploits attempt to take advantage of weaknesses in the very protocols that **are** allowed through our perimeter firewalls, and once the Web server has been compromised, this can often be used as a springboard to launch additional attacks on other internal servers. Once a “rootkit” or “back door” has been installed on a server, the hacker has ensured that he will have unfettered access to that machine at any point in the future.

Firewalls are also typically employed only at the network perimeter. However, many attacks, intentional or otherwise, are launched from within an organisation. Virtual private networks, laptops, and wireless networks all provide access to the internal network that often bypasses the firewall. Intrusion detection systems may be effective at detecting suspicious activity, but do not provide *protection* against attacks. Recent worms such as Slammer and Blaster have such fast propagation speeds that by the time an alert is generated, the damage is done and spreading fast.

Intrusion Prevention Systems (IPS)

The inadequacies inherent in current defences has driven the development of a new breed of security products known as *Intrusion Prevention Systems* (IPS). This is a term which has provoked some controversy in the industry since some firewall and IDS vendors think it has been “hijacked” and used as a marketing term rather than as a description for any kind of new technology.

Whilst it is true that firewalls, routers, IDS devices and even AV gateways all have intrusion prevention technology included in some form, we believe that there are sufficient grounds to create a new market sector for true *Intrusion Prevention Systems*.

These systems are proactive defence mechanisms designed to detect malicious packets within normal network traffic (something that the current breed of firewalls do not actually do, for example) and stop intrusions dead, blocking the offending traffic automatically before it does any damage rather than simply raising an alert as, or after, the malicious payload has been delivered.

Within the IPS market place, there are two main categories of product: *Host IPS* and *Network IPS*, with the latter being further sub-divided into *Content-Based* and *Rate-Based* (or *Attack Mitigation*) systems.

Host IPS (HIPS)

As with Host IDS systems, the Host IPS relies on agents installed directly on the system being protected. It binds closely with the operating system kernel and services, monitoring and intercepting system calls to the kernel or APIs in order to prevent attacks as well as log them.

It may also monitor data streams and the environment specific to a particular application (file locations and Registry settings for a Web server, for example) in order to protect that application from generic attacks for which no “signature” yet exists.

One potential disadvantage with this approach is that, given the necessarily tight integration with the host operating system, future OS upgrades could cause problems.

Since a Host IPS agent intercepts all requests to the system it protects, it has certain prerequisites - it must be very reliable, must not negatively impact performance, and must not block legitimate traffic. Any HIPS that does not meet these minimum requirements should never be installed in a host, no matter how effectively it blocks attacks.

Network IPS (NIPS)

The Network IPS combines features of a standard IDS, an IPS and a firewall, and is sometimes known as an *In-line IDS* or *Gateway IDS (GIDS)*. The next-generation firewall - the *deep inspection firewall* - also exhibits a similar feature set, though we do not believe that the deep inspection firewall is ready for mainstream deployment just yet.

As with a typical firewall, the NIPS has at least two network interfaces, one designated as *internal* and one as *external*. As packets appear at the either interface they are passed to the detection engine, at which point the IPS device functions much as any IDS would in determining whether or not the packet being examined poses a threat.

However, if it should detect malicious traffic, in addition to raising an alert, it will discard the packet(s) and mark that flow as bad. As the remaining packets that make up that particular TCP session arrive at the IPS device, they are discarded immediately.

Legitimate packets are passed through to the second interface and on to their intended destination. A useful side effect of some NIPS products is that as a matter of course - in fact as part of the initial detection process - they will provide “*packet scrubbing*” functionality to remove protocol inconsistencies resulting from varying interpretations of the TCP/IP specification (or intentional packet manipulation).

Thus any fragmented packets, out-of-order packets, or packets with overlapping IP fragments will be re-ordered and “cleaned up” before being passed to the destination host, and illegal packets can be dropped completely.

One thing to watch out for - don't let the "reactive" IDS vendors kid you into believing that they have *intrusion prevention* capabilities just because they can send TCP reset commands or re-configure a firewall when they detect an attack (a worrying piece of FUD that we have noticed in some IDS marketing literature recently).

The problem here is that unless the attacker is operating on a 2400 baud modem, the likelihood is that by the time the IDS has detected the offending packet, raised an alert, and transmitted the TCP Resets - and especially by the time the two ends of the connection have received the Reset packets and acted on them (or the firewall or router has had time to activate new rules to block the remainder of the flow) - the payload of the exploit has long since been delivered….. *game over*! Our guess is that there are not many crackers using 2400 baud modems these days….

A true IPS device, however, is sitting in-line - **all** the packets have to pass through it. Therefore, as soon as a suspicious packet has been detected - and **before** it is passed to the internal interface and on to the protected network, it can be dropped. Not only that, but now that flow has been flagged as suspicious, **all** subsequent packets that are part of that session can also be dropped with very little additional processing. Oh, and for good measure, some products are also capable of sending *TCP Resets* or *ICMP Unreachable* messages to the attacking host.

Rate-Based IPS (Attack Mitigator)

Most NIPS products are basically IDS engines that operate in-line, and are thus dependent on protocol analysis or signature matching to recognise malicious content within individual packets (or across groups of packets). These can be classed as *Content-Based IPS* systems.

There is, however, a second breed of Network IPS that ignores packet content almost completely, instead monitoring for anomalies in network traffic that might characterise a flood attempt, scan attempt, and so on. These devices are capable of monitoring traffic flows in order to determine what is considered "normal", and applying various techniques to determine when that traffic deviates from normal. This is not always as simple as watching for high-volumes of a specific type of traffic in a short space of time, since they must also be capable of detecting "stealth "attacks, such as low-rate connection floods and slow port scan attempts.

Since these devices are concerned more with anomalies in traffic flow than packet contents, they are classed as *Rate-Based IPS* systems - and are also known as *Attack Mitigators*, as they are so effective against DOS and DDOS attacks.

Implementation Challenges

There are a number of challenges to the implementation of an IPS device that do not have to be faced when deploying passive-mode IDS products. These challenges all stem from the fact that the IPS device is designed to work in-line, presenting a potential choke point and single point of failure.

If a passive IDS fails, the worst that can happen is that some attempted attacks may go undetected. If an in-line device fails, however, it can seriously impact the performance of the network.

Perhaps latency rises to unacceptable values, or perhaps the device fails closed, in which case you have a self-inflicted Denial of Service condition on your hands. On the bright side, there will be no attacks getting through! But that is of little consolation if none of your customers can reach your e-commerce site.

Even if the IPS device does not fail altogether, it still has the potential to act as a bottleneck, increasing latency and reducing throughput as it struggles to keep up with up to a Gigabit or more of network traffic. Devices using off-the-shelf hardware will certainly struggle to keep up with a heavily loaded Gigabit network, especially if there is a substantial signature set loaded, and this could be a major concern for both the network administrator - who could see his carefully crafted network response times go through the roof when a poorly designed IPS device is placed in-line - as well as the security administrator, who will have to fight tooth and nail to have the network administrator allow him to place this unknown quantity amongst his high performance routers and switches.

As an integral element of the network fabric, the Network IPS device must perform much like a network switch. It must meet stringent network performance and reliability requirements as a prerequisite to deployment, since very few customers are willing to sacrifice network performance and reliability for security. A NIPS that slows down traffic, stops good traffic, or crashes the network is of little use.

Dropped packets are also an issue, since if even one of those dropped packets is one of those used in the exploit data stream it is possible that the entire exploit could be missed. Most high-end IPS vendors will get around this problem by using custom hardware, populated with advanced FPGAs and ASICs - indeed, it is necessary to design the product to operate as much as a switch as an intrusion detection and prevention device.

It is very difficult for any security administrator to be able to characterise the traffic on his network with a high degree of accuracy. What is the average bandwidth? What are the peaks? Is the traffic mainly one protocol or a mix? What is the average packet size and level of new connections established every second - both critical parameters that can have detrimental effects on some IDS/IPS engines? If your IPS hardware is operating “on the edge”, all of these are questions that need to be answered as accurately as possible in order to prevent performance degradation.

Another potential problem is the good old *false positive*. The bane of the security administrator’s life (apart from the script kiddie, of course!), the false positive rears its ugly head when an exploit signature is not crafted carefully enough, such that legitimate traffic can cause it to fire accidentally. Whilst merely annoying in a passive IDS device, consuming time and effort on the part of the security administrator, the results can be far more serious and far reaching in an in-line IPS appliance.

Once again, the result is a self-inflicted Denial of Service condition, as the IPS device first drops the “offending” packet, and then potentially blocks the entire data flow from the suspected hacker. If the traffic that triggered the false positive alert was part of a customer order, you can bet that the customer will not wait around for long as his entire session is torn down and all subsequent attempts to reconnect to your e-commerce site (if he decides to bother retrying at all, that is) are blocked by the well-meaning IPS.

Another potential problem with any Gigabit IPS/IDS product is, by its very nature and capabilities, the amount of alert data it is likely to generate. On such a busy network, how many alerts will be generated in one working day? Or even one hour? Even with relatively low alert rates of ten per second, you are talking about 36,000 alerts every hour. That is 864,000 alerts each and every day. The ability to tune the signature set accurately is essential in order to keep the number of alerts to an absolute minimum. Once the alerts have been raised, however, it then becomes essential to be able to process them effectively. Advanced alert handling and forensic analysis capabilities - including detailed exploit information and the ability to examine packet contents and data streams - can make or break a Gigabit IDS/IPS product.

Of course, one point in favour of IPS when compared with IDS is that because it is designed to prevent the attacks rather than just detect and log them, the burden of examining and investigating the alerts - and especially the problem of rectifying damage done by successful exploits - is reduced considerably.

Requirements for effective prevention

Having pointed out the potential pitfalls facing anyone deploying these devices, what features are we looking for that will help us to avoid such problems?

- **In-line operation -** only by operating in-line can an IPS device perform true protection, discarding all suspect packets immediately and blocking the remainder of that flow
- **Reliability and availability -** should an in-line device fail, it has the potential to close a vital network path and thus, once again, cause a DoS condition. An extremely low failure rate is thus very important in order to maximise up-time, and if the worst should happen, the device should provide the option to fail open or support fail-over to another sensor operating in a fail-over group (see below). In addition, to reduce downtime for signature and protocol coverage updates, an IPS must support the ability to receive these updates without requiring a device re-boot. When operating inline, sensors rebooting across the enterprise effectively translate into network downtime for the duration of the reboot
- **Resilience -** as mentioned above, the very minimum that an IPS device should offer in the way of High Availability is to fail open in the case of system failure or power loss (some environments may prefer this default condition to be “fail closed” as with a typical firewall, however - the most flexible products will allow this to be user-configurable). Active-Active stateful fail-over with cooperating in-line sensors in a fail-over group will ensure that the IPS device does not become a single point of failure in a critical network deployment
- **Low latency -** when a device is placed in-line, it is essential that its impact on overall network performance is minimal. Packets should be processed quickly enough such that the overall latency of the device is as close as possible to that offered by a layer 2/3 device such as a switch, and no more than a typical layer 4 device such as a firewall or load-balancer.
- **High performance -** packet processing rates must be at the rated speed of the device under real-life traffic conditions, and the device must meet the stated performance with all signatures enabled.

Headroom should be built into the performance capabilities to enable the device to handle any increases in size of signature packs that may occur over the next three years. Ideally, the detection engine should be designed in such a way that the number “signatures” (or “checks”) loaded does not affect the overall performance of the device.

- **Unquestionable detection accuracy -** it is imperative that the quality of the signatures is beyond question, since false positives can lead to a Denial of Service condition. The user MUST be able to trust that the IDS is blocking only the user selected malicious traffic. New signatures should be made available on a regular basis, and applying them should be quick (applied to all sensors in one operation via a central console) and seamless (no sensor reboot required)
- **Fine-grained granularity and control -** fine grained granularity is required in terms of deciding exactly which malicious traffic is blocked. The ability to specify traffic to be blocked by attack, by policy, or right down to individual host level is vital. In addition, it may be necessary to only alert on suspicious traffic for further analysis and investigation
- **Advanced alert handling and forensic analysis capabilities -** once the alerts have been raised at the sensor and passed to a central console, someone has to examine them, correlate them where necessary, investigate them, and eventually decide on an action. The capabilities offered by the console in terms of alert viewing (real time and historic) and reporting are key in determining the effectiveness of the IPS product.

The NSS Intrusion Prevention Group Test

The NSS Group conducted the first comprehensive IPS test of its kind, now updated in this Edition. This exhaustive review will give readers a complete perspective of the capabilities, maturity and suitability of the products tested for their particular needs.

As part of its extensive IPS/Attack Mitigator test methodologies (see section on *Testing Methodology* later in this report for detailed methodologies, updated for this latest test) The NSS Group subjects each product to a brutal battery of tests that verify the stability and performance of each IPS tested, determine the accuracy of its security coverage, and ensure that the device will not block legitimate traffic.

If a particular IPS has been designated as *NSS Approved*, customers can be confident that the device will not significantly impact network/host performance, cause network/host crashes, or otherwise block legitimate traffic.

To assess the complex matrix of IPS/Attack Mitigator performance and security requirements, the NSS Group has developed a specialised lab environment that is able to exercise every facet of an IPS product. The test suite contains over 800 individual tests that evaluate IPS products in three main areas: *performance and reliability, security accuracy,* and *usability*.

This thorough review should give readers a complete perspective of the capabilities, maturity and suitability of the products tested for their particular needs.

Performance

Any IPS is expected to be reliable (not crash), to never block legitimate traffic, and to not unduly affect network or host system performance.

The latency and throughput of a Network IPS (NIPS) or Attack Mitigation device must be on a par with other equipment in the network on which it is deployed, and in this respect, an in-line NIPS must strive to perform much more like a switch than a typical passive security device, especially when it is necessary to install more than one NIPS in the same data path.

Detection/Blocking Performance Under Load

This group of tests verifies that the IPS does not adversely impact legitimate traffic, even when new TCP connections are being created rapidly. We also verify that the sensor is capable of detecting and blocking exploits when subjected to increasing loads of background traffic up to the maximum bandwidth supported as claimed by the vendor. An IPS that misses attacks under load can be evaded. An IPS that adversely affects legitimate background traffic will not stay in-line for long.

A fixed number of exploits are launched with zero background traffic to ensure the sensor is capable of detecting our baseline attacks. Once that has been established, increasing levels of varying types of background traffic are generated **through** the IPS device in order to determine the point at which the sensor begins to miss attacks.

All tests are repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic (or up to the maximum rated throughput of the device in 25 per cent increments should this be less than 1Gbps). The test is conducted with UDP, HTTP, and mixed-protocol traffic and includes packet rates up to 453,000 packets per second and connection rates up to 20,000 connections per second.

Latency & User Response Times

In any network environment latency is important. Latency may impose an upper bound on throughput and it also has an impact on interactive applications, thus affecting user response time. As such, it is important to understand the impact of latency introduced by a NIPS and to determine the maximum acceptable delay, which will be different for each network.

There is a direct relationship between latency introduced by a networking device and the maximum throughput allowed by that device on a single TCP connection. There is a critical value for the *round trip time* (RTT) of a packet in each network, and if the latency is below this critical value, TCP throughput will be unaffected - instead, it is the line speed of the underlying network which becomes the bottleneck. Above this critical value, however, TCP throughput is negatively impacted. To be specific, the maximum throughput achievable for any given TCP connection in a zero loss network is expressed as:

throughput = window / RTT

where *window* is the maximum TCP window size (64 Kbytes by default) and RTT is the round trip time in the network.

This equation tells us that the throughput of a TCP connection is inversely proportional to network latency (note that this is TCP throughput for *one* connection - the aggregate bandwidth is not affected by latency). In other words, if you double latency, you halve throughput.

Consider adding a NIPS in an internal Gigabit network where the RTT is 200 microseconds. The critical value for RTT in a Gigabit network is 500 microseconds (below which it may no longer be possible to achieve 1Gbps of throughput), which means the NIPS can add a maximum of 300 microseconds to the RTT without affecting the network. In this particular case, therefore, for an internal, high speed deployment, the administrator may determine that his chosen IPS device needs to be capable of sub-300 microsecond latency under normal traffic loads.

Of course, the latency of an IPS device may vary significantly based on packet size, complexity of the protocol, presence of attack traffic, or simply the makeup of the normal traffic passing through it. For example, Gigabit segments, will rarely carry only a single TCP connection. Rather, a saturated Gigabit segment could be supporting hundreds, if not thousands of TCP connections, and this multiplexing eases the impact of latency on the overall throughput on the segment.

Although each of these connections carries only a fraction of the total throughput, a few connections tend to dominate. The maximum latency for a NIPS is then determined by the utilisation of the fastest connection. For example, in a Gigabit Ethernet segment carrying 10,000 TCP connections the fastest connection might have a throughput of 250Mbps. In this case, the critical value for round trip latency is as high as 2 milliseconds.

Assuming the latency without the NIPS is 300 microseconds, an administrator may therefore determine that his chosen NIPS device must be capable of 1700 microsecond round trip latency (850 microseconds in each direction).

Such critical value calculations are important when TCP connections achieve maximum throughput, which is true for large data transfers. For smaller data transfers, and non-TCP applications like NFS, latency has a more direct impact on user experience - response time is directly proportional to latency. That is, *doubling latency doubles response time.* In these situations, the latency of the network in which a NIPS is deployed determines the acceptable latency of the NIPS.

Consider deploying a hypothetical NIPS with 1 millisecond one-way latency in the following scenarios:

- In internal corporate LANs, the round trip latency could be in the 200-300 microsecond range. Deploying our hypothetical NIPS would increase the maximum round trip latency to 2.3 milliseconds, an increase of just over 700 per cent. The time to copy a large group of files, for example, would increase by a factor of seven.
- In inter-campus corporate networks connected over a MAN, the latency could be in the 500-1000 microsecond range (or less). Deploying our hypothetical NIPS would increase the maximum round trip latency to 3 milliseconds, a minimum increase of 300 per cent. The time to copy a large group of files, for example, would increase by at least factor of three.

- Internet facing connections experience round-trip latency from 10-100 milliseconds. Deploying our hypothetical NIPS would increase the round trip latency by 1-10 per cent, which would have only a minor impact on the user experience.

The latency of the NIPS must therefore be evaluated in the context of the network in which it is deployed. For example, to protect networks that are accessed over the public Internet, one-way NIPS latencies in the 1-2 millisecond range would be acceptable. Whereas for NIPS deployments on MAN/WAN links, NIPS latencies of well under 1 millisecond would be essential. And as we have already mentioned, for deployments on internal networks where latencies are a few hundred microseconds, NIPS latencies of less than 300 microseconds would be more appropriate.

Network administrators have laboured long and hard to reduce latency within the corporate network to an absolute minimum. Core network devices such as switches are frequently chosen as much on their performance - packet loss and latency under all load conditions - as any other feature. Given that Network IPS devices are operating in-line, it is not surprising that they will be evaluated in a similar way.

For this reason, part of The NSS Group methodology uses very similar testing techniques to those we would normally employ when testing switches (in order to determine *packet latency*), in **addition** to measuring *application latency*. This group of tests determine the effect the IPS sensor has on the traffic passing through it under various load conditions. High packet latency will lower TCP throughput. High application latency will create a negative user experience.

Bi-directional network latency of a range of differently-sized UDP packets is measured under three test conditions: with no load, with 500 Mbps of HTTP traffic (or half the rated load of the device if this is less than 1Gbps), and while the device is under a heavy SYN flood attack (up to 10 per cent of the rated throughput of the sensor).

Spirent Avalanche and Reflector devices are also used to generate HTTP sessions through the device in order to gauge how any increases in latency will impact the user experience in terms of failed connections and increased Web response times. This "*application latency*" is measured both with no background load and while the device is under attack.

Stability & Reliability

These tests verify the stability of the IPS device under various extreme conditions. Long-term stability is critical for an in-line IPS device, where failure can produce network outages.

In the first part of this test, we expose the external interface of the sensor to a constant stream of attacks over an extended period of time. The device is configured to block and alert, and thus this test provides an indication the effectiveness of both the blocking and alert handling mechanisms. A continuous stream of exploits mixed with some legitimate sessions is transmitted through the sensor at a maximum rate of 90 per cent of the claimed throughput of the device for eight hours with no additional background traffic.

The device is expected to remain operational and stable throughout this test, blocking 100 per cent of recognisable exploits, raising an alert for each, and passing 100 per cent of legitimate traffic. If any recognisable exploits are passed - caused by either the volume of traffic or the IPS device failing open for any reason - this will result in a FAIL. If any legitimate traffic is blocked - caused by either the volume of traffic or the IPS device failing closed for any reason - this will also result in a FAIL.

In the second part of the test we stress the protocol stack of the device under test by exposing it to malformed traffic from the ISIC test tool for eight hours. The device is expected to remain operational and capable of detecting and blocking exploits throughout the test to attain a PASS.

We scan the management interface for open ports and active services and report on known vulnerabilities. We also stress the protocol stack of the management interface of the NIPS by exposing it to malformed traffic from the ISIC test tool. The device is expected to remain (a) operational and capable of detecting and blocking exploits, and (b) capable of communicating in both directions with the management server/console throughout the test to attain a PASS. We also note whether the sensor detects the ISIC attacks even though targeted at the management port.

Security Effectiveness

Detection Accuracy & Breadth

This group of tests verifies that the NIPS will not block legitimate traffic (*Accuracy*) and is capable of detecting and blocking a wide range of common exploits (*Breadth*). Although *breadth* is extremely important, *accuracy* is critical because a NIPS that blocks legitimate traffic will not remain in-line for long.

We have a number of trace files of normal traffic with “suspicious” content, together with several “neutered” exploits that have been rendered completely ineffective. The IPS attains a “PASS” for each test case if it does **not** raise an alert and does **not** block the traffic. Whilst it is not possible to validate completely the entire signature set of any IPS, this test demonstrates how accurately the IPS detects and blocks a wide range of common exploits, port scans, and Denial of Service attempts.

This test is repeated twice: the first run with blocking disabled on the IPS in order to determine which attacks are detected and how accurately they are detected (*Attack Recognition Rating*); the second run with blocking enabled in order to determine which attacks are blocked successfully regardless of how they are detected or what alerts are raised (*Attack Blocking Rating*).

Following the initial test run, each vendor is provided with a list of CVE references of the attacks missed and is allowed 48 hours to produce an updated signature set. This updated signature set must be released to the general public as a standard signature/product update before the report is published - this ensures that vendors do not attempt to code signatures just for this test.

Naturally, Rate-Based IPS devices will not respond to the same attack traffic as Content-Based devices, and so for those the Detection Accuracy tests involve detecting and mitigating a wide range of rate-based attacks such as port scans, SYN floods, connection floods, and so on.

We note which of these are mitigated completely, which are mitigated partially, and which require the use of built-in firewall capabilities.

Resistance To Evasion Techniques

These tests verify that the IPS is capable of detecting and blocking basic exploits when subjected to varying common evasion techniques. An IPS that cannot detect attacks subjected to these “script kiddie” evasion techniques is easily bypassed.

The tests consist of four parts (only the third is applicable to Rate-Based devices):

- ***Baselines -*** *This establishes that the IPS is capable of detecting and blocking a number of common basic attacks (our baseline suite) in their normal state, with no evasion techniques applied.*
- ***Packet Fragmentation and Stream Segmentation -*** *The baseline HTTP attacks are repeated, running them through fragroute using 19 evasion techniques.*
- ***URL Obfuscation -*** *The baseline HTTP attacks are repeated, this time applying 9 URL obfuscation techniques made popular by the Whisker Web server vulnerability scanner.*
- ***Miscellaneous Evasion Techniques -*** *Certain baseline attacks are repeated, and are subjected to 7 protocol- or exploit-specific evasion techniques, including altering default ports, inserting spaces in FTP command lines, inserting non-text Telnet opcodes in FTP data streams, and RPC record fragging.*

For each of the evasion techniques, we note if (i) the attempted attack is blocked successfully (the primary aim of any IPS device), (ii) the attempted attack is detected and an alert raised in **any** form, and (iii) if the exploit is successfully “decoded” to provide an accurate alert relating to the original exploit, rather than alerting purely on anomalous traffic detected as a result of the evasion technique itself.

Stateful Operation

If the IPS is tracking TCP session state, then it has the potential to introduce denial of service when the session table becomes full (too many connections) or if it can’t keep up with the creation of new sessions (too many connections per second). As with latency and bandwidth, the number of connections supported by the IPS and its connection per second rate should be matched to the network.

For example, a fully saturated Gigabit Ethernet link can handle 22,000 5KByte transfers per second. Assuming each connection lasts 20 seconds, the IPS should be able to handle 448,000 simultaneous connections. These numbers scale proportionately for slower networks. Any IPS that doesn’t offer these capabilities will impact performance of Web or e-commerce servers.

The aim of this section is to be able to determine whether the IPS is capable of monitoring stateful sessions established through the device at various traffic loads without either losing state or incorrectly inferring state.

An IPS that does not maintain TCP session state can flood the management console with false-positive alerts. Although this should not directly impact the IPS blocking function, it can make it very hard to perform forensic analysis of the attacks. In addition, if the default condition of the sensor is to block all traffic for which it does not believe there is a current connection in place, then an inability to maintain state under extreme conditions could result in the sensor blocking legitimate traffic by mistake.

In the first part of this test, we transmit a number of packets taken from capture files of valid exploits, but without first establishing a valid session with the target server. In order to receive a “PASS” in this test, no alerts should be raised for any of the actual exploits. However, each packet should be blocked if possible since it represents a “broken” or “incomplete” session.

In part two, we test whether the sensor is capable of preserving state across increasing numbers of open connections, as well as continuing to detect and block new exploits while not blocking legitimate traffic when the state tables are filled. Various numbers of TCP sessions from 10,000 to 1,000,000 (one million) are tested.

This test is run in both the out-of-box configuration and then repeated after applying any tuning recommended by the vendor (if applicable) to increase the size of the state tables.

Usability

After quantitatively evaluating the network performance and security effectiveness of the IPS, we qualitatively evaluate the features and usability of the product.

This evaluation provides the reader with valuable insight into product features, how easy it is to install the IPS and perform common, day-to-day operations with the management console. Areas evaluated include *installation, configuration, policy editing, alert handling,* and *reporting and analysis*.

FORTINET FORTIGATE-800

Executive Summary

Fortinet's FortiGate series of ASIC-accelerated Antivirus Firewalls are real-time network protection systems designed to detect and eliminate the most damaging, content-based threats from e-mail and Web traffic such as viruses, worms, intrusions, spam and inappropriate Web content. In addition to providing application level protection, the FortiGate systems deliver a range of network-level services, including firewall, VPN, intrusion detection/prevention and traffic shaping.

A range of models are available covering all sizes of installation from SOHO to service providers. All FortiGate Antivirus Firewalls employ the FortiASIC content processing chip and the FortiOS operating system.

The FortiGate-800 Antivirus Firewall features four 10/100/1000Mbps Ethernet ports for networks running at Gigabit speeds, and four user-definable 10/100Mbps ports that provide granular security through multi-zone capabilities, allowing administrators to segment their network into zones and create policies between zones.

The FortiGate-800 firewall is rated at 600Mbps, but Fortinet conservatively rates the device at 400Mbps once IDS/IPS protection module is enabled. At this bandwidth rating, the FortiGate-800 has horsepower to spare, and couples outstanding performance with very low latency when under load.

We also found the FortiGate-800 to be to be very stable and reliable, coping with our extensive reliability tests with ease and without succumbing to common evasion techniques.

Out of the box, policy management, alert management and reporting is extremely limited, and so we would recommend the use of Fortinet's own or third-party management and reporting tools to get the most from this product.

Architecture

The Fortinet appliance-based IPS offering consists of the following components:

FortiGate-800 Appliance

The FortiGate-800 is a 1U rack mount appliance based on Fortinet's FortiASIC Content Processor chip. The appliance contains a single Pentium M processor and 1 GB of RAM, a specification designed to accommodate the rated bandwidth of the appliance (600Mbps for firewall, and 400Mbps for IDS/IPS).

The range of FortiGate appliances (various capacities to handle SOHO to enterprise/carrier environments) are intended to deliver complete, real-time network protection services at the network edge by incorporating the following services in a single device:

- *Firewall*
- *Anti Virus*

- *VPN*
- *IDS/IPS*
- *Web content filtering*
- *Anti Spam*
- *Traffic shaping*

The FortiGate-800 features four 10/100/1000Mbps copper Ethernet ports for networks running at Gigabit speeds (labelled *Internal, External, DMZ* and *HA*), and four user-definable 10/100Mbps copper ports. It is not necessary to stick to the designations on these ports, since their function is not fixed - thus it is possible to connect the *HA* port to another subnet on the network if High Availability is not required. Note also that since the FortiGate is effectively a firewall, unlike other dedicated IPS devices it is not necessary to configure the in-line ports in pairs.

If required, *Internal* and *External* can be one in-line port pair, and *DMZ* and *HA* could be another. However, it is equally possible to have the *External*, *DMZ* and *HA* ports all act as “external” networks, all feeding traffic to the single *Internal* port, and each protected by a different security policy. This is a very flexible implementation, and multiple ports provide granular security through multi-zone capabilities, allowing administrators to segment their network into zones and create different policies between those zones.

Up to 20,000 security policies and 256 schedules can be created, providing incredible flexibility for the administrator to tailor policies to individual networks, VLANS, ranges of IP addresses, and even individual hosts - and all varied by the time of day or day of the week. Note that it is impossible to apply a different **IPS** policy to different networks - the same global IPS configuration applies to **every** policy where IPS has been enabled. The level of granularity within policies is such that it is possible to enable/disable (and provide some high-level configuration for) each of the separate security services only (see list above for services which can be activated).

The High-Availability (HA) port allows two or more FortiGate-800 appliances to be configured in stateful clusters for improved scalability and uptime. All Fortinet devices support both *active-active* and *active-passive* failover modes, and failover is fully stateful. When all devices in a cluster are operational, performance is enhanced via load-sharing.

For those environments where uptime is critical, we would recommend using an HA configuration since a single FortiGate appliance will fail closed, and no bypass mode is available. Although larger appliances in the range also offer multiple hot-swap cooling fans and power supplies, the FortiGate-800 has fixed fans and single, fixed power supply.

All the ports are mounted on the front panel of the appliance, along with a useful LCD display with control buttons. This can be used for limited management and monitoring functions without the need to attach a separate screen and keyboard.

Note that, unusually, there is no dedicated management port on the FortiGate appliances. Instead, it is necessary to either manage over one of the active ports, or dedicate one port to management traffic only and firewall it accordingly. Normally, we would prefer to see a dedicated management port on the front panel with all the necessary physical segmentation and protection already implemented, though Fortinet’s solution does provide an extra port for monitoring traffic, of course.

Virus and attack updates can be downloaded automatically via a schedule set on the device. The appliance can also be configured to accept a “push” packet from the central Fortinet update server, which will prompt it to override the schedule and contact the server for an immediate update.

Command and control of the device can be provided via a Web-based GUI, or via an extremely powerful Command Line Interface (CLI) accessed via a direct serial connection, Telnet or SSH. The latter provides the means not only to perform simple management tasks, but also to perform more extensive global operations which are simply not available via the browser interface. For those who prefer the Cisco-like CLI, complete control of the device can be effected in this way.

Web Manager

HTTP and HTTPS access is provided to the FortiGate appliance in order to use the browser-based configuration and management utility. Web Manager provides an intuitive means to configure the majority of the parameters used to manage the FortiGate-800.

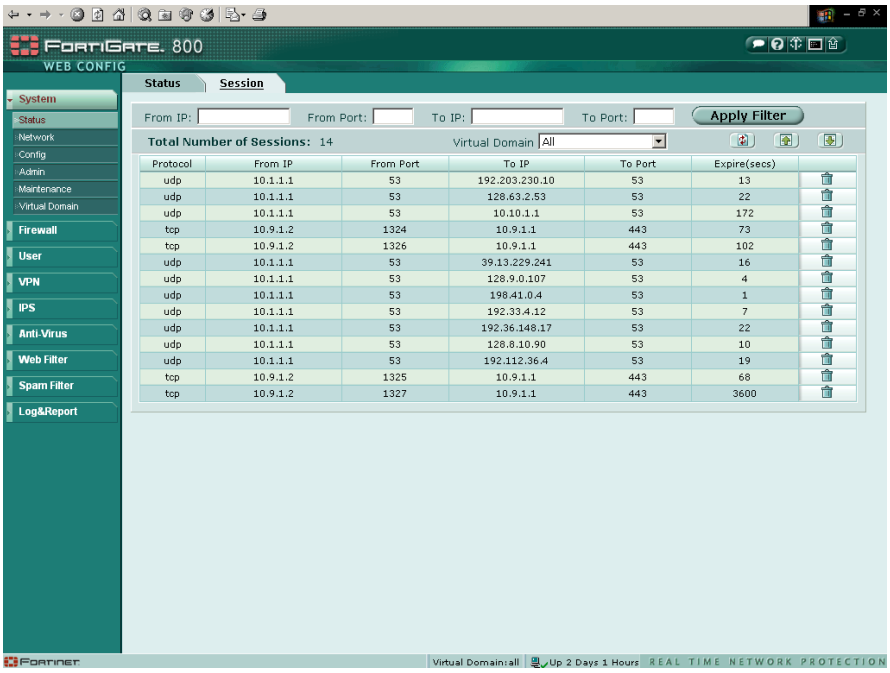

Figure 1 - FortiGate: Web Manager interface

A simple two-tier architecture is employed, and policies are stored directly on the appliance - there is no management server provided by default with the FortiGate appliances. This means that by default, the Web GUI is intended to manage single devices, or small numbers of devices. Organisations with larger deployments need to consider purchasing the optional FortiManager product.

FortiManager

The *FortiManager* system is an integrated management and monitoring tool that enables enterprises and service providers to manage large numbers of FortiGate appliances.

It minimises the administrative effort required to deploy, configure, monitor, and maintain the range of network protection services provided by FortiGate devices, simplifying the maintenance of security policies across multiple, dispersed FortiGate installations.

The robust, hardened FortiManager Server platform centralises configuration and monitoring of all FortiGate network protection functions, providing a central point for logging events and monitoring system status, traffic and threat activity.

FortiManager helps ensure consistent definition and application of polices across the full line of FortiGate Antivirus Firewalls, and provides a central point for monitoring system status and logging traffic and threat activity. Deployed as a multi-tiered system, which includes the FortiManager Server appliance and the Java-based FortiManager Console, the FortiManager System scales to support multiple system administrators and hundreds of FortiGate units.

Role-based administration capabilities make the FortiManager System suitable for large enterprises and for service providers offering managed security services. Different administrators can be restricted to specific management domains and specific functions. In addition, device grouping enables collections of FortiGate units to be aggregated into independent management domains to control administrative access and simplify policy deployment.

Logging & Reporting

The *FortiLog* family of turnkey logging and reporting appliances and the *FortiReporter Security Analyser* software package are dedicated solutions that aggregate and analyse log data securely from multiple FortiGate appliances.

The systems provide network administrators with a comprehensive view of network usage and security information, supporting the needs of enterprises and service providers responsible for discovering and addressing vulnerabilities across dispersed FortiGate installations. They minimize the effort required to monitor and maintain acceptable use policies, to identify attack patterns and attackers, and to comply with governmental regulations regarding privacy and disclosure of security breaches. They accept and process a full range of log records provided by FortiGate devices, including traffic, event, virus, attack, content filtering, and email filtering data.

FortiLog

The *FortiLog* family includes the FortiLog-100, 400, and 800 appliances which provide varying levels of storage and performance to meet a range of needs. Log records are transmitted from FortiGate units to FortiLog systems using encrypted VPN tunnels to ensure security. Capacities reach up to 360GB of log data and RAID levels (0, 1, and 5) can be selected to support desired trade-offs between capacity and data assurance. Built-in log analysis provides a central point for consistent analysis of network utilisation, web activity and attack activity across multiple FortiGate systems.

FortiReporter

FortiReporter Security Analyser is a cost-effective, browser-based, analysis, reporting and monitoring solution that generates reports across all FortiGate platform functionalities and provides IT administrators and security professionals with insight into network usage and attack activities.

FortiReporter Security Analyser is a software-only solution that provides comprehensive reports covering the full range of network and security activity, including virus and worm activity, bandwidth usage, network attacks, Web usage, and protocol usage. The FortiReporter system can collect and analyze data from all FortiGate models as well as from over 30 additional network and security devices from 3rd party vendors.

Performance

The aim of this section is to verify that the sensor is capable of detecting and blocking exploits when subjected to increasing loads of background traffic up to the maximum bandwidth supported as claimed by the vendor.

For each type of background traffic, we also determine the maximum load the IPS can sustain before it begins to drop packets/miss alerts. It is worth noting that devices which demonstrate 100 per cent blocking but less than 100 per cent detection in these tests will be prone to blocking **legitimate** traffic under similar loads.

The FortiGate-800 was tested up to 400Mbps, the rated speed of the appliance with the IPS module enabled (the firewall-only capability is rated at 600Mbps). Performance at all levels of our load tests was impeccable, with 100 per cent of all attacks being detected and blocked under all load conditions. We would thus have no hesitation in rating the FortiGate-800 as a true 400Mbps device as claimed by Fortinet - indeed, in a live network we feel this rating is conservative.

Basic latency figures were well within acceptable limits for a device of this type at all traffic loads and with all packet sizes, ranging from 249µs with 100Mbps of 256 byte packets, to 280µs with 400Mbps of 1000 byte packets (these figures are also well within those claimed by the vendor).

Strangely, once we placed the device under load using HTTP traffic, latency figures actually **improved**. When we loaded the FortiGate-800 with 200Mbps (half the rated speed of the device) of genuine HTTP traffic, latency ranged from 138µs with 256 byte packets to 198µs with 1000 byte packets. This is due to the use of NAPI drivers for the network cards, which switch from interrupt-driven to polling mode when under heavy load.

40Mbps of SYN flood traffic had a predictable effect, increasing latency to 188µs with 256 byte packets and to 216µs with 1000 byte packets (still **less** than when the device is under a pure UDP load). HTTP response times also increased only slightly during the SYN flood tests, from 214ms under normal load to 219ms with the SYN flood.

The FortiGate-800 performed consistently and completely reliably throughout our tests. Under eight hours of extended attack (comprising millions of exploits mixed with genuine traffic) it continued to block 100 per cent of attack traffic, whilst passing 100 per cent of legitimate traffic.

Exposing the sensor interface to an extended run of ISIC-generated traffic had no adverse effect, and the device continued to detect and block all other exploits throughout and following the ISIC attack.

Please refer to the *Testing Methodology* section for full details of the methodology used and performance results.

Security Effectiveness

We installed one sensor with the latest signature pack, and created an IPS Protection Profile with every attack signature and every anomaly detection method enabled. The SYN flood and UDP flood thresholds were adjusted accordingly to differentiate between our “normal” test loads and our attack traffic.

We then created two firewall policies (*Internal* to *External*, and *External* to *Internal*) which were set to permit **all** traffic in **both** directions, and applied the IPS Protection Profile to each. Only the IPS protection module was active - Anti Virus, Web Filtering, Web Category Filtering, Spam Filtering and Content Logging were all disabled. In other words, the FortiGate was configured as a typical dedicated IPS appliance.

Signature recognition (with blocking disabled) was acceptable out of the box, though at 75 per cent was low for a pure IPS device. The recognition rate was increased to 95 per cent after the application of a signature pack update which was provided to us in 48 hours. The quality of the new signatures seemed to be as high as the existing ones, and performance of the box was not affected at all by the large update.

Blocking performance was slightly higher than detection throughout the tests due to the fact that the firewall module silently drops certain packets (notably several of the DOS fragment packets) which means it is simply not possible for the IPS module to detect them. Blocking performance was increased from a creditable 80 per cent to a perfect 100 per cent following the application of the signature update.

We noted a minimum of “noise”, with very few test cases raising multiple alerts for a single exploit. Most of our “false negative” (modified exploit) cases were detected correctly out of the box, and were all detected accurately following the signature update.

A major concern in deploying an IPS is the blocking of legitimate traffic. We noted only one false positive alert from our test suite out of the box, and this was corrected following the signature update.

The FortiGate-800 arrives with a default policy configured with a number of signatures disabled (though the majority are enabled), and thus it would be necessary for the customer to tune the device for his own environment. The PASS and BLOCK actions seem to be set intelligently throughout by default to provide a set of “recommended settings”, and we would be reasonably confident in deploying the FortiGate-800 with the default policy.

Resistance to known evasion techniques was excellent, with the FortiGate-800 achieving almost a clean sweep across the board in our evasion tests.

Fragroute, Whisker, ADMmutate and even *RPC record fragging* all failed to trick FortiGate into ignoring valid attacks.

Note that not only were the fragmented and obfuscated attacks blocked successfully, but almost every one of them was decoded accurately as well.

Out of the box, the FortiGate-800 handled 1 million open connections easily (no tuning was necessary to complete all our stateful tests).

Please refer to the *Testing Methodology* section for full details of the methodology used and performance results.

Usability

This part of the test procedure consists of a subjective evaluation of the features and capabilities of the product, and covers *installation, configuration, policy editing, alert handling,* and *reporting and analysis*.

Installation

Installation of the FortiGate-800 is very straightforward, and can be accomplished via a simple Wizard under the browser-based GUI, via the Command Line Interface (CLI), or via the front panel-mounted control buttons and LCD. The first decision to be made is which operating mode should be configured for the device: *NAT/Route Mode* or *Transparent Mode*.

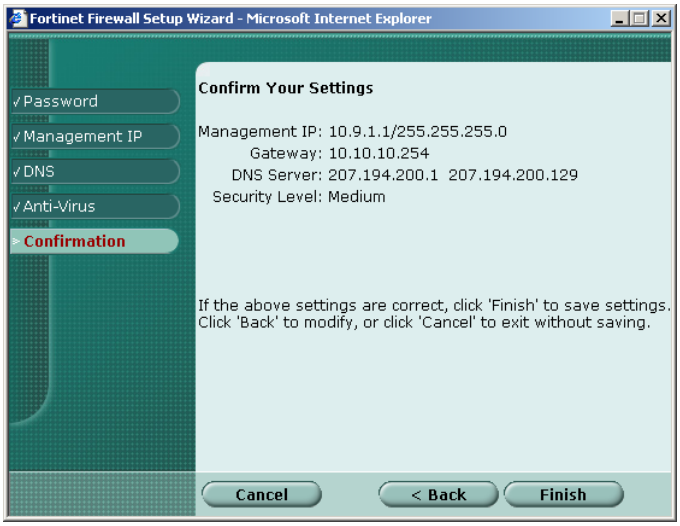

Figure 2 - FortiGate: Setup wizard

In *NAT/Route Mode*, the FortiGate-800 is visible to the network, since each of the internal and external interfaces are configured with valid IP addresses. Because of this, each of the interfaces are on different subnets, and the minimum configuration involves the *Internal* and *External* interfaces. Optionally, it is also possible to configure the DMZ and HA interfaces (Gigabit), and interfaces 1 to 4 (100Mbps).

NAT/Route mode is typically used when the FortiGate-800 is deployed as a gateway between private and public networks, and in its default NAT/Route mode configuration, the unit functions as a firewall. Users on the internal network can access the Internet while the FortiGate-800 blocks all other traffic. More complex security policies can then be defined to configure antivirus protection, content filtering, Network Intrusion Prevention (NIPS), and Virtual Private Networks (VPNs).

Security policies control whether communications through the FortiGate-800 operate in *NAT* mode or in *route* mode. In NAT mode, the FortiGate-800 performs *Network Address Translation* before IP packets are sent to the destination network. In route mode, no translation takes place. By default, the unit has a single NAT mode policy that allows users on the internal network to securely access and download content from the Internet. No other traffic is possible until the administrator configures additional security policies.

In *Transparent Mode*, the FortiGate-800 is invisible to the network, behaving as a simple “bump in the wire”. All of its interfaces are on the same subnet, and only a management IP address needs to be configured to allow configuration changes (note that management needs to be performed over one of the active interfaces, or one interface needs to be dedicated to management tasks and firewall policies implemented manually).

Transparent Mode is the typical configuration for dedicated IPS devices, and this is how we had the FortiGate-800 configured for our tests. With no IP addresses to configure, it is a simple matter to install a Transparent Mode device anywhere on a network with no reconfiguration of network addresses or routing tables on either side of it. As with NAT/Route Mode, in its default Transparent Mode configuration, the FortiGate functions as a firewall. It has a single security policy that allows users on the internal network to securely download content from the external network, and no other traffic is possible until the administrator has configured additional security policies.

In our tests we used only the *Internal* and *External* ports to connect the device in-line in our test rig, but up to eight network segments can be connected to the FortiGate unit to control traffic between these network segments (with management permitted on one or more of those segments). Even with a dedicated management port and one port used for HA connectivity, that still leaves six ports free.

There is an installation method to suit everyone - those who love GUIs, those who cannot manage without a Command Line Interface, and those who like pushing buttons on the front panel.

- **Web-based manager & Setup Wizard -** The FortiGate web-based manager Setup Wizard guides the administrator through the initial configuration steps. It is used to configure the administrator password, the internal, external and DMZ interface addresses, the default gateway address, and the DNS server addresses. Optionally, the Setup Wizard can be used to configure the internal server settings for NAT/Route mode.
- **Command Line Interface (CLI) -** The CLI is a full-featured management tool accessed via a serial cable connected to the front panel serial port (or via SSH/Telnet once the device has been installed). It can be used to configure the administrator password, the interface addresses, the default gateway address, and the DNS server addresses, and the excellent *Quick Start Guide* provides enough information to allow the administrator to have the device up and running in this mode. Beyond that, the device can be managed entirely via the CLI if required, and those who are used to the Cisco CLI will feel at home on the FortiGate. For those not so well versed, there is an extensive *CLI Reference Guide* available.

- **Control Buttons & LCD -** The control buttons and LCD are located on the front panel of the FortiGate-800. These can be used to configure the internal, external and DMZ interface addresses, and the default gateway address. To configure the other interface addresses, and the DNS server addresses, the Web-based manager or the CLI are required. The front panel buttons thus provide the ideal way to provide initial connectivity to the FortiGate appliance without having to adjust IP addresses on your management console PC.

During configuration, it is possible to define the services which are available on each interface for management purposes. Thus, it is possible to enable or disable HTTP/HTTPS for the Web-based manager, or Telnet/SSH for the CLI.

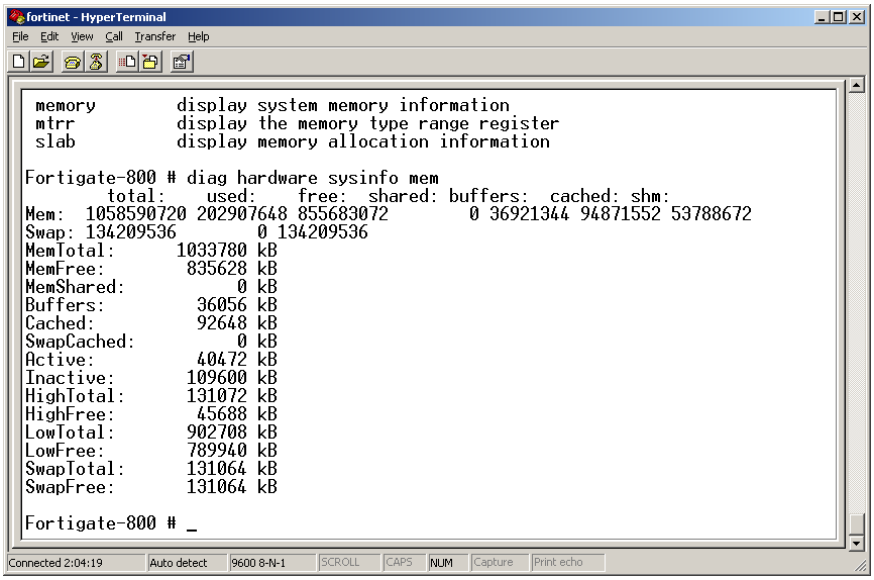

Figure 3 - FortiGate: CLI

A range of excellent documentation is available including the aforementioned *Quick Start Guide* and *CLI Reference Guide*, as well as a comprehensive *FortiGate-800 Installation & Configuration Guide*. Separate manuals are also available for *FortiManager, FortiLog, Content Protection, VPN* and *NIDS*, and all documentation is provided as PDF files only. Both the *NIDS* documentation and *the FortiGate-800 Installation & Configuration Guide* as supplied to us were actually out of date and slightly confusing in places, however, since they referred exclusively to the older NIDS functionality, and not at all to the newer NIPS functionality, which is slightly different when it comes to configuration.

Overall, the level of detail was generally fairly good, and we found coverage of all the main features to be reasonably comprehensive and very clear.

Configuration

Aside from the command line, the Web Manager is the main utility for managing and configuring the FortiGate-800. This communicates directly with a single FortiGate appliance, and all configuration and policy information is stored on the appliance. For this reason it obviously has a very device-centric view - it is impossible, for example, to group multiple devices together and apply a single security policy to the group.

This clearly does not scale well for management of multiple devices, but for those sites with a single FortiGate it is adequate. For sites with large numbers of Fortinet appliances, the optional (extra cost) FortiManager product would be essential.

In all other respects, we found the Web Manager to be intuitive, and very straightforward to use. Multiple administrators can be created, each with a unique Access Profile that restricts read and write access to individual system modules, such as *System Configuration, Log & Report, Security Policy*, and so on. Each administrator can also be restricted to using a specific host or range of hosts from which they can access the FortiGate appliance.

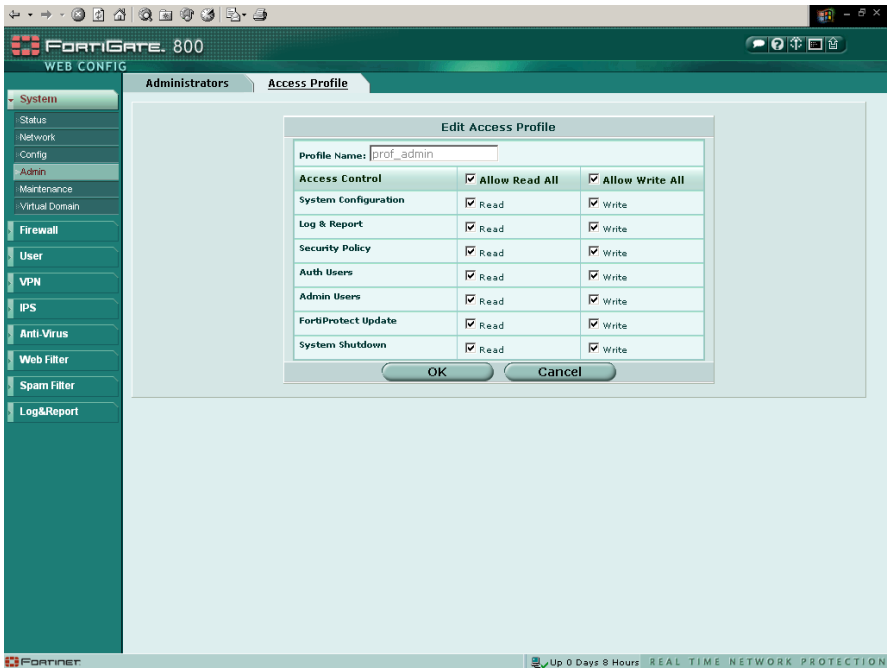

Figure 4 - FortiGate: Configuring User Access Profiles

This level of granularity is adequate for many environments, but we would like to see it extended to restricting access to the major security modules within the FortiGate appliance, making it possible to restrict one administrator to AV configuration, one to IPS configuration, one to Web Content Filtering administration, and so on. More importantly - especially in a managed services environment or a large corporate deployment where separate administrators are responsible for individual subnets - we would like to see the ability to control access to individual security policies, and individual subnets or port pairs.

Once logged in, the administrator is presented with a column of tabs down the left side of the screen covering the major components of the device:

- **System -** *including Status screen, network configuration, admin user management, maintenance operations and virtual domains*
- **Firewall -** *including policy creation, address ranges/groups, service definitions, schedules (different policies can be applied at different times of the day, or day of the week) and protection profiles (allowing configuration of the protection modules including Anti Virus, Web Filtering, Web Category Filtering, Spam Filtering, IPS, and Content Logging, and their application to firewall rules)*

- **User -** *to define local users and groups, as well as LDPA directories and RADIUS servers*
- **VPN -** *to define IPSec policies and manage certificates*
- **IPS -** *to manage signatures and anomalies*
- **Anti Virus -** *to configure file blocking and quarantine activity*
- **Web Filter -** *covering content blocking, URL blocking, URL exempt lists, Web category blocking and script filtering (Java applets, Cookies and ActiveX)*
- **Spam Filter -** *covering creation of IP black lists and white lists, RBL and ORDBL servers, e-mail address black lists and white lists, MIME header black lists and white lists, and banned word lists*
- **Log & Report -** *providing access to log configuration and basic log files in memory and on disk for Traffic Management, Events, Attacks, Anti Virus, Web Filter, Spam Filter and Content Filter.*

The *System Status* screen provides useful summary information on the current state of the appliance, including up-time, disk capacity, CPU usage, memory usage, active sessions, network utilisation, recent virus detections and recent attacks. It also lists the current firmware version, AV definitions and attack definitions, and provides the means to update them via a file on the local system. These can also be updated automatically via the *FortiProtect Distribution Network*, and the appliance can be configured to check automatically at scheduled intervals, or accept “push updates”. With the latter, the FortiProtect server sends a push packet to cause the appliance to connect and request an immediate update - thus although the initial “push” comes from the FortiProtect server, it is still the FortiGate appliance which actually establishes the secure outbound connection.

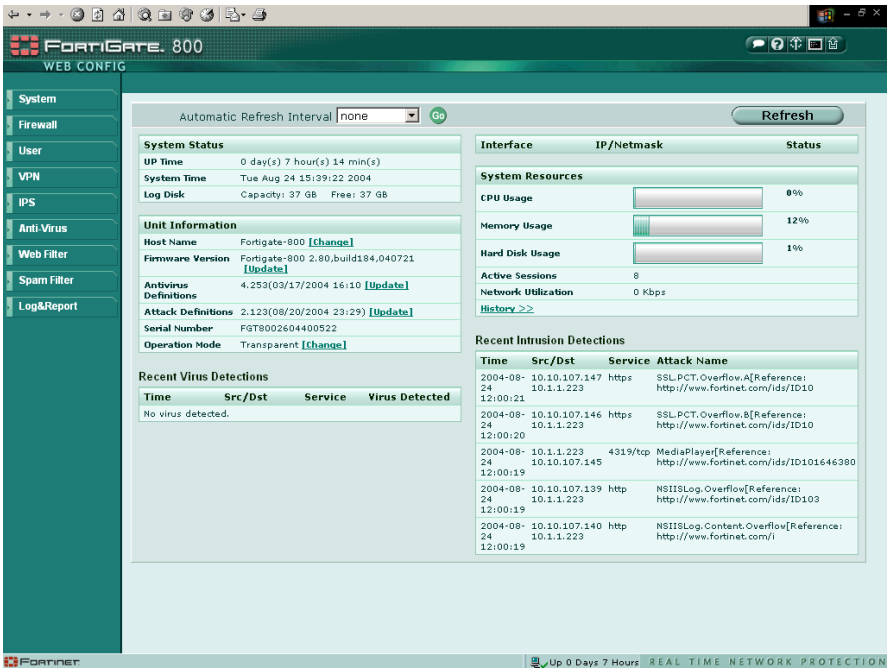

Figure 5 - FortiGate: System Status screen

Also available on the System Maintenance tab is the *Backup & Restore* option. This provides the means for the administrator to backup and restore key configuration files, such as black list, white lists, address lists, certificates and the entire system configuration.

With regard to the remaining options, for the purposes of this test, we will concentrate purely on the IPS capabilities, and the firewall functionality wherever it pertains directly to IPS.

For anyone used to configuring firewalls, the FortiGate-800 will be familiar territory. For those who are used to configuring dedicated IPS devices, however, there are one or two conventions that are challenged.

This is neither a "*good thing*" nor a "*bad thing*" - it is simply the way the device operates. Most IPS products these days have some form of firewall at the lowest level in the stack in order to drop packets quickly that match certain conditions (i.e. once a session has been marked as bad, packets from that session can be dropped by the firewall without having to be analysed further). The FortiGate is simply more "up front" about the firewall capability (it is a firewall first and foremost), and simply assigns IPS protection policies to individual firewall rules.

Thus, the first task an administrator needs to perform is to *create Firewall Policies* between the *Internal* and *External* ports, allowing traffic to pass in both directions (if required). At this point, of course, FortiGate has the advantage over most dedicated IPS devices, since it can permit or deny traffic as a firewall before it even reaches the IPS detection engine, thus potentially improving performance. For example, traffic could be permitted only from or to a specific range of IP addresses. Or specific protocols (Telnet, for example) could be denied throughout the network. Or traffic could be permitted only during working hours and denied altogether at weekends. All of this is under the control of the firewall policies before traffic is even passed to the IPS engine.

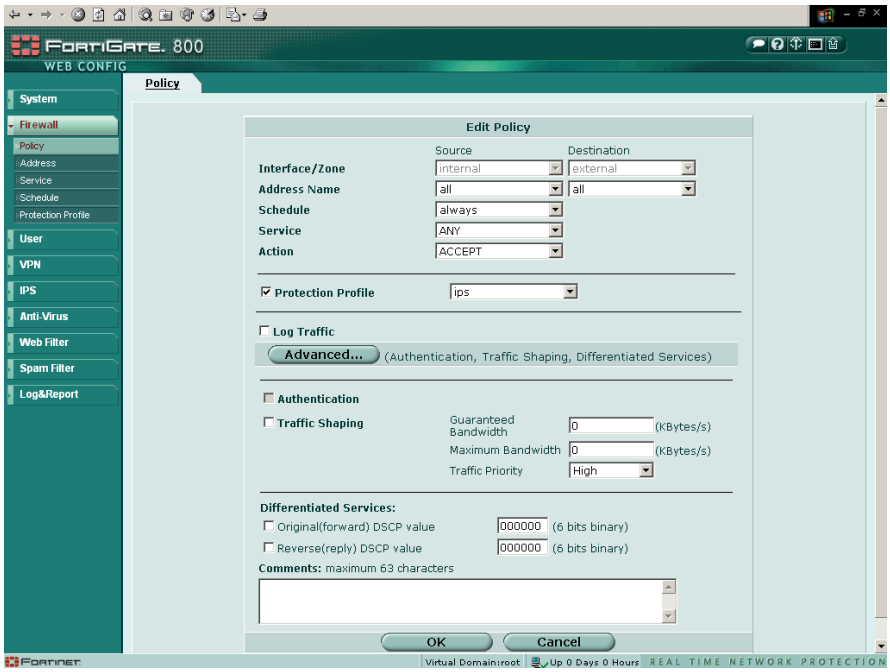

Figure 6 - FortiGate: Defining Firewall Policies

The administrator then creates one or more *Protection Profiles* in order to determine which of the protection modules (Anti Virus, Web Filtering, Web Category Filtering, Spam Filtering, IPS and Content Logging) are active, and which are disabled, and how each active module is configured.

For example, in the AV module the administrator gets to choose which protocols are scanned for viruses, which protocols have file blocking enabled, and which can be quarantined. The IPS module has only two check boxes - to enable/disable *IPS Signatures* and to enable/*disable IPS Anomalies.*

Within the firewall configuration, therefore, we are simply defining **which** of the protection modules are to be applied to **which** traffic streams on **which** physical/logical interfaces. Once those have been applied (Fortinet claims a maximum of 20,000 policies can be supported, and 256 schedules), the main tabs along the left side of the screen described earlier are used to configure those protection modules in detail, and that will be covered in the following *Policy Management* section.

In Transparent mode, The FortiGate device can apply firewall policies and protection services, such as virus scanning or IPS, to traffic on an IEEE 802.1 VLAN trunk. The FortiGate unit operating in Transparent mode can be inserted into the trunk without making changes to the network. In a typical configuration, the *Internal* interface accepts VLAN packets on a VLAN trunk from a VLAN switch or router connected to internal VLANs, whilst the *External* interface forwards tagged packets through the trunk to an external VLAN switch or router. This external switch or router could be connected to the Internet. The FortiGate unit can be configured to apply different policies for traffic on each VLAN in the trunk.

FortiGate also supports the concept of *Zones*, which are virtual groupings of ports and/or VLAN sub-interfaces designed to simplify firewall policy creation. For example, it is possible to group together three ports into a single logical zone - this would mean that only a single firewall rule would be needed to control traffic for that zone, instead of one for each interface.

Policy Management

Although the term “policy” does exist in FortiGate terminology, it does not, as you might expect, apply to the overall collection of configuration parameters which can be saved and deployed to an appliance on a typical dedicated IPS device.

When using the Web Manager, the “policy” which controls the device to which the Web Manager is attached (bear in mind it can only access one device at a time) is stored on the FortiGate appliance itself, and is generally referred to as the “system configuration”. Therefore, the only way to save one particular “policy” for recall at a later date is to backup the entire system configuration, and restore it as required (this forces a system reboot, however, during which the device - and thus the networks behind it - are unavailable).

As mentioned in the previous section, activating the IPS section of the policy is simply a matter of checking the *IPS Signature* and *IPS Anomaly* boxes in the *Protection Profile*, and then assigning the Protection Profile to one or more *Firewall Policies*. Having done that, the FortiGate will immediately begin protecting the network traffic covered by the Firewall Policies.

In our case, it was set to protect all traffic, but it could easily be restricted to individual subnets, or even individual hosts.

Unfortunately, the detailed IPS configuration (i.e. which signatures are enabled/disabled, whether the action is to block or pass traffic, whether alerts should be logged or not, etc.) is global across **all** Firewall Policies, and it is only possible to globally enable or disable **all** signatures and/or **all** anomalies at the Firewall Policy level. It would be much more flexible and powerful if it were possible to create multiple separate IPS policies and assign those to separate Firewall Policies. That would provide a "virtual IDS" capability, with different protection policies applied to different address ranges, hosts, physical ports, logical port groups or VLANs. Maybe in a future release?

The default IPS configuration out of the box implements recommended settings for all of the signatures and anomalies. Those for whom there is little chance of raising false positive alerts are set to "block" traffic, the rest are set to "alert" only. Most are set to create log events when an alert is raised, although some of the "noisier" flood-related signatures are set to drop traffic silently, with no logging.

Figure 7 - FortiGate: Configuring Anomalies

For most networks, the Anomaly thresholds are probably set to sensible levels. We were required to do some tuning to handle the extremely high connection setup rates imposed by our Spirent Avalanche test equipment, but this was easily accomplished by setting the anomalies to "PASS" traffic, running the Avalanche tests, and noting the peak thresholds reached as reported in the logs. Having set the thresholds accordingly (only four signatures were affected) we then returned the Anomaly settings to their "RECOMMENDED" settings and we were ready to go.

Configuring the IPS settings in detail is accomplished via two tabs - *Anomalies* and *Signatures*.

Whereas the *Anomalies* consist mainly of bad traffic (fragmented packet attacks, Ping of Death, and so on) and rate-based exploits (SYN Floods, UDP Floods, etc), the *Signatures* consist of application-level exploits - there are over 1300 in total in the current signature pack.

They are grouped together to identify their target (backdoors, Apache, Finger, FTP, IIS, and so on), and each group can be expanded to display the signatures within. It is easy to see which are enabled via the bright green tick marks against each one, and the same icon is used to highlight those with logging enabled. Also against each signature is its revision number, and the current action, which can be chosen from the following list:

- ***Pass** - pass the packet*
- ***Drop** - drop the packet*
- ***Reset** - attempt to send TCP RST to client and server*
- ***Reset Client** - attempt to send TCP RST to client*
- ***Reset Server** - attempt to send TCP RST to server*
- ***Drop Session** - drop the packet and mark the rest of the session as "bad"*
- ***Pass Session** - pass the packet and mark the rest of the session as "good"*
- ***Clear Session** - drop the packet, mark the rest of the session as "bad", and attempt to send TCP RST to client and server*

Changing any of the default signature settings causes a green button to appear alongside the changed signature on-screen. Clicking on this button causes all of the settings to revert to the recommended settings originally applied by default - a nice touch.

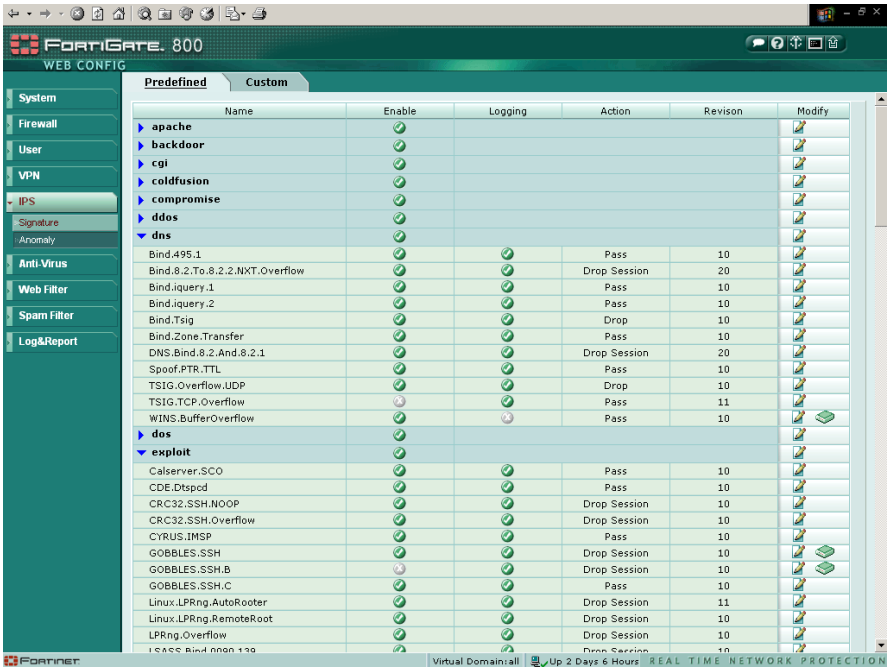

Figure 8 - FortiGate: Configuring Signatures

In spite of the signature groupings, it is extremely difficult to locate specific signatures for maintenance operations. If one signature is cropping up too often in your logs, for example, you have to know exactly in which group it is located, otherwise a long trawl through 1300 entries is the result. The unique ID displayed in the logs is not even displayed alongside the signatures in the Web Manager.

There really needs to be a search facility to make it easier to locate individual signatures or groups of signatures.

Nor is it possible via the GUI to make mass changes to groups of signatures. If you want to turn off blocking, for example, for all signatures, this can only be accomplished via the command line. Luckily, if you wish to reinstate all signatures to their default settings, this too is a single operation at the command line. However, should you wish to turn off blocking for FTP or IIS signatures only, this has to be performed signature by signature via the GUI. It *is* possible to enable and disable entire groups of signatures with a single check box, but this would disable them completely - not the same as “pass the traffic but log it for me”.

All in all, policy handling is extremely basic in the current release and is far more difficult than it should be when attempting to search and make mass changes to groups of signatures (a common enough operation for the average administrator). More flexibility is required here.

There is a potential performance impact when all signatures are enabled (though having all enabled in our tests did not adversely affect the rated performance of 400Mbps), so it makes sense to tune the IPS settings according to the expected traffic on your network - if you don’t have Apache servers, you could turn off the Apache signature group (as long as you are not interested in seeing *attempted* exploits), thus allowing you to squeeze more bandwidth out of the appliance.

Finally, it is possible to create custom signatures and apply them via the GUI. Each signature, however, must be created and entered longhand into the GUI, crafted from a complex (at least for lay-users) Snort-like signature language. As with most IPS/IDS products, custom signature creation is not for the fainthearted, but at least it is catered for with the FortiGate.

Alert Handling

The FortiGate device can be configured to log network activity from routine configuration changes and traffic sessions to emergency events. It is also possible to configure the FortiGate to send alert email messages to inform system administrators about events such as network attacks, virus incidents, and firewall and VPN events.

The administrator can configure the logs that he wants to record and the message categories that he requires in each log via the *Log Filter* tab. He can also configure the FortiGate unit to send alert email to up to three e-mail addresses when there are virus incidents, block incidents, network intrusions, and other firewall or VPN events or violations. It is possible to record logs to one or more of:

- *A computer running a syslog server*
- *A computer running a WebTrends firewall reporting server*
- *The FortiGate hard disk - the maximum size and rollover frequency can be set in the Log Config tab within the GUI. When a log file is rolled over, it can also be automatically uploaded to a remote FTP server if required.*
- *Memory (restricted to the last 128 events)*

Log entries can be displayed via the GUI, each entry containing the date and time, attack ID, source and destination IP address, source and destination port, source and destination interface, protocol and alert description. Unfortunately, in the current release this is all displayed on a single line in typical syslog fashion, and is thus extremely difficult to read.

At the very least this should be presented in a “spreadsheet” format, with sortable columns of data - we were informed that this is planned for a future release.

However, given the current cluttered and confusing presentation, that there is no way to re-arrange or sort the columns of data, that there is only a basic search facility, and that there is no detailed packet or context data to indicate why the event was raised, forensic analysis is virtually impossible with the software as it stands. However, it should be remembered that this criticism is made whilst evaluating the FortiGate as a dedicated IPS device. The fact remains that it is **not** a dedicated IPS device - it is a multi-function security appliance that includes far more than just IDS/IPS. In that context, IPS forensic analysis becomes less of a priority.

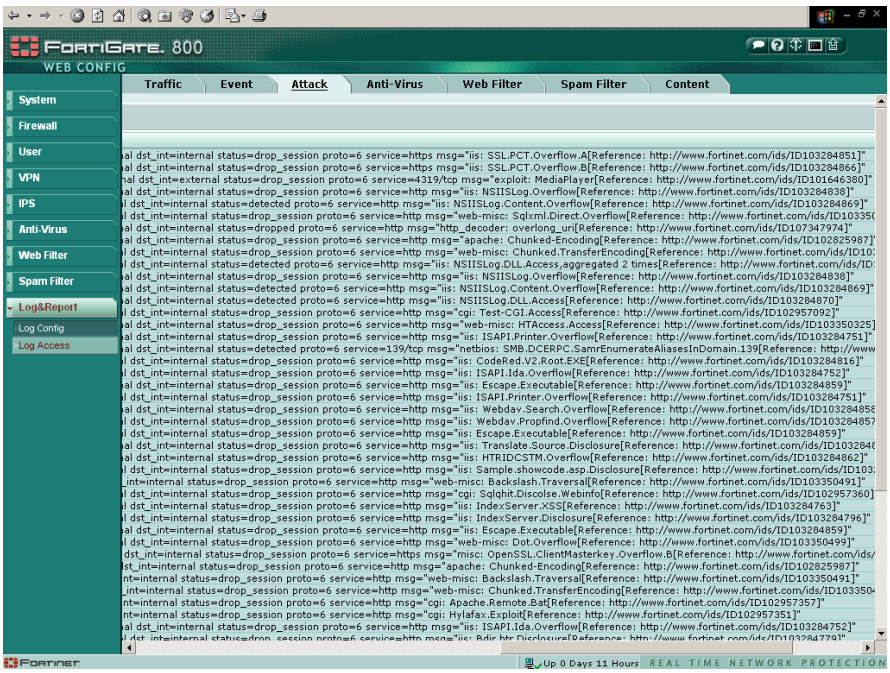

Figure 9 - FortiGate: Viewing Alerts

Whichever way you look at it, however, third party tools or the optional Fortinet reporting tools are a must.

Duplicate log events raised within a given time-frame are aggregated and reported as a single event (which includes a count of the total number of events raised) to reduce clutter in the logs.

Reporting and Analysis

There are no reporting or analysis tools built into the base product. Optional logging, reporting and analysis products are available at extra cost in the form of FortiLog and FortiReporter.

Verdict

Performance

The FortiGate-800 appears to have been over-engineered for its rated 400Mbps.

It proved itself capable of achieving at least that throughput in all of our extreme tests (and with all signatures enabled), and much higher in a real-world scenario when used as a dedicated IPS device (clearly performance may be affected if other security modules are activated). We would have no hesitation in rating the FortiGate-800 as a true 400Mbps IPS device.

Basic latency figures were well within acceptable limits for a device of this type at all traffic loads and with all packet sizes, ranging from 249µs with 100Mbps of 256 byte packets, to 280µs with 400Mbps of 1000 byte packets (these figures are also well within those claimed by the vendor).

Strangely, once we placed the device under load using HTTP traffic, latency figures actually **improved**. When we loaded the FortiGate-800 with 200Mbps (half the rated speed of the device) of genuine HTTP traffic, latency ranged from 138µs with 256 byte packets to 198µs with 1000 byte packets. This is due to the use of NAPI drivers for the network cards, which switch from interrupt-driven to polling mode when under heavy load.

40Mbps of SYN flood traffic had a predictable effect, increasing latency to 188µs with 256 byte packets and to 216µs with 1000 byte packets (still **less** than when the device is under a pure UDP load). HTTP response times also increased only slightly during the SYN flood tests, from 214ms under normal load to 219ms with the SYN flood.

These figures when under attack are excellent, and SYN Flood mitigation was handled well throughout the test, with the flood being completely mitigated once the initial threshold had been passed.

Overall, latency figures were considered to be very good for a device of this type.

The FortiGate-800 also performed consistently and completely reliably throughout our tests, continuing to block attack traffic in a consistent manner whilst passing 100 per cent of the legitimate traffic, even when under extended attack. Exposing the sensor interface to ISIC-generated traffic had no adverse effect, and the device continued to detect and block all other exploits throughout and following the ISIC attack.

Security Effectiveness

Signature recognition (with blocking disabled) was quite low out of the box (75 per cent), but was increased to 95 per cent after the application of a signature pack update which was provided to us in 48 hours.

Blocking performance was slightly higher than detection throughout the tests due to the fact that the firewall module silently drops certain packets which means it is simply not possible for the IPS module to detect them. Blocking performance was increased from a creditable 80 per cent to a perfect 100 per cent following the application of the signature update. This has to be recognised as a prodigious feat, increasing coverage from 80 per cent to perfect score in just 48 hours.

All of our “false negative” (modified exploit) cases were detected accurately following the signature update, and none of our false positive cases triggered once the update had been applied.

Resistance to known evasion techniques was excellent, with the FortiGate-800 achieving almost a clean sweep across the board in our evasion tests. Not only were the fragmented and obfuscated attacks blocked successfully, but almost all of them were decoded accurately as well.

Usability

The Web Manager included out of the box is an extremely basic browser-based GUI that has a long way to go before it could be described as an enterprise-class management solution for an IPS device. When considering the FortiGate-800 as a pure IPS device, the Web Manager is severely lacking in the areas of policy definition and deployment, searching, making mass changes to signatures, reporting and forensic analysis.

However, it should be remembered that the company does provide alternative offerings (albeit at additional cost) which should address most of our concerns in this area if policy definition and forensic analysis/reporting are important to you. For those customers wishing to deploy multiple appliances or with more advanced reporting and analysis requirements, these alternative offerings should be investigated.

If you have a single device to manage, however, and are more concerned with that device's ability to block attacks effectively rather than tell you exactly how it did it, then the basic embedded Web Manager will probably give you everything you need.

The other key point to bear in mind here, of course, is that the FortiGate-800 is **NOT** actually a dedicated IPS device. It is a multi-function security appliance that includes several other security modules, all of which need to be managed via the same interface. From a usability point of view, therefore, Fortinet has actually done a reasonable job in making all of those modules configurable via a single interface in a straightforward and intuitive way.

We found that with only occasional reference to the excellent on-line help, and no recourse to the extensive user guides at all, we were able to install, configure and manage the entire system with no problems - that is more than can be said of many of the products which visit our labs.

And whilst it may not be for everyone, the provision of a familiar and very comprehensive Command Line Interface allowed us to complete many of the tasks made too convoluted by the simplistic GUI.

The lack of pure IPS features - and related complexity - should certainly be balanced against the addition of the other security modules (firewall, anti virus, anti spam, content filtering, etc.) and multiple Gigabit/100Mbps ports at such a low cost, and the overall simplicity with which they can all be configured from a single interface.

Contact Details

Company name: Fortinet, Inc.

E-mail: sales@fortinet.com

Internet: www.fortinet.com

Address:
920 Stewart Drive,
Sunnyvale,
CA 94085
USA

Tel: +1 408 235 7700

Fax: +1 408 235 7737

APPENDIX A – TEST RESULTS

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

The Test Environment

The network is 100/1000Mbit Ethernet with CAT 5e cabling and Cisco 6500-Series switches (these have a mix of fibre and copper Gigabit interfaces). All devices are expected to be provided as appliances - if software-only, the supplier pre-installs the software on the recommended hardware platform. The sensor is configured as a perimeter device during testing (i.e. as if installed behind the main Internet gateway/firewall). There is no firewall protecting the target subnet.

Traffic generation equipment - such as the machines generating exploits, Spirent Avalanche and Spirent Smartbits *transmit* port - is connected to the “external” network, whilst the “receiving” equipment - such as the “target” hosts for the exploits, Spirent Reflector and Spirent Smartbits *receive* port - is connected to the internal network. The device under test is connected between two “gateway” switches - one at the edge of the external network, and one at the edge of the external network.

All “normal” network traffic, background load traffic and exploit traffic will therefore be transmitted **through** the device under test, from external to internal. 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 ensure that the total amount of traffic never exceeds 1Gbps (which would invalidate the test run).

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

Section 1 – Detection Engine

The aim of this section is to verify that the sensor is capable of detecting and blocking a wide range of common exploits accurately, whilst remaining resistant to false positives. All tests in this section are completed with **no background network load**. The latest signature pack is acquired from the vendor, and sensors are deployed with **all** available attack signatures enabled (some audit/informational signatures may be disabled).

Test 1.1 - Attack Recognition

Whilst it is not possible to validate completely the entire signature set of any sensor, this test attempts to demonstrate how accurately the sensor detects and blocks a wide range of common exploits, port scans, and Denial of Service attempts. These are updated/changed for every new test, and all exploits are run with no load on the network and no IP fragmentation.

Our attack suite contains over 100 basic exploits (plus variants) covering the following areas:

- ***Test 1.1.1 -*** *Backdoors (standard ports and random ports)*
- ***Test 1.1.2 -*** *DNS/WINS*
- ***Test 1.1.3 -*** *DOS*
- ***Test 1.1.4 -*** *False negatives (common exploits which have been modified to remove or alter obvious "triggers" - this ensures that the signatures are coded for the underlying vulnerability rather than a particular exploit)*
- ***Test 1.1.5 -*** *Finger*
- ***Test 1.1.6 -*** *FTP*
- ***Test 1.1.7 -*** *HTTP*
- ***Test 1.1.8 -*** *ICMP (including unsolicited ICMP response)*
- ***Test 1.1.9 -*** *Reconnaissance*
- ***Test 1.1.10 -*** *RPC*
- ***Test 1.1.11 -*** *SSH*
- ***Test 1.1.12 -*** *Telnet*
- ***Test 1.1.13 -*** *Database*
- ***Test 1.1.14 -*** *Mail*
- ***Test 1.1.15 -*** *Voice*

A wide range of vulnerable target operating systems and applications are used, and the majority of the attacks are successful, gaining root shell or administrator privileges on the target machine.

We expect all the attacks to be reported in as straightforward and clear a manner as possible (i.e. an "RDS MDAC attack" should be reported as such, rather than a "Generic IIS Attack"). Wherever possible, attacks should be identified by their assigned CVE reference. It will also be noted when a response to an exploit is considered too "noisy", generating multiple similar or identical alerts for the same attack. Finally, we will note whether the device blocks the attack packet only or the entire "suspicious" TCP session.

This test is repeated twice: the first run with blocking disabled on the sensor (monitor mode only) in order to determine which attacks are detected and how accurately they are detected (*Attack Recognition Rating*); the second run with blocking enabled in order to determine which attacks are blocked successfully regardless of how they are detected or what alerts are raised (*Attack Blocking Rating*)

The "**default**" *Attack Recognition Rating-Detect Only* (ARRD) and *Attack Recognition Rating-Block* (ARRB) are each expressed as a percentage of detected/blocked exploits against total number of exploits launched with the default signature set as received by NSS. This demonstrates how effective the sensor can be when simply deploying the default configuration.

Following the initial test run, each vendor is provided with a list of CVE references of the attacks missed, and is then allowed 48 hours to produce an updated signature set. This updated signature set **must** be released to the general public as a standard signature/product update before the report is published - this ensures that vendors do not attempt to code signatures just for this test.

The sensor is then exposed to a second round of identical tests and the "**custom**" ARRD/ARRB is determined. This demonstrates how effective the vendor is at responding to a requirement for new or updated signatures.

Both the *default* and *custom* ARRD/ARRB figures are reported.

Test 1.2 - Resistance To False Positives

The aim of this test is to demonstrate how likely it is that a sensor raises a false positive alert - particularly critical for IPS devices.

We have a number of trace files of normal traffic with "suspicious" content, together with several "neutered" exploits which have been rendered completely ineffective. If a signature has been coded for a specific piece of exploit code rather then the underlying vulnerability, or if it relies purely on pattern matching, some of these false alarms could be alerted upon.

The product attains a "PASS" for each test case if it does **not** raise an alert and does **not** block the traffic. Raising an alert on any of these test cases is considered a "FAIL", since none of the "exploits" used in this test represents a genuine threat. A "FAIL" would thus indicate the chance that the sensor could block legitimate traffic inadvertently.

- ***Test 1.2.1 -*** *False positives*

Section 2 – Evasion

The aim of this section is to verify that the sensor is capable of detecting and blocking basic exploits when subjected to varying common evasion techniques.

Test 2.1 - Baselines

The aim of this test is to establish that the sensor is capable of detecting and blocking a number of common basic attacks (our baseline suite) in their normal state, with no evasion techniques applied. Note that common/older attacks have been chosen deliberately for this particular test to ensure that ALL products tested have signatures in place for the evasion tests.

- ***Test 2.1.1 -*** *Baseline attack replay*

Test 2.2 - Packet Fragmentation and Stream Segmentation

The baseline HTTP attacks are repeated, running them through fragroute using various evasion techniques, including:

- ***Test 2.2.1 -*** *IP fragmentation - ordered 8 byte fragments*
- ***Test 2.2.2 -*** *IP fragmentation - ordered 24 byte fragments*
- ***Test 2.2.3 -*** *IP fragmentation - out of order 8 byte fragments*
- ***Test 2.2.4 -*** *IP fragmentation - ordered 8 byte fragments, duplicate last packet*
- ***Test 2.2.5 -*** *IP fragmentation - out of order 8 byte fragments, duplicate last packet*
- ***Test 2.2.6 -*** *IP fragmentation - ordered 8 byte fragments, reorder fragments in reverse*

- ***Test 2.2.7 -*** *IP fragmentation - ordered 16 byte fragments, fragment overlap (favour new)*
- ***Test 2.2.8 -*** *IP fragmentation - ordered 16 byte fragments, fragment overlap (favour old)*
- ***Test 2.2.9 -*** *TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with invalid TCP checksums*
- ***Test 2.2.10 -*** *TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with null TCP control flags*
- ***Test 2.2.11 -*** *TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with requests to resync sequence numbers mid-stream*
- ***Test 2.2.12 -*** *TCP segmentation - ordered 1 byte segments, duplicate last packet*
- ***Test 2.2.13 -*** *TCP segmentation - ordered 2 byte segments, segment overlap (favour new)*
- ***Test 2.2.14 -*** *TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with out-of-window sequence numbers*
- ***Test 2.2.15 -*** *TCP segmentation - out of order 1 byte segments*
- ***Test 2.2.16 -*** *TCP segmentation - out of order 1 byte segments, interleaved duplicate segments with faked retransmits*
- ***Test 2.2.17 -*** *TCP segmentation - ordered 1 byte segments, segment overlap (favour new)*
- ***Test 2.2.18 -*** *TCP segmentation - out of order 1 byte segments, PAWS elimination (interleaved dup segs with older TCP timestamp options)*
- ***Test 2.2.19 -*** *IP fragmentation - out of order 8 byte fragments, interleaved duplicate packets scheduled for later delivery*
- ***Test 2.2.20 -*** *TCP segmentation - ordered 16 byte segments, segment overlap (favour new (Unix))*

For each of the evasion techniques, we note if (i) the attempted attack is blocked successfully (the primary aim of any IPS device), (ii) the attempted attack is detected and an alert raised in **any** form, and (iii) if the exploit is successfully “decoded” to provide an accurate alert relating to the original exploit, rather than alerting purely on anomalous traffic detected as a result of the evasion technique itself.

Test 2.3 - URL Obfuscation

The baseline HTTP attacks are repeated, this time applying various URL obfuscation techniques made popular by the Whisker Web server vulnerability scanner, including:

- ***Test 2.3.1 -*** *URL encoding*
- ***Test 2.3.2 -*** */./ directory insertion*
- ***Test 2.3.3 -*** *Premature URL ending*
- ***Test 2.3.4 -*** *Long URL*
- ***Test 2.3.5 -*** *Fake parameter*
- ***Test 2.3.6 -*** *TAB separation*
- ***Test 2.3.7 -*** *Case sensitivity*
- ***Test 2.3.8 -*** *Windows \ delimiter*
- ***Test 2.3.9 -*** *Session splicing*

For each of the evasion techniques, we note if (i) the attempted attack is blocked successfully, (ii) the attempted attack is detected and an alert raised in **any** form, and (iii) if the exploit is successfully "decoded" to provide an accurate alert relating to the original exploit, rather than alerting purely on anomalous traffic detected as a result of the evasion technique itself.

Test 2.4 - Miscellaneous Evasion Techniques

Certain baseline attacks are repeated, and are subjected to various protocol- or exploit-specific evasion techniques, including:

- ***Test 2.4.1 -*** *Altering default ports/passwords for backdoors*
- ***Test 2.4.2 -*** *Inserting spaces in FTP command lines*
- ***Test 2.4.3 -*** *Inserting non-text Telnet opcodes in FTP data stream*
- ***Test 2.4.4 -*** *Polymorphic mutation (ADMmutate)*
- ***Test 2.4.5 -*** *Altering protocol and RPC PROC numbers*
- ***Test 2.4.6 -*** *RPC record fragging (MS-RPC and Sun)*
- ***Test 2.4.7 -*** *HTTP exploits to non-standard port*

For each of the evasion techniques, we note if (i) the attempted attack is blocked successfully, (ii) the attempted attack is detected and an alert raised in **any** form, and (iii) if the exploit is successfully "decoded" to provide an accurate alert relating to the original exploit, rather than alerting purely on anomalous traffic detected as a result of the evasion technique itself.

Section 3 – Stateful Operation

The aim of this section is to be able to determine whether the sensor is capable of monitoring stateful sessions established through the device at various traffic loads without either losing state or incorrectly inferring state.

Test 3.1 - Stateless Attack Replay (Mid-Flows)

This test determines whether the sensor is resistant to stateless attack flooding tools - these utilities are used to generate large numbers of false alerts on the protected subnet using valid source and destination addresses and a range of protocols.

The main characteristic of many flooding tools is the fact that they generate single packets containing "trigger" patterns without first attempting to establish a connection with the target server. Whilst this can be effective in raising alerts with some stateless protocols such as UDP and ICMP, they should never be capable of raising an alert for exploits based on stateful protocols such as FTP and HTTP.

In this test, we transmit a number of packets taken from capture files of valid exploits, but without first establishing a valid session with the target server. We also remove the session tear down and acknowledgement packets so that the sensor can not "infer" that a valid connection was made.

In order to receive a "PASS" in this test, no alerts should be raised for any of the actual exploits (although "mid-flow" alerts are permitted).

However, each packet should be blocked if possible since it represents a "broken" or "incomplete" session.

- ***Test 3.1.1 -*** *Stateless attack replay*

Test 3.2 - Simultaneous Open Connections (default settings)

This test determines whether the sensor is capable of preserving state across increasing numbers of open connections, as well as continuing to detect and block new exploits when the state tables are filled. It also attempts to determine whether or not the sensor will block legitimate traffic once state tables are filled. This test is run using the default sensor settings (no tuning of sensor parameters).

A legitimate HTTP session is opened and the first packet of a two-packet exploit is transmitted. The Spirent Avalanche (on the "external" interface of the sensor) then opens various numbers of TCP sessions from 10,000 to 1,000,000 (one million) with the Spirent Reflector (on the "internal" interface of the sensor). 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 first session established, 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.

At each step, we ensure that the sensor is still capable of detecting and blocking freshly-launched exploits once all the connections are open, as well as confirming that the device does not block legitimate traffic (perhaps as a result of state tables filling up). We then launch further exploits whilst the Avalanche/Reflector devices "churn" connections at the maximum level set, ensuring that the sensor is still capable of detecting and blocking freshly-launched exploits as old connections are torn down and new ones recreated constantly.

- ***Test 3.2.1 - Attack Detection:*** *This test ensures that the sensor continues to detect new exploits as the number of open sessions is increased in stages from 10,000 to 1,000,000*
- ***Test 3.2.2 - Attack Blocking:*** *This test ensures that the sensor continues to block new exploits as the number of open sessions is increased in stages from 10,000 to 1,000,000*
- ***Test 3.2.3 - State Preservation:*** *This test ensures that the sensor maintains the state of pre-existing sessions as the number of open sessions is increased in stages from 10,000 to 1,000,000*
- ***Test 3.2.4 - Legitimate Traffic Blocking:*** *This test ensures that the sensor does not begin to block legitimate traffic as the number of open sessions is increased in stages from 10,000 to 1,000,000*

Test 3.3 - Simultaneous Open Connections (after tuning)

Test 3.2 is repeated after any tuning recommended by the vendor (if applicable) to increase the size of the state tables.

- ***Test 3.3.1 - Attack Detection:*** *As Test 3.2.1 following tuning*
- ***Test 3.3.2 - Attack Blocking:*** *As Test 3.2.2 following tuning*
- ***Test 3.3.3 - State Preservation:*** *As Test 3.2.3 following tuning*
- ***Test 3.3.4 - Legitimate Traffic Blocking:*** *As Test 3.2.4 following tuning*

Section 4 – Detection/Blocking Performance Under Load

The aim of this section is to verify that the sensor is capable of detecting and blocking exploits when subjected to increasing loads of background traffic up to the maximum bandwidth supported as claimed by the vendor.

The latest signature pack is acquired from the vendor, and sensors are deployed with **all** available attack signatures enabled (some audit/informational signatures may be disabled). Each sensor is configured to **detect and block** suspicious traffic.

Our “attacker” host launches a fixed number of exploits at a target host on the subnet being protected by the device under test. The Adtech network monitor is configured to monitor the switch SPAN port consisting of normal, exploit and background traffic, and is capable of reporting the total number of exploit packets seen on the wire as verification.

A fixed number of exploits are launched with zero background traffic to ensure the sensor is capable of detecting our baseline attacks. Once that has been established, increasing levels of varying types of background traffic are generated **through** the sensor in order to determine the point at which the sensor begins to miss attacks - all tests are repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic (or up to the maximum rated throughput of the device should this be less than 1Gbps).

At all stages, the Adtech network monitor verifies both the overall traffic loading and the total number of exploits seen on the target subnet. An additional confirmation is provided by the target host which reports the number of exploits which actually made it through.

The *Attack Blocking Rate* (ABR) at each background load is expressed as a percentage of the number of exploits blocked by the sensor (when in blocking mode) against the number verified by the Adtech network monitor and target host. The *Attack Detection Rate* (ADR) at each background load is expressed as a percentage of the number of exploits detected by the sensor (with blocking mode disabled) against the number verified by the Adtech network monitor and target host.

For each type of background traffic, we also determine the maximum load the sensor can sustain before it begins to drop packets/miss alerts. It is worth noting that devices which demonstrate 100 per cent ABR (blocking) but less than 100 per cent ADR (detection) in these tests will be prone to blocking **legitimate** traffic under similar loads.

Test 4.1 - UDP Traffic To Random Valid Ports

This test uses UDP packets of varying sizes generated by a **Smartbits SMB6000** with LAN-3301A 10/100/1000Mbps **TeraMetrics** cards installed.

A constant stream of the appropriate mix of packets - with variable source IP addresses and ports transmitting to a single fixed IP address/port - is transmitted through the sensor (bi-directionally, maximum of 1Gbps).

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 packets per second (pps) figures are verified by the Adtech Gigabit network monitoring tool before each test begins. Multiple tests are run and averages taken where necessary.

This traffic does not attempt to simulate any form of “real world” network condition. The aim of this test is purely to determine the raw packet processing capability of the sensor, and its effectiveness at passing “useless” packets quickly in order to pass potential attack packets to the detection engine. The range of packet sizes has been selected to mirror the maximum, minimum and average packet sizes used in our HTTP stress tests.

- ***Test 4.1.1 - 256 byte packets - maximum 453,000 packets per second:*** *This test is roughly equivalent to a 40,000 connections per second test in our HTTP stress tests (in terms of packet size and packets per second rate), and has been included to provide an indication of the packet processing performance under the most extreme conditions for most devices - it is unlikely that any real-life network will ever see network loads of over 450,000 256-byte packets per second unless under severe DOS conditions. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic.*
- ***Test 4.1.2 - 550 byte packets - maximum 220,000 packets per second:*** *This test has been included to provide a comparison with our “real world” packet mixes, since the average packet size is similar. No sessions are created during this test and there is very little for the detection engine to do in the way of protocol analysis. This test provides a reasonable indication of the ability of a device to process packets from the wire on an “average” network, and we would expect all products to demonstrate good performance levels. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic.*
- ***Test 4.1.3 - 1000 byte packets - maximum 122,000 packets per second:*** *This test is the complete opposite of the 256 byte packet test, in that we would expect every single product to be capable of returning 100 per cent detection rates across the board when using only 1000 byte packets. We have included this test mainly to demonstrate how easy it is to achieve good results using large packets – beware of test results that **only** quote performance figures using similar (or larger) packet sizes. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic.*

Test 4.2 - HTTP “Maximum Stress” Traffic With No Transaction Delays

HTTP is the most widely used protocol in most normal networks, as well as being one of the most widely exploited. The number of potential HTTP exploits for the protocol makes a pure HTTP network something of a torture test for the average sensor.

The use of multiple Spirent Communications **Avalanche 2500** and **Reflector 2500** devices allows us to create true “real world” traffic at speeds of up to 4.2 Gbps as a background load for our tests. Our Avalanche configuration is capable of simulating over 5 million users, with over 5 million concurrent sessions, and over 200,000 HTTP requests per second.

By creating genuine session-based traffic with varying session lengths, the sensor is forced to track valid 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” as it is possible to achieve in a lab environment, whilst ensuring absolute accuracy and repeatability.

The aim of this test is to stress the HTTP detection engine and determine how the sensor copes with detecting and blocking exploits under network loads of varying average packet size and varying connections per second.

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.

- ***Test 4.2.1 -*** *Max 2,500 new connections per second - average packet size 1000 bytes - maximum 120,000 packets per second. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic. With relatively low connection rates and large packet sizes, we expect all sensors to achieve 100% blocking rates throughout this test.*
- ***Test 4.2.2 -*** *Max 5,000 new connections per second - average packet size 540 bytes - maximum 225,000 packets per second. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic. With average connection rates average packet sizes, this is a good approximation of a real-world production network, and we expect all sensors to perform well in this test.*
- ***Test 4.2.3 -*** *Max 10,000 new connections per second - average packet size 440 bytes - maximum 275,000 packets per second. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic. With average packet sizes coupled with very high connection rates, this is a strenuous test for any sensor, and represents a very heavily used production network.*
- ***Test 4.2.4 -*** *Max 20,000 new connections per second - average packet size 360 bytes - maximum 320,000 packets per second. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic. With small packet sizes and extremely high connection rates this is an extreme test for any sensor. Not many sensors will perform well at all levels of this test.*

Test 4.3 - HTTP “Maximum Stress” Traffic With Transaction Delays

This test is identical to Test 4.2 except that we introduce a 10 second delay in the server response for each transaction. This has the effect of maintaining a high number of open connections throughout the test, thus forcing the sensor to utilise additional resources to track those connections.

- ***Test 4.3.1 -*** *Max 5,000 new connections per second - average packet size 540 bytes - maximum 225,000 packets per second - 10 second transaction delay - maximum 50,000 open connections. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic. With average connection rates average packet sizes, this is a good approximation of a real-world production network, and we expect all sensors to perform well in this test.*
- ***Test 4.3.2 -*** *Max 10,000 new connections per second - average packet size 440 bytes - maximum 275,000 packets per second - 10 second transaction delay - maximum 100,000 open connections. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic. With average packet sizes coupled with very high connection rates, this is a strenuous test for any sensor, and represents a very heavily used production network.*

Test 4.4 - Protocol Mix Traffic

Whereas 4.2 and 4.3 provide a pure HTTP environment with varying connection rates and average packet sizes, the aim of this test is to simulate more of a “real world” environment by introducing additional protocols whilst still maintaining a precisely repeatable and consistent background traffic load (something rarely seen in a real world environment).

The result is a background traffic load that, whilst less stressful than previous tests, is closer to what may be found on a heavily-utilised “normal” production network.

- ***Test 4.4.1 -*** *72% HTTP traffic (540 byte packets) + 20% FTP traffic + 6% UDP traffic (256 byte packets). Max 4000 new connections per second - average packet size 540 bytes - maximum 215,000 packets per second - maximum 750 open connections. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic. With lower connection rates, average packets sizes and a common protocol mix, this is a good approximation of a heavily-used production network, and we expect all sensors to perform well throughout this test.*

Test 4.5 - “Real World” Traffic

This is as close as it is possible to come to a true “real world” environment under lab conditions. For this test we eliminate the Reflector device and substitute an IIS Web server installed on a dual-Xeon server with Gigabit interface and 4GB RAM. This server holds a copy of The NSS Group Web site, and is capable of handling a full 1Gbps of traffic. We then capture a typical client browsing session on the NSS Group Web site, accessing a mixture of menu pages, lengthy text-based reports and multiple graphical images (screen shots) and have Avalanche replay multiple identical sessions from up to **20 new users per second**.

It should be noted that whereas the goal of the previous tests is a very predictable, consistent and repeatable background load that never varies, the nature of this test means that traffic is slightly more “bursty” in nature.

- ***Test 4.5.1 - Pure HTTP Traffic (simulated browsing session on NSS Web site):*** *Max 4700 new connections per second - 20 new users per second - average packet size 560 bytes - maximum 210,000 packets per second.*

*Repeated with 250Mbps, 500Mbps, 750Mbps and 950Mbps of background traffic. With genuine server responses to genuine **browser** sessions consisting **of multiple transactions per** session, this is a typical "real world" background load, albeit pure HTTP. Although the Web server and the network are extremely busy at the higher traffic loads, the "normal" connection rates and packet sizes should enable most sensors to perform well at all load levels in this test.*

- ***Test 4.5.2 - Protocol Mix (72% HTTP traffic (simulated browsing sessions as 4.5.1)) + 20% FTP traffic + 6% UDP traffic (256 byte packets)):** Max 3700 new connections per second - average packet size 560 bytes - maximum 205,000 packets per second - maximum 1,500 open connections.*

  *Repeated with 250Mbps, 500Mbps, 750Mbps and 950Mbps of background traffic. With genuine server responses to genuine browser sessions consisting of multiple **transactions** per session, mixed with FTP and UDP traffic, this is a typical "real world" background load. Although the Web server and the network are extremely busy at the higher traffic loads, the "normal" connection rates and packet sizes should enable most sensors to perform well at all load levels in this test.*

To gauge the effects of varying (smaller) packet sizes, connection rates and transaction delays, the results of tests 4.2 - 4.4 should be examined.

Section 5 – Latency & User Response Times

The aim of this section is to determine the effect the sensor has on the traffic passing through it under various load conditions.

Should a device impose a high degree of latency on the packets passing through it, a network or security administrator would need to think carefully about how many devices could be installed in a single data path before user response times became unacceptable or the combination of devices caused excessive timeouts. We also determine the effect of high levels of normal HTTP traffic and a basic DOS attack on the average latency and user response times.

Test 5.1 - Latency

We use Spirent SmartFlow software and The Smartbits SMB6000 with Gigabit TeraMetrics cards to create multiple traffic flows through the appliance and measure the basic throughput, packet loss, and latency through the sensor. This test - whilst not indicative of real-life network traffic - provides an indication of how much the sensor affects the traffic flow through it. This data is particularly useful for network administrators who need to gauge the effect of any form of in-line device which is likely to be placed at critical points within the corporate network.

SmartFlow runs through several iterations of the test varying the traffic load from 250Mbps to 1Gbps bi-directionally (or up to the maximum rated throughput of the device should this be less than 1Gbps) in steps of 250Mbps. This is repeated for a range of packet sizes (256 bytes, 550 bytes and 1000 bytes) of UDP traffic with variable IP addresses and ports. At each iteration of the test, SmartFlow records the number of packets dropped, together with average and maximum latency.

- ***Test 5.1.1 - Latency With No Background Traffic:*** *SmartFlow traffic is passed across the infrastructure switches and through the device (the latency of the basic infrastructure is known and is constant throughout the tests). The packet loss and average latency are recorded at each packet size and each load level from 250Mbps to 1Gbps (in 250Mbps steps).*
- ***Test 5.1.2 - Latency With Background Traffic Load:*** *The Avalanche and Reflector are configured to generate a fixed amount of background HTTP traffic through the sensor (up to 50 per cent of the maximum rated bandwidth of the device under test - maximum 500Mbps - maximum 2,500 new connections per second - average packet size 540 bytes - maximum 112,500 packets per second).*

  A 250Mbps bi-directional load of SmartFlow traffic at various packet sizes (256 bytes, 540 bytes and 1000 bytes) is then passed across the infrastructure switches and through the device and the packet loss and average latency are recorded.
- ***Test 5.1.3 - Latency When Under Attack:*** *The Spirent WebSuite software is used to generate a fixed load of DOS/DDOS traffic of 10 per cent of the maximum rated bandwidth of the device under test (maximum 100Mbps). A 250Mbps bi-directional load of SmartFlow traffic at various packet sizes (256 bytes, 540 bytes and 1000 bytes) is then passed across the infrastructure switches and through the device and the packet loss and average latency are recorded. The device should be configured to detect/block/mitigate the DOS attack by the most efficient method available.*

Test 5.2 - User Response Times

Avalanche and Reflector devices are used to generate HTTP sessions through the device in order to gauge how any increases in latency will impact the user experience in terms of failed connections and increased Web response times.

- ***Test 5.2.1 - Web Response With No Background Traffic:*** *The Avalanche and Reflector are configured to generate HTTP traffic through the sensor (up to 50 per cent of the maximum rated bandwidth of the device under test - maximum 500Mbps - maximum 2,500 new connections per second - average packet size 540 bytes - maximum 112,500 packets per second).*

  The minimum, maximum and average page response times and number of failed connections are recorded by Avalanche to provide an indication of the expected response times under normal traffic conditions.
- ***Test 5.2.2 - Web Response When Under Attack:*** *The Avalanche and Reflector are configured to generate HTTP traffic through the sensor as for Test 5.2.1. The Spirent WebSuite software is then used to generate DOS/DDOS traffic up to 10 per cent of the maximum rated bandwidth of the device under test (maximum 100Mbps).*

  The minimum, maximum and average page response times and number of failed connections are recorded by Avalanche to provide an indication of the expected response times when the device is under attack.

Section 6 – Stability & Reliability

These tests attempt to verify the stability of the device under test under various extreme conditions. Long term stability is particularly important for an in-line IPS device, where failure can produce network outages.

- ***Test 6.1.1 - Blocking Under Extended Attack:*** *For this test, we expose the external interface of the device to a constant stream of alerts over an extended period of time. The device is configured to block and alert, and thus this test provides an indication the effectiveness of both the blocking and alert handling mechanisms. A continuous stream of exploits mixed with some legitimate sessions is transmitted through the device at a maximum of 100Mbps (max 50,000 packets per second, average packet sizes in the range of 120-350 bytes) for 8 hours with no additional background traffic. This is not intended as a stress test in terms of traffic load - 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 per cent of recognisable exploits, raising an alert for each. Results are presented as a simple PASS/FAIL. If any recognisable exploits are passed - caused by either the volume of traffic ***or*** *the sensor failing open for any reason - this will result in a FAIL.*

- ***Test 6.1.2 - Passing Legitimate Traffic Under Extended Attack:*** *This test is identical to 6.1.1, where we expose the external interface of the device to a constant stream of alerts over an extended period of time. The device is expected to remain operational and stable throughout this test, and to pass 100 per cent of legitimate traffic. Results are presented as a simple PASS/FAIL. If any legitimate traffic is blocked - caused by either the volume of traffic* ***or*** *the sensor failing closed for any reason - this will result in a FAIL.*

- ***Test 6.1.3 - ISIC/ESIC/TCPSIC/UDPSIC/ICMPSIC:*** *This test attempts to stress the protocol stack of the device under test by exposing it to traffic from the ISIC test tool. The ISIC test tool host is connected directly to the external interface of the sensor, and the ISIC target directly to the internal interface. ISIC traffic is transmitted through the sensor (without passing through any other network equipment) and the effects noted. Traffic load is a maximum of 350Mbps and 60,000 packets per second (average packet size is 690 bytes). Results are presented as a simple PASS/FAIL - the device is expected to remain operational and capable of detecting and blocking exploits throughout the test to attain a PASS.*

Section 7 – Management and Configuration

The aim of this section is to determine the features of the management system, together with the ability of the management port on the device under test to resist attack.

Test 7.1 - Management Port

Clearly the ability to manage the alert data collected by the sensor is a critical part of any IDS/IPS system. For this reason, an attacker could decide that it is more effective to attack the management interface of the device than the detection interface.

Given access to the management network, this interface is often more visible and more easily subverted than the detection interface, and with the management interface disabled, the administrator has no means of knowing his network is under attack.

- ***Test 7.1.1 - Open ports:*** *We will scan the open ports and active services on the management interface and report on known vulnerabilities.*
- ***Test 7.1.2 - ISIC/ESIC/TCPSIC/UDPSIC/ICMPSIC:*** *This test attempts to stress the protocol stack of the management interface of the device under test by exposing it to traffic from the ISIC test tool. The ISIC test tool host is connected directly to the management interface of the IPS sensor, and that interface is also the target. ISIC traffic is transmitted to the management interface of the IPS device (without passing through any other network equipment) and the effects noted.*

  Traffic load is a maximum of 350Mbps and 60,000 packets per second (average packet size is 690 bytes). Results are presented as a simple PASS/FAIL - the device is expected to remain (a) operational and capable of detecting and blocking exploits, and (b) capable of communicating in both directions with the management server/console throughout the test to attain a PASS.
- ***Test 7.1.3 -*** *We note whether the ISIC attacks themselves are detected by the sensor even though targeted at the management port.*

Fortinet FortiGate-800 Test Results

Section 1 - Detection Engine

Test 1.1 – Attack Recognition	Attacks	Default ARRD	Default ARRB	Custom ARRD	Custom ARRB
Test 1.1.1 - Backdoors	7	5	5	7	7
Test 1.1.2 - WINS/DNS	3	2	2	3	3
Test 1.1.3 - DOS	10	6	9	7[1]	10
Test 1.1.4 - False negatives (modified exploits)	14	10	10	14	14
Test 1.1.5 - Finger	4	3	3	4	4
Test 1.1.6 - FTP	5	4	4	5	5
Test 1.1.7 - HTTP	43	39	39	43	43
Test 1.1.8 - ICMP	2	0	2	0[1]	2
Test 1.1.9 - Reconnaissance	8	5	5	8	8
Test 1.1.10 - RPC	9	5	5	9	9
Test 1.1.11 - SSH	1	1	1	1	1
Test 1.1.12 - Telnet	1	1	1	1	1
Test 1.1.13 - Database	1	1	1	1	1
Test 1.1.14 - Mail	1	1	1	1	1
Test 1.1.15 - Voice	1	0	0	1	1
Total	**110**	**83 / 110**	**88 / 110**	**105 / 110**	**110 / 110**
		75%	**80%**	**95%[1]**	**100%**

Test 1.2 – Resistance to False Positives	Pass/Fail
Test 1.2.1 - Suspicious FTP traffic	PASS
Test 1.2.2 - HTTP “exploit” using incorrect method	PASS
Test 1.2.3 - Retrieval of Web page containing “suspicious” URLs	PASS
Test 1.2.4 - Simple SMTP QUIT command	PASS
Test 1.2.5 - Normal NetBIOS copy of “suspicious” files	PASS
Test 1.2.6 - Normal NetBIOS traffic	PASS
Test 1.2.7 - POP3 e-mail containing “suspicious” URLs	PASS
Test 1.2.8 - POP3 e-mail with “suspicious” DLL attachment	PASS
Test 1.2.9 - POP3 e-mail with “suspicious” Web page attachment	PASS
Test 1.2.10 - SMTP e-mail transfer containing “suspicious” URLs	PASS
Test 1.2.11 - SMTP e-mail transfer with “suspicious” DLL attachment	PASS
Test 1.2.12 - SMTP e-mail transfer with “suspicious” Web page attachment	PASS
Test 1.2.13 - SNMP V3 packet with invalid parameter	PASS
Test 1.2.14 - Fake DNS /bin/sh buffer overflow	PASS
Test 1.2.15 - Inter-firewall communication traffic	PASS
Test 1.2.16 - Fake SQL Slammer traffic	PASS[2]
Test 1.2.17 - File copy of GIF file (contains bytes which look like NOP sled)	PASS
Total Passed	**17 / 17**

Section 2 - IPS Evasion

Test 2.1 – Evasion Baselines	Detected?	Blocked?
Test 2.1.1 - NSS Back Orifice ping	YES	YES
Test 2.1.2 - Back Orifice connection	YES	YES
Test 2.1.3 - FTP CWD root	YES	YES
Test 2.1.4 - ISAPI printer overflow	YES	YES
Test 2.1.5 - Showmount export lists	YES	YES
Test 2.1.6 - Test CGI probe (/cgi-bin/test-cgi)	YES	YES
Test 2.1.7 - PHF remote command execution	YES	YES
Total	**7 / 7**	**7 / 7**

Test 2.2 – Packet Fragmentation/Stream Segmentation	Detected?	Decoded?	Blocked?
Test 2.2.1 - IP fragmentation - ordered 8 byte fragments	YES	YES	YES
Test 2.2.2 - IP fragmentation - ordered 24 byte fragments	YES	YES	YES
Test 2.2.3 - IP fragmentation - out of order 8 byte fragments	YES	YES	YES
Test 2.2.4 - IP fragmentation - ordered 8 byte fragments, duplicate last packet	YES	YES	YES
Test 2.2.5 - IP fragmentation - out of order 8 byte fragments, duplicate last packet	YES	YES	YES
Test 2.2.6 - IP fragmentation - ordered 8 byte fragments, reorder fragments in reverse	YES	YES	YES
Test 2.2.7 - IP fragmentation - ordered 16 byte fragments, fragment overlap (favour new)	YES	YES	YES
Test 2.2.8 - IP fragmentation - ordered 16 byte fragments, fragment overlap (favour old)	YES	YES	YES
Test 2.2.9 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with invalid TCP checksums	YES	YES	YES
Test 2.2.10 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with null TCP control flags	YES	YES	YES
Test 2.2.11 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with requests to resync sequence nos. mid-stream	YES	YES	YES
Test 2.2.12 - TCP segmentation - ordered 1 byte segments, duplicate last packet	YES	YES	YES
Test 2.2.13 - TCP segmentation - ordered 2 byte segments, segment overlap (favour new)	YES	YES	YES
Test 2.2.14 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with out-of-window sequence numbers	YES	YES	YES
Test 2.2.15 - TCP segmentation - out of order 1 byte segments	YES	YES	YES
Test 2.2.16 - TCP segmentation - out of order 1 byte segments, interleaved duplicate segments with faked retransmits	YES	YES	YES
Test 2.2.17 - TCP segmentation - ordered 1 byte segments, segment overlap (favour new)	YES	YES	YES
Test 2.2.18 - TCP segmentation - out of order 1 byte segments, PAWS elimination (interleaved dup segments with older TCP timestamp options)	YES	YES	YES
Test 2.2.19 - IP fragmentation - out of order 8 byte fragments, interleaved duplicate packets scheduled for later delivery	YES	YES	YES
Test 2.2.20 - TCP segmentation - ordered 16 byte segments, segment overlap (favour new (Unix))	YES	YES	YES
Total	**20 / 20**	**20 / 20**	**20 / 20**

Test 2.3 – URL Obfuscation	Detected?	Decoded?	Blocked?
Test 2.3.1 - URL encoding	YES	YES	YES
Test 2.3.2 - /./ directory insertion	YES	YES	YES
Test 2.3.3 - Premature URL ending	YES	YES	YES
Test 2.3.4 - Long URL	YES	YES	YES
Test 2.3.5 - Fake parameter	YES	YES	YES
Test 2.3.6 - TAB separation	YES	YES	YES
Test 2.3.7 - Case sensitivity	YES	YES	YES
Test 2.3.8 - Windows \ delimiter	YES	YES	YES
Test 2.3.9 - Session splicing	YES	YES	YES
Total	**9 / 9**	**9 / 9**	**9 / 9**

Test 2.4 – Miscellaneous Obfuscation Techniques	Detected?	Decoded?	Blocked?
Test 2.4.1 - Altering default ports	YES	YES	YES
Test 2.4.2 - Inserting spaces in FTP command lines	YES	NO	YES
Test 2.4.3 - Inserting non-text Telnet opcodes in FTP data stream	YES	YES	YES
Test 2.4.4 - Polymorphic mutation (ADMmutate)	YES	YES	YES
Test 2.4.5 - Altering protocol and RPC PROC numbers	YES	YES	YES
Test 2.4.6 - RPC record fragging (MS-RPC and Sun)	YES	YES	YES
Test 2.4.7 - HTTP exploits to port <> 80	NO	NO	NO
Total	**6 / 7**	**5 / 7**	**6 / 7**

Section 3 - Stateful Operation

Test 3.1 – Stateless Attack Replay	Alert?	Blocked?	Pass/Fail
Test 3.1.1 - Stateless Web exploits	NO	YES[3]	PASS
Test 3.1.2 - Stateless FTP exploits	NO	YES[3]	PASS

Test 3.2 – Simultaneous Open Connections (default settings)							
Number of open connections	10,000	25,000	50,000	100,000	250,000	500,000	1,000,000
Test 3.2.1 - Attack Detection	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Test 3.2.2 - Attack Blocking	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Test 3.2.3 - State Preservation	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Test 3.2.4 - Legitimate traffic blocking	PASS	PASS	PASS	PASS	PASS	PASS	PASS

Test 3.3 – Simultaneous Open Connections (after tuning)							
Number of open connections	10,000	25,000	50,000	100,000	250,000	500,000	1,000,000
Test 3.3.1 - Attack Detection	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Test 3.3.2 - Attack Blocking	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Test 3.3.3 - State Preservation	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Test 3.3.4 - Legitimate traffic blocking	PASS	PASS	PASS	PASS	PASS	PASS	PASS

Section 4 - Detection/Blocking Performance Under Load

Test 4.1 – UDP traffic to random valid ports		100Mbps	200Mbps	300Mbps	400Mbps	Max
Test 4.1.1 - 256 byte packet test - max 338,000pps (400Mbps)	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	
Test 4.1.2 - 550 byte packet test - max 880,000pps (400Mbps)	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	
Test 4.1.3 - 1514 byte packet test - max 48,800pps (400Mbps)	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	

Test 4.2 – HTTP "maximum stress" traffic with no transaction delays		100Mbps	200Mbps	300Mbps	400Mbps	Max
Test 4.2.1 - Max 1000 connections per second - ave packet size 1000 bytes - max 48,800 packets per second	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	
Test 4.2.2 - Max 2000 connections per second - ave packet size 540 bytes - max 90,000 packets per second	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	
Test 4.2.3 - Max 4000 connections per second - ave packet size 440 bytes - max 110,000 packets per second	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	
Test 4.2.4 - Max 8000 connections per second - ave packet size 360 bytes - max 128,000 packets per second	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	

Test 4.3 – HTTP "maximum stress" traffic with transaction delays		100Mbps	200Mbps	300Mbps	400Mbps	Max
Test 4.3.1 - Max 2000 connections per second - ave packet size 540 bytes - max 90,000 packets per second - 10 sec delay - max 20,000 open connections	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	
Test 4.3.2 - Max 4000 connections per second - ave packet size 440 bytes - max 110,000 packets per second - 10 sec delay - max 40,000 open connections	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	

Test 4.4 – Protocol mix		100Mbps	200Mbps	300Mbps	400Mbps	Max
Test 4.4.1 - 72% HTTP (540 byte packets) + 20% FTP + 6% UDP (256 byte packets). Max 1600 connections per second - ave packet size 540 bytes - max 86,000 packets per second - max 300 open connections	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	

Test 4.5 – Real World traffic		100Mbps	200Mbps	300Mbps	400Mbps	Max
Test 4.5.1 - Pure HTTP (simulated browsing session on NSS Web site). Max 1900 connections per second - 8 new users per second - ave packet size 560 bytes - max 84,000 packets per second	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	
Test 4.5.2 - Protocol mix - 72% HTTP (simulated browsing sessions as 2.5.1) + 20% FTP + 6% UDP (256 byte packets). Max 1500 connections per second - ave packet size 560 bytes - max 82,000 packets per second - max 600 open connections	Detected	100%	100%	100%	100%	400Mbps
	Blocked	100%	100%	100%	100%	

Section 5 - Latency & User Response Times

Test 5.1 – Latency	Packet Size	250Mbps	500Mbps	750Mbps	1Gbps
Test 5.1.1 Average latency (µs) with no background traffic	256	248.79	250.01	250.08	251.99
	550	260.21	262.71	264.47	266.41
	1000	262.34	266.35	269.01	279.83
Test 5.1.2 Average latency (µs) with background traffic (200Mbps HTTP traffic, max 2500 connections per second - ave packet size 540 bytes - max 45,000 packets per second)	256	137.73			
	550	163.90			
	1000	198.44			
Test 5.1.3 Average latency (µs) when under attack (40Mbps SYN flood)	256	187.90			
	550	188.02			
	1000	215.81			

Test 5.2 – User Response Times	Attempted Trans	Failed Trans	Min Page Response	Max Page Response	Ave Page Response
Test 5.2.1 - Web page response (ms) with no background traffic (200Mbps HTTP traffic, max 2500 connections per sec - ave packet size 540 bytes - max 45,000 packets per sec)	630065	0	202	237	204
Test 5.2.2 - Web page response (ms) when under attack (200Mbps HTTP traffic, max 2500 connections per sec - ave packet size 540 bytes - max 45,000 packets per sec PLUS 40Mbps SYN flood)	623857	0	202	18666	219

Section 6 - Stability & Reliability

Test ID	Result
Test 6.1.1 - Blocking Under Extended Attack	100%
Test 6.1.2 - Passing legitimate traffic under extended attack	100%
Test 6.1.3 - ISIC/ESIC/TCPSIC/UDPSIC/ICMPSIC	PASS

Section 7 - Management Interface

Test ID	Result
Test 7.1.1 - Open Ports	PASS
Test 7.1.2 - ISIC/ESIC/TCPSIC/UDPSIC/ICMPSIC	PASS
Test 7.1.3 - ISIC attacks detected against management interface?	YES

Notes:

1. “Missing” exploits were actually blocked silently by the firewall - not possible to detect
2. Blocked initially - clean following signature update
3. The firewall module blocks all mid-flows - this is not configurable

Section 1: Detection Engine

We installed one sensor with the latest signature pack, and created an IPS Protection Profile with every attack signature and every anomaly detection method enabled. The SYN flood and UDP flood thresholds were adjusted accordingly to differentiate between our “normal” test loads and our attack traffic.

We then created two firewall policies (*Internal* to *External*, and *External* to *Internal*) which were set to permit **all** traffic in **both** directions, and applied the IPS Protection Profile to each. Only the IPS protection module was active - Anti Virus, Web Filtering, Web Category Filtering, Spam Filtering and Content Logging were all disabled. In other words, the FortiGate was configured as a typical dedicated IPS appliance.

Signature recognition (with blocking disabled) was acceptable out of the box, though at 75 per cent was one of the lowest we have seen in the labs for a pure IPS device. However, the Fortinet engineers rose to the challenge and increased the recognition rate to 95 per cent after the application of a signature pack update which was provided to us in 48 hours. The quality of the new signatures seemed to be as high as the existing ones, and performance of the box was not affected at all by the large update.

Blocking performance was slightly higher than detection throughout the tests due to the fact that the firewall module silently drops certain packets (notably several of the DOS fragment packets) which means it is simply not possible for the IPS module to detect them. Blocking performance was increased from a creditable 80 per cent to a perfect 100 per cent following the application of the signature update.

We noted a minimum of “noise”, with very few test cases raising multiple alerts for a single exploit. Most of our “false negative” (modified exploit) cases were detected correctly out of the box, and were all detected accurately following the signature update.

A major concern in deploying an IPS is the blocking of legitimate traffic. We noted only one false positive alert from our test suite out of the box, and this was corrected following the signature update.

The FortiGate-800 arrives with a default policy configured with a number of signatures disabled, and thus it would be necessary for the customer to tune the device for his own environment. The PASS and BLOCK actions seem to be set intelligently throughout by default to provide a set of “recommended settings”, and we would be reasonably confident in deploying the FortiGate-800 with the default policy.

It is a straightforward matter **using the CLI** to set all signatures to pass/log for initial testing, and to reset them all to the suggested defaults using a single command. This cannot be achieved via the GUI, however.

Section 2: IPS Evasion

Resistance to known evasion techniques was excellent, with the FortiGate-800 achieving almost a clean sweep across the board in our evasion tests. *Fragroute, Whisker, ADMmutate* and even *RPC record fragging* all failed to trick FortiGate into ignoring valid attacks.

It is not yet possible, however, to run servers (like HTTP, FTP, etc) on non-standard ports and successfully detect exploits against them - this capability will be added in the next release.

Note that not only were the fragmented and obfuscated attacks blocked successfully, but almost every one of them was decoded accurately as well.

Section 3: Stateful Operation

Out of the box, the FortiGate-800 handled 1 million open connections easily (no tuning was necessary to complete all our stateful tests).

Default operation of the device is to allow all traffic when the state tables are full or resources are low - this means that it is technically possible to evade the FortiGate once the state tables are full, since it will allow attack traffic through at that point. However, all packets for such new connections are still subjected to packet-based inspection if possible instead of passing them through without discrimination.

The default action can be configured via the CLI to block all traffic by default when resources are low - an excellent feature which allows the administrator to make this difficult choice between security and availability.

Section 4: Detection/Blocking Performance Under Load

Note that the FortiGate-800 was tested as a 400Mbps IPS device. This is the rated speed of the appliance with the IPS module enabled (the firewall-only capability is rated at 600Mbps). Performance at all levels of our load tests was impeccable, with 100 per cent of all attacks being detected and blocked under all load conditions.

We would thus have no hesitation in rating the FortiGate-800 as a true 400Mbps device as claimed by Fortinet - indeed, in a live network we feel this rating is conservative.

Section 5: Latency & User Response Times

Basic latency figures were within acceptable limits for a device of this type at all traffic loads and with all packet sizes, ranging from 249µs with 100Mbps of 256 byte packets, to 280µs with 400Mbps of 1000 byte packets (these figures are also well within those claimed by the vendor). Latency with 128 byte packets was also good, with zero packet loss across all loads.

Behaviour throughout the tests with no background traffic was unpredictable, however, with widely varying latency figures at almost all network loads and all packet sizes. It was necessary for us to make several runs and average the results in order to obtain meaningful figures. Increases in average latency as the load was increased from 100Mbps to 400Mbps were marginal across all packet sizes.

Strangely, once we placed the device under load using HTTP traffic, latency figures actually **improved** dramatically and became extremely stable and repeatable between runs.

When we loaded the FortiGate-800 with 200Mbps (half the rated speed of the device) of genuine HTTP traffic, latency ranged from 138µs with 256 byte packets to 198µs with 1000 byte packets. This is due to the use of NAPI drivers for the network cards, which switch from interrupt-driven to polling mode when under heavy load.

40Mbps of SYN flood traffic had a predictable effect, increasing latency to 188µs with 256 byte packets and to 216µs with 1000 byte packets (still **less** than when the device is under a pure UDP load). HTTP response times also increased only slightly during the SYN flood tests, from 214ms under normal load to 219ms with the SYN flood.

These figures when under attack are excellent, and SYN Flood mitigation was handled well throughout the test, with the flood being completely mitigated once the initial threshold had been passed.

Section 6: Stability & Reliability

The FortiGate-800 performed consistently and completely reliably throughout our tests. Under eight hours of extended attack (comprising millions of exploits mixed with genuine traffic) it continued to block 100 per cent of attack traffic, whilst passing 100 per cent of legitimate traffic.

Exposing the sensor interface to an extended run of ISIC-generated traffic had no adverse effect, and the device continued to detect and block all other exploits throughout and following the ISIC attack.

Section 7: Management Interface

Open ports on the management interface are restricted to TCP/443 (HTTPS), TCP/444 (SNPP), TCP/80 (HTTP), TCP/22 (SSH) and TCP/23 (Telnet). These are configurable within the firewall module, and typically only 22, 443 and 444 are enabled to allow communication with the management console. Access can also be restricted to trusted hosts only, following which port scans failed completely from any other PC on the management network.

The extended ISIC attack against the management interface had no effect on the appliance and its ability to detect and block attacks. Alerts for some of the ISIC traffic were raised - not as wide a range as when we attacked the detection interfaces, but enough to inform the administrator that the management port was under attack.

The sensor continued to work perfectly throughout and following the ISIC attack, and there were no residual stability problems.