Internet Security Systems
Proventia G200 Rev A
Technical Evaluation

An NSS Group Report

First published January 2004 (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

The NSS Group Limited is registered in England & Wales, Reg No. 3233843
Registered Office: Montagu House, 81 High Street, Huntingdon, Cambs, PE29 3NY, England
Tel +44 (0)7005 802 953

TABLE OF CONTENTS

TABLE OF FIGURES

The NSS Group

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

Based in the UK with separate 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 throughout Europe and the United States.

The Group consists of two wholly-owned subsidiaries :

- *NSS Network Testing Laboratories*
- *Network Security Services*

NSS Network Testing Laboratories are available to vendors and end-users for fully independent testing of networking, communications and security hardware and software.

NSS Network Testing Laboratories also operates certification schemes for vendors and certification bodies, and currently provides certification of firewalls, VPN's, crypto products 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

Network Security Services provides a range of security-related services to vendors and end-users including security policy definition, IDS, firewall and VPN implementation, network security auditing and analysis, and penetration testing.

Foreword

The NSS Group is pleased to present the results of the first comprehensive *Intrusion Prevention System* (IPS) test of its kind.

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.

Contrary to recent analyst claims, we do not believe that "IDS is dead" or that "IPS is stillborn". So-called "*deep inspection firewalls*" **may** be where the industry is heading in the long term, but they are simply not ready for prime-time deployments at this point in time. Until they are, 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.

Please note that we fully recognise that each of the above product-types could be considered as "intrusion prevention" systems in some respect, along with Anti Virus gateways, desktop firewalls, and other products designed to "prevent" malicious activity on a network or individual host. However, we also believe that the marketing terms for each of these products are well established, and that the grounds for creating a new market segment - referred to as *Intrusion Prevention Systems* (IPS) - specifically for those products which are evaluated as part of this report is valid. We have defined what **we** consider to be IPS (both host- and network-based) in the introductory text to this report.

You might not like the marketing hype, but the time for quibbling over the terminology is over - it is now time to get down to the serious issue of evaluating the technology behind it.

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

Overall, the suite contains over **750 individual tests**, many of which are run multiple times, to provide the most thorough and complete evaluation of IPS products available anywhere today.

We believe that our IPS test methodology will become the *de facto* standard for testing in-line intrusion prevention 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 1)** report can be viewed on-line at *www.nss.co.uk/ips*.

Bob Walder

INTRODUCTION

In a recent 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*.

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 a malicious packet, in addition to raising an alert, it will discard the packet 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.

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 has conducted the first comprehensive IPS test of its kind. 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 test methodology (see section on *Testing Methodology* later in this report for detailed methodology) 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 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 750 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) 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 1.48 million 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 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.

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:

- ***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*.

ISS PROVENTIA G200 REVISION A

Executive Summary

Proventia G Series intrusion prevention appliances are designed to proactively block malicious attacks from entering the network, including denial of service, intrusions and malicious code, backdoors and hybrid threats like MS Blaster, SQL Slammer, Nimda and Code Red. Proventia G Series blocks attacks in real-time, minimising the need for active administrator involvement in most security events.

Built upon the Proventia code base and modified for in-line appliance configurations, the Proventia G200 appliance features a pre-installed network agent on a standard Intel platform. The device features two 10/100/1000Mbps ports for management and for sending TCP Resets when monitoring in passive mode, and a dual copper port 100/1000Mbps network card which is capable of operating in pass-through mode when power is not applied to the appliance.

Proventia G Series appliances analyse over 100 network protocols and include over 2,500 checks, and can operate in three modes: *Active* (inline, blocking), *Passive* (not inline, no blocking), and *Simulation* (inline, no blocking).

The Proventia G200 is designed for 200Mbps networks and within that range offers outstanding performance coupled with surprisingly low latency under normal traffic conditions. The G200 will not cause bottlenecks in any normal 200Mbps environment unless subjected to heavy DOS attacks, which it nevertheless mitigates effectively.

We also found the Proventia G200 to be to be very stable and reliable, coping with our extensive reliability tests with ease and without blocking any legitimate traffic or succumbing to common evasion techniques.

The SiteProtector management system has been well designed to handle management and configuration of large numbers of IPS and IDS sensors across the enterprise. Alert handling is powerful and flexible, especially when combined with the optional SiteProtector SecurityFusion module.

Architecture

The Internet Security Systems appliance-based IPS offering consists of the following components:

Intrusion Protection Appliance

The Proventia G200 appliance is a 1U rack mount server chassis based on a standard Intel platform. Details of the processor and memory configuration are not available, but they are designed to accommodate the rated bandwidth of the appliance (200Mbps in this case).

The device includes redundant internal cooling fans (not hot-swappable) and a RAID configuration for redundant data storage. Only a single power supply is available, however.

There are two built-in copper 10/100/1000Mbps ports on the rear panel, one used as the management interface, and the other dedicated to sending TCP Reset/Kills when the device is being used in passive IDS mode. Two more copper 10/100/1000Mbps ports are provided for inline or passive monitoring - one 200Mbps segment can be monitored in inline mode, whilst two 100Mbps segments can be monitored in passive mode. One nice fault tolerance feature is that when there is no power applied to the device, the dual-port card has an automatic inline bypass, ensuring that traffic is passed normally when the device is otherwise unavailable. Note that this is not configurable - it is not possible to have this device fail closed.

The Proventia G200 uses the Linux-based Proventia detection engine together with a custom driver specially built for Proventia hardware. The system is hardened and completely locked down so that the only way it can be accessed is from SiteProtector or SSH (or directly at the appliance via keyboard and monitor). Workgroup Manager is not supported with Proventia.

The system is designed to provide all initial configuration functions - such as IP address assignment, management console assignment, and password change - via a simple text-based local interface to which the administrator is restricted following login. It is also possible to switch monitoring modes, examine and clear the dynamic blocking table, and reboot or shut down the appliance via this menu. All other day-to-day administrative tasks, such as policy changes, can be accomplished via the SiteProtector manager installed nearby or at a remote location.

Proventia Network Agent

The Proventia Linux-based sensor software (which ISS refers to as an *Agent*) runs on the Proventia hardware in order to monitor a network segment, analyse the traffic flow and look for intrusions and signs of network abuse.

Sitting in front of the Agent in terms of the logical path taken by the traffic being analysed is a dynamic packet filtering firewall (it is actually fully integrated with the Agent). This is a very simple firewall which can have rules created for it by the administrator within SiteProtector as well automatically by the Agent whenever dynamic blocking is enabled. The idea is to provide a “fast path” for certain traffic - if rules are matched by the firewall first (either to permit traffic or to block it) then the packets can be passed directly to the internal interface or dropped instantly without ever having to be passed through the relatively slower detection engine. This helps to maintain performance through the device.

The inline appliance supports three modes of operation :

- ***Passive monitoring -*** *Behaving as a typical passive mode IDS, in this mode, TCP Reset/Kill is the only response that can modify network traffic (sent via the separate RSKill interface). The drop and dynamic blocking responses are disabled in this mode, as are firewall rules. This mode can be used to tune the appliances for subsequent inline protection.*
- ***Inline protection -*** *This mode includes all the functionality of passive monitoring mode. In addition, all firewall rules are enabled, so any packets that match a Drop and DropAndReset firewall action are dropped and not processed any further by the appliance.*

The drop and dynamic blocking responses are enabled, and result in malicious packets being dropped when invoked.

- ***Inline simulation -*** *This mode includes all the functionality of the inline protection mode, but only alerts are raised and no traffic is dropped or blocked. In addition, firewall rule actions Drop and DropAndReset are also disabled. Any events that have blocking responses enabled still have alerts raised, but with an indication that the events did not block because of the mode. In inline simulation mode, the appliance does not reset TCP connections by default. This mode can be used to tune policies in a production environment without the risk of adversely affecting network traffic, or verify that the appliances are not disrupting the network or blocking legitimate traffic*

When an intrusion is detected, Proventia can respond in a number of ways, including:

- *Recording the date, time, source, and target of the event*
- *Recording the content of the event*
- *Notifying the network administrator*
- *Reconfiguring the firewall*
- *Dropping the offending packet and terminating the session*

The drop response includes the following options:

- ***ConnectionWithReset:*** *drops all packets on the connection in which the event occurred and sends TCP reset packets.*
- ***Connection:*** *drops all packets on the connection in which the event occurred.*
- ***Packet:*** *drops the packet that triggered the event only.*

It is also possible to enable a *dynamic blocking* response for the signature in the active policy. In this case, the appliance blocks any subsequent packets that meet the same criteria as specified in the response by setting dynamic rules in the firewall module. This means that any subsequent packets which match the specified criteria are dropped immediately by the firewall without having to be processed by the potentially slower *protection engine*. Dynamic blocking criteria include:

- *Target IP address*
- *Target port*
- *Attacker IP address*
- *Attacker port*
- *ICMP code*
- *ICMP type*

Proventia understands over 100 network and application layer protocols such as HTTP, FTP, SMTP, SNMP, RPC, SMB, NetBIOS as well as many Trojan communication protocols. Over 150 protocols are understood, if not completely decoded.

Proventia's protocol analysis capabilities do not rely on merely checking for textbook (RFC) compliance since anomalies are difficult (and sometimes impossible) to define due to RFC ambiguities and differences in interpretation of the standards. Instead, Proventia focuses on analysing protocols to recognise context and then employs pattern matching techniques where appropriate to positively identify malicious content.

In addition, Proventia can also perform full IP packet de-fragmentation, as well as TCP and HTTP session reassembly (tracking two hundred and fifty thousand simultaneous sessions out of the box - more if tuned appropriately), and is largely immune to evasion techniques such as polymorphic shell code mutation, Unicode URL encoding, packet overlapping and RPC record fragging.

Event consolidation logic helps to reduce the total number of unique events that the sensor generates by logically combining large numbers of identical events into a single alert.

Proventia listens to both the transmitting and receiving data streams which allows it to capture server responses for many attacks, such as HTTP, FTP, RPC and DNS events. The ability to report return codes and other server-side responses to the alert console helps guide security operators to quickly recognise the outcome of an attack, and is thus an important tool for prioritising incident response.

Proventia has very strong signature coverage. In addition to the Proventia protocol analysis module, now referred to as the *Protection Engine*, Proventia, like RealSecure, has the ability to import most of the published rules from Snort, via the aptly named *Trons* module.

The aim of this is not to replace Snort, nor is it to enable sites to include the entire Snort signature set (which would seriously impact the performance of Proventia), but to allow users to write their own custom signatures using a relatively simple rules definition language that is familiar to many in the security industry.

SiteProtector

SiteProtector provides a central console geared towards large scale enterprise management and event correlation across multiple network, server, desktop, and assessment agents.

Features include:

- *A scalable three-tiered management architecture*
- *Centralised command, control, and event management of network, server, and desktop sensors*
- *Centralised database-driven security analysis*
- *Simplified sensor deployment*
- *Remote, secure, roles-based user interface*
- *Logical “asset-centric” view of security data*
- *Group command and control of sensors*
- *Group-oriented data analysis*
- *Internet Scanner and System Scanner product support*
- *Third-party product integration*
- *“SecurityFusion” real-time event correlation and attack verification*
- *Automation of security update process*

Deployment Manager

The *Deployment Manager* can be used to install all of the SiteProtector components from a centralised computer anywhere on the network.

Once the Deployment Manager has been installed on one PC, the CD is no longer required to install either SiteProtector itself, or any of the other components of a Proventia system. Instead, a simple Web-based installation capability is provided which can be accessed from any host on the network.

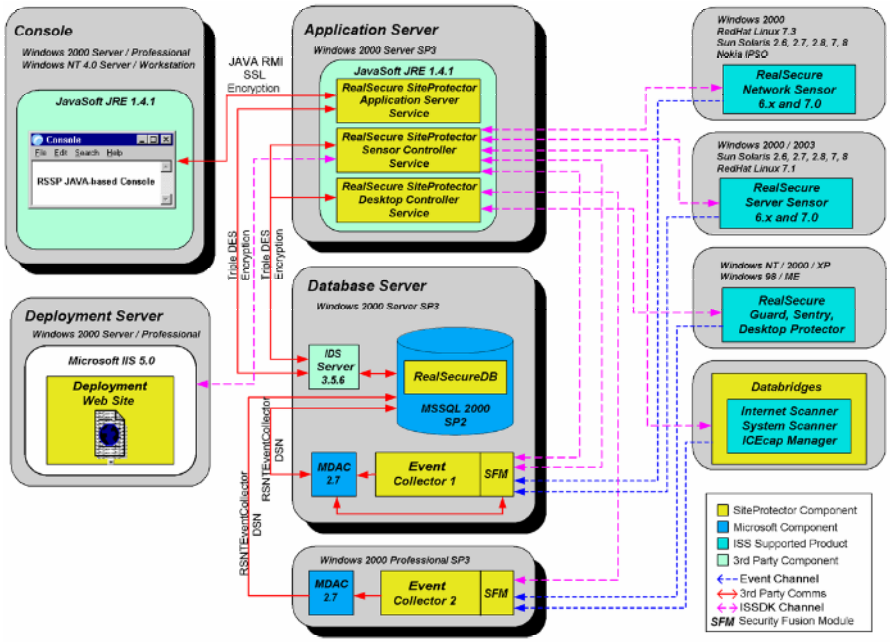

Figure 1 - Proventia: SiteProtector architecture

The Deployment Manager can be used to install the Basic or Custom SiteProtector installation, in addition to the following components:

- ***SiteProtector components -*** *Site database, Application Server and Sensor Controller, Event Collector, SiteProtector Console, Desktop Controller*
- ***Add-on components -*** *SiteProtector SecurityFusion module, Internet Scanner Databridge (which is no longer necessary when using Internet Scanner 7.0 for distributed scanning with SiteProtector), System Scanner Databridge*
- ***Components that are managed by SiteProtector -*** *Internet Scanner, RealSecure Network 10/100 and Gigabit, RealSecure Server, Proventia appliances, RealSecure Desktop*

Application Server

The *Application Server* enables communication between the SiteProtector Console and the Proventia Site database.

Sensor Controller

The *Sensor Controller* sends commands to the sensors, such as the command to start or stop collecting events. The Sensor Controller and the Application Server are installed on the same host.

Proventia Site Database

The *Proventia Site Database* stores the data that the Event Collector collects from sensors. This is based on Microsoft SQL Server for large deployments, or MSDE for smaller installations.

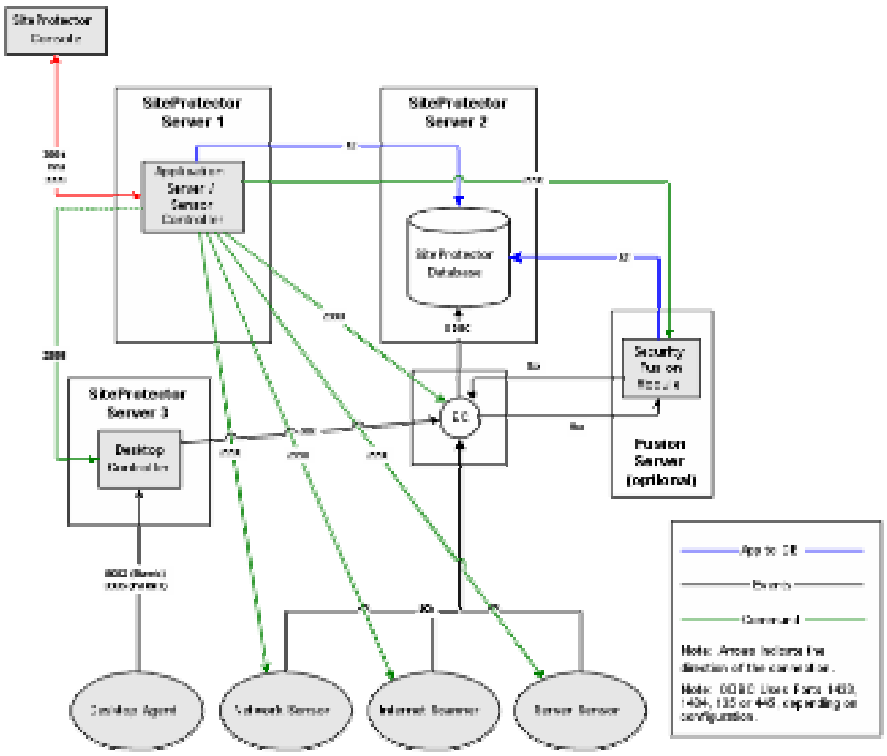

Figure 2 - Proventia: SiteProtector scales across multiple sites

Event Collector

The *Event Collector* pulls data from sensors and stores the data in the database. Typically only one Event Collector is installed on a SiteProtector Site, but up to five Event Collectors can be installed per Site when required for performance reasons.

SiteProtector SecurityFusion Module

The *SiteProtector SecurityFusion* module is an optional (extra-cost) module that correlates data from multiple sources, sources, including Proventia, RealSecure Network and Server agents, and Internet Scanner or System Scanner instances. This automated correlation escalates critical attacks and reduces false alarms.

SiteProtector Console

The *Console* is the graphical user interface (GUI) for the SiteProtector installation. With the Console, the administrator can perform a variety of activities, such as monitoring events and scheduling scans.

The specific tasks that can be performed using the SiteProtector Console depends on role assigned to the administrator.

The new Console is Java-based rather than the C++ code of the old Workgroup Manager, and one of the big advantages of that is the ability to spawn multiple windows within the same Console, each performing a different task. So while that large report is loading, it is still possible to be defining and deploying a new policy.

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 Proventia G200 was tested up to 200Mbps, the rated speed of the appliance. 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 Proventia G200 as a true 200Mbps device as claimed by ISS.

Latency figures were excellent for a device of this type at all traffic loads and with all packet sizes, ranging from 83.48µs with 50Mbps of 64 byte packets, to 135.62µs with 200Mbps of 1514 byte packets. Behaviour throughout the tests was predictable, with remarkably constant latency figures at all network loads with 440 and 1514 byte packets. Latency on 64 byte packets increased by 20 per cent as the load of 64 byte packets was increased from 50Mbps to the 200Mbps level.

Another surprise was that latency barely increased at all as we loaded the device with 100Mbps (half the rated speed of the device) of genuine HTTP traffic. However, 10Mbps of SYN flood traffic did have a serious negative impact on the Proventia. Although the SYN flood was effectively mitigated, the latency was increased to the point where we noted a significant number of failed connections in our HTTP traffic. The Proventia G200 is better as a straight IPS than as a DOS/DDOS mitigation device.

Apart from the DOS/DDOS condition, however, latency figures were considered to be acceptable for a device of this type, and much lower than we expected from a device using commercially-available hardware components.

The Proventia G200 performed consistently and reliably throughout our tests. Under eight hours of extended attack (comprising millions of exploits mixed with genuine traffic) it continued to pass legitimate traffic whilst blocking attack traffic in a consistent manner.

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, reporting to a single SiteProtector server. We used the *In-line Evaluation* policy, which has **all** attack signatures enabled (apart from port probes) and some key audit signatures.

Signature recognition (with blocking disabled) was excellent out of the box (95 per cent), and was increased to 100 per cent after the application of a signature pack update which was provided to us in less than 48 hours.

Blocking performance was one percentage point higher than detection performance out of the box, and was also increased to 100 per cent after the application of the signature pack update.

We noted a minimum of “noise” in this release, with few test cases raising multiple alerts for a single exploit. All our “false negative” (modified exploit) cases were detected correctly, demonstrating that the Proventia signatures are designed to detect the underlying vulnerability rather than a specific exploit.

A major concern in deploying an IPS is the blocking of legitimate traffic. There was one false positive alert out of the box, which was caused by the MDAC heap overflow signature being a little too generic and covering both the POST and GET conditions. Once this had been split into two signatures, we could block the POST and alert on (or ignore) the GET, thus achieving a clean 100 per cent.

The *Default-Blocking* policy includes over 180 built-in rules for out-of-the-box blocking against hybrid threats, including MS Blaster, SQL Slammer, Nimda, and Code-Red. Only certain critical signatures are configured to *block*, with the majority set to *alert only*. Given the quality of the signatures noted during these tests, we would be reasonably confident in deploying the Proventia G200 in a live network with the *Default-Blocking* policy activated.

Note that the Proventia can be configured in *Simulation* mode to enable to administrator to see the effects of applying a particular policy without the device actually blocking traffic.

Resistance to known evasion techniques was excellent, with Proventia being one of the few products to collect a clean sheet across the board in our evasion tests. *Fragroute, Whisker, ADMmutate* and even *RPC record fragging* all failed to trick Proventia into ignoring valid attacks.

Note that not only were the fragmented and obfuscated attacks blocked successfully, but every one of them was decoded accurately as well. This is the level of performance to which we would like to see all IDS and IPS products aspire.

Out of the box, the Proventia G200 handled up to 250,000 open connections - this will be increased to 500,000 connections with a future update pack. Although 500,000 connections is the maximum number of open connections officially supported by this device, it wasn’t until we loaded it beyond the 900,000 level that we started to see problems occur with the device running low on resources.

Default operation of the device is to allow all traffic when the state tables are full or resources are low - this naturally means that it is possible to evade the Proventia once the state tables are full, since it will allow attack traffic through at that point.

Changing a couple of parameters in the policy enables the administrator to alter this default behaviour to block all traffic by default when resources are low and to block all traffic for which there is no session established. This is how we configured the device for our tests in order to render the Proventia immune to stateless attack reply tools such as *Stick* and *Snot*.

The downside to this configuration is, of course, that when resources are low or the state tables are full, legitimate traffic can be blocked in error.

The fact is that there is no right or wrong way to do this - all IPS devices have to make the choice whether to risk denying legitimate traffic or allowing malicious traffic once they run low on resources. ISS has done extremely well here in allowing the administrator to choose for himself which of these risks he would prefer to take, and configure the Proventia accordingly.

Interestingly, the sensor did continue to track state on our “half open” exploits right up to 1 million open connections, since Proventia is designed to ignore new connections once the limit is exceeded, but does not age out old ones. We actually prefer old connections to be aged out, since the dropping of new connections does provide an easier method of evasion for the attacker when the device is configured to allow traffic from “unknown” connections. It would be nice to see this behaviour made configurable too.

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

As you would expect from an appliance that is designed to be as close to “plug and play” as possible, installation is very straightforward. Initial configuration is performed at the appliance, the simple configuration menu being presented automatically when logging in to the admin account. The admin interface on the Proventia G200 is designed to disallow any OS modifications, and the Root account should never be used - ISS will not support any configurations that are not accomplished from the admin account.

The simple, text-based menu enables the administrator to configure network settings, date and time, admin password, manually clear the dynamic blocking table, allow or disallow SiteProtector access, restart the protection Agent, configure the mode of operation (passive, inline or simulation), reboot and shutdown the appliance. A recovery CD provides the means to completely re-install the appliance software should that prove necessary (following disk corruption, for example).

Once configured, the Proventia appliance can only be accessed via a serial cable using a communications application such as Hyperterminal, keyboard and monitor, SSH, or SiteProtector.

Documentation is generally excellent for the Proventia range. It is provided as PDF files or hard copy, and includes an *Installation Guide* and *User Guide* for each of the components. The level of detail is good, and we found coverage of all the main features to be reasonably comprehensive and very clear. The *SiteProtector Strategy Guide* is an extremely useful additional manual which provides best practice guidelines and suggestions for securing a network, offering a guide to deployment and use in a range of organisation sizes and complexities.

Configuration

SiteProtector provides scalable, centralised security management and data analysis capabilities for all Proventia appliances, RealSecure Network, Server, and Desktop agents, and ISS' scanning applications. SiteProtector simplifies Proventia deployments through unified command, control and monitoring, the aim being to reduce security management demands on network traffic, staff or other operational resources.

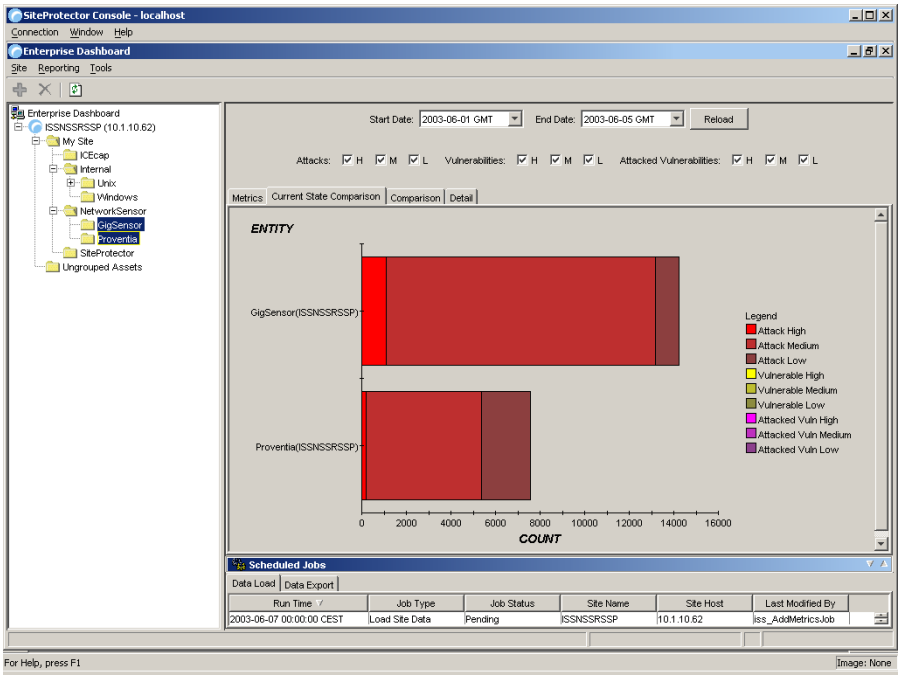

Figure 3 - Proventia: Enterprise Dashboard

The first goal of SiteProtector is to provide the administrator with a more logical view of the security boundaries under his control. Thus, rather than view individual sensors by machine name or IP address, they can be grouped together logically into departments or physical sites.
This makes it much more meaningful when viewing alerts or applying security policies.

Security assets can belong to more than one group for maintenance and reporting purposes, but can be “subscribed” to a single group for policy application (it is also possible to override this and apply policies on a per-sensor basis). This has been very well thought out, and provides one of the most flexible policy deployment capabilities we have seen.

Administrators can manage a wide range of sensors - both network and server-based - across the corporate network from a single Console if required. It is also possible, of course, to provide multiple Consoles to different administrators, and each one can be assigned different roles. For example, an administrator may control, configure and maintain the SiteProtector system and its components at the same time that localised security analysts perform command and control over the sensor infrastructure.

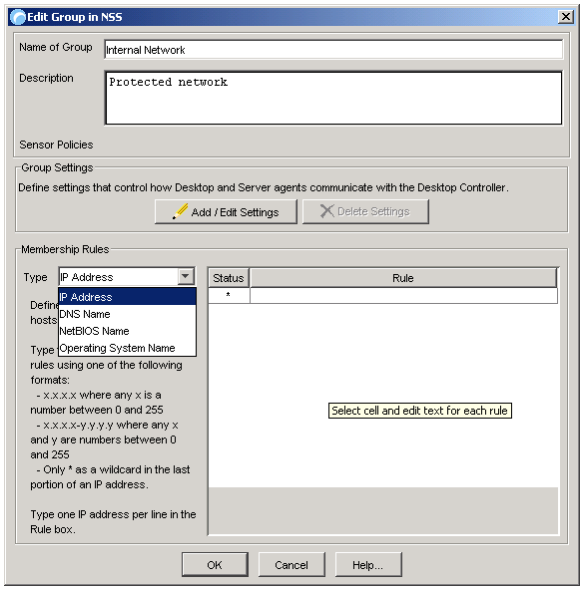

Figure 4 - Proventia: Creating groups in SiteProtector

However, these analysts remain restricted from configuring the SiteProtector system itself. Operators with viewing privileges cannot perform command and control functions, but may view the aggregated security information contained within a SiteProtector environment.

As with previous versions of the software, users are designated by assigning Windows users to specially created groups, which designate *Administrator*, *Operator* or *Analyst*. Within SiteProtector, these users can then be permitted or denied access to individual resources - groups or sensors - within the Site. Users can also be permitted global access to all Sites with a single login, or be forced to authenticate to each site individually as they drill down when performing analysis.

As new sensors are installed they can be registered manually or automatically (via a scanning operation) with the SiteProtector Console. As new assets (hosts or sensors) are registered, it is possible to auto-group them according to pre-defined rules (such as domain name, IP address range, and so on).

Since default policies can be applied at group level, it is possible to install software on a sensor (via the *Deployment Manager*), have that sensor register itself, assign it to a group and apply a policy without any direct action being required from the administrator. This is a nice feature.

Command and control of a group of sensors can be achieved with a single click by selecting multiple sensors and right-clicking, or by applying management operations at group or site level, in which case all sensors beneath are affected simultaneously.

Context-sensitive menus appear following a right click to provide the option to start and stop sensors, apply policies, apply global responses, apply updates, remove updates, or run vulnerability scans (if Internet Scanner is installed). Whenever a command and control operation is selected, the option is provided to run once or to schedule multiple repeated operations.

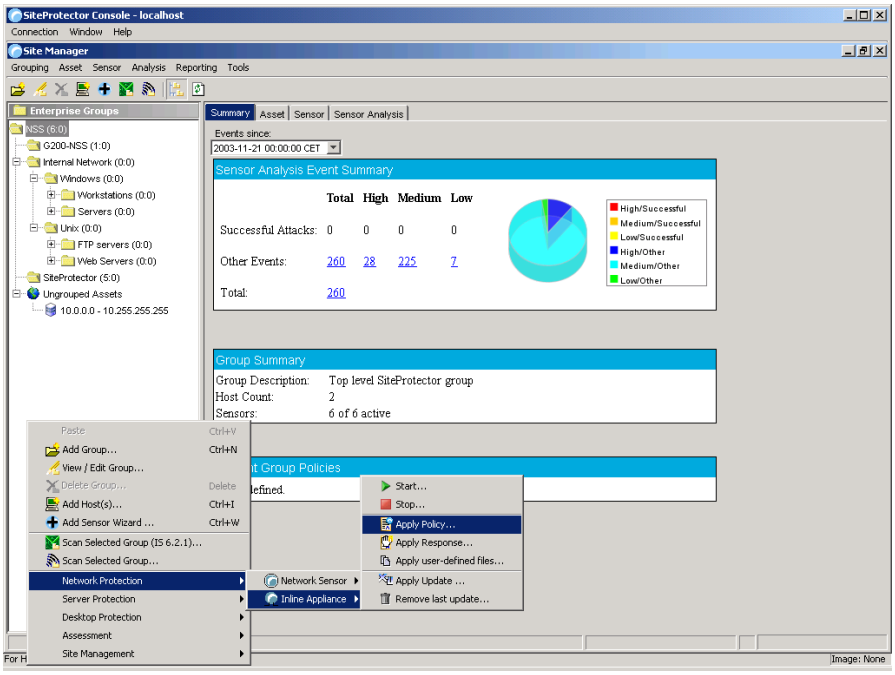

Figure 5 - Proventia: Managing groups within SiteProtector Console

This would allow the administrator to have different security policies applied across a range of sensors on weekdays to those applied at weekends, for example. The potential time savings for large-scale deployments are enormous, and the ability to organise assets by multiple nested groups within multiple sites and apply management operations at any level of the hierarchy make the system extremely scalable.

The latest release of SiteProtector has improved the database management capabilities by adding automated backups configured in the user interface by right clicking on the database, and sophisticated automated purging by event categorisation type. This latter feature enables the administrator to set a time interval for keeping exceptions, incidents and un-categorized events, and to set granular filters on specific events to define exceptions to the automated purging regime.

Policy Management

The most important configuration task, of course, is to assign a security policy, and five default - and fixed - inline policies are provided with the system (additional policies are available for managing other ISS components from the same Console). Different policies are defined with different modes of protection in mind.

For example, one is for detection only (no blocking enabled), whilst others are designated as blocker policies - one with audits enabled and one with attacks only. There is also an "*Evaluation*" policy which has all attacks enabled (apart from port probes) along with certain key audit signatures.

The majority of these signatures are configured to detect only, but those where ISS has determined that there is no risk of false positives are enabled in blocking mode.

There is no performance impact when all signatures are enabled. Even if a signature is turned off in a policy with Proventia, it is ignored for alerting purposes only, but is still included in the attack detection phase. Thus the Proventia effectively *always* runs with *all* signatures enabled.

The Policy Editor used in SiteProtector is the one remaining legacy of the old Workgroup Manager product, and is beginning to look a little “tired”. It is also showing signs of straining to accommodate the BlackICE integration, although the changes required since the last release in order to accommodate inline appliances have been integrated seamlessly.

It is certainly easy enough to create new policies - simply select one of the existing policies and click on “*Derive New Policy*”, give it a name, and you are free to create your own policy using that as a template. There are seven tabs on the policy definition screen, including:

- *Security Events*
- *Connection Events*
- *User Defined Events*
- *Packet Filters*
- *Event Filters*
- *Firewall Rules*

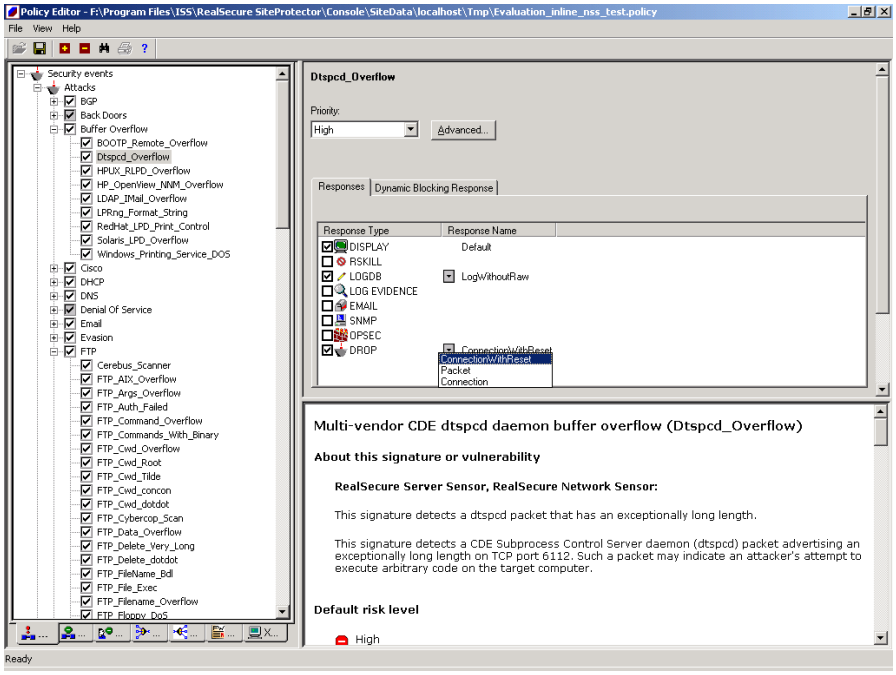

Figure 6 - Proventia: Policy Editor

The *Security Events* tab lists all the available attacks in the signature database, and these can be enabled or disabled by clicking on a check box next to each one.

The available signatures are split into two categories - *Attacks* and *Audits*. *Attacks* are those events which are deemed serious enough to warrant an alert (DOS attacks, Trojans, Web exploits, etc) on any network.

Audits are events which may be considered suspicious but may be prone to false positives if enabled on certain networks. For example, one Audit signature monitors all SNMP activity, which is only of use if you do not run SNMP on your network. Other Audit signatures monitor “reconnaissance” events such as FTP SYST, SMTP EHLO, or Bind Version requests.

Most companies would begin with the *Evaluation* policy, which includes all the serious attack signatures and some key audit signatures (this is the one we used for our tests). This can then be tuned to remove any false alarms and add in any Audit events which may be useful on a particular network. The *Attacks and Audits* policy would provide the most complete coverage out of the box, but would provide far too many false alarms and superfluous data on most networks.

Selecting an individual attack brings up a detailed description of the attack (including its effect and how to counteract it) and a configuration screen. There is a wide range of attacks available, and new ones are added at regular intervals through the efforts of the Internet Security Systems X-Force team.

The Web-based update procedure is very straightforward and easy to use, and allows the administrator to download new signature files (known as *X-Press Updates* (*XPU)*) to the network and update individual Proventia installations from those, all of which can be accomplished from the Console.

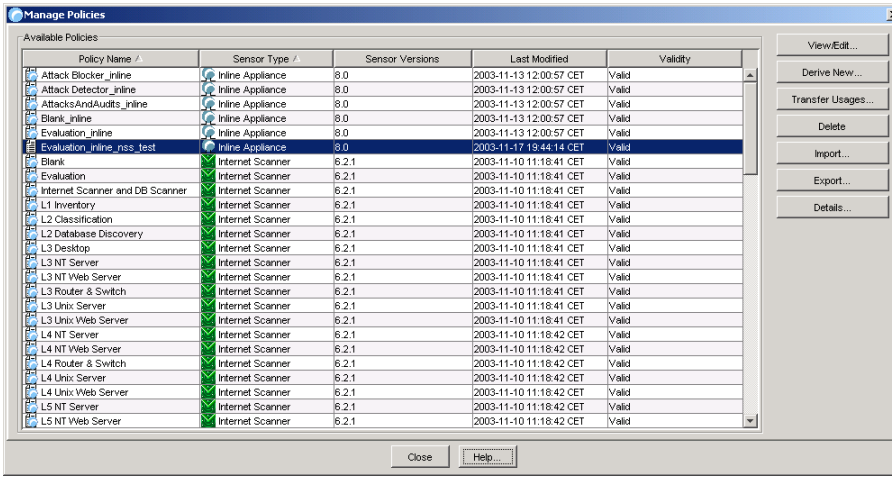

Figure 7 - Proventia: Managing Policies

Unfortunately, new signatures are only included on the XPU tab, and not in the main Security Events tab, which makes them very difficult to find unless you know exactly where they are. This problem should be eliminated by the use of the *Search* function, but for some reason this never searches beyond the first occurrence of the search string entered. Policy definition can thus be more difficult than it should be. It would be nice to see the Search function extended to produce search results consisting of multiple signatures, and also allow searching on additional criteria.

The attack priority sets the priority flag which can be used for filtering and reporting, and this can be escalated automatically by the Fusion module if it determines that an attack was successful.

For each event there are also a number of responses available, including:

- *Notify Console*
- *Log to database*
- *Log raw data*
- *Log evidence*
- *Send e-mail notification*
- *Kill connection*
- *View session*
- *OPSEC*
- *Send an SNMP trap*
- *Block*

Kill connection resets the IP connection to terminate the attack immediately, whilst the OPSEC option allows Proventia to automatically reconfigure any OPSEC-compliant firewall to prevent further attacks. Both of these would be used mainly when the device is configured to operate in passive IDS mode. When operating in inline mode, the ability to drop packets and block sessions is more useful, since it provides the means to drop the offending packet immediately and block all subsequent traffic that is a part of that same TCP flow.

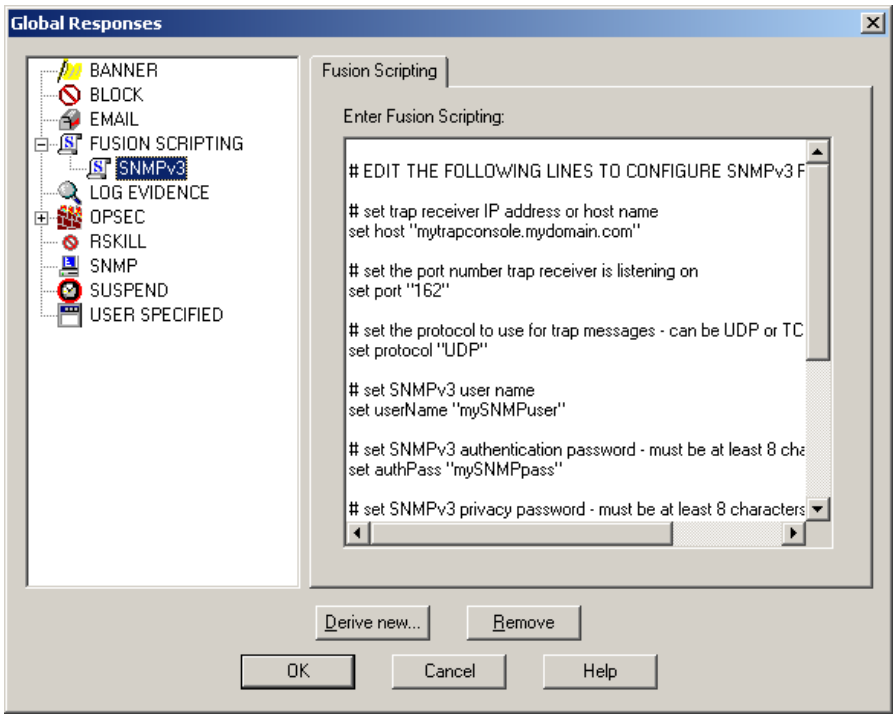

Figure 8 - Proventia: Configuring Global Responses

Dynamic blocking is an extension of this, that enabled the administrator to determine that all subsequent traffic from the offending host or to the target that is under attack is blocked automatically and immediately. This facility needs to be used carefully since it has the potential to cause a DOS condition if any of the addresses in the attack packet are spoofed. However, it does provide the means to effectively mitigate heavy attacks from one particular source or targeted at one particular server.

Response settings become the defaults for individual signatures, but these can be overridden in bulk by applying different response settings at sensor, Group or Site level.

There is also an *Advanced* properties screen that theoretically provides the means to edit any additional parameters that might apply to a specific attack signature, as well as define how multiple events should be handled and propagated from sensor to Console.

Both the Event propagation control - the means by which rapidly occurring attacks can be consolidated into a single alert to prevent network floods - and tuning parameters for individual signatures tend not to be found here any more, however, since following the integration of the BlackICE technology they are now made available via a set of parameters that are applied at the sensor level. No doubt this dialogue box will disappear in future releases once the new common policy editor is released and customers have migrated from the 6.5 sensors.

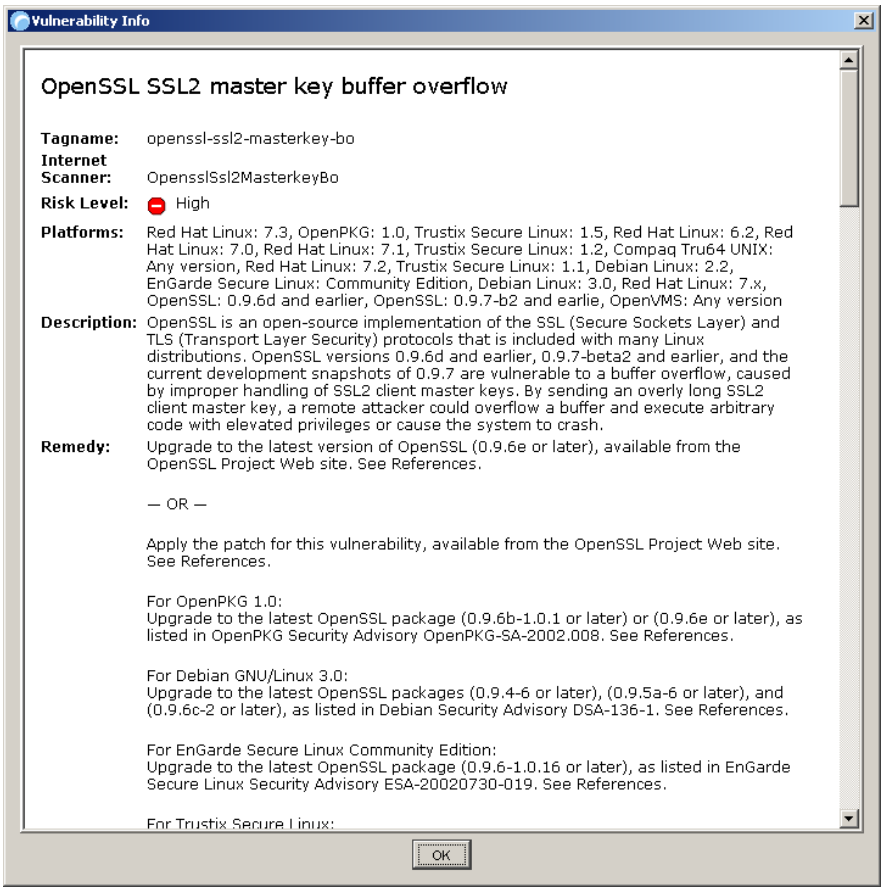

Figure 9 - Proventia: Vulnerability information stored against each signature

Event coalescing worked extremely well in our tests, with each alert generated containing a repeat count to show how many attacks of that type were actually detected. This count remains accurate until the sensor is under extreme loads, at which point a “pre-coalescer” cuts in to drop a percentage of identical packets before they are passed to the parsing engine.

Proventia can be used to monitor more than just security problems by using the *Connection Events*. These are generic events such as HTTP, FTP or SMTP activities, and can be filtered by source or destinations address, source or destination port, or protocol. One possible example of how this could be used would be if company policy forbids the use of Telnet.

Proventia could be configured to monitor for TCP port 23 and log the source and destination addresses of the perpetrators, perhaps blocking the process completely too.

User-defined signatures allow the administrator to apply regular expression string matching on certain fields within a packet (login name, URL data, e-mail subject, etc.) to determine suspicious content. This could be used as a "quick and dirty" virus checker by creating a signature that raises an alert every time it detects a packet with a specific string in the e-mail subject field, or to monitor and alert on attempts to login as Root.

User-defined signatures containing regular expressions can be a performance drain. However, this tends to be more of an issue in Gigabit appliances, and it was not necessary to turn off RegExp parsing of custom signatures in order to achieve 200Mbps detection rates.

Note that this is the limit of signature customisation via the Console interface – neither RealSecure nor BlackICE have ever been particularly flexible when it comes to signature customisation (especially of the built in signatures, which remain virtually untouchable in this latest release). However, Proventia does include the *Trons* module which provides the means to add completely custom rules using the Snort rules definition syntax.

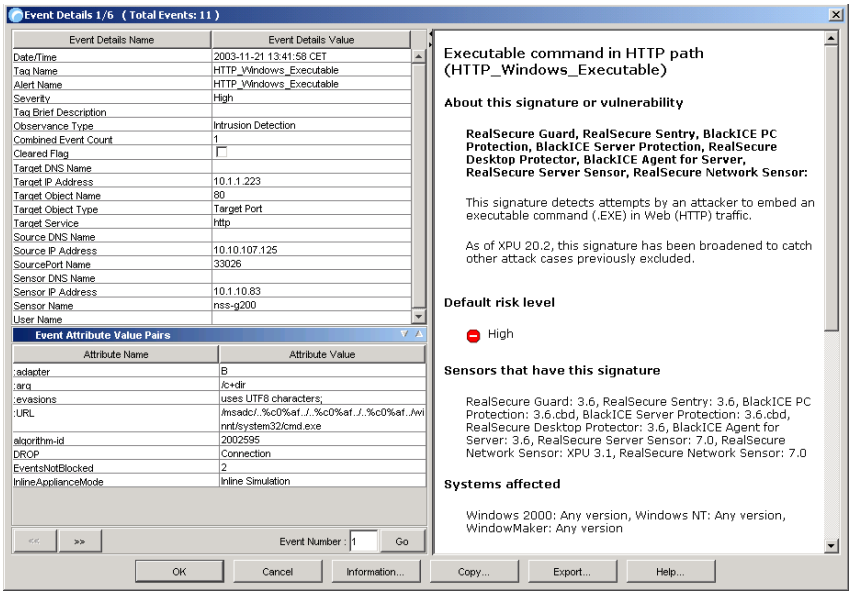

Figure 10 - Proventia: Viewing Event details

Although neither TCP reassembly nor any of the standard Snort plug-ins and pre-processors are supported, Trons does allow the administrator to add brand new signatures using a familiar "language" that is reasonably easy to learn and simple to use. And, of course, there is an entire library of signatures ready made on the Snort Web site, though Internet Security Systems does not recommend you add them all.

Apart from a significant overlap in functionality between the Snort and built-in signatures, there would be a heavy performance impact. Because the Trons module is currently completely separate from the Proventia PAM (this will not always be the case in future releases), the signature parsing path is not exactly optimised, and thus Trons signatures should be kept to a minimum.

We found it very straightforward to add Snort signatures to our Proventia sensor, and the resulting alerts are displayed in the SiteProtector Console alongside all the built-in alerts, but including Snort-specific data in the Event Inspector window, such as the Snort SID, external reference, version number, and so on.

Finally, if there are genuine events which are raising false alarms via Proventia, the *Packet Filter* tab allows the administrator to define which packets should be ignored for alerting purposes, based on protocol, port or IP address. All packets matching one of the defined packet filters are rejected before being processed by the parsing engine.

Where it is still required to disregard specific events that have been detected by the sensor - perhaps because they are known to be false positives when coming from a particular host - then the *Event Filters* can be used. Here, the administrator can specify an event name, together with source and destination IP address and port, and any event detected by the sensor matching these criteria is simply ignored.

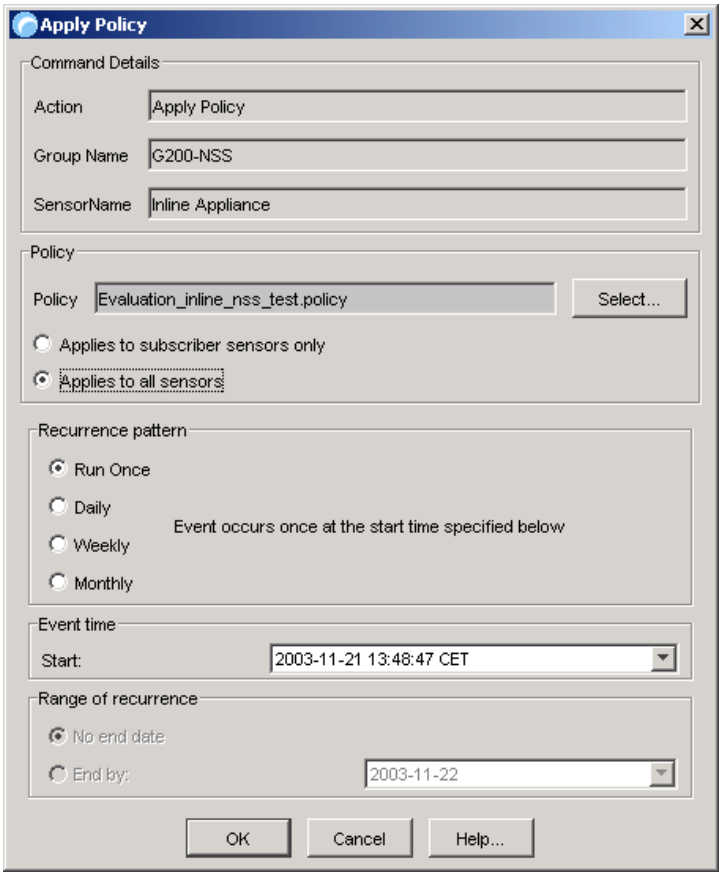

Figure 11 - Proventia: Applying Policies

Once policies have been defined they are applied to individual sensors, Groups or Sites via the SiteProtector Console. Multiple policies can be defined for different functions (i.e. DMZ or internal network) and applied to a range of sensors via the asset hierarchy. Policy application can be performed as a one-off process or scheduled for regular runs, thus allowing administrators to apply different policies at different times (days and nights, weekdays and weekends, and so on).

If the administrator makes a change to an existing policy, SiteProtector will determine which sensors are using that policy and provide a list to the administrator allowing him to deploy the policy to all sensors in one go, or just a selection of them (by deselecting checkboxes).

The existing Policy Editor in SiteProtector 2.0 is actually one of the few remaining legacies from the old Workgroup Manager Console. This will be replaced in the near future by a new Java-based *Common Policy Editor* that will provide a single interface for all types of security policies to be applied via SiteProtector.

Alert Handling

Once a sensor is up and running with a policy applied, alerts are logged locally on the sensor (if required) and transmitted in real time to the designated Event Collector, from where they are stored in the database. At user-defined intervals, the SiteProtector Console retrieves new events and displays them in the *Analysis* pane in a “spreadsheet” format.

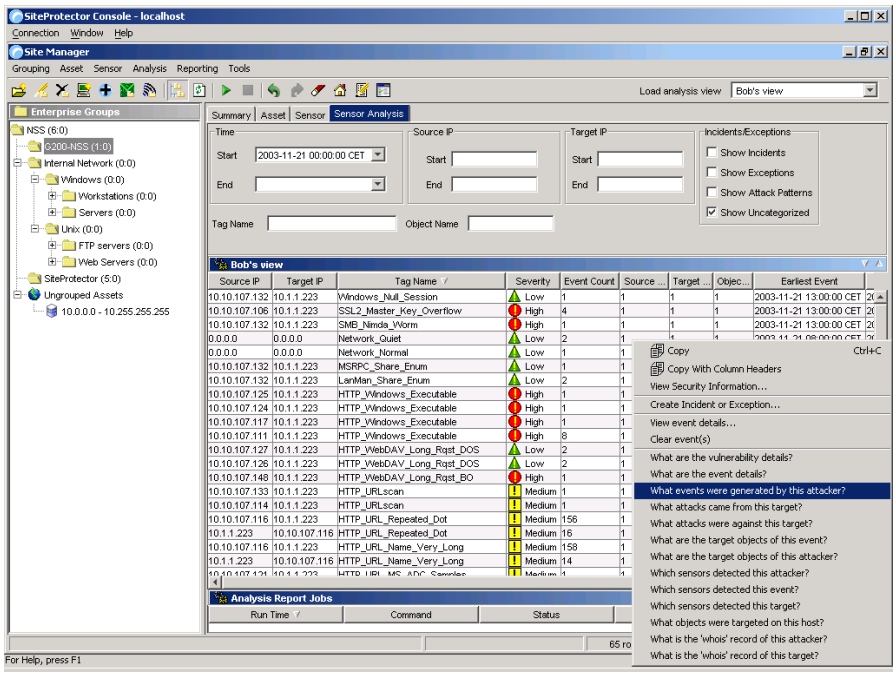

Figure 12 - Proventia: Viewing security alerts

Each row of the spreadsheet displays the complete alert details, including date and time, source and destination IP address, source and destination port, severity, status, URL/command that cause the alert, and so on. Right clicking on an alert provides the ability to view all alert details in a single window, which makes it more readable. It is also possible to call up the detailed X-Force information on that alert, but the ability to view detailed packet contents is missing.

A number of other options on the right-click menu guide the administrator through the process of “drilling down” through the data to get to the most important and relevant information. This is done by providing a number of plain English “questions” such as “*What are the event details?*”, “*What events were generated by this attacker?*” and “*What attacks were against this target?*”.

A number of filter options are provided along the top of the screen, allowing the administrator to home in on the data that is of interest by specifying source and target IP addresses (or ranges), ports, date and time stamps, and so on.

One extremely useful feature is the ability to create “baselines”. At any given point in time, the current analysis view can be “frozen”, which maintains the totals and counts (such as event count, target count, and so on) on screen. Any increases or decreases in those totals are then shown in red as plus or minus figures from the baseline, making it easy to spot trends during an investigation.

The baseline enables the administrator to determine, at a glance, how events have changed in an analysis view. If, for example, he notes that one IP address or tag name is associated with an unusually high *increase* in the number of events, he can investigate it to determine whether it represents a threat.

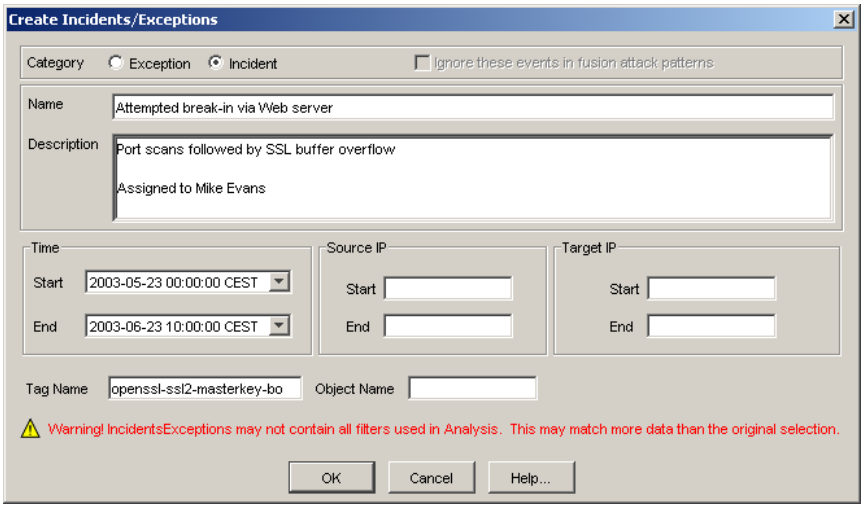

Figure 13 - Proventia: Creating Incidents and Exceptions

Different analysis views can be loaded directly from a drop-down menu, each one providing a different data layout (a different “report format” if you will), and these are almost infinitely customisable via the ability to define column layouts and filters to be applied to the underlying data. Once the administrator has fine-tuned the analysis to his satisfaction it can be saved for subsequent recall.

Events that are deemed unimportant can be designated as “*Exceptions*” and they will subsequently not appear in analysis views. This does not prevent them from being detected in the first place, merely from being displayed for analysis - it would be nice if there were a right-click option to enable the signature that caused the alert to be disabled in the policy.

Should the administrator determine that a group of events are related (perhaps a port scan, followed by a buffer overflow, followed by attacks launched from the compromised machine) they can be groups together as an “*Incident*”. These events are then removed from the normal analysis display and are shown only in the Incident analysis views.

This allows the administrator successively to reduce the data displayed, resulting in the ability to “find the needle in the haystack”.

Incidents are tracked as a unit from that point on, and the incident can be annotated with actions taken to resolve it. This is an extremely useful feature, and would be even more useful if it were possible to flag each incident with a status (pending, resolved, under investigation, etc) and an owner.

SiteProtector's modular format enables plug-in enhancements for a wide range of security management needs, and the *SecurityFusion* module is the first plug-in module for SiteProtector. This extra-cost module uses data correlation and analysis to rapidly and automatically derive the likelihood of a successful attack from aggregated vulnerability assessment information. Visual cues in the SiteProtector Console indicate attacks with a high probability of success, with automatic escalation criteria for critical security events.

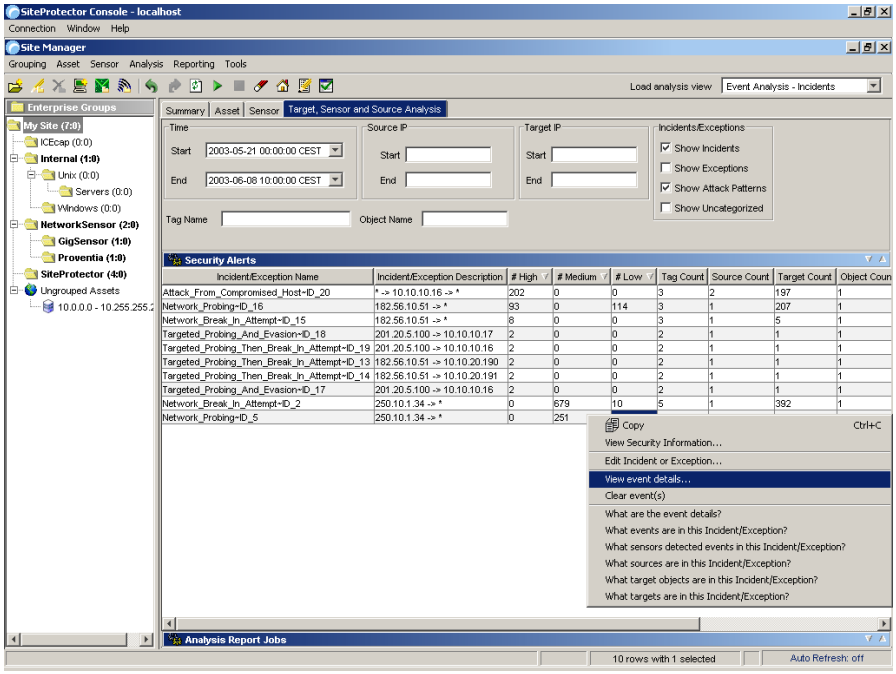

Figure 14 - Proventia: Investigating Incidents created by the SecurityFusion module

For example, the module can escalate important events by generating additional responses outside the Console (such as email or SMTP). Alternatively, it can de-emphasise less important events by reducing alert priority or by selectively preventing an event from being displayed or logged. All escalation and de-escalation options are fully customisable.

During testing, we noted that a specific alert - a DNS Zone Transfer - was escalated from the normal event priority of *medium* to *high* because Internet Scanner had previously determined that the particular host against which the transfer was made was potentially vulnerable to spurious zone transfer requests. On the other hand, it would be possible to "downgrade" an alert from high to medium or low if the attempted attack - say an FTP exploit - was against a host with no vulnerable FTP service running.

This approach can help to reduce false alarms, and can certainly reduce the load on the administrator since it would be possible to record all suspicious events for trend reporting and forensic analysis, whilst only alerting on events where there is a real chance of an exploit targeting a vulnerable host.

The potential downside, of course, is the necessity to run Internet Scanner at sufficiently regular intervals to make the data correlation effective, but SiteProtector makes it easy to schedule Scanner runs in order to achieve this. In addition to the escalation and downgrading of alerts, Fusion also records its assessment of the alert (failure, likely to have succeeded, etc.) in the alert details, making it available in analysis views and reports.

The most valuable feature of Fusion, however, is the ability to analyse and group alerts into Incidents. If Fusion detects a pattern of events similar to the one we mentioned earlier - a port scan to a particular host, followed by a potentially successful exploit, followed by further port scans and exploits launched **from** that host (indicating that it had probably been compromised) - then it would group all of these alerts together into a single Incident.

Because alerts within Incidents are removed from the normal analysis review, the result is a much smaller number if unclassified events to deal with. And by focussing on the Incident analysis first, of course, the administrator can be sure of spending his time dealing with genuine problems. Naturally, when operating inline it still remains within the administrators control whether to block the individual events or to alert only.

The alert-handling capabilities of SiteProtector 2.0 are a huge improvement over previous RealSecure management consoles, and this is one of the better IDS consoles we have seen in our labs.

Reporting and Analysis

ISS has eliminated the very basic, fixed reports that were available in Workgroup Manager and provided instead the almost infinitely customisable analysis tool that we discussed in the *Alert Handling* section.

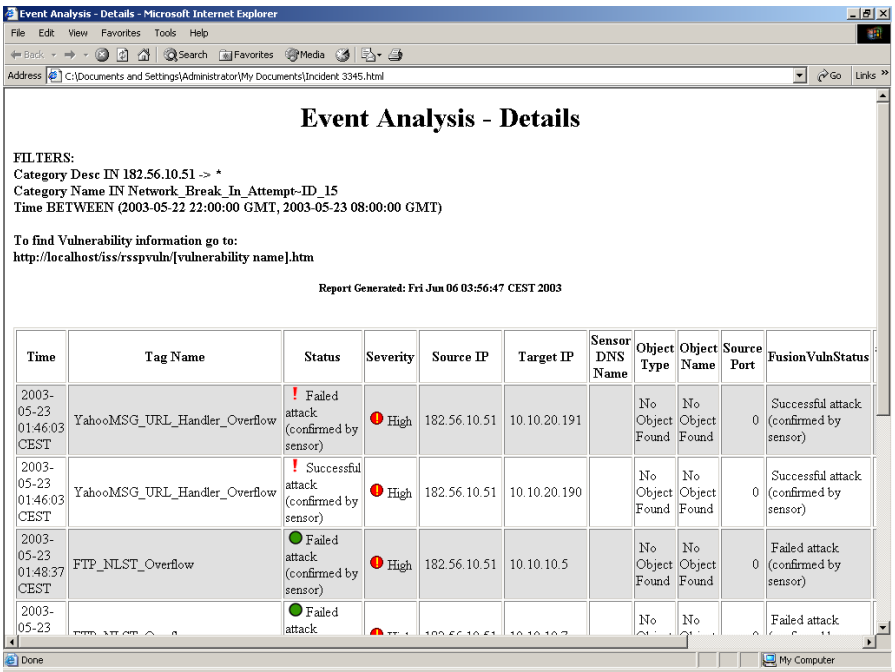

Figure 15 - Proventia: Typical report

A number of very useful basic views are provided out of the box, including *Attacker, Details, Event name, Incidents, Sensor,* and *Target.*

However, the real power of the system lies in the fact that it is possible to create a limitless supply of customised views by specifying your own column layouts and filters (and not a Crystal Report in sight!). These custom views can be saved for later recall, and can also be saved as reports in PDF, HTML or CSV format.

When saving HTML files, an index is automatically created of all reports saved, making it easy to publish management reports to a Web site if required. All reports can also be scheduled to run at regular intervals, and query performance has been improved significantly in the latest release.

In addition to the analysis views and text-based reports, SiteProtector also provides the Enterprise Dashboard. The Enterprise Dashboard allows the administrator to:

- *View a high-level graphical summary of alert activity for a site (this is the default view in the new release)*
- *View metrics and graphs for an enterprise or a site*
- *Export reports*
- *Create groups for specific sensors or assets monitored*
- *Perform high-level command and control functions, such as associating users and groups, which determines the specific sites, groups, and subgroups that users can access*
- *Drill down to examine site data in more detail*

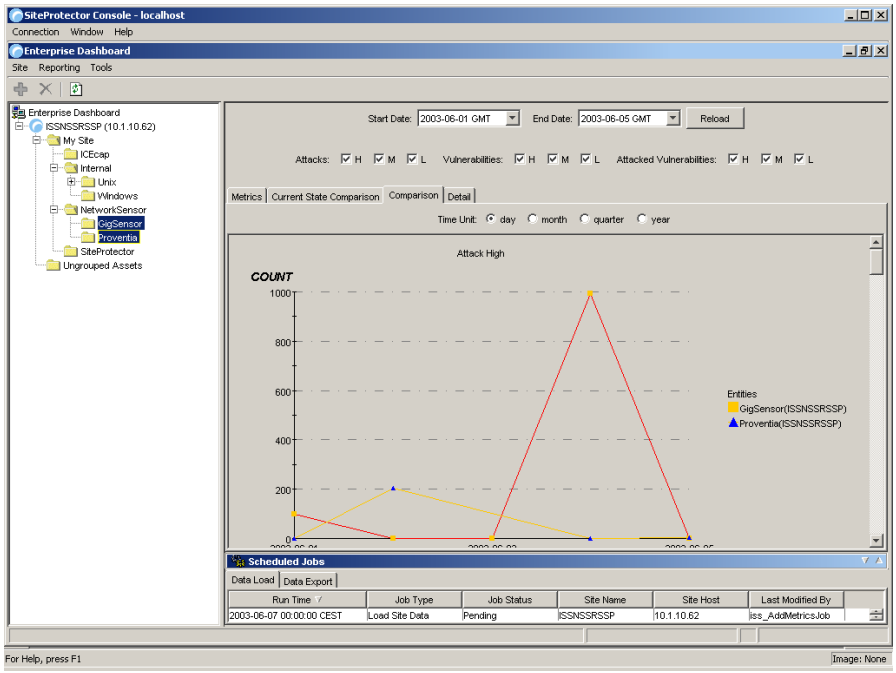

Figure 16 - Proventia: Sensor comparison in the Enterprise Dashboard

The *Metrics and Trends* pane enables you to view security results for specific assets, sensors, and sites, and to create reports that display trends. The information in the Metrics and Trends pane is based on the filters you select in the Filters pane. This displays information on the following tabs:

- ***Metrics -*** *Volume of events by type of event for the group selected. The administrator can select multiple groups for the display.*

- ***Current State Comparison -*** *Bar chart of event volume by priority and type of event. It is possible to select multiple groups for the display.*
- ***Comparison -*** *Trends for the type of events by category and priority. Trend lines are displayed for each selected group, and it is possible to select multiple groups for the display.*
- ***Detail -*** *Trended stacked chart showing the volume of events. The administrator can select multiple groups for the display, and data is displayed for the dates chosen in the time filters.*
- ***Configuration****—Information is shown in tabular format about the selected groups. Information includes name of group, its site, and the applicable policy or X-Press Update.*

Even though the detailed packet contents are not stored as part of the Proventia database, it is possible to store detailed packet logs of all traffic seen by the sensor (packet logging) or just packets associated with specific signatures (evidence logging).

When packet logging is enabled, Proventia records all system traffic into log files, the size of the files and rotation characteristics controlled via the *Advanced Properties* tab for each sensor. It is important to note that packet logging keeps track of **all** system traffic, not just intrusions, which can result in some large files and would obviously have quite an impact on performance.

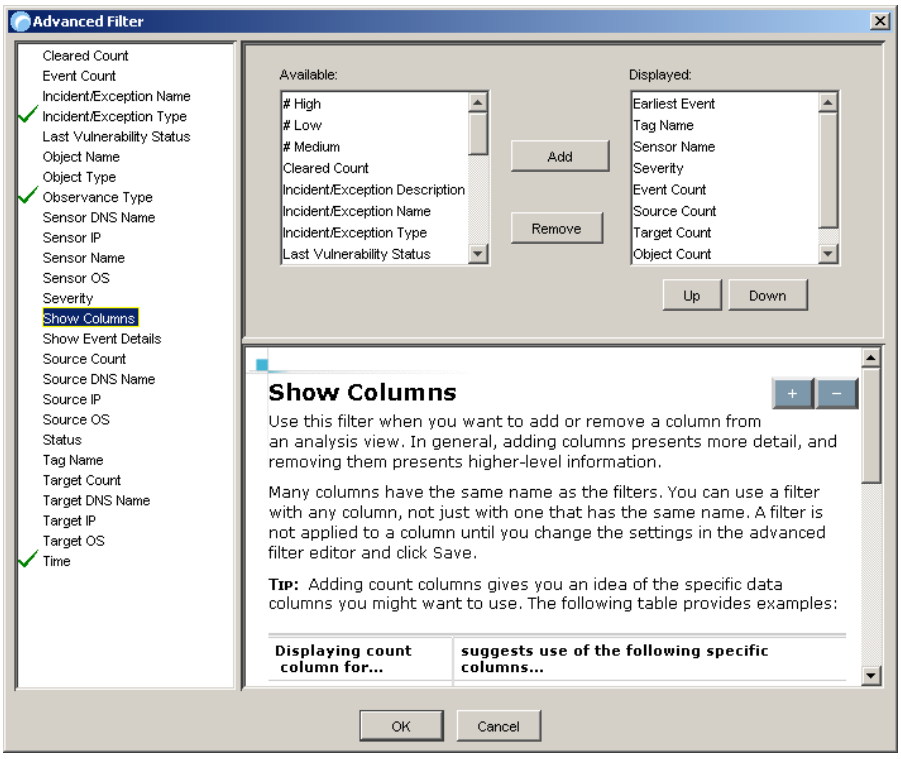

Figure 17 - Proventia: Creating custom filters for reporting

If it is required to be more selective in what is recorded, evidence files can be used. These are controlled via the global or per-signature settings in the policy files. Whenever a particular attack is detected that is flagged as "*Log Evidence*", Proventia captures network traffic specific to the attack in progress and stores that information in an evidence file.

Both Packet and Evidence Logs are encoded as *Sniffer* or *tcpdump* trace files, and will require a decoding application (which is not included) to view the contents.

Unfortunately, these log files must be retrieved manually from each sensor (although they can be retrieved centrally via the Console) and there is nothing to tie the individual alerts in the database to specific evidence files, making forensic analysis more difficult than it should be.

Verdict

Performance

Clearly a lot of effort has been put into designing the Proventia hardware to cope with the bandwidth claimed (200Mbps on the G200) and in producing custom drivers for the built in network cards. This care allowed Proventia to detect and block 100 per cent of attacks under all of our test conditions. We would have no hesitation in rating the Proventia G200 as a true 200Mbps device.

Latency figures were excellent for a device of this type at all traffic loads and with all packet sizes, ranging from 83.48µs to 135.62µs depending on traffic load and packet size. Behaviour throughout the tests was predictable, with remarkably constant latency figures at all network loads with 440 and 1514 byte packets, and only a slight increase with 64 byte packets as the load was increased. In addition, there was hardly any increase in latency when we loaded the device with 100Mbps (half the rated speed of the device) of genuine HTTP traffic.

However, 20Mbps of SYN flood traffic did have a serious negative impact on the latency performance of the Proventia, although the SYN flood was effectively mitigated. The Proventia G200 is better as a straight IPS than as a DOS/DDOS mitigation device.

Overall, latency figures were considered to be acceptable for a device of this type, and much lower than we expected from a device using commercially-available hardware components.

The Proventia G200 also performed consistently and reliably throughout our tests, continuing to pass legitimate traffic whilst blocking attack traffic in a consistent manner even when under extended attack. 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.

Security Effectiveness

Signature recognition (with blocking disabled) was excellent out of the box (95 per cent), and was increased to 100 per cent after the application of a signature pack update which was provided to us in less than 48 hours.

Blocking performance was one percentage point higher than detection performance out of the box, and was also increased to 100 per cent after the application of the signature pack update.

All our “false negative” (modified exploit) cases were detected correctly, and none of the false positive test cases triggered once the signature update had been applied. Resistance to known evasion techniques was also excellent, with Proventia being one of the few products to collect a clean sheet across the board in our evasion tests. Not only were the fragmented and obfuscated attacks blocked successfully, but every one of them was decoded accurately as well. This is the level of performance to which we would like to see all IDS and IPS products aspire.

We liked the fact that the device can be quickly switched to *Simulation* mode to enable the administrator to see the effects of applying a particular policy without the device actually blocking traffic. We also liked the fact that it is possible to configure the device to block **or** permit traffic by default when state tables are full or when it runs low on resources.

Usability

With the removal of the aged *Workgroup Manager* Console and its replacement with the more scalable and flexible *SiteProtector*, Proventia has taken a huge leap forward.

The only significant portion of “legacy” code remaining in the user interface is the Policy Editor, which is certainly showing signs of the strain involved in making two quite different technologies - BlackICE and RealSecure - work together as seamlessly as possible. This should be replaced in the not too distant future with the new Java-based *Common Policy Editor* which, as its name implies, will provide a common policy management interface across the entire Proventia range.

The old Workgroup Manager Console was one of the most comprehensive and easy to use out of the box and for a long time was the benchmark against which all other IDS consoles were measured. With SiteProtector, ISS has done it again, producing one of the best consoles we have seen in our labs to date. It is worth pointing out that this is no longer an extra cost option, but is included in the price of the sensor (the SecurityFusion module remains an extra-cost option, however).

Sensor and policy management are amongst the most straightforward and scalable that we have seen to date. Policies can be applied to multiple sensors at the click of a mouse, and the use of rules to auto-group assets together with the ability to apply policies at site or group levels means it is very easy to install and activate sensors with little or no administrator intervention. Currently, Proventia is probably one of the easiest products to deploy across a large, distributed network that we have seen.

Alert handling, too, is excellent, with SiteProtector attempting to simplify the life of the administrator via automatic impact analysis and event correlation across vulnerability assessment tools and network sensors.

The result should be the ability to reduce the number of critical alerts appearing at the Console to a more manageable level in even the largest installations by grouping them together and tracking them as “incidents”. The ability to report and annotate at an incident level is useful, as is the ability to **manually** correlate multiple events into a single incident if required.

Built in reports have given way to infinitely customisable views which provide the means to drill down into the event data from almost any angle. The resulting views can be saved and recalled on demand or run at regular intervals via a scheduler, and the Dashboard provides high level graphical summaries. It is nice to see a company producing a genuine **advance** in reporting and analysis tools rather than simply relying on Crystal Reports and a few basic pre-defined templates.

Contact Details

Company name: Internet Security Systems, Inc.

E-mail: sales@iss.net

Internet: www.iss.net

Address:
6303 Barfield Road
4th Floor
Atlanta
GA 30328
USA

Tel: +1 404-236-2600 or 1-800-776-2362
Fax: +1 404-236-2614

APPENDIX A – TEST RESULTS

The aim of this procedure (based on V1.0 of the NSS Group Network IPS Testing Methodology) 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 a mix of Allied Telesyn AT-9816GB and AT-9812T switches (these have a mix of fibre and copper Gigabit interfaces). All IPS devices are expected to be provided as appliances - if software-only, the supplier pre-installs the software on the recommended hardware platform. The IPS 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 Smartbits *receive* port - is connected to the internal network. The IPS 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 IPS 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 IPS 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. All exploits are run with no load on the network and no IP fragmentation.

Our attack suite contains over 100 exploits covering the following areas:

- ***Test 1.1.1 -*** *Backdoors (standard ports and random ports)*
- ***Test 1.1.2 -*** *DNS*
- ***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*

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 IPS 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 IPS device could block legitimate traffic inadvertently.

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

Section 2 – IPS 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.

- ***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*

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*
- ***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*

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 IPS 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 such as *Stick* and *Snot* - 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 tools such as *Stick* and *Snot* 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. 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.

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 IPS 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 IPS sensor). The initial HTTP session is then completed with the second half of the exploit and the session is closed. If the IPS 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 - an IPS 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 IPS engine 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). This test is run using the default sensor settings.

- ***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 IPS device. 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 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 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 IPS 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 IPS device bi-directionally (maximum of 1Gbps, or 500Mbps in each direction). 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, and the aim of this test is purely to determine the raw packet processing capability of the IPS device, and its effectiveness at passing “useless” packets quickly in order to pass potential attack packets to the detection engine.

- ***Test 4.1.1 - 64 byte packets - maximum 1,480,000 packets per second:*** *The "torture test" - it is unlikely that any real-life network will ever see network loads of almost 1.5 million 64-byte packets per second. This test therefore measure packet processing performance under the most extreme conditions. Repeated with 250Mbps, 500Mbps, 750Mbps and 1000Mbps of background traffic.*
- ***Test 4.1.2 - 440 byte packets - maximum 260,000 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 - 1514 byte packets - maximum 81,720 packets per second:*** *This test is the complete opposite of the 64 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 1514 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 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 IPS sensor.

The use of the Spirent Communications Gigabit **Avalanche** and **Reflector** allows us to create true "real world" traffic at speeds of up to 2.2 Gbps as a background load for our IPS tests. Our Avalanche configuration is capable of simulating over 2.5 million users, with over 2.5 million concurrent sessions, and almost 100,000 HTTP requests per second.

By creating genuine session-based traffic with varying session lengths, the IPS 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 1200 bytes - maximum 100,000 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 IPS 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 230,000 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 IPS sensors to perform well in this test.*
- ***Test 4.2.3 -*** *Max 10,000 new connections per second - average packet size 440 bytes - maximum 280,000 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 IPS sensor, and represents a very heavily used production network.*
- ***Test 4.2.4 -*** *Max 20,000 new connections per second - average packet size 350 bytes - maximum 360,000 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 IPS 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 230,000 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 IPS sensors to perform well in this test.*
- ***Test 4.3.2 -*** *Max 10,000 new connections per second - average packet size 440 bytes - maximum 280,000 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 IPS 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 (560 byte packets) + 20% FTP traffic + 6% UDP traffic (256 byte packets). Max 380 new connections per second - average packet size 555 bytes - maximum 22,000 packets per second - maximum 136 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 IPS 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 P4 SuperMicro server with Gigabit interface. This server holds a copy of The NSS Group Web site, and is capable of handling 950Mbps 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 **25 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 much more "bursty" in nature.

- ***Test 4.5.1 - Pure HTTP Traffic (simulated browsing session on NSS Web site):*** *Max 100 new connections per second - 25 new users per second - average packet size 1000 bytes - maximum 110,000 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 IPS 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 550 new connections per second - average packet size 900 bytes - maximum 130,000 packets per second - maximum 11,000 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 IPS sensors to perform well at all load levels in this test.*

Note that the average packet sizes during this test do not match the various studies on Internet packet size distribution (we consider the average packet size to be in the region of 440-550 bytes).

However, given that this is a test which utilises real browsing sessions against a real server on a real network, the resulting packet distribution should be considered valid.

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 IPS 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.

Test 5.1 - Latency

We use Spirent SmartFlow software and The SmartBits SMB6000 with Gigabit TeraMetrics cards to create multiple traffic flows through the IPS 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 (500Mbps in each direction, 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 (64 bytes, 440 bytes and 1518 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 IPS 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 115,000 packets per second). A 250Mbps load (125Mbps bi-directional) of SmartFlow traffic at various packet sizes (64 byte, 440 byte and 1514 byte) 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 load (125Mbps bi-directional) of SmartFlow traffic at various packet sizes (64 byte, 440 byte and 1514 byte) is then passed across the infrastructure switches and through the device and the packet loss and average latency are recorded.

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 IPS 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 115,000 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 IPS 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 IPS 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 IPS device 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 IPS device 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 IPS sensor, and the ISIC target directly to the internal interface. ISIC traffic is transmitted through 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 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. We also note whether the ISIC attacks themselves are detected by the sensor even though targeted at the management port.*

ISS Proventia G200 Revision A 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	5	5	5	5	5
Test 1.1.2 - DNS	2	2	2	2	2
Test 1.1.3 - DOS	12	11	12	12	12
Test 1.1.4 - False negatives (modified exploits)	13	12	12	13	13
Test 1.1.5 - Finger	4	4	4	4	4
Test 1.1.6 - FTP	5	5	5	5	5
Test 1.1.7 - HTTP	40	37	37	40	40
Test 1.1.8 - ICMP	2	2	2	2	2
Test 1.1.9 - Reconnaissance	8	8	8	8	8
Test 1.1.10 - RPC	3	3	3	3	3
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
Total	**98**	**93 / 98**	**94 / 98**	**98 / 98**	**98 / 98**

Test 1.2 – Resistance to False Positives	Pass/Fail
Test 1.2.1 - Audiogalaxy FTP traffic	PASS
Test 1.2.2 - MDAC heap overflow using GET instead of POST	PASS[1]
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 request ID	PASS
Test 1.2.14 - Fake DNS /bin/sh buffer overflow	PASS
Test 1.2.15 - Inter-firewall communication traffic (looks like FW-1 RDP header firewall bypass)	PASS
Test 1.2.16 - Fake SQL Slammer traffic	PASS
Test 1.2.17 - File copy of GIF file (contains bytes which look like MS Blaster 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 numbers 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
Total	**19 / 19**	**19 / 19**	**19 / 19**

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	YES	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	YES	YES	YES
Total	**6 / 6**	**6 / 6**	**6 / 6**

Section 3 - Stateful Operation

Test 3.1 – Stateless Attack Replay	**Alert?**	**Blocked?**	**Pass/Fail**
Test 3.1.1 - Stateless Web exploits	NO	NO[2]	PASS
Test 3.1.2 - Stateless FTP exploits	NO	NO[2]	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	YES	YES	YES	YES	YES	NO[2]	NO[2]
Test 3.2.2 - Attack Blocking	YES	YES	YES	YES	YES	NO[2]	NO[2]
Test 3.2.3 - State Preservation	YES	YES	YES	YES	YES	YES	YES
Test 3.2.4 - Block legitimate traffic	NO	NO	NO	NO	NO	NO[2]	NO[2]

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	YES	YES	YES	YES	YES	YES	NO[3]
Test 3.3.2 - Attack Blocking	YES	YES	YES	YES	YES	YES	YES[3]
Test 3.3.3 - State Preservation	YES	YES	YES	YES	YES	YES	YES
Test 3.3.4 - Block legitimate traffic	NO	NO	NO	NO	NO	NO	YES[3]

Section 4 - Detection/Blocking Performance Under Load

Test 4.1 – UDP traffic to random valid ports		50Mbps	100Mbps	150Mbps	200Mbps	Max
Test 4.1.1 - 64 byte packet test - max 296,000pps (200Mbps)	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	
Test 4.1.2 - 440 byte packet test - max 52,000pps (200Mbps)	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	
Test 4.1.3 - 1514 byte packet test - max 16,300pps (200Mbps)	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	

Test 4.2 – HTTP "maximum stress" traffic with no transaction delays		50Mbps	100Mbps	150Mbps	200Mbps	Max
Test 4.2.1 - Max 250 connections per second - ave packet size 1200 bytes - max 20,000 packets per second	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	
Test 4.2.2 - Max 1000 connections per second - ave packet size 540 bytes - max 46,000 packets per second	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	
Test 4.2.3 - Max 2000 connections per second - ave packet size 440 bytes - max 56,000 packets per second	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	
Test 4.2.4 - Max 4000 connections per second - ave packet size 350 bytes - max 72,000 packets per second	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	

Test 4.3 – HTTP "maximum stress" traffic with transaction delays		50Mbps	100Mbps	150Mbps	200Mbps	Max
Test 4.3.1 - Max 1000 connections per second - ave packet size 540 bytes - max 46,000 packets per second - 10 sec delay - max 10,000 open connections	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	
Test 4.3.2 - Max 2000 connections per second - ave packet size 440 bytes - max 56,000 packets per second - 10 sec delay - max 20,000 open connections	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	

Test 4.4 – Protocol mix		50Mbps	100Mbps	150Mbps	200Mbps	Max
Test 4.4.1 - 72% HTTP (540 byte packets) + 20% FTP + 4% UDP (256 byte packets). Max 760 connections per second - ave packet size 555 bytes - max 42,000 packets per second - max 700 open connections	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	

Test 4.5 – Real World traffic		50Mbps	100Mbps	150Mbps	200Mbps	Max
Test 4.5.1 - Pure HTTP (simulated browsing session on NSS Web site). Max 30 connections per second - 7 new users per second - ave packet size 1000 bytes - max 24,000 packets per second	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	
Test 4.5.2 - Protocol mix - 72% HTTP (simulated browsing sessions as 2.5.1) + 20% FTP + 4% UDP (256 byte packets). Max 112 connections per second - ave packet size 900 bytes - max 27,000 packets per second	Detected	100%	100%	100%	100%	200Mbps
	Blocked	100%	100%	100%	100%	

Section 5 - Latency & User Response Times

Test 5.1 – Latency	Packet Size	50Mbps	100Mbps	150Mbps	200Mbps
Test 5.1.1 Average latency (µs) with no background traffic	64	83.48	83.22	83.04	100.98
	440	100.37	94.53	92.84	94.24
	1514	135.62	134.00	132.64	130.44
Test 5.1.2 Average latency (µs) with background traffic (100Mbps HTTP traffic, max 500 connections per second - ave packet size 540 bytes - max 23,000 packets per second)	64	86.25			
	440	100.61			
	1514	134.06			
Test 5.1.3 Average latency (µs) when under attack (20Mbps SYN flood)	64	282636.10			
	440	675364.60			
	1514	424570.90			

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 (100Mbps HTTP traffic, max 500 connections per sec - ave packet size 540 bytes - max 23,000 packets per sec)	297728	0	1	<1	<1
Test 5.2.2 - Web page response (ms) when under attack (100Mbps HTTP traffic, max 500 connections per sec - ave packet size 540 bytes - max 23,000 packets per sec PLUS 20Mbps SYN flood)	334484	82049	<1	13.614	3.876

Section 6 - Stability & Reliability

Test ID	Result
Test 6.1.1 - Blocking Under Extended Attack	PASS
Test 6.1.2 - Passing legitimate traffic under extended attack	PASS
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?	FAIL

Notes:

1. Originally the GET & POST versions of this exploit were part of the same signature. Since the GET is an audit-only event and POST is an actual exploit, it was necessary to enable this signature in blocking mode, thus causing a false positive. Separating into two separate signatures allowed us to alert only on the GET and block the POST, thus eliminating the false positive alert.
2. The Proventia G200 can be configured so that packets for unknown connections are blocked by default. When the state tables are full or the sensor runs low on resources, packets which appear to be from unknown connections will then be blocked (the default out of the box is to allow them through the sensor). The downside to this mode of operation is that although suspected attack traffic will be blocked when state tables are full, so will new connections for legitimate traffic
3. Resource issues occurred at 1 million connections - in excess of 900,000 connections we noted a large number of failed connections - thus legitimate traffic was being dropped/blocked. Because we changed the default settings for this test to block packets for unknown connections, no attack traffic was allowed through when this occurred. Note that the device is only rated to 500,000 connections by ISS

Section 1: Detection Engine

We installed one sensor with the latest signature pack, reporting to a single SiteProtector server. We used the *In-line Evaluation* policy, which has **all** attack signatures enabled (apart from port probes) and some key audit signatures.

Signature recognition (with blocking disabled) was excellent out of the box (95 per cent), and was increased to 100 per cent after the application of a signature pack update which was provided to us in less than 48 hours (and which was simultaneously made available to the ISS customer base).

Blocking performance was one percentage point higher than detection performance out of the box, and was also increased to 100 per cent after the application of the signature pack update.

We noted a minimum of “noise” in this release, with few test cases raising multiple alerts for a single exploit. All our “false negative” (modified exploit) cases were detected correctly, demonstrating that the Proventia signatures are designed to detect the underlying vulnerability rather than a specific exploit.

A major concern in deploying an IPS is the blocking of legitimate traffic. There was one false positive alert out of the box, which was caused by the MDAC heap overflow signature being a little too generic and covering both the POST and GET conditions. Once this had been split into two signatures, we could block the POST and alert on (or ignore) the GET, thus achieving a clean 100 per cent.

The *Default-Blocking* policy includes over 180 built-in rules for out-of-the-box blocking against hybrid threats, including MS Blaster, SQL Slammer, Nimda, and Code-Red. Only certain critical signatures are configured to *block*, with the majority set to *alert only*. Given the quality of the signatures noted during these tests, we would be reasonably confident in deploying the Proventia G200 in a live network with the *Default-Blocking* policy activated.

Note that the Proventia can be configured in *Simulation* mode to enable to administrator to see the effects of applying a particular policy without the device actually blocking traffic.

Section 2: IPS Evasion

Resistance to known evasion techniques was excellent, with Proventia being one of the few products to collect a clean sheet across the board in our evasion tests. *Fragroute, Whisker, ADMmutate* and even *RPC record fragging* all failed to trick Proventia into ignoring valid attacks.

Note that not only were the fragmented and obfuscated attacks blocked successfully, but every one of them was decoded accurately as well. This is the level of performance to which we would like to see all IDS and IPS products aspire.

Section 3: Stateful Operation

Out of the box, the Proventia G200 handled up to 250,000 open connections - this will be increased to 500,000 connections with a future update pack. Although 500,000 connections is the maximum number of open connections officially supported by this device, it wasn’t until we loaded it beyond the 900,000 level that we started to see problems occur with the device running low on resources. Default operation of the device is to allow all traffic when the state tables are full or resources are low - this naturally means that it is possible to evade the Proventia once the state tables are full, since it will allow attack traffic through at that point.

Changing a couple of parameters in the policy enables the administrator to alter this default behaviour to block all traffic by default when resources are low and to block all traffic for which there is no session established. This is how we configured the device for our tests in order to render the Proventia immune to stateless attack reply tools such as *Stick* and *Snot*.

The downside to this configuration is, of course, that when resources are low or the state tables are full, legitimate traffic can be blocked in error.

The fact is that there is no right or wrong way to do this - all IPS devices have to make the choice whether to risk denying legitimate traffic or allowing malicious traffic once they run low on resources. ISS has done extremely well here in allowing the administrator to choose for himself which of these risks he would prefer to take, and configure the Proventia accordingly.

Interestingly, the sensor did continue to track state on our “half open” exploits right up to 1 million open connections, since Proventia is designed to ignore new connections once the limit is exceeded, but does not age out old ones. We actually prefer old connections to be aged out, since the dropping of new connections does provide an easier method of evasion for the attacker when the device is configured to allow traffic from “unknown” connections. It would be nice to see this behaviour made configurable too.

Section 4: Detection/Blocking Performance Under Load

Note that the Proventia G200 was tested as a 200Mbps device. 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 have no hesitation in rating the Proventia G200 as a true 200Mbps device.

Section 5: Latency & User Response Times

Latency figures were excellent for a device of this type at all traffic loads and with all packet sizes, ranging from 83.48µs with 50Mbps of 64 byte packets, to 135.62µs with 200Mbps of 1514 byte packets. Behaviour throughout the tests was predictable, with remarkably constant latency figures at all network loads with 440 and 1514 byte packets. Latency on 64 byte packets increased by 20 per cent as the load of 64 byte packets was increased from 50Mbps to the 200Mbps level.

Another surprise was that latency barely increased at all as we loaded the device with 100Mbps (half the rated speed of the device) of genuine HTTP traffic. However, 20Mbps of SYN flood traffic did have a serious negative impact on the Proventia. Although the SYN flood was effectively mitigated, the latency was increased to the point where we noted a significant number of failed connections in our HTTP traffic. The Proventia G200 is better as a straight IPS than as a DOS/DDOS mitigation device.

Apart from the DOS/DDOS condition, however, latency figures were considered to be acceptable for a device of this type, and much lower than we expected from a device using commercially-available hardware components.

Section 6: Stability & Reliability

The Proventia G200 performed consistently and reliably throughout our tests. Under eight hours of extended attack (comprising millions of exploits mixed with genuine traffic) it continued to pass legitimate traffic whilst blocking attack traffic in a consistent manner.

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 22/TCP (SSH), 9901/TCP (SAMBA-SWAT) and 2998/TCP (ISS-REALSEC) to allow communication between the SiteProtector server and the Proventia appliance. Firewalls rules were put in place to ensure these ports were only visible to the management server PC. 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 or its ability to detect and block attacks and communicate with the management server. However, no alerts were raised at any time to inform the administrator that the management port was under attack.

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