Symantec SNS 7160 V4.0.0.9
Technical Evaluation

An NSS Group Report

First published July 2005 (Version 1.0)

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

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

TABLE OF CONTENTS

TABLE OF FIGURES

The NSS Group

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

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

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

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

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

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

Foreword

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

As with the first two Editions, 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.

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

Overall, the brand new test suite contains over **800 individual tests**, many of which are run multiple times, to provide the most thorough and complete evaluation of IPS products available anywhere today. The NSS Group has developed advanced testing methodologies for both *Rate-Based IPS* and *Content-Based IPS* products, since these devices are often very different in operation, although all products tested in this edition of the report are content-based.

It is worth pointing out that not every product submitted for testing receives an *NSS Approved* award. Standards are very high, and only those appearing in this report have received ***NSS Approved*** awards. For this latest edition, **ten** vendors submitted a total of **twelve** products for testing, and **eight** of these passed our stringent testing to receive **NSS Approved**. It is heartening to note that this is a much-improved success ratio over Edition 2.

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

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

Bob Walder

INTRODUCTION

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

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

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

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

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

Intrusion Prevention Systems (IPS)

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

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

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

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

Host IPS (HIPS)

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

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

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

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

Network IPS (NIPS)

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

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

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

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

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

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

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

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

Rate-Based IPS (Attack Mitigator)

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

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

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

Detection Methods

At one time, most Network IDS/IPS products based their alerts purely on pattern matching packet contents against a database of known signatures. Then came a new breed of offerings that approached the problem in a completely different way - by doing a full protocol analysis on the data stream. Others began to use heuristics or anomaly-based analysis to determine when an attempted attack had taken place.

Today, most IDS/IPS employ a mixture of these detection methods in a single product, though some will be more biased towards one method than another.

According to Cisco, there are five main methods of attack identification (*source: Cisco Systems, The Science of Intrusion Detection System Attack Identification*):

Pattern Matching

Pattern matching in its most basic form is concerned with the identification of a fixed sequence of bytes in a single packet. In addition to the tell-tale byte sequence, most IPS will also match various combinations of the source and destination IP address or network, source and destination port or service, and the protocol. It is also often possible to tune the signature further by specifying a start and end point for inspection within the packet, or a particular combination of TCP flags.

The more specific these parameters can be, the less inspection needs to be carried out against each packet on the wire. However, this approach can make it more difficult for systems to deal with protocols that do not live on well defined ports and, in particular, Trojans, and their associated traffic, which can usually be moved at will.

Although it is often quite simple to define a signature for a particular exploit, basic pattern matching can often be too specific, sometimes requiring multiple signatures to be defined for minor variations in exploits. They are also prone to false positives, since legitimate traffic can often contain the relatively small set of criteria supposedly used to determine when an attack is taking place.

This method is usually limited to inspection of a single packet and, therefore, does not apply well to the stream-based nature of network traffic such as HTTP sessions. This limitation gives rise to easily implemented evasion techniques.

Stateful Pattern Matching

Stateful pattern matching offers a slightly more sophisticated approach, since it takes the context of the established session into account, rather than basing its analysis on a single packet.

Stateful IPS products must consider arrival order of packets in a TCP stream and should handle matching patterns across packet boundaries. Thus, if the exploit string to be matched is *foobar*, and the exploit is split across two packets, with *foo* in one and *bar* in another, the simple packet matching IPS will miss the attack, since it will not be able to match the complete string. The stateful IPS, however, will maintain the session context and reassemble the traffic stream, once again making the complete string available to the detection engine.

This requires more resources than simple pattern matching, since the IPS now has to allocate large amounts of memory and processing power to track a potentially large number of open sessions for as long as possible. This approach does make IPS evasion that much more difficult, though far from impossible.

Direction of traffic is also important here, both in terms of quality of detection and performance.

Client-to-server traffic inspection is the process of applying detection mechanisms to the “request side” portion of a communication - for example, in HTTP this could be the “GET” request coming from a client.

Client-to-server traffic inspection is typically activated to protect all traffic whether internally or externally generated. As the size of the traffic in terms of byte count is relatively small, the processing load placed on the IPS will be lower.

Server-to-client traffic inspection is the process of finding an attack in the "response side" portion of a communication - for example, in HTTP the server-to-client traffic could be the web page and content returned from the server as a result of a "GET" request. Server-to-client traffic, as in this example, is often much larger than the client-to-server traffic in terms of byte count. As a result, the processing load that is placed on an IPS is greater for server-to-client traffic.

Some vendors do not implement server-to-client signatures at all. Often this is for performance reasons, but sometimes it is a design decision by those vendors who also offer HIPS products, which are often better placed to detect the types of exploits executed by malicious response traffic as opposed to request traffic. Some vendors do include server-to-client signatures, but recommend they are disabled when performance is paramount. Bi-directional detection can have a significant impact on performance in some cases - those products which can handle this situation with zero or minimal impact on performance are worth closer inspection (although this level of performance often comes with a higher price tag).

It should be noted that there are situations where disabling server-to-client signatures is reasonably safe, and - happily - these are usually the situations where the highest levels of performance are demanded. Typically, this would be where an IPS is deployed within the network perimeter, where it is unlikely that purely internal HTTP response traffic is likely to be malicious. Perimeter defences would normally be deployed with both client-to-server and server-to-client signatures enabled, but perimeter devices rarely have the same performance requirements as internal ones.

Protocol Decode

Protocol decode IPS take a radically different approach to simple pattern matching IPS products - though sometimes not quite as radically different as the marketing folks would have you believe. With this technique, the IPS detection engine performs a full protocol analysis, decoding and processing the packet contents in the same way that the target client or server application would. It also tends to be stateful.

Although this may seem like using a sledgehammer to crack a nut, it does have the advantage of highlighting anomalies in packet contents much more quickly than doing an exhaustive search of a signature database. It also has the advantage of greater flexibility in capturing attacks that would be very difficult - if not impossible - to catch using pure pattern-matching techniques, as well as new variations of old attacks. These are attacks which - although changing only slightly from variant to variant - would normally require a new signature in the database for the "traditional" IPS architecture, but which would be detected automatically by a complete protocol analysis.

One of the first things the protocol decode engine does is to apply rules defined by the appropriate RFCs to look for violations. This can help to detect certain anomalies such as binary data in an HTTP request, or a suspiciously long piece of data where it should not be - a sign of a possible buffer overflow attempt.

One simple example of how this might work concerns searching Telnet login strings for one of the many well-known login names that rootkits tend to leave behind on the system. A pattern matching system might scan *all* Telnet traffic for *all* these patterns, in which case the more patterns you add, the slower it becomes (not *always* the case, but a reasonable assumption for the purposes of this example).

In contrast, a protocol analysis system will decode the Telnet protocol and extract the login name. It can then perform an efficient search in a binary-search tree or a hash table for just the login name, which should scale much better as new signatures are added.

In theory, therefore, protocol decoding should offer more efficient processing of traffic and improved scalability as more signatures are added, compared to a pure pattern matching solution. In reality, pattern matching solutions rarely opt for a “brute force” approach (there are some extremely intelligent and efficient pattern matching mechanisms available), and so the differences are not always as marked as the marketing people would like us to believe.

Note also, that pattern matching and protocol decoding are not mutually exclusive, as some would lead you to believe. A protocol analysis IPS can only go so far with its protocol decodes before it too will be forced to perform some kind of pattern matching, albeit against a theoretically smaller subset of “signatures”.

One major downside, of course, is that if a completely new type of exploit does surface, it is likely that the developer will have to create new protocol decode code to handle it, whereas the pattern matching approach can allow the administrator to develop a custom signature much more quickly on site.

Protocol decoding does offer a number of advantages, however. It minimises the chance for false positives if the protocol is well defined and enforced (although false positives can be higher if the RFC is ambiguous), and can also be more broad and general to allow the IPS to detect minor variations of an exploit without having to implement separate signatures.

You may see this technique referred to in several different ways:

- *Protocol decode*
- *Protocol Anomaly Detection*
- *Protocol validation*

Each of these terms, if strictly applied, could use a slightly different approach to the problem. For example, we would expect a *protocol decode* engine to perform the sort of additional pattern matching and length checking mentioned above on the field contents in order to detect specific exploits or buffer overflows.

Pure *protocol validation* or *Protocol Anomaly Detection* engines, however, might go no further than decoding just enough to be able to determine if the packet follows the RFC to the letter. If not, they will raise an alert - but in allowing a packet to pass, they cannot be sure that the contents will not contain a means of exploit that just happens to conform with the RFC.

Beware the marketing hype in this particular area – no matter what architecture is used, the performance figures and detection rates in a live deployment will speak for themselves.

Heuristic Analysis

Heuristic-based signatures use some kind of algorithmic logic on which to base their alarm decisions. These algorithms are often statistical evaluations of the type of traffic being presented.

A good example of this type of signature is one that would be used to detect a port sweep. This signature looks for the presence of a threshold number of unique ports being touched on a particular machine. The signature may further restrict itself through the specification of the types of packets that it is interested in (that is, SYN packets). Additionally, there may be a requirement that all the probes must originate from a single source, and even that valid SYN ACK packets must be seen to be returned by the host being probed.

Signatures of this type will react differently on different networks, and can be a significant source of false positives if not tuned correctly, requiring some threshold manipulations to make them conform to the utilisation patterns on the network they are monitoring. This type of signature may be used to look for very complex relationships as well as the simple statistical example given.

Anomaly Analysis

The final approach is to forget about trying to identify the attacks directly, and concentrate instead on ignoring everything that is considered “normal”. This is known as “*anomaly-based*” IPS, and the basic principle is that, having identified what could be considered “normal” traffic on a network, then anything that falls outside those bounds could be considered an “intrusion” - or at the very least, something worthy of note. This is generally better suited to passive IDS rather than in-line IPS devices, given its propensity for false positives.

The primary strength of anomaly detection is its ability to recognise previously unseen attacks, since it is no longer concerned with knowing what an attack looks like - merely with knowing what does not constitute normal traffic. Its drawbacks, of course, include the necessity of training the system to separate noise from natural changes in normal network traffic (the installation of a new - perfectly legitimate - application somewhere on the network, for example).

Changes in standard operations may cause false alarms while intrusive activities that appear to be normal may cause missed detections. It is also difficult for these systems to name types of attacks, and this technology has a long way to go before it could be considered ready for “prime time”.

Which Detection Method Is The Best?

Which detection method to choose is a difficult question, and in all honesty, it is not one with which most of those evaluating these products should concern themselves.

Adequate performance to handle the traffic to which the sensor will be exposed, accuracy of alerts, low incidence of false positives, and centralised management and reporting/analysis tools are far more important than how the packets are processed.

In some instances, the lines blur between methodologies to the point where they become almost indistinguishable.

For example, most protocol decode analysis engines alert the user to the presence of protocol violations that are not directly related to any known attack but are “anomalous” (for example, length-based buffer overflow detection). Therefore, in this instance the engine has attributes of an anomaly-based system.

As we have already mentioned, most protocol analysis systems are also reduced to performing some form of pattern-matching process following the protocol decode. Likewise, even the most basic pattern-matching systems perform some form of protocol analysis - even if it is only for a limited range of protocols. In truth, almost all Network IPS systems are already adopting a hybrid architecture.

By and large, therefore, the *pattern-matching vs. protocol decode* debate is one of religion - something for the marketing departments to shout about. Why should the average user care what happens under the hood as long as the product does what it claims to do - detect and prevent intrusions?

Implementation Challenges

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

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

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

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

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

Dropped packets are also an issue, since if even one of those dropped packets is one of those used in the exploit data stream it is possible that the entire exploit could be missed.

Most high-end IPS vendors will get around this problem by using custom hardware, populated with advanced FPGAs and ASICs - indeed, it is necessary to design the product to operate as much as a switch as an intrusion detection and prevention device.

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

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

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

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

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

Requirements for effective prevention

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

- **In-line operation -** only by operating in-line can an IPS device perform true protection, discarding all suspect packets immediately and blocking the remainder of that flow

- **Reliability and availability -** should an in-line device fail, it has the potential to close a vital network path and thus, once again, cause a DoS condition. An extremely low failure rate is thus very important in order to maximise up-time, and if the worst should happen, the device should provide the option to fail open or support fail-over to another sensor operating in a fail-over group (see below). In addition, to reduce downtime for signature and protocol coverage updates, an IPS must support the ability to receive these updates without requiring a device re-boot. When operating inline, sensors rebooting across the enterprise effectively translate into network downtime for the duration of the reboot
- **Resilience -** as mentioned above, the very minimum that an IPS device should offer in the way of High Availability is to fail open in the case of system failure or power loss (some environments may prefer this default condition to be “fail closed” as with a typical firewall, however - the most flexible products will allow this to be user-configurable). Active-Active stateful fail-over with cooperating in-line sensors in a fail-over group will ensure that the IPS device does not become a single point of failure in a critical network deployment
- **Low latency -** when a device is placed in-line, it is essential that its impact on overall network performance is minimal. Packets should be processed quickly enough such that the overall latency of the device is as close as possible to that offered by a layer 2/3 device such as a switch, and no more than a typical layer 4 device such as a firewall or load-balancer.
- **High performance -** packet processing rates must be at the rated speed of the device under real-life traffic conditions, and the device must meet the stated performance with all signatures enabled. Headroom should be built into the performance capabilities to enable the device to handle any increases in size of signature packs that may occur over the next three years. Ideally, the detection engine should be designed in such a way that the number “signatures” (or “checks”) loaded does not affect the overall performance of the device.
- **Unquestionable detection accuracy -** it is imperative that the quality of the signatures is beyond question, since false positives can lead to a Denial of Service condition. The user MUST be able to trust that the IDS is blocking only the user selected malicious traffic. New signatures should be made available on a regular basis, and applying them should be quick (applied to all sensors in one operation via a central console) and seamless (no sensor reboot required)
- **Fine-grained granularity and control -** fine grained granularity is required in terms of deciding exactly which malicious traffic is blocked. The ability to specify traffic to be blocked by attack, by policy, or right down to individual host level is vital. In addition, it may be necessary to only alert on suspicious traffic for further analysis and investigation
- **Advanced alert handling and forensic analysis capabilities -** once the alerts have been raised at the sensor and passed to a central console, someone has to examine them, correlate them where necessary, investigate them, and eventually decide on an action. The capabilities offered by the console in terms of alert viewing (real time and historic) and reporting are key in determining the effectiveness of the IPS product.

The NSS Intrusion Prevention Group Test

The NSS Group conducted the first comprehensive IPS test of its kind, now updated in this Edition.

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

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

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

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

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

Performance

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

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

Detection/Blocking Performance Under Load

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

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

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

Latency & User Response Times

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

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

throughput = window / RTT

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

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

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

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

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

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

Such critical value calculations are important when TCP connections achieve maximum throughput, which is true for large data transfers.

For smaller data transfers, and non-TCP applications like NFS, latency has a more direct impact on user experience - response time is directly proportional to latency. That is, *doubling latency doubles response time.* In these situations, the latency of the network in which a NIPS is deployed determines the acceptable latency of the NIPS.

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

- In internal corporate LANs, the round trip latency could be in the 200-300 microsecond range. Deploying our hypothetical NIPS would increase the maximum round trip latency to 2.3 milliseconds, an increase of just over 700 per cent. The time to copy a large group of files, for example, would increase by a factor of seven.
- In inter-campus corporate networks connected over a MAN, the latency could be in the 500-1000 microsecond range (or less). Deploying our hypothetical NIPS would increase the maximum round trip latency to 3 milliseconds, a minimum increase of 300 per cent. The time to copy a large group of files, for example, would increase by at least factor of three.
- Internet facing connections experience round-trip latency from 10-100 milliseconds. Deploying our hypothetical NIPS would increase the round trip latency by 1-10 per cent, which would have only a minor impact on the user experience.

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

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

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

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

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

This “*application latency*” is measured both with no background load and while the device is under attack.

Stability & Reliability

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

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

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

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

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

Security Effectiveness

Detection Accuracy & Breadth

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

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

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

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

Naturally, Rate-Based IPS devices will not respond to the same attack traffic as Content-Based devices, and so for those the Detection Accuracy tests involve detecting and mitigating a wide range of rate-based attacks such as port scans, SYN floods, connection floods, and so on. We note which of these are mitigated completely, which are mitigated partially, and which require the use of built-in firewall capabilities.

Resistance To Evasion Techniques

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

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

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

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

Stateful Operation

If the IPS is tracking TCP session state, then it has the potential to introduce denial of service when the session table becomes full (too many connections) or if it can’t keep up with the creation of new sessions (too many connections per second).

As with latency and bandwidth, the number of connections supported by the IPS and its connection per second rate should be matched to the network.

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

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

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

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

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

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

Usability

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

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

SYMANTEC SNS 7160 V4.0.0.9

Executive Summary

The Symantec Network Security 7100 Series is a family of integrated intrusion prevention appliances designed to detect and prevent attacks across multiple network segments at up to Gigabit speeds. A range of models are available covering various sizes of installation from small business to large enterprise, and all use the same software version.

The SNS 7160 under test here is a dedicated 2U appliance designed to monitor and protect multiple network segments at Gigabit speeds. The device sports twelve copper 10/100/1000Mbps ports, of which eight are used for monitoring, three for sending TCP resets when monitoring in passive-mode, and one for management. The eight monitoring ports can be configured in various combinations of in-line pairs (in blocking or non-blocking mode), or single passive-mode ports.

Overall, the performance of the SNS 7160 was very good, easily handling the rated 1Gbps of traffic under normal network conditions. Attack recognition and blocking ability were excellent out of the box, and improved to a perfect 100 per cent following an update, whilst resistance to all common evasion techniques tested was also perfect.

Latency was also very good for a device of this type, allowing the SNS 7160 to be deployed anywhere in the network - either at the perimeter or in the core.

We found the SNS 7160 to be very stable and reliable under extended attack, and the handling of high-levels of DOS/DDOS attacks (such as SYN floods) was excellent.

Out of the box, policy management, alert management and reporting is first-rate. We found the management system to be powerful, scalable, and intuitive, with an advanced event correlation capability that simplifies the processing of large numbers of alerts and some good reporting and analysis tools.

Architecture

The *Symantec Network Security* (SNS) 7100 Series offering is available as a range of dedicated, purpose-built hardware appliances which all employ a common core architecture that provides detection, analysis, storage, and response functionality (referred to by Symantec as *Intrusion Mitigation Unified Network Engine*, or *IMUNE*). The main components of the SNS 7100 system are as follows:

Symantec Network Security Console

The administrative and management component of SNS is called the *Symantec Network Security Console*. It communicates over an encrypted and authenticated link to ensure that authorised administrators may log in from any secure or insecure network. The Network Security Console manages all operations, including policy management and application, drill-down incident analysis with full packet capture, detailed event descriptions, application of security updates and patches, and allows event annotations and incident marking for tracking.

The Network Security Console provides an interface from which the administrator can monitor events and devices, edit sensor parameters, configure protection policies and response rules, apply policies, and view log data. It is possible to generate reports and view them immediately in the Network Security Console, or they can be scheduled to be generated automatically and transmitted via e-mail or Secure Copy (SCP) .

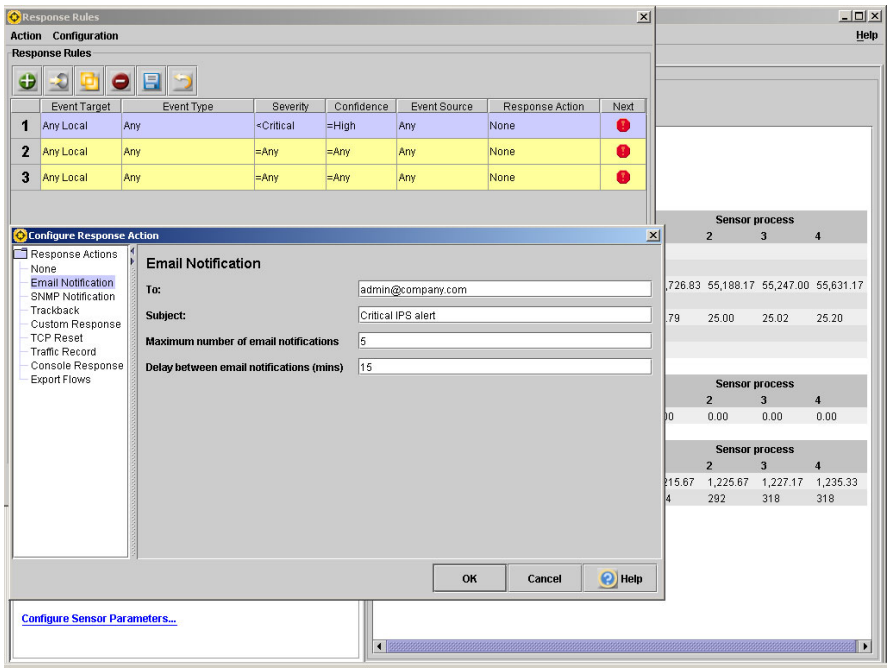


Figure 1 - SNS 7160: Configuring Response Rules in the Console

Four pre-defined roles provide granular management. Roles are sets of permissions for specific management operations, and each user's login identity determines their role and permission assignment during an administrative session.

SNS installs with a *SuperUser* account, and the SuperUser can create additional login accounts in the following user groups:

- ***SuperUsers -*** *Full administrative capabilities, including permissions to configure user accounts, topology objects, response rules, custom signatures, configurable parameters, reporting, and redundancy to the master node.*
- ***Administrators -*** *Partial administrative capabilities*
- ***StandardUsers -*** *Full read-only capabilities, allowed to view all information in the Network Security Console*
- ***RestrictedUsers -*** *Partial read-only capabilities*

It is not possible to restrict access to specific policies or interfaces on a per-user basis. The SNS console enables encrypted and authenticated communication using password-protected 1024-bit Diffie-Hellman key exchange and 256 bit AES encryption.

Sensor Software

The 7100 Series Appliance contains a variety of tools and techniques that work together to gather attack information, analyse the attacks, and initiate responses appropriate to specific attack circumstances.

The following diagram illustrates the basic architecture:

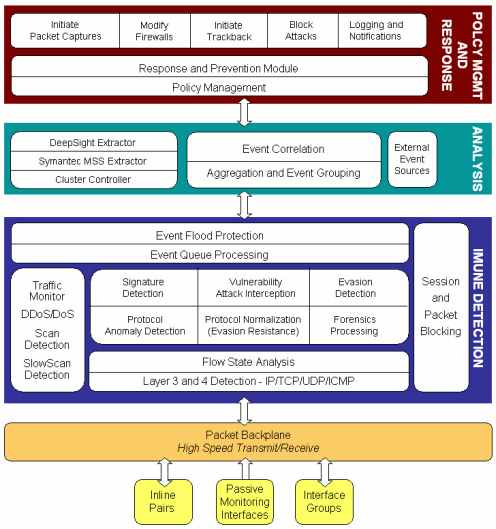


Figure 2 - SNS 7160: Architecture

The components of the SNS appliance architecture include:

- **Alert Manager -** The SNS Alerting Manager provides three types of alerts or notifications: a console action alert, an e-mail alert, and an SNMP trap alert.
- **Sensor Manager -** The Sensor Manager maintains a pool of sub-processes to manage sensor-related functionality. This includes sensor processes for event detection, traffic recording, and FlowChaser sub-processes that handle network device configuration, starting and stopping.
- **Administration Service -** All communication across the network passes through the QSP Proxy, an administration service with 256-bit AES encryption and pass phrase authentication. This ensures that all communication between the Network Security Console and the master node, and between appliance nodes within a cluster, are properly authenticated and encrypted. In addition, this service enforces role-base administration and thus prevents any circumvention of established access policy.
- **Analysis Framework -** Because SNS appliances have access to information collected by other SNS nodes - and even other IDS systems deployed on the network - attack analysis and response can be coordinated across a distributed network, enabling a rapid global attack response.

  The SNS *Analysis Framework* (AF) aggregates event data on possible attacks from all event sources meaning that multiple related events are correlated and presented as a single “incident” at the console. This makes it much easier for the administrator to sort high priority alerts from those which are less important. The Analysis Framework also performs statistical correlation analysis on events to identify event patterns that vary significantly from usual network activity and to identify individual events that are highly related, such as a port scan followed closely by an intrusion attempt. The grouping of several low priority alerts together in a single incident - based on event type, source or destination - thus makes it easier to spot when several exploits are being employed one after the other in a concerted attack.

When an event is received from the sensors, the tuneable Analysis Framework considers its characteristics to decide if it is part of an existing incident. If it is, it is added to that incident - if not, a new incident is created. All further analysis is then based upon the current incidents and not on the individual events. If no new events are added to an incident for a predetermined period of time, the incident expires and is moved to the historical incident list.

Incidents are extremely valuable in the determination of malicious activities because individual events may or may not be important by themselves, until other related events take place and build a profile of activity. For example, a port scan is not an important enough event to send a notification at 3:00 in the morning. But, rather than throwing away that information, an incident will hang on to it to see if more activities follow.

An invalid login on an open port also is not necessarily a significant event. Incidents allow these activities to be associated such that the priority may be increased. After a certain number of invalid logins, the priority for the incident may be high enough to take action. This allows for a much more granular approach to attack response, so that no one is bothered when the threat severity is low, but resources can be allocated rapidly when necessary.

- **Databases -** SNS uses multiple databases to store information about attacks, the network topology, and configuration information, including:
    - ***Topology database:*** *Stores information about local network devices and interfaces and the network configuration. Symantec Network Security uses this data to direct the FlowChaser toward the area of the network in which an attack occurs.*
    - ***Protection Policy database:*** *Stores the pre-defined protection policies that installed with the product and those added through LiveUpdate, as well as any user-defined signatures.*
    - ***Response Rule database:*** *Stores the rules that define the actions to take when an attack is identified, the priority to give to the attack incidents, and the necessity for further investigation of the attack.*
    - ***Configuration database:*** *Stores configurable parameters that SuperUsers and Administrators can use to configure tasks at the node level and to configure detection at the sensor level.*
    - ***Incident and Event databases:*** *Stores information about events and incidents. The event log can be signed periodically to verify that the log has not been tampered with or altered*
    - ***LiveUpdate database:*** *Stores data relevant for LiveUpdate.*
    - ***User database*****:** Stores information about each user login account.
- **Event Stream Provider -** The *Event Stream Provider* (ESP) prevents flood invasions by categorising events and treating them as a single unit when it sends them to the Analysis Framework to be analysed. The ESP can also monitor bi-directional traffic.

  When the ESP monitors an event, it generates a message hash and a destination hash of the event. The ESP uses the values of these two hashes to select the queue into which to place the event. The ESP then pushes the queued events to the Analysis Framework at a rate configured by the administrator.

The ESP checks each queue in turn and pushes one event to the Analysis Framework from each queue containing events.

Given this architecture, if multiple identical events bombard the network, as in a DoS attack, SNS will assign the same message and destination hashes for all of the DoS attack events and place them in the same queue. This prevents the Analysis Framework from becoming overloaded with the DoS events, because the ESP sends only one event at a time from each queue. If there is an attack hidden beneath the DoS attack, the message and destination hashes for the events related to the hidden attack will differ from the DoS event hashes.

The ESP, therefore, places these events in separate queues. The Analysis Framework can then analyse the events related to the hidden attack.

Sensor - SNS 7100 Series sensors can be deployed in either in-line or passive mode, and using interface groups or single monitoring interfaces. SNS detects a variety of threats using a layered detection model that Symantec refers to as its IMUNE architecture. This architecture includes *protocol anomaly detection* (PAD), Vulnerability Attack Interception (VAI), Signature Detection, evasion detection, Denial-of-Service and Scan Detection, and traffic monitoring.

Independent of the deployment mode of a particular sensor, SNS applies the same detection strategy and protection, tuned to maximise detection while retaining network performance and reliability. For example, using in-line mode, the sensor tunes itself to minimise latency and maximise throughput across a pair of interfaces.

Using interface groups, the sensor correctly adjusts itself to compensate for the fact that a single network session may be conducted using multiple, asymmetric links that have packet-based asymmetry. Using single monitoring interfaces, the sensor batches process packets to maximise detection coverage.

- **Smart Agents -** SNS Smart Agents provide a bridge between SNS and other intrusion detection/prevention and firewall products, allowing users to centralise management of events and incidents from the Network Security Console.

  Smart Agents enable SNS to collect data from third-party hosts and network IPS/IDS products in real time, as well as collecting event data from external sensors such as *Symantec Decoy Server*. This data is then sent to be analysed, aggregated, and correlated with all other SNS events.

- **FlowChaser -** FlowChaser serves as a data source in coordination with TrackBack, a response mechanism that traces a DoS attack or network flow back to its source, or to the edges of an administrative domain. FlowChaser receives network flow data from multiple devices, such as SNS sensors and network routers.

7100 Series Appliance

The SNS 7100 Series is available in three models :

- ***SNS 7120 -*** *Monitors up to four 10/100 Base-T network segments and provides a maximum bandwidth license of 200Mbps, and in-line mode maximum bandwidth of 100Mbps*

- ***SNS 7160 -*** *Monitors up to eight 10/100/1000 Base-T network segments. It provides a maximum bandwidth license of 2Gbps, and in-line mode maximum bandwidth of 1Gbps*
- ***SNS 7161 -*** *Monitors up to four 1000 Base-SX fibre-optic network segments, and four 10/100/1000 Base-T network segments. It provides a maximum bandwidth license of 2Gbps, and in-line mode maximum bandwidth of 1Gbps*

The device submitted for testing was the SNS 7160, a purpose-built, dedicated 2U appliance designed to monitor and protect multiple network segments at Gigabit speeds. The SNS 7160 runs an optimised, hardened operating system with limited user services to further increase security and performance.

The rear panel sports twelve copper 10/100/1000Mbps ports, of which eight are used for monitoring, three for sending TCP resets when monitoring in passive-mode, and one for management. The eight monitoring ports can be configured in various combinations of in-line pairs (in blocking or non-blocking mode), or single passive-mode ports.

Also on the rear panel are dual power sockets (the supplies and fans are not hot-swappable), master power switch, USB ports, serial console port and Compact Flash slot. The serial console can be used for initial configuration of the appliance and for command line access to the operating system utilities and file systems. The Compact Flash can be used for initial configuration and for full backup and restore. Slave appliance nodes can also be configured initially from Compact Flash.

The front panel sports a number of useful LED indicators, plus an LCD screen and push buttons. The screen can display two lines of sixteen characters each, and there are six buttons: four arrow buttons and two function buttons labelled **s** (start) and **e** (enter). The LCD panel can be used for initial configuration of the appliance if required. After initial configuration, the LCD screen displays system statistics in a rotating sequence, and provides a menu of tasks including stopping and starting SNS, rebooting or shutting down the appliance, and changing the IP address. Command and control of the device can thus be provided via a Java-based Console, SSH connection, direct serial connection, or the front panel LCD.

In-line mode Protection Policies are configurable so that the administrator can choose to block and alert on designated events. It is a one-click operation to switch between blocking mode and alert-only in the Network Security Console - Symantec refers to this as "*One-Click to Prevention*" as it enables users to move rapidly between blocking and non-blocking modes very quickly by circumventing multiple, time-consuming editing steps found on some other products.

In blocking mode, all network traffic is examined by the SNS detection software before it enters the network, and when a blocking-enabled event is detected, the offending packet is dropped. For TCP/IP traffic, a reset is also sent to both sides (client **and** server) of the TCP connection (this will be a configurable option from the next release).

In in-line alert-only mode, the SNS detection software still analyses all packets as they enter the network, but will neither block packets nor reset connections.

The advantage of in-line alerting mode over operating in passive mode (attached to a switch SPAN port or network tap) is that it is possible to enable blocking with a single mouse-click from the Network Security Console - it is not necessary to halt network traffic while changing cabling and configuration to switch between in-line alerting and blocking modes, and nor is it necessary to create and manage multiple policies for blocking and non-blocking operation.

Interface grouping enables up to four monitoring interfaces to be grouped together as a single logical interface. This is especially useful in environments where packet-level asymmetry is present.

By default, the 7100 Series appliances fail closed, meaning that network traffic is blocked when the appliance fails or is shut down or become temporarily unavailable during, say, software upgrades.

This behaviour can be modified with the addition of an In-line Bypass unit, a custom fail-open device available from Symantec specifically for the 7100 Series appliances. Two and four segment copper bypass units are available for the SNS 7120, SNS 7160 and the copper-interfaces of the SNS 7161, and single segment fibre bypass units are available for the fibre interfaces of the SNS 7161.

SNS Clusters

Within a network, multiple SNS nodes can work together across multiple network segments and network locations - this is called a SNS *cluster*. Attack data is correlated across all SNS nodes in the cluster.

By default, the first SNS installation in a cluster is designated to be the *primary master* node, and all subsequent SNS installations are designated to be *slave* nodes. Changes made on a master node will propagate automatically to all other nodes in a cluster.

All SNS nodes in the cluster can be administered from a single Network Security Console. From this same interface, the system administrator can also view all attack activity being logged by **any** SNS node in the cluster.

One point worth bearing in mind is that, out of the box, SNS is not implemented as a “traditional” three-tier management architecture. Instead, it is designed as a peer-to-peer architecture that provides centralised management and decentralised event storage and analysis. There are several benefits to this approach:

- *Ease of deployment - There is no dedicated management server or middle layer to deploy and manage. This is especially advantageous in smaller deployments - and particularly in single-node deployments - where the infrastructure needed to support a full three-tier system can add significant overhead to what should be a fairly small, easy-to-manage system*
- *Data availability - Both configuration and alert data are stored locally on the appliance, so if the Master crashes the data is still safe and available for later use.*
- *Flexibility – Users with large environments can configure a Master node that is essentially dedicated to managing Consoles, reporting, updates, etc., enabling users to create a 3-tier system as their needs for scalability and performance demand*

The clustering feature takes care of replicating topology and configuration data throughout the system, whilst viewing alerts and running reports requires the central Network Security Console to retrieve data from multiple sensors each time a query is run (the main disadvantage of a peer-to-peer implementation).

Symantec also has a separate three-tier architecture management system which is available at low cost (the user only pays for the CD-ROM and print documentation media - about $30). The main reasons an SNS user would deploy SESA are:

- *Centralised and consolidated event data*
- *Integration of SNS event data with data from other Symantec and third-party security products.*
- *Event data integration and analysis using Symantec Security Information Manager (SSIM).*

Fail Over Groups

SNS provides the ability use backup SNS nodes to ensure constant coverage in the case of unforeseen failures.

The backup node will process the same data, perform the same analysis, and evaluate the same response policies as the dominant SNS node, but it will not execute duplicate responses. If the Master node fails, the next node in the fail over group becomes acting Master, providing continuous event viewing for the Console. Monitoring in this state is read-only, however - to edit policies, apply updates, or make other changes in the cluster, the user must promote the acting Master (or another node in the group) to Master. When the original Master is returned to operation, it will automatically rejoin the cluster as a Slave node

To communicate with the backup nodes, SNS uses a fail over heartbeat protocol which is transported using AES-encrypted, authenticated tunnels.

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 SNS 7160 was tested up to 1Gbps, the rated speed of the device when deployed in-line, and performance at almost all levels of our load tests was perfect, with 100 per cent of all attacks being detected and blocked under most load conditions.

The one exception was Test 4.2.4 where we determined that there was a limit of 17,000 HTTP connection per second (equating to approximately 850Mbps) meaning the test could not be completed. Beyond this limit, legitimate connections began to fail, but all attack traffic was still blocked successfully.

However, we would happily confirm Symantec's 1Gbps rating for this device under all normal network conditions.

Basic latency figures were very good for a device of this type at all traffic loads and with all packet sizes, ranging from 155µs with 250Mbps of 256 byte packets, to 259µs with 1Gbps of 1000 byte packets.

Behaviour throughout the tests with no background traffic was very predictable, with minimal increases in latency as traffic levels increased from 250Mbps to 1Gbps across each packet size.

Placing the device under a half load of 500Mbps of HTTP traffic, we noted slightly larger increases of 66 per cent with 256 byte packets (155µs to 258µs), 45 per cent with 550 byte packets (184µs to 267µs), and 32 per cent with 1000 byte packets (222µs to 293µs).

However, all results remained below the "magic figure" of 300 microseconds, our limit for deploying in-line devices in the core of the network. HTTP response times were also very good, meaning the SNS 7160 could be situated anywhere on a Gigabit network, either internally or at the perimeter.

SYN Flood mitigation is a recent addition to this product, and 100Mbps of SYN flood traffic had no significant effect on the SNS 7160. Latency increased by only a few microseconds across all packet sizes (HTTP response times were barely affected), and the SYN Flood was mitigated almost completely.

The SNS 7160 performed consistently and completely reliably throughout our tests. Under eight hours of extended attack (comprising millions of exploits mixed with genuine traffic) it continued to block 100 per cent of attack traffic, whilst passing 100 per cent of legitimate traffic.

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

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

Security Effectiveness

We installed one sensor with the latest signature pack, and created a Protection Policy with every attack signature enabled, plus some key audit/information only signatures.

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

We noted a minimum of "noise", with very few test cases raising multiple alerts for a single exploit, and the accuracy of the exploit descriptions was high. Performance in our "false negative" tests was good out of the box, and there is every indication that the majority of signatures are written for the underlying vulnerability rather than specific exploits. Specific exploit signatures are also included where appropriate, however, to provide more accurate identification for the administrator.

A major concern in deploying an IPS is the blocking of legitimate traffic. The SNS 7160's resistance to false positives was perfect in our tests.

The SNS 7160 arrives with a number of default policies configured for different environments and with sensible PASS and BLOCK actions set for appropriate signatures (i.e. where the confidence level of a signature is VERY HIGH then the action will generally be set to BLOCK).

Resistance to known evasion techniques was excellent, with the SNS 7160 achieving a clean sweep across the board in all our evasion tests. *Fragroute, Whisker, ADMmutate* and even *RPC record fragging* all failed to trick SNS into ignoring valid attacks. Not only were the fragmented and obfuscated attacks blocked successfully, but all but one of them were decoded accurately as well.

Out of the box, the SNS 7160 handled 100,000 open connections without tuning. Following a simple tuning operation (a single parameter) the device handled just over 1 million connections (the maximum allowed). Default operation of the device is to age out old connections when the state tables are full or resources are low, and this behaviour is not configurable.

Stateless "exploits" are not alerted upon (this is correct behaviour in order to be resistant to *Stick* and *Snot* tools) and mid-flows are not blocked by default. It is not possible to configure the device to block mid-flows.

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

Usability

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

Installation

Installation of the SNS 7160 is very straightforward, mainly thanks to the front-panel LCD screen (although compact flash configuration provides an even simpler method when deploying large numbers of appliances across an organisation).

Using this, it is possible to configure the management IP address and port number, time zone, date and time, whether it is a master or slave node, and the relevant passwords. Care needs to be taken over the choice of master or slave node, however, since it is not possible to change this following installation without starting again from scratch (which involves re-applying any patches and updates).

The Network Security Console software can be installed on any suitable host (Windows, Macintosh or Linux) with access to the management network, and once it is running it is a simple matter to connect to the device and apply the license. Without a three-tier management system, it is necessary to connect to each master node individually using a separate instance of the Console. However, most organisations will undoubtedly make use of the SNS Cluster capability, in which case the Console is connected to the master node, and all slave nodes are added to the topology allowing policies and alerts to be replicated cluster-wide.

Each SNS node is installed without a Protection Policy and thus will not pass network traffic by default (we are assuming throughout this test that the device has been installed in in-line mode and without a bypass unit). Once the Console has access to an SNS node, it is a simple matter to define an in-line pair from the available interfaces and apply one of the default Protection Policies in order to start monitoring traffic. By default, blocking is not enabled.

The documentation set for the SNS 7160 is extensive, including:

- ***Symantec Network Security 7100 Series Implementation Guide -*** *This guide explains how to install, configure, and perform key tasks on the SNS 7100 Series (printed and PDF)**.***
- ***Symantec Network Security Administration Guide (printed and PDF) -*** *This guide provides the main reference material, including detailed descriptions of the SNS features, infrastructure, and how to configure and manage effectively (printed and PDF).*
- ***Symantec Network Security 7100 Series: Models 7160 and 7161 Getting Started Card -****This card provides the minimum procedures necessary for installing, configuring, and starting to operate the SNS 7100 Series appliance (printed and PDF).*
- ***Symantec Network Security User Guide -*** *This guide provides basic introductory information about SNS software (PDF only).*

We found all of the documentation to be very well written and extremely useful. For example, in addition to step-by-step instructions, each section contains detailed information on the reason for, and use of, each of the menu options. This is high quality documentation, and Symantec is to be praised for continuing to offer both hard-copy *and* electronic versions of most of its manuals.

Configuration

The Java-based Network Security Console contains three main tabs that provide a view of the *Devices*, *Incidents*, and *Policies*:

- **Devices -** Provides a hierarchical tree view of the network topology, with a detailed summary of each device.
- **Incidents -** Provides detailed descriptions of incidents and events taking place in the monitored network, and can be drilled down to reveal detailed packet information.
- **Policies -** Provides the tools to create, manage, and apply user-defined signatures, signature variables, and Protection Policies.

SNS maintains a network topology database, and the *Devices* tab is the administrator's view into that database. This tab displays the current network topology in tree form, and contains objects representing SNS nodes, monitoring interfaces, Smart Agents, routers, network segments, and other aspects of the network.

In a large distributed implementation, the population of the network topology tree, if done properly, is probably the most arduous task faced by the SNS administrator. There is no auto-discovery process, and so all the information needs to be entered by hand, and maintained in the same way as new sensors, routers, hubs and subnets are created or relocated.

The network can be mapped logically (such as by department) or physically (such as by building or network segment), and it is necessary to include nodes for all devices and device interfaces that SNS should monitor (including external devices and Smart Agents for event aggregation), as well as all those through which SNS should be able to track flows. However, if it is not required to take advantage of some of the more advanced features of SNS, then only the sensor details need to be configured in the topology tree.

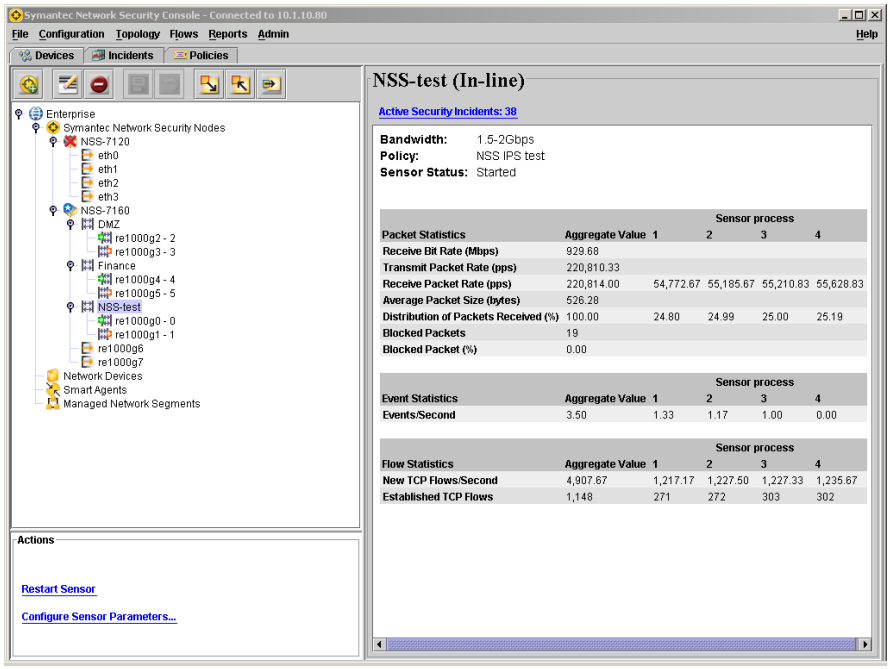


Figure 3 - SNS 7160: The Devices tab in the Console

The topology database must be populated on a master node, which subsequently synchronises the databases on all other nodes in the cluster to the master's topology database. This means that even for the largest distributed network, the topology database needs defining in one place only.

Types of objects which can be included in the topology database include:

- ***Locations**: Objects that represent physical or logical groups of one or more network segments. The installation procedure automatically creates the first location object, named Enterprise by default.*
- ***SNS nodes**: The object category for both software and appliance nodes.*
    - ***Software nodes**: Objects that represent the SNS software installed on a designated computer.*
    - ***7100 Series nodes**: Objects that represent the SNS 7100 Series appliances.*
- ***Network devices**: The object category for both routers and router interfaces.*
    - ***Routers**: Objects that represent devices that store data packets and forward them along the most expedient route. SNS monitors this connection between hosts or networks.*

    - ***Interfaces**: Objects that represent boundaries across which separate elements can communicate. Interfaces provide the point of contact between SNS and routers.*

- ***Smart Agents**: Objects that represent the entry point for event data from Symantec Decoy Server, Symantec Network Security Smart Agents, and other third-party sensors.*
- ***Interfaces**: Objects that represent boundaries across which separate elements can communicate. Interfaces provide the point of contact between SNS and the network devices.*
    - ***Monitoring interfaces**: Objects that represent dedicated ports that mirror incoming or outgoing traffic on a software or appliance node.*
    - ***In-line pairs**: Objects that represent pairs of interfaces on a 7100 Series appliance node that are directly in the network traffic path. For a given flow, one interface connects to inbound traffic and the other to outbound traffic. Only in-line pairs can be configured to block malicious traffic.*
    - ***Interface groups**: Objects that represent groups of two to four interfaces on a 7100 Series appliance node that share a common sensor. Interface groups are used to monitor asymmetrically routed network environments.*

Note that many of these are optional, and only the appliance and interface details are actually required to have the system up and running. At least one in-line pair (or single passive-mode port) needs to be configured for a node before a Protection Policy can be applied.

The *master node* is created by default during the install process - additional nodes can be added as independent single nodes or as *slave nodes* in a cluster. A slave node is synchronised with a master node within a cluster or group of SNS nodes, whereas a single node behaves like a master node in a cluster of one.

Selecting any SNS node in the tree brings up a status display in the right hand pane which exposes one of the most complete sets of statistics we have seen in any GUI. This display includes detailed packet, event and flow statistics on a per-node and even per-process basis, providing a wealth of information which is useful when troubleshooting or tweaking settings for maximum performance.

It is also possible to define external sensor nodes - Symantec Decoy Server deception hosts or other Symantec Smart Agents for example - from which SNS can accept and correlate security events.

Once the topology database has been populated, the bulk of the configuration work is done. All that is then required is to create Protection and Response Policies and apply them to the configured in-line pairs - this is covered in the following section.

As far as configuration goes, there is not much else to do. There are a couple of parameter screens at a node and sensor level which enable the administrator to modify advanced settings such as maximum log size, sensor alert delays (to prevent flooding), and maximum events per incident, but most of the entries here are accompanied by dire warnings about what happens if you modify them and get it wrong. Thus, for an initial deployment these can be safely ignored.

However, we do like the way such advanced tuning parameters are exposed in the GUI, and the descriptions for each one are very clear in the potential effect each parameter has on performance, memory usage, and so on. These are also very well documented in the *Administration Guide*.

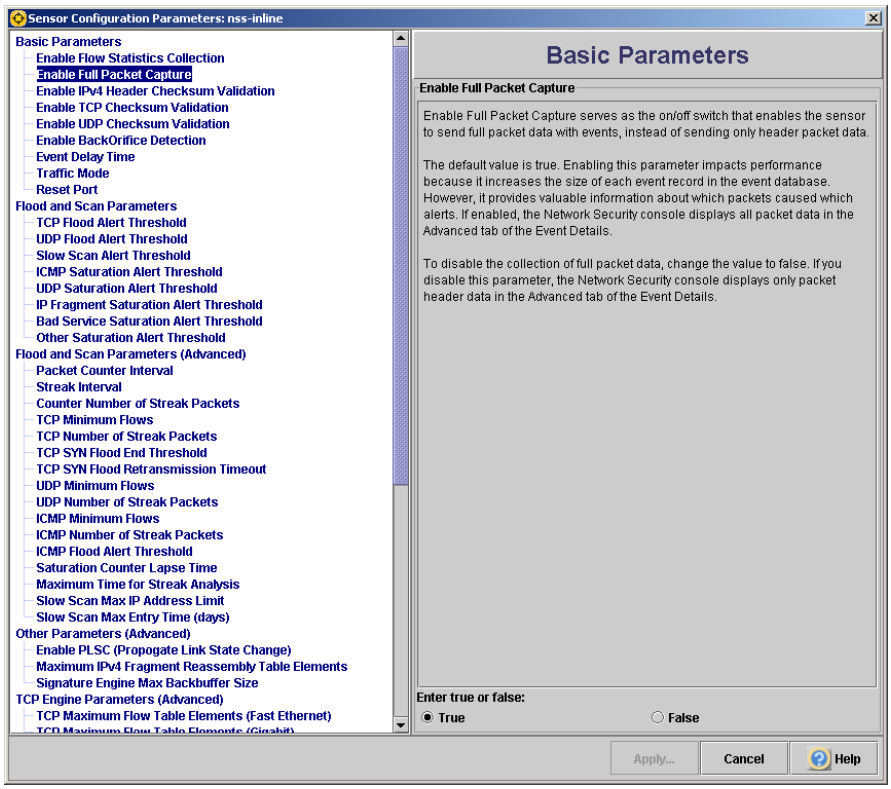


Figure 4 - SNS 7160: Configuring Sensor parameters

Having configured the topology, all the changes are propagated to every slave node in the cluster automatically. Changes only need to be made in one place for them to be available from every administration console in the organisation.

There is not much to do in the way of configuring clusters. All that is really required to make two nodes part of a cluster is to allocate the same management port and password to both of them (though each will have a different node ID).

The administrator authenticates to the Console using a login name and password, and specifies the IP address and management port of the master node. All other nodes using the same port and password combination are automatically available to be managed from that console.

If it is required to manage another cluster, it is necessary to initiate another instance of the console, and log in to a separate master node with a separate management port and set of login credentials.

Four different user levels provide varying degrees of access to the Console and SNS configuration data, but it is not possible to restrict access to individual nodes, interfaces or Protection Policies on a per-user basis, something which is becoming a key requirement in an enterprise-level management system.

Backup and restore capabilities are provided to allow the administrator to backup entire cluster or individual node configurations to disk or compact flash. These can be restored later to the same cluster/node, or a different one in case of failure.

Policy Management

One of the major improvements over early versions of the management console is the support for *Protection Policies*.

There is no way to restrict access to individual Policies or nodes on a per-user basis, however, which is something of an omission. Given sufficient rights, every administrator has access to every policy and can apply those policies to any sensor.

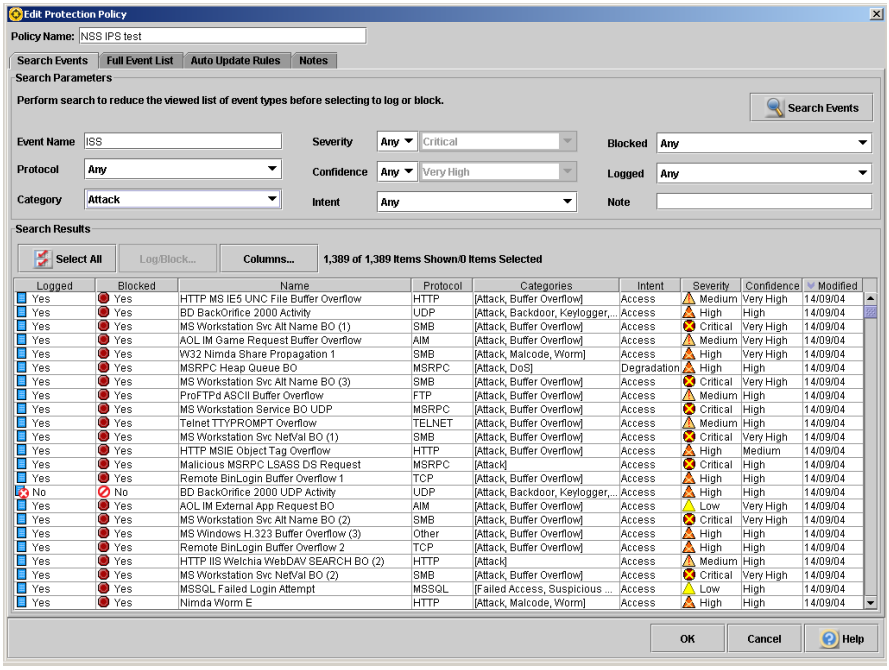


Figure 5 - SNS 7160: Editing Protection Policies

In addition, *Response Policies* can optionally be created to enhance the existing response actions and customise how the SNS system responds to individual alerts. Any number of *Response Rules* can be created, each one matching on source, destination, event type and priority.

When a response rule matches, an alert is sent to the SNS Administration Console and one of several additional actions can be taken:

- ***E-Mail alert -*** *Sends an e-mail message to the administrator. A throttle can be put on the number of e-mail alerts sent in a given time period for a specific policy*
- ***SNMP alert -*** *Sends a trap to a central network management console*
- ***TrackBack -*** *The TrackBack response is designed to track attacks back to their sources. This capability is especially important for tracking denial-of-service attacks that must be traced to their source in order to shut them down most effectively. TrackBack automatically tracks a data stream to its source within the cluster, or, if the source is outside the cluster, to its entry point into the cluster. It does this by gathering information from routers or its own sensor resources.*

- ***Custom Response -*** *Runs a command or script*
- ***TCP Reset -*** *Tears down the suspect session by sending a TCP reset to both source and destination hosts when monitoring in passive mode*
- ***Traffic Record -*** *SNS is already capable of capturing the entire packet that triggered the alert. The Traffic Record response allows it to initiate traffic recording upon detection of a specified event. Traffic parameters including protocol type (TCP, UDP, ICMP), source and destination port and IP address, as well as capture size and/or time period, can all be used for filters. This allows security specialists to capture everything from the full pipe stream down to a few packets in the same connection.*
- ***Console Response -*** *Execute a command at the Console (to sound an alarm or run an external alerting program, for example)*
- ***Setting Export Flow Response Action -*** *The export flow response action exports matching flows stored in the flow data store. The action is based on the characteristics of the triggering events, which are specified by parameters that the administrator provides when creating the rule. The administrator can use Export Flow to specify the event characteristics of the triggering event, and flows that match the specified characteristics are exported and saved.*

Any number of Response Rules can be created, and although each rule can only have one response, the rule base is traversed from top to bottom whenever an alert is raised. This means that multiple rules - triggering multiple responses - can be matched for a single alert. It would be nice, however - and much easier - to allow each rule to have more than one response allocated to it.

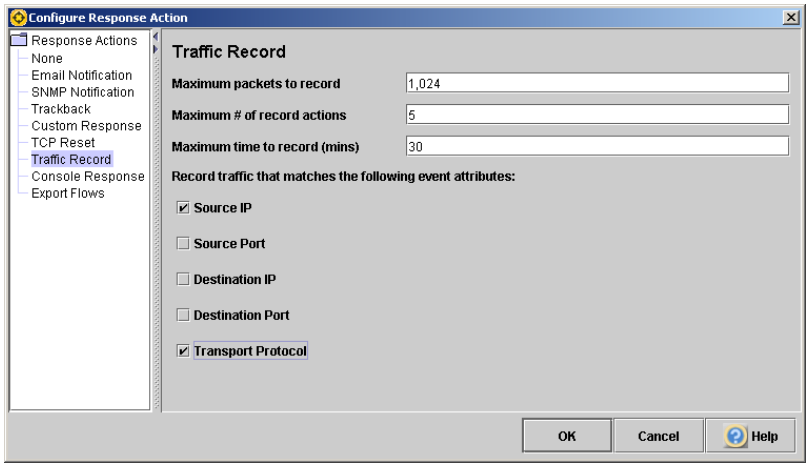


Figure 6 - SNS 7160: Configuring Response Rules

In addition to Response Rules, SNS can respond to network traffic according to *Flow Alert Rules*. These respond to traffic flows that violate defined security policies on monitored networks, and can be configured to notify the administrator when a sensor or router detects flows that match specific criteria. SNS collects data about network flows from various devices, optimising the data to enable response actions such as TrackBack and to notify the administrator about illegal flows.

To trace an attack back to the source (or at least to the edges of the administrative domain), SNS uses a component called FlowChaser to retrieve the Netflow data provided by Cisco routers to accomplish flow detection and analysis of the flow.

The Cisco Netflow technology is a proprietary source of device information that provides the metering base for a key set of applications including network traffic accounting, usage-based network billing, network planning, network monitoring and data mining capabilities for both service provider and enterprise customers. SNS' FlowChaser uses the Netflow data to provide TrackBack with additional traffic shape information that is analysed to quickly determine the ingress point of a DDoS attack within the network infrastructure.

Out of the box, Symantec provides a number of default Protection Policies which are split into "*detection only*" and "*prevention*" policies.

Prevention policies have a subset of the signatures configured to block traffic, whilst detection policies perform no blocking. A number of policies are provided in each category, each tuned for a different deployment location (DMZ, for example) or for different level of threat (*Web server, worm protection, all exploits, exploits and audits*, and so on).

Although the default Protection Policies can be applied as they stand, many administrators will be more comfortable creating their own. It is not possible to change the default policies, but any one of them can be cloned as a starting point for a custom policy, making it easy to be up and running quickly.

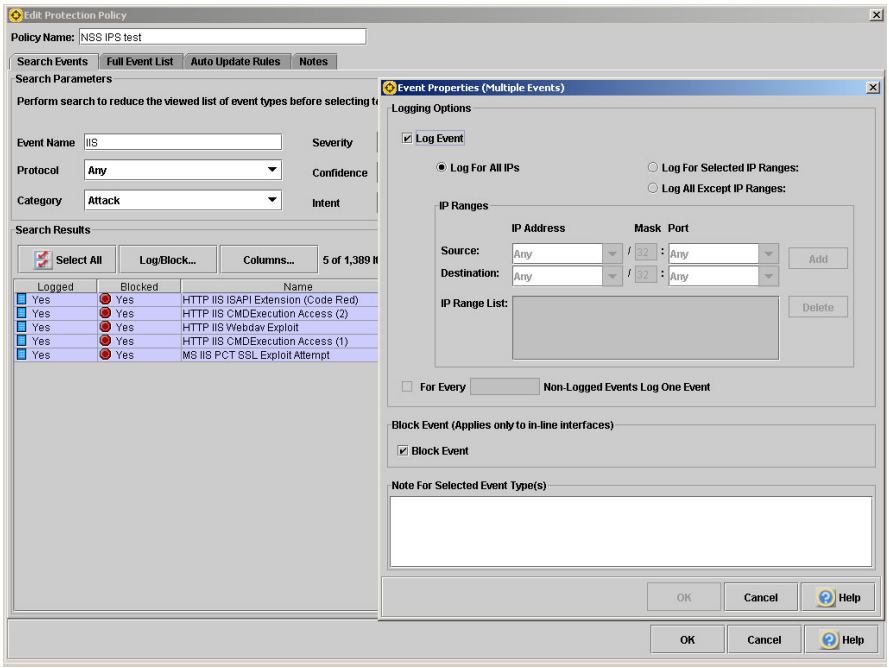


Figure 7 - SNS 7160: Searching and bulk editing signatures in Policy Editor

Policy editing is very straightforward thanks to the advanced search tab, which presents an extensive set of search criteria in the top half of the window, and the results of the search in the lower half. Search criteria include:

- **Event Name** - search on complete or partial event names. It is possible to search for specific events by entering a complete event name, or for all *IIS* or *FTP* events by entering just those characters, for example
- **Protocol** - protocol is selected from a pull-down list
- **Category** - the event category can be selected from a range of options in a pull-down list, including *Attack, Audit, Backdoor, Trojan, Worm,* etc.

- **Severity** - the severity level can be selected from *Critical, High, Medium, Low* or *Informational*
- **Confidence** - confidence level is selected from *Very High, High, Medium, Low,* or *Very Low*. This indicates the confidence that the Symantec signature writers place in the accuracy and resistance to false positives of any given signature
- **Intent** - intention is selected from a range of options in a pull-down list, including *Access, Degradation, Reconnaissance*, etc. This describes the ultimate intent of the exploit (degrade performance, subvert to gain administrator/root access, probe services and open ports, and so on).
- **Blocked** - select those signatures where *blocking* has been set
- **Logged** - select those signatures where *logging* has been set
- **Note** - within the policy editor it is possible to annotate individual events. This field can be used to search for the contents of the annotations

Any number of these can be selected together, providing the means to select a very precise subset of the available signatures. The results are available in the lower half of the screen (a summary of the number selected out of the total available is displayed above this) and it is possible to select all of the results for bulk editing operations such as setting all to block, or log, in a single operation. It is also possible to set the same annotation for a group of signatures which would make it simple to re-select the same group at a later date, since it is possible to search on just the annotation.

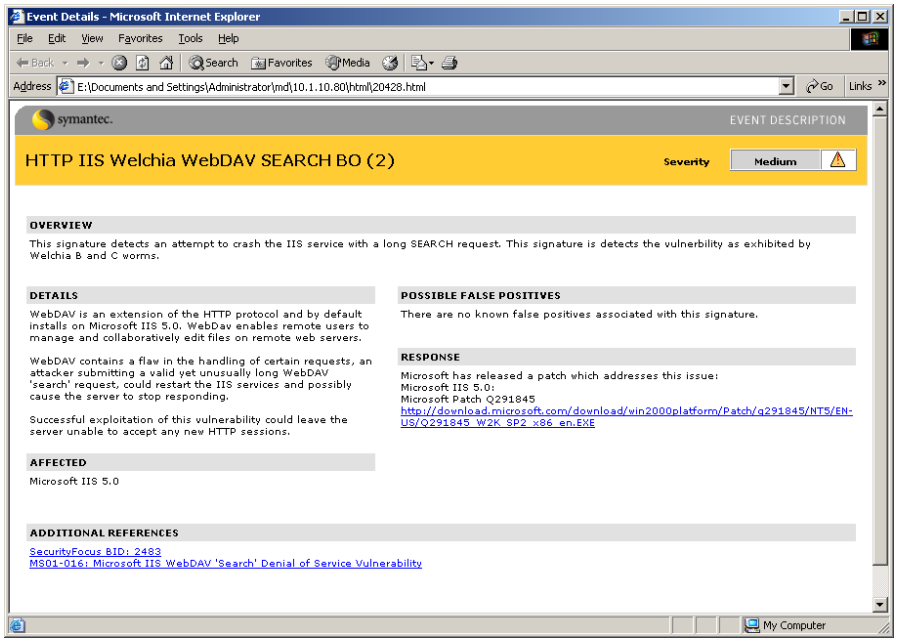


Figure 8 - SNS 7160: View detailed DeepSight-based information

A right-click on any signature provides instant access to the *log/block* configuration screen, as well as the ability to view the detailed event description. Buttons above the search results provide one-click means to *select all* and access the *log/block* configuration screen for the group selected.

It is also possible to sort the signatures using any of the columns headers (by *Severity*, *Protocol*, or *Date Modified*, for example) and then use the normal Windows selection keystrokes to select a subset of the signatures displayed.

This provides a simple means to select all of those signatures provided in the latest LiveUpdate (via *Date Modified*), or all of those currently set to *Block*, or all *FTP* signatures, and so on.

Individual signatures, or groups, can be edited to log all occurrences, or just those which fall within a specified range of source/destination IP addresses an ports. It is also possible to set an aggregation counter, such that for every user-specified number of non-logged events, a single event is logged (ideal for those events of which the administrator thinks he may see an excessively large number).

Every signature can have a custom annotation/note applied to it, which provides a good way for the administrator to document policies (why a specific signature has been disabled, or why it is set to block but not log, for example) or to create custom groups of signatures by virtue of the fact that they all share the same annotation.

Finally, it is possible to set events to block or pass traffic. Blocking traffic is the only prevention option configurable at this level, since the Response Rules mentioned earlier handle other responses at a global level. Although the rules provide an extremely flexible and powerful way of defining responses across all signatures, it would be nice to be able to override these on a per-signature basis - perhaps you might want to enable packet capture temporarily for a single signature, for example.

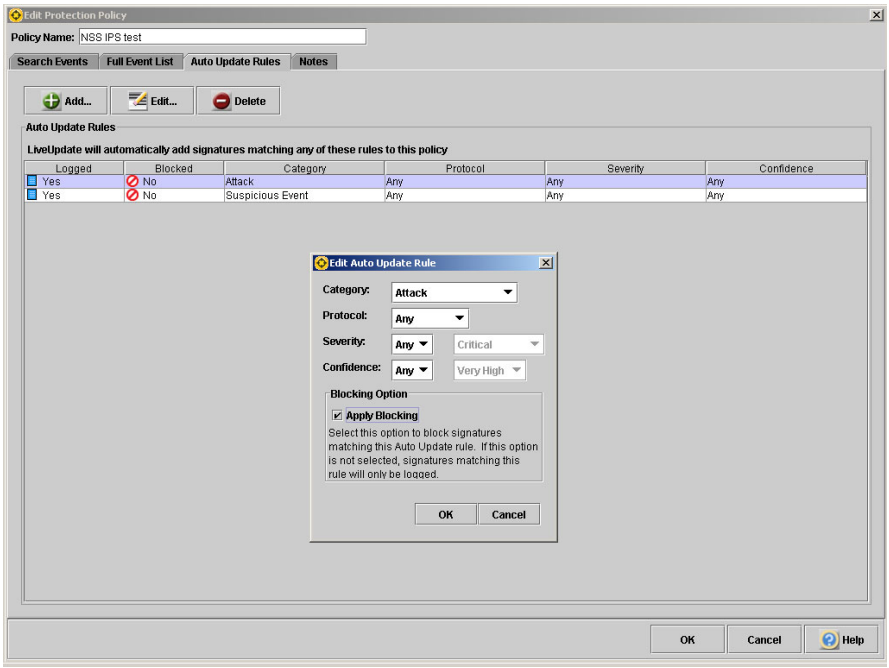


Figure 9 - SNS 7160: Auto-Update Rules

The Block response sends TCP resets to both client and server, in addition to dropping the offending packet and blocking all subsequent packets from the same flow. In the version of software tested, this behaviour was fixed, but as of patch 10 for the SNS 7100, it is possible for administrators to disable the automatic transmissions of resets in both directions upon a blocked event.

In addition to the signature selection and search tabs, there are two more available: *Notes* and *Auto-Update Rules*. Notes provides a free-format text area to enable the administrator to create documentation at a policy level.

Auto Update Rules allows the policy to be updated automatically each time a new set of signatures are acquired.

SNS uses the standard Symantec LiveUpdate mechanism (which will be familiar to users of its Anti Virus products) to provide access to new and updated software and signatures automatically. These updates can be downloaded directly from the Symantec servers, or "mirrored" LiveUpdate servers can be created on an internal network to conserve bandwidth when there are multiple appliances installed.

Of course, normally it would be necessary to review new signatures and add them to Protection Policies manually, but the *Auto-Update Rules* provide a means to streamline this process.

The administrator can create multiple rules per Policy based on signature *Category, Protocol, Severity* and *Confidence*, and where new signatures match those conditions, they will be added to the Policy automatically. A further option allows the new signature to be switched into *blocking* mode immediately, or activated in *logging* mode only. Thus, even if the LiveUpdate occurs in the middle of the night, SNS can immediately start logging and/or blocking the matching events.

Should the administrator need to check on and review new signatures, it is a simple matter to use the search capabilities in the Policy Editor to sort and search for signatures with a specific *Date Modified*, but otherwise there is nothing contained within a signature to say with which LiveUpdate it arrived.

For many administrators, this mechanism could provide an almost "install and forget" means of maintaining their Protection Policies, and this capability is currently only available from a couple of products in the market.

Another nice feature of SNS 7100 is the "*One-Click to Prevention*" feature. Where a blocking policy is applied to a sensor, it is possible to disable and enable all blocking at the touch of a button. When disabled, the sensor continues to detect events as normal, but will only log alerts and no longer block traffic. Once the administrator is confident in the accuracy of the policy, the blocking capability can be re-enabled instantly.

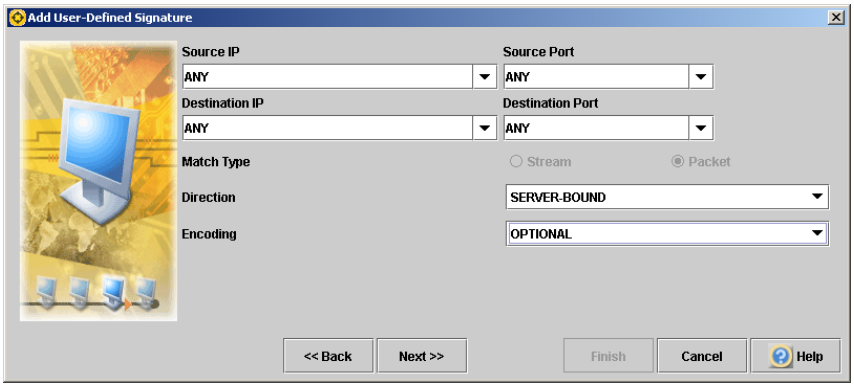


Figure 10 - SNS 7160: User-Defined Signatures

Any number of user-defined signatures can be created on a per-sensor basis, and there is a very simple Wizard available to help with this process. Signatures created in this manner are called "*basic signatures*", and are restricted to matching simple patterns at given offsets within a packet, as well as matching on protocol, source and destination IP address, and port.

For more advanced requirements, there is a complete signature writing language available, using complex structures, functions, regular expressions, and so on. These signatures are compiled and imported via the GUI. User-defined signatures are synchronised across clusters so that each node has the title, severity, and definition of the user-defined signature. The signature writing “language” has undergone radical enhancement in recent engine updates, and now provides the capability of creating some extremely advanced signatures.

Once the signatures are loaded into the database, they are available for selection in all Policies. Although the SNS Policy Editor does not support the concept of signature groups, all user-defined signatures have the same SNS_GEN prefix to their name, so it is possible to use the search capabilities to search for all signatures starting with SNS_GEN and process them as a group in that way.

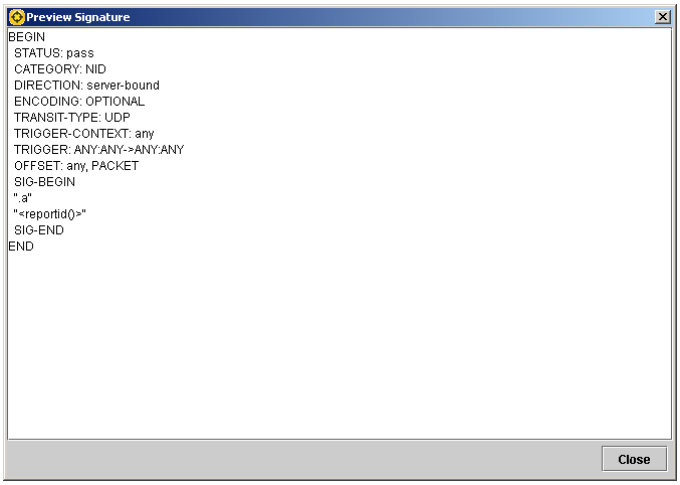


Figure 11 - SNS 7160: Signature writing language example (basic signature)

Once the Policy has been completed, it is necessary to click on the *Apply* button to save it. This not only saves the Policy itself, but also automatically applies it to all of the Nodes and Sensors which already have that Policy applied.

Unfortunately, there is no way to save the edited Policy under a different name at this point. This means that if the administrator’s original intention was to clone an existing Policy and amend it rather than edit the original, or if he feels that he may have made a mistake and wishes to save under a temporary name while he investigates further, there is no option but to discard the changes (there is a *Revert* option to do this) and start again.

It is possible to make multiple changes to several different Policies, and *Apply* them all on one single operation. However, when changes to several Policies are made in the same session, it is a shame that there is no on-screen indication of which nodes are awaiting application of changed policies or even which policies have been changed.

One other significant shortcoming of the current system is that there is no history/audit capability to be able to track *who* changed *what* on *which* Policy or Node, or monitor the date of last time a policy was applied, by whom, and so on. Rollback capabilities and policy versioning (tracking all versions of a Policy through multiple edits) are also missing from the current version.

However, one area which is very strong is the actual deployment of the finished Protection Policies. Selecting any Policy and clicking on the "*Set to Interfaces*" button brings up the ubiquitous topology tree again with a checkbox against each hierarchical level and each individual node down to the interface level.

By clicking one or more of the checkboxes (clicking on one of the upper levels, such as a physical location or department name, causes all the nodes beneath it in the tree to be checked automatically) allows the administrator to apply a policy enterprise wide - or to an individual node - in a single operation.

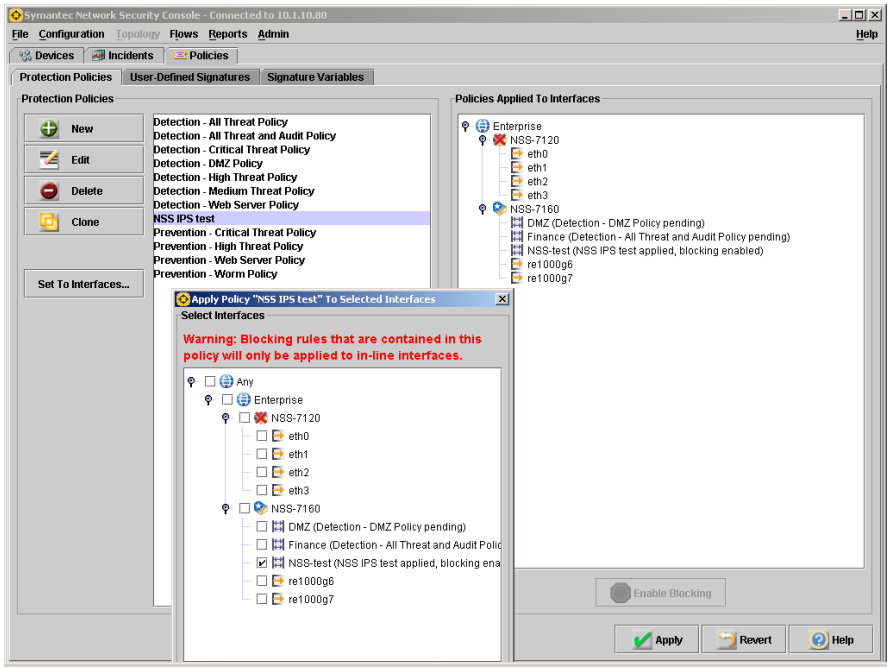


Figure 12 - SNS 7160: Applying Protection Policies

Once the policies have been applied, the right hand pane of the *Protection Policies* screen shows the same hierarchical topology tree with a display showing which Protection Policy is currently applied to which interface - very useful.

Alert Handling

There are two tabs used for monitoring and alert handling in the SNS Console:

- ***Incidents -*** *Displays both active and idle incidents. Selecting an incident in the upper pane causes the lower pane to display information about the related events within that incident*
- ***Devices -*** *Displays the topology tree in the left-hand pane. Selecting an object in the topology tree displays related information in the right-hand pane, including a link to security incidents that are currently active on that object.*

  Also displayed are useful real-time statistics which change depending on what is selected in the topology tree. When a specific interface or in-line pair is selected, for example, the statistics include received packets, blocked packets, average packet size, data rate, current active flows, and so on.

The *Incidents* tab provides the main view on current and previous incidents, both active and idle. Incidents are actually a collection of individual events, grouped together by the SNS correlation engine based on a sophisticated algorithm that takes into account things like event type, event name, source IP, destination IP, and so on.

The result is a vastly reduced amount of information for the administrator to sift through in order to determine the most important events to be investigated. Configuration parameters provide the means to tune the weightings given to each of the above criteria in order to customise the way events are correlated to a certain extent, making the correlation feature even more usable.

By intelligent use of the weightings, it is possible to have SNS group events together in completely different ways (i.e. all events of a specific name grouped as an incident regardless of source or destination, or all events targeted at a particular host grouped together) or even effectively turn off correlation altogether.

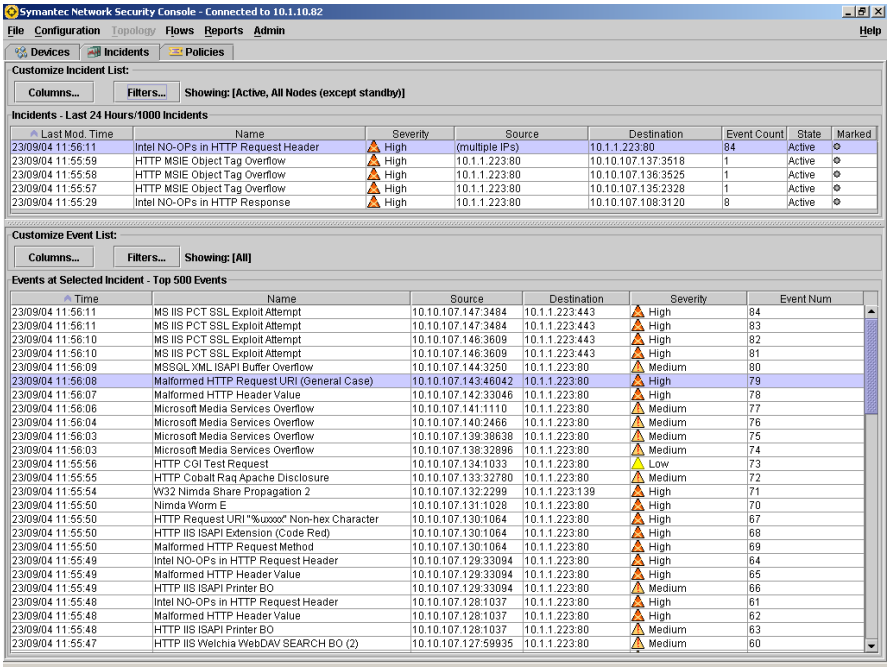


Figure 13 - SNS 7160: Viewing Incidents and Events

Even though individual event types can be quite different, SNS is reasonably adept at determining which of them are related, and raising the priority and response status depending on how an incident develops. The top level heading of each incident always reflects the highest priority event within that incident.

For example, low priority port scans and other reconnaissance probes are usually followed by more invasive probes, and eventually by a serious exploit attempt. SNS will consider port scans as low priority, and - depending on the response policy in place - will usually not alert the administrator directly. As the intruder moves on to more invasive techniques, however, the overall priority of the incident may be raised to a level which causes an e-mail or pager alert to be raised. This correlation is performed across all nodes in a cluster, and this makes SNS one of the more useful products in terms of alert handling.

Active incidents are those to which new events are still being added. Incidents to which no new events have been added for a given amount of time are considered idle, so SNS closes them. The display can be filtered to exclude closed events if required, leading to an uncluttered display of data which is of immediate interest. The closed events remain available for further analysis, however, and can be restored to the main display at the click of a mouse button.

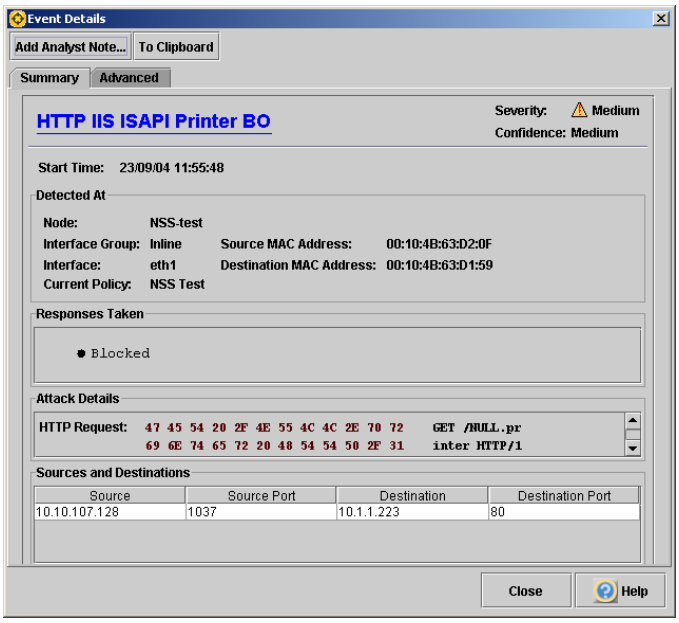


Figure 14 - SNS 7160: Viewing Event details

The Incidents table allows the administrator to view incidents detected by all SNS nodes or only those detected by a particular SNS node, and it is possible to restrict individual Consoles to viewing events from certain SNS nodes only. Information displayed includes *date and time, event type, source, destination, event count, device name, location, marked, node number, node name* and *current state (active or idle).*

To keep track of which incidents have been viewed or processed by the administrator, it is possible to select the *Mark* Incident check box in the *Incident Details* dialogue (or by right-clicking on an incident). The *Marked* column of the incident will then display an indicator.

Selecting any incident from the *Incident Table* lists all the events that make up that incident in the bottom half of the Incidents screen. A single line is displayed for each event, consisting of *date and time, event type, name, source, destination, severity, confidence, event number (within that incident), device name, interface group, VLAN ID* and whether or not it was *blocked*. Not all of the above fields need be displayed - it is possible to select the columns to be displayed on-screen and, as with incidents, additional filters can quickly and easily be applied to further refine the data displayed.

The name of an incident and its severity are determined from the highest priority event within it, and this is a function of the event's *severity* and *confidence*. *Severity* is a measure of the potential damage that an attack can cause, whilst the *confidence* indicates the level of certainty that a particular event is actually an attack.

If the incident is merely suspicious, or if there is a propensity for false positives, then its assigned reliability level is low. If SNS collects more data on the incident to substantiate its reliability, the reliability is adjusted upward automatically.

Double-clicking any individual event brings up the event details dialogue. This repeats all the event details from the list on the first tab, along with a list of the source and destination IP addresses and ports that have been involved in that incident. If flow status collection is enabled on the monitored interfaces, then SNS can also display information about network flows to and from each address.

The *Advanced* tab shows detailed packet content data, whilst the long description shows in-depth exploit info, with cross reference links such as CVE, BugTraq, etc. SNS provides an extremely intuitive means to drill down quickly from high level data to packet contents, allowing an administrator to investigate incident and events in detail and with a high degree of effectiveness.

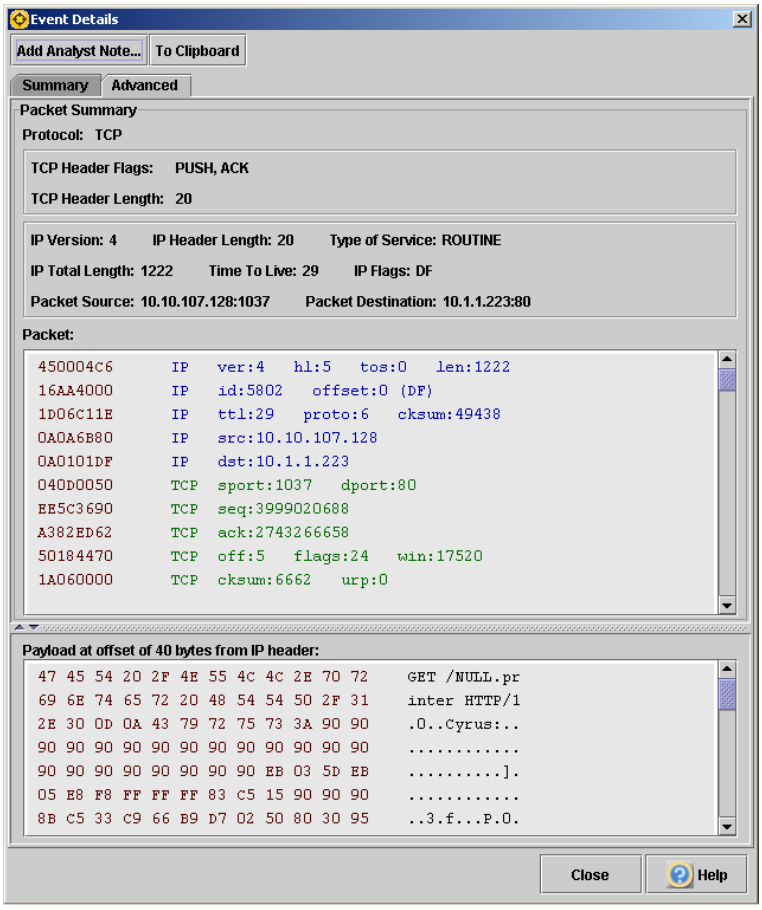


Figure 15 - SNS 7160: Viewing packet data for an Event

Another extremely useful event management feature is the ability to annotate incidents and events. Administrators can add multiple comments to incidents and events by clicking the *Annotation* button in the *Incident Details* or *Event Details* dialogue respectively.

Each annotation is time stamped and lists the author of the annotation. If there are multiple annotations for an event, they can be sorted by time stamp in ascending or descending order.

The Annotation box in the Annotation dialogue displays a default template which prompts for information, such as ticket number, owner and the last action taken in response to the event.

Filters can quickly and easily be applied to refine the incident and event data displayed by eliminating (or only displaying) marked events, by eliminating (or only displaying) annotated events, or by eliminating (or only displaying) events from a particular SNS node.

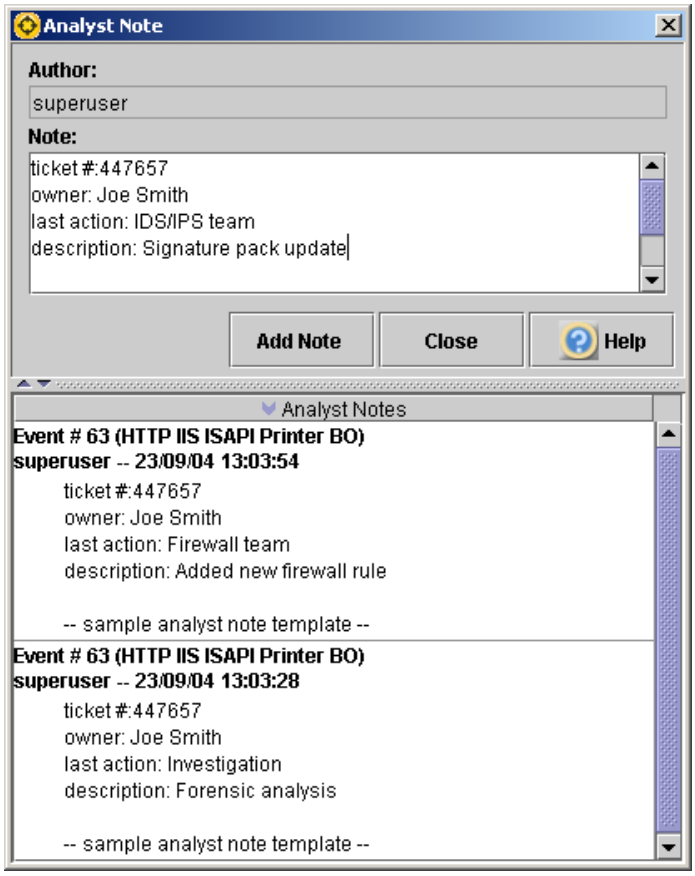


Figure 16 - SNS 7160: Annotating Events

Note that all the event and incident data is actually stored on the individual SNS nodes and is only transferred to the console as required for alerting and reporting. This can make some operations very slow as a console attempts to gather large numbers of incidents and events from multiple SNS nodes across a network.

The upside to this approach is that with no single central database there is no single point of failure or single performance bottleneck. Another advantage for those organisations that have round the clock incident monitoring on a "*follow the sun*" basis (i.e. in different time zones) is that any administrator logging on to a cluster from any console around the global network has instant access to the identical data that was being worked on by all the other administrators before him.

All in all, the incident handling capabilities of SNS are very impressive. The ability to handle large quantities of data are vital in a high-bandwidth systems, especially when it is possible to include events from third party sensors, and SNS acquits itself with honours in this area.

Reporting and Analysis

Summary reporting is well catered for within SNS, with a range of built in graphical and text reports, though it is not possible to customise them in any way (other than to select a date range). For more detailed reporting, data can be exported to a SQL database.

Any SNS node can generate daily e-mail reports that contain detailed incident log information for all SNS nodes in the cluster. The report time period is always the last 24 hours leading up to the report generation time.

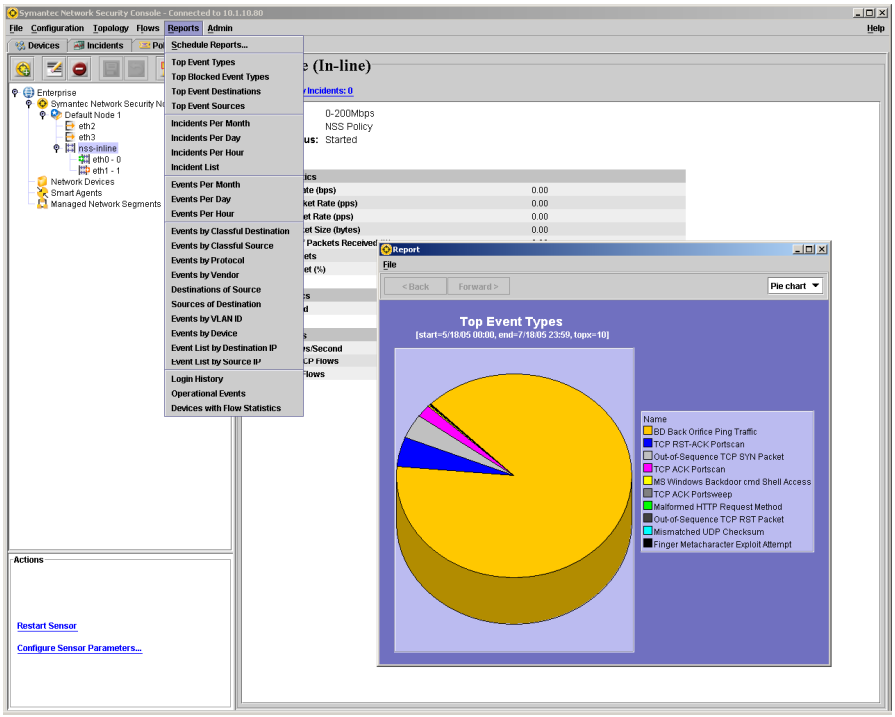


Figure 17 - SNS 7160: Running reports

The e-mail reports include the following information:

- *Report start time*
- *Node number of the SNS that generated the report*
- *SNS login history with the source IP and time of the login event.*
- *Event types that occurred most frequently during the report period, and the number of times each event type occurred (up to 20 unique event types).*
- *Event source IP addresses followed by the number of times each IP address occurred (up to 20 unique addresses).*
- *Event destination IP addresses followed by the number of times each IP address occurred (up to 20 unique addresses).*
- *Highest severity incidents followed by the source and destination IP addresses, as well as the time the incident occurred (up to 20 separate incidents).*

Reports generated at the Console consist of data collected from all SNS nodes in the cluster. The data collection is performed as the report is running, and so can be quite slow when large numbers of nodes are involved.

Most reports are presented in graphical format initially, with a choice of bar charts, column charts or pie charts. Right-clicking any one of the columns of data on a chart allows the administrator to drill down to the next level. If, for example, the report is *Incidents Per Month*, right-clicking one of the individual columns drills down to that particular day and presents incidents per hour.

Right-clicking on one of the columns on that graph will drill down to the incidents that occurred within that hour, from where it is possible to generate other drill-down reports, such as a *Top Event Types* report to view the types of events that occurred most frequently during that hour. Eventually, it is possible to drill down to lists of the individual incidents and events (though it is not possible to directly access the individual event details from this point, which is a shame).

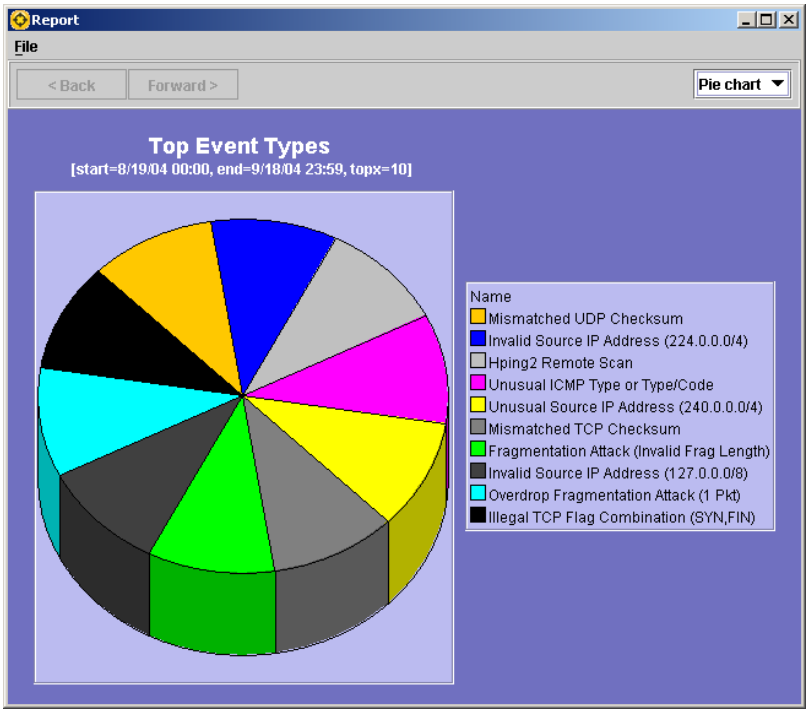


Figure 18 - SNS 7160: Typical graphical report

This approach makes it very easy to filter out unnecessary data and quickly drill down to the data which is of most interest, providing a more focussed level of detail. All console reports can be printed directly or saved as HTML, PostScript or PDF files if required.

Multiple report schedules to be created to run reports - either singly or as groups - at regular intervals. Scheduled reports run directly on any SNS node in a cluster, and the finished reports can be e-mailed to an administrator or secure copied to a report server elsewhere on the network. Regardless of which node is running the reports, SNS gathers data from all nodes to generate a comprehensive cluster-wide report.

Reports available include:

- ***Top Event Types/Top Blocked Event Types -*** *lists the event types - such as SYN flood, Telnet DoS and Port scan - that occurred most frequently during the time period specified, and the number of times each event type occurred. Separate reports are available to cover all events or blocked events only.*

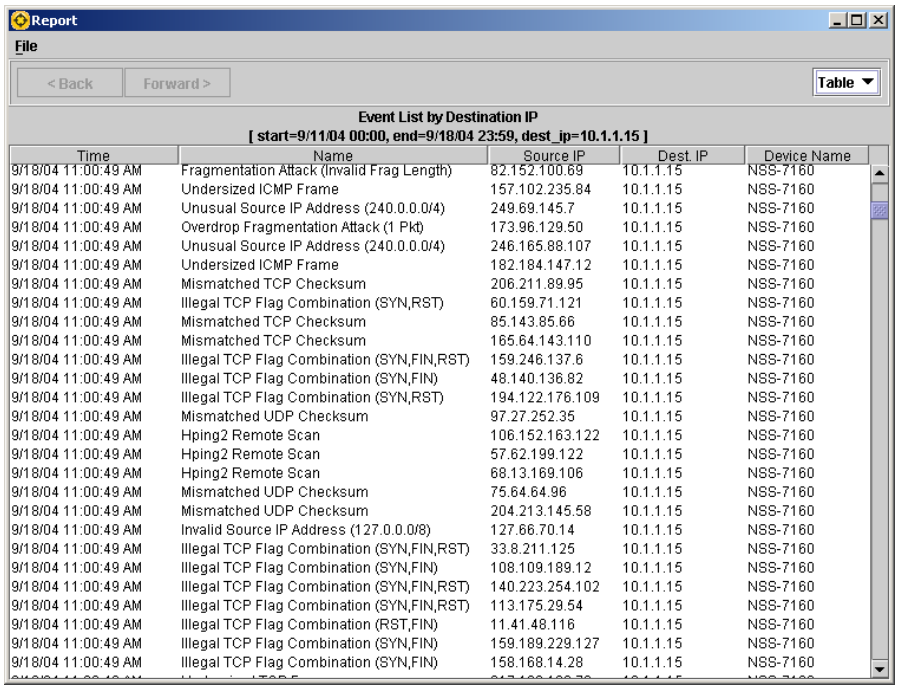
- ***Top Event Sources/Destinations -*** *lists the most frequently occurring source/destination IP addresses of detected events.*
- ***Incidents Per Month/Day/Hour -*** *displays the total number of incidents that occurred during a specified time period.*
- ***Incident List -*** *for each incident that occurred during the report period specified, this report lists the incident start date and time, event type to which the incident is mapped, the name of the device where SNS detected the incident, and the number of the SNS node that detected the incident.*
- ***Events Per Month/Day/Hour -*** *displays the total number of events detected in the time period specified.*
- ***Events by Classful Source/Destination -*** *sorts events by their source/destination IP addresses and presents a count of the number of addresses that are from class A, class B and class C networks.*
- ***Events by Protocol -*** *lists the number of events detected that exploit each particular protocol, such as ICMP, UDP, TCP, or IP.*
- ***Events by Vendor -*** *lists the number of events detected per vendor (i.e. when multiple third-party IDS are used for data collection).*
- ***Destinations of Source -*** *lists the destination IP address(es) for any event source IP address specified, along with the number of times each address was the destination for the source address.*
- ***Sources of Destination -*** *lists the source IP address(es) for any event destination IP address specified, along with the number of times each address was the source for the destination address.*
- ***Events by VLAN ID -*** *lists all events grouped by VLAN IDs.*
- ***Events by Device -*** *lists all events for all devices and interfaces in the network topology.*
- ***Event List by Destination/Source IP -*** *lists all the events contained in the selected incident or time period, as well as the event end time, the event source and destination IP addresses, and the name of device where the event was detected.*
- ***SNS Login History -*** *lists the user login times, IP addresses from which the user logged in, and the type of user that logged in (either an administrator with full read/write privileges or a user with limited privileges).*
- ***SNS Operational Events -*** *lists operational events such as user logins, communication errors, response actions, and license status notification.*
- ***Devices with Flow Statistics -*** *lists names for devices on which the Flow Status Collection sensor mode is enabled, and the number of the SNS node where the sensor is located.*

Although most top-level report types are also available as drill-down reports within other top-level reports, there are some reports within the Console which are **only** available as drill-down reports, and not available from a menu. These include:

- ***Incident Details -*** *lists individual incident details*
- ***Event List-*** *For the incident selected, data is displayed within the Incident List report.*
- ***Event Details -*** *Displays the data within any Event List report*
- ***Sources/Destinations of Events -*** *lists the source/destination IP address(es) for the event selected.*
- ***Sources/Destinations of Event -*** *lists all of the source/destination IP addresses for the selected event.*

- ***Flows by Source/Destination Address -*** *lists the source/destination addresses of flows found on devices with the Flow Status Collection sensor mode enabled.*
- ***Flows by Source/Destination Port -*** *lists the source/destination ports of flows found on devices with Flow Status Collection sensor mode enabled.*
- ***Flows by Protocol -*** *lists the protocols of flows found on devices with Flow Status Collection sensor mode enabled.*

FlowChaser serves as a data source in coordination with *SNS TrackBack*, a response mechanism that traces a DoS attack or network flow back to its source. The FlowChaser database can be queried for flows by port and arbitrary address. The Network Security Console displays both current flow data and exported flow data, and provides secondary query options from the results page.


Report

File

< Back | Forward > | Table

Event List by Destination IP

[start=9/11/04 00:00, end=9/18/04 23:59, dest_ip=10.1.1.15]

Time	Name	Source IP	Dest. IP	Device Name
9/18/04 11:00:49 AM	Fragmentation Attack (Invalid Frag Length)	82.152.100.69	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Undersized ICMP Frame	157.102.235.84	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Unusual Source IP Address (240.0.0.0/4)	249.69.145.7	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Overdrop Fragmentation Attack (1 Pkt)	173.96.129.50	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Unusual Source IP Address (240.0.0.0/4)	246.165.88.107	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Undersized ICMP Frame	182.184.147.12	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Mismatched TCP Checksum	206.211.89.95	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (SYN,RST)	60.159.71.121	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Mismatched TCP Checksum	85.143.85.66	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Mismatched TCP Checksum	165.64.143.110	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (SYN,FIN,RST)	159.246.137.6	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (SYN,FIN)	48.140.136.82	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (SYN,RST)	194.122.176.109	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Mismatched UDP Checksum	97.27.252.35	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Hping2 Remote Scan	106.152.163.122	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Hping2 Remote Scan	57.62.199.122	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Hping2 Remote Scan	68.13.169.106	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Mismatched UDP Checksum	75.64.64.96	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Mismatched UDP Checksum	204.213.145.58	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Invalid Source IP Address (127.0.0.0/8)	127.66.70.14	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (SYN,FIN,RST)	33.8.211.125	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (SYN,FIN)	108.109.189.12	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (SYN,FIN,RST)	140.223.254.102	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (SYN,FIN,RST)	113.175.29.54	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (RST,FIN)	11.41.48.116	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (SYN,FIN)	159.189.229.127	10.1.1.15	NSS-7160
9/18/04 11:00:49 AM	Illegal TCP Flag Combination (SYN,FIN)	158.168.14.28	10.1.1.15	NSS-7160

Figure 19 - SNS 7160: Typical text report

Like the *FlowChaser*, *Query Current Flows*, and *Query Exported Flows*, the *Traffic Playback Tool* provides another way to search recorded data outside of the SNS reporting system. When a response rule is set to record events of a particular description, the Traffic Playback Tool can then be used to replay and scrutinise the records of those events. The recorded file is in PCAP format, which is easily accessed using commonly available tools such as Ethereal.

Verdict

Performance

The SNS 7160 is rated at 1Gbps when deployed in-line, and performance at almost all levels of our tests was very good, with 100 per cent of all attacks being detected and blocked under all but the most extreme (20,000 connections per second) load conditions.

Even when pushed beyond its limits (17,000cps), the SNS 7160 continued to block all malicious traffic successfully. We would happily confirm Symantec's 1Gbps rating for this device under normal network conditions.

Basic latency figures were within acceptable limits for a device of this type at all traffic loads and with all packet sizes. Behaviour throughout the tests with no background traffic and with a half load of 500Mbps of HTTP traffic was very predictable, and HTTP response times were also good.

All latency results remained below the "magic figure" of 300 microseconds, our limit for deploying in-line devices in the core of the network. This means the SNS 7160 could be situated anywhere on a Gigabit network, either internally or at the perimeter.

SYN Flood mitigation is a recent addition to this product, and 100Mbps of SYN flood traffic had no significant effect on the SNS 7160. Latency increased by only a few microseconds across all packet sizes (HTTP response times were barely affected), and the SYN Flood was mitigated almost completely.

The SNS 7160 performed consistently and completely reliably throughout our tests. Under eight hours of extended attack (comprising millions of exploits mixed with genuine traffic) it continued to block 100 per cent of attack traffic, whilst passing 100 per cent of legitimate traffic.

Exposing the sensor interface to 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).

It was increased to a perfect 100 per cent after the application of a signature pack update which was provided to us within 48 hours. Blocking performance was identical throughout the tests.

We noted a minimum of "noise", with very few test cases raising multiple alerts for a single exploit, and the accuracy of the exploit descriptions was high. Performance in our "false negative" tests was good out of the box, and there is every indication that the majority of signatures are written for the underlying vulnerability rather than specific exploits. Specific exploit signatures are also included where appropriate, however, to provide more accurate identification for the administrator.

Signature recognition capabilities have improved considerably since engine update 4 (the version tested was engine update 5) at which point the Symantec signature writing language gained numerous new features allowing the creation of much more complex signatures.

It is worth noting that the power of this language is available to administrators when writing custom signatures (though it is a shame that the built-in Symantec signatures are not open to examination).

There were no false positives noted during our testing, and resistance to known evasion techniques was perfect, with the SNS 7160 achieving a clean sweep across the board in all our evasion tests.

Not only were the fragmented and obfuscated attacks blocked successfully, but all but one of them were decoded accurately as well.

Out of the box, the SNS 7160 handled 100,000 open connections, and just over 1 million connections after tuning. Default operation of the device is to age out old connections when the state tables are full or resources are low, and this behaviour is not configurable. It would be nice to see a choice offered to the administrator of whether to age out old connections or block new ones.

Usability

The Java-based GUI is one of the strong points of the SNS product line, sporting an excellent correlation engine and good reporting and analysis capabilities across multiple sensors. The ability to correlate data from third-party IDS products makes it particularly attractive in larger enterprise deployments.

The one area to show the biggest improvement in recent releases is the policy management. Early versions of SNS did not use policies at all, and so it is good to see that not only has the concept of Protection Policies finally been introduced, but that it has been implemented so well. The policy editor was extremely easy to use, and the searching and bulk editing capabilities were excellent. Slight improvement is still required in the areas of version tracking and rollback of policy deployment, but all in all we found policy management to be good.

SNS' greatest strength, however, lies in its incident management capabilities. The tremendous challenge of sorting through volumes of network traffic requires an organisation to more efficiently manage the flow of incident information. SNS' correlation engine combines multiple events from multiple sensors into single incidents, making it much easier to sort genuine alerts from the irrelevant. The recently introduced feature allowing fine-tuning of the correlation mechanism makes this even more useful.

The ability to annotate, mark and sort incidents with SNS provides the security specialist with the means to effectively identify specific incidents that have been resolved and remove them from the current “view”.

This provides a form of incident “To-Do” list, and clearly indicates incidents that require immediate attention. Annotation capabilities allow an organisation to more thoroughly track the progress of investigation into an incident, even when that investigation is spread across multiple administrators and multiple time zones. These types of features are typical of those companies with a strong IDS pedigree, and not often seen in the newer IPS products.

Reporting and analysis are well catered for with a wide range of graphical reports and the ability to drill down from those reports into the more detailed event lists behind them and, eventually, right down to the individual event details.

Overall, Symantec is still amongst the leaders in terms of event handling, reporting and analysis. Given the potential volumes of traffic that will be seen on high speed networks, and the resulting volume of alerts that could be generated, the ability to more effectively manage those alerts is essential. SNS has one of the better approaches to this problem, and is worthy of consideration for any large scale IPS/IDS deployment.

Contact Details

Company name: Symantec Corporation

Internet: www.symantec.com

Address:
20330 Stevens Creek Blvd.
Cupertino
CA 95014

Tel: +1 (408) 517-8000

APPENDIX A – TEST RESULTS

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

The Test Environment

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

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

All “normal” network traffic, background load traffic and exploit traffic will therefore be transmitted **through** the device under test, from external to internal. The same traffic is mirrored to a single SPAN port of the external gateway switch, to which an Adtech network monitoring device is connected. The Adtech AX/4000 monitors the same mirrored traffic to ensure that the total amount of traffic never exceeds 1Gbps (which would invalidate the test run).

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

Section 1 – Detection Engine

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

Test 1.1 - Attack Recognition

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

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

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

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

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

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

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

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

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

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

Test 1.2 - Resistance To False Positives

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

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

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

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

Section 2 – Evasion

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

Test 2.1 - Baselines

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

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

Test 2.2 - Packet Fragmentation and Stream Segmentation

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

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

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

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

Test 2.3 - URL Obfuscation

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

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

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

Test 2.4 - Miscellaneous Evasion Techniques

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

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

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

Section 3 – Stateful Operation

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

Test 3.1 - Stateless Attack Replay (Mid-Flows)

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

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

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

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

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

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

Test 3.2 - Simultaneous Open Connections (default settings)

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

A legitimate HTTP session is opened and the first packet of a two-packet exploit is transmitted. The Spirent Avalanche (on the "external" interface of the sensor) then opens various numbers of TCP sessions from 10,000 to 1,000,000 (one million) with the Spirent Reflector (on the "internal" interface of the sensor). The initial HTTP session is then completed with the second half of the exploit and the session is closed. If the sensor is still maintaining state on the first session established, the exploit will be recorded. If the state tables have been exhausted, the exploit string will be seen as a non-stateful attack, and will thus be ignored.

Both halves of the exploit are required to trigger an alert - a product will fail the test if it fails to generate an alert after the second packet is transmitted, or if it raises an alert on either half of the exploit on its own.

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

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

Test 3.3 - Simultaneous Open Connections (after tuning)

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

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

Section 4 – Detection/Blocking Performance Under Load

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

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

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

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

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

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

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

Test 4.1 - UDP Traffic To Random Valid Ports

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

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

Each packet contains dummy data, and is targeted at a valid port on a valid IP address on the target subnet. The percentage load and packets per second (pps) figures are verified by the Adtech Gigabit network monitoring tool before each test begins. Multiple tests are run and averages taken where necessary.

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

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

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

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

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

By creating genuine session-based traffic with varying session lengths, the sensor is forced to track valid sessions, thus ensuring a higher workload than for simple packet-based background traffic. This provides a test environment that is as close to “real world” as it is possible to achieve in a lab environment, whilst ensuring absolute accuracy and repeatability.

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

Each transaction consists of a single HTTP GET request and there are no transaction delays (i.e. the Web server responds immediately to all requests). All packets contain valid payload (a mix of binary and ASCII objects) and address data, and this test provides an excellent representation of a live network (albeit one biased towards HTTP traffic) at various network loads.

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

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

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

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

Test 4.4 - Protocol Mix Traffic

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

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

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

Test 4.5 - “Real World” Traffic

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

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

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

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

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

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

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

Section 5 – Latency & User Response Times

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

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

Test 5.1 - Latency

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

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

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

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

Test 5.2 - User Response Times

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

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

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

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

Section 6 – Stability & Reliability

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

- ***Test 6.1.1 - Blocking Under Extended Attack:*** *For this test, we expose the external interface of the device to a constant stream of alerts over an extended period of time. The device is configured to block and alert, and thus this test provides an indication the effectiveness of both the blocking and alert handling mechanisms. A continuous stream of exploits mixed with some legitimate sessions is transmitted through the device at a maximum of 100Mbps (max 50,000 packets per second, average packet sizes in the range of 120-350 bytes) for 8 hours with no additional background traffic. This is not intended as a stress test in terms of traffic load - merely a reliability test in terms of consistency of blocking performance.*

  The device is expected to remain operational and stable throughout this test, and to block 100 per cent of recognisable exploits, raising an alert for each. Results are presented as a simple PASS/FAIL. If any recognisable exploits are passed - caused by either the volume of traffic ***or*** *the sensor failing open for any reason - this will result in a FAIL.*

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

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

Section 7 – Management and Configuration

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

Test 7.1 - Management Port

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

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

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

  Traffic load is a maximum of 350Mbps and 60,000 packets per second (average packet size is 690 bytes). Results are presented as a simple PASS/FAIL - the device is expected to remain (a) operational and capable of detecting and blocking exploits, and (b) capable of communicating in both directions with the management server/console throughout the test to attain a PASS.

  Test 7.1.3 - *We note whether the ISIC attacks themselves are detected by the sensor even though targeted at the management port.*

Symantec SNS 7160 V4.0.0.9

Section 1 - Detection Engine

Test 1.1 – Attack Recognition	Attacks	Default ARRD	Default ARRB	Custom ARRD	Custom ARRB
Test 1.1.1 - Backdoors	7	7	7	7	7
Test 1.1.2 - WINS/DNS	3	2	2	3	3
Test 1.1.3 - DOS	10	10	10	10	10
Test 1.1.4 - False negatives (modified exploits)	14	9	9	14	14
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	43	43	43	43	43
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	9	9	9	9	9
Test 1.1.11 - SSH	1	1	1	1	1
Test 1.1.12 - Telnet	1	1	1	1	1
Test 1.1.13 - Database	1	1	1	1	1
Test 1.1.14 - Mail	1	1	1	1	1
Test 1.1.15 - Voice	1	1	1	1	1
Total	**110**	**104 / 110**	**104 / 110**	**110 / 110**	**110 / 110**
		95%	**95%**	**100%**	**100%**

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

Section 2 - IPS Evasion

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

Test 2.2 – Packet Fragmentation/Stream Segmentation	Detected?	Decoded?	Blocked?
Test 2.2.1 - IP fragmentation - ordered 8 byte fragments	YES	YES	YES
Test 2.2.2 - IP fragmentation - ordered 24 byte fragments	YES	YES	YES
Test 2.2.3 - IP fragmentation - out of order 8 byte fragments	YES	YES	YES
Test 2.2.4 - IP fragmentation - ordered 8 byte fragments, duplicate last packet	YES	YES	YES
Test 2.2.5 - IP fragmentation - out of order 8 byte fragments, duplicate last packet	YES	YES	YES
Test 2.2.6 - IP fragmentation - ordered 8 byte fragments, reorder fragments in reverse	YES	YES	YES
Test 2.2.7 - IP fragmentation - ordered 16 byte fragments, fragment overlap (favour new)	YES	YES	YES
Test 2.2.8 - IP fragmentation - ordered 16 byte fragments, fragment overlap (favour old)	YES	YES	YES
Test 2.2.9 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with invalid TCP checksums	YES	YES	YES
Test 2.2.10 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with null TCP control flags	YES	YES	YES
Test 2.2.11 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with requests to resync sequence nos. mid-stream	YES	YES	YES
Test 2.2.12 - TCP segmentation - ordered 1 byte segments, duplicate last packet	YES	YES	YES
Test 2.2.13 - TCP segmentation - ordered 2 byte segments, segment overlap (favour new)	YES	YES	YES
Test 2.2.14 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with out-of-window sequence numbers	YES	YES	YES
Test 2.2.15 - TCP segmentation - out of order 1 byte segments	YES	YES[1]	YES
Test 2.2.16 - TCP segmentation - out of order 1 byte segments, interleaved duplicate segments with faked retransmits	YES	YES[1]	YES
Test 2.2.17 - TCP segmentation - ordered 1 byte segments, segment overlap (favour new)	YES	YES[1]	YES
Test 2.2.18 - TCP segmentation - out of order 1 byte segments, PAWS elimination (interleaved dup segments with older TCP timestamp options)	YES	NO	YES
Test 2.2.19 - IP fragmentation - out of order 8 byte fragments, interleaved duplicate packets scheduled for later delivery	YES	YES	YES
Test 2.2.20 - TCP segmentation - ordered 16 byte segments, segment overlap (favour new (Unix))	YES	YES	YES
Total	**20 / 20**	**19 / 20**	**20 / 20**

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

Test 2.4 – Miscellaneous Obfuscation Techniques	Detected?	Decoded?	Blocked?
Test 2.4.1 - Altering default ports	YES	YES	YES
Test 2.4.2 - Inserting spaces in FTP command lines	YES	YES[2]	YES
Test 2.4.3 - Inserting non-text Telnet opcodes in FTP data stream	YES	YES	YES
Test 2.4.4 - Polymorphic mutation (ADMmutate)	YES	YES	YES
Test 2.4.5 - Altering protocol and RPC PROC numbers	YES	YES	YES
Test 2.4.6 - RPC record fragging (MS-RPC and Sun)	YES	YES	YES
Test 2.4.7 - HTTP exploits to port <> 80	YES	YES	YES
Total	**7 / 7**	**7 / 7**	**7 / 7**

Section 3 - Stateful Operation

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

Test 3.2 – Simultaneous Open Connections (default settings)							
Number of open connections	10,000	25,000	50,000	100,000	250,000	500,000	1,000,000
Test 3.2.1 - Attack Detection	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Test 3.2.2 - Attack Blocking	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Test 3.2.3 - State Preservation	PASS	PASS	PASS	PASS	FAIL[3]	FAIL[3]	FAIL[3]
Test 3.2.4 - Legitimate traffic blocking	PASS	PASS	PASS	PASS	PASS	PASS	PASS

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

Section 4 - Detection/Blocking Performance Under Load

Test 4.1 – UDP traffic to random valid ports		250Mbps	500Mbps	750Mbps	1Gbps	Max
Test 4.1.1 - 256 byte packet test - max 453,000pps	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	
Test 4.1.2 - 550 byte packet test - max 220,000pps	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	
Test 4.1.3 - 1514 byte packet test - max 122,000pps	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	

Test 4.2 – HTTP "maximum stress" traffic with no transaction delays		250Mbps	500Mbps	750Mbps	1Gbps	Max
Test 4.2.1 - Max 2500 connections per second - ave packet size 1000 bytes - max 120,000 packets per second	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	
Test 4.2.2 - Max 5000 connections per second - ave packet size 540 bytes - max 225,000 packets per second	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	
Test 4.2.3 - Max 10000 connections per second - ave packet size 440 bytes - max 275,000 packets per second	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	
Test 4.2.4 - Max 20000 connections per second - ave packet size 360 bytes - max 320,000 packets per second	Detected	100%	100%	100%	N/A[4]	850Mbps
	Blocked	100%	100%	100%	N/A[4]	

Test 4.3 – HTTP "maximum stress" traffic with transaction delays		250Mbps	500Mbps	750Mbps	1Gbps	Max
Test 4.3.1 - Max 5000 connections per second - ave packet size 540 bytes - max 225,000 packets per second - 10 sec delay - max 50,000 open connections	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	
Test 4.3.2 - Max 10000 connections per second - ave packet size 440 bytes - max 275,000 packets per second - 10 sec delay - max 100,000 open connections	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	

Test 4.4 – Protocol mix		250Mbps	500Mbps	750Mbps	1Gbps	Max
Test 4.4.1 - 72% HTTP (540 byte packets) + 20% FTP + 6% UDP (256 byte packets). Max 4000 connections per second - ave packet size 540 bytes - max 215,000 packets per second - max 750 open connections	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	

Test 4.5 – Real World traffic		250Mbps	500Mbps	750Mbps	1Gbps	Max
Test 4.5.1 - Pure HTTP (simulated browsing session on NSS Web site). Max 4700 connections per second - 20 new users per second - ave packet size 560 bytes - max 210,000 packets per second	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	
Test 4.5.2 - Protocol mix - 72% HTTP (simulated browsing sessions as 2.5.1) + 20% FTP + 6% UDP (256 byte packets). Max 3700 connections per second - ave packet size 560 bytes - max 205,000 packets per second - max 1,500 open connections	Detected	100%	100%	100%	100%	1Gbps
	Blocked	100%	100%	100%	100%	

Section 5 - Latency & User Response Times

Test 5.1 – Latency	Packet Size	250Mbps	500Mbps	750Mbps	1Gbps
Test 5.1.1 Average latency (µs) with no background traffic	256	155.55	169.08	194.67	212.02
	550	184.66	195.30	213.61	230.72
	1000	222.75	238.17	247.47	259.77
Test 5.1.2 Average latency (µs) with background traffic (500Mbps HTTP traffic, max 2500 connections per second - ave packet size 540 bytes - max 112,500 packets per second)	256	258.33			
	550	267.16			
	1000	293.64			
Test 5.1.3 Average latency (µs) when under attack (100Mbps SYN flood)	256	169.22			
	550	194.70			
	1000	228.63			

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 (500Mbps HTTP traffic, max 2500 connections per sec - ave packet size 540 bytes - max 112,500 packets per sec)	1556517	0	201	224	206
Test 5.2.2 - Web page response (ms) when under attack (500Mbps HTTP traffic, max 2500 connections per sec - ave packet size 540 bytes - max 112,500 packets per sec PLUS 100Mbps SYN flood)	1556398	0	201	247	207

Section 6 - Stability & Reliability

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

Section 7 - Management Interface

Test ID	Result
Test 7.1.1 - Open Ports	PASS
Test 7.1.2 - ISIC/ESIC/TCPSIC/UDPSIC/ICMPSIC	PASS
Test 7.1.3 - ISIC attacks detected against management interface?	NO

Notes:

1. Requires change to “TCP out of order segs per flow” parameter from 64 to 128 in order to decode accurately. Without this change, evasion attempt is still blocked, but reported as “too many out of order TCP segments”
2. Decoded as “Telnet control sequence in FTP control connection”
3. Maximum 100,000 open connections out of the box
4. Not possible to run this test - maximum HTTP connection rate is 17,000cps

Section 1: Detection Engine

We installed one sensor with the latest signature pack, and created a Protection Policy with every attack signature enabled, plus some key audit/information only signatures.

Signature recognition (with blocking disabled) was excellent out of the box (95 per cent), and was increased to a perfect 100 per cent after the application of a signature pack update which was provided to us within 48 hours. Blocking performance was identical throughout the tests.

We noted a minimum of “noise”, with very few test cases raising multiple alerts for a single exploit, and the accuracy of the exploit descriptions was high. Performance in our “false negative” tests was good out of the box, and there is every indication that the majority of signatures are written for the underlying vulnerability rather than specific exploits. Specific exploit signatures are also included where appropriate, however, to provide more accurate identification for the administrator.

Signature recognition capabilities have improved considerably since engine update 4 (the version tested was engine update 5) at which point the Symantec signature writing language gained numerous new features allowing the creation of much more complex signatures.

It is worth noting that the power of this language is available to administrators when writing custom signatures (though it is a shame that the built-in Symantec signatures are not open to examination).

A major concern in deploying an IPS is the blocking of legitimate traffic. The SNS 7160’s resistance to false positives was perfect in our tests.

The SNS 7160 arrives with a number of default policies configured for different environments and with sensible PASS and BLOCK actions set for appropriate signatures (i.e. where the confidence level of a signature is VERY HIGH then the action will generally be set to BLOCK).

Section 2: IPS Evasion

Resistance to known evasion techniques was excellent, with the SNS 7160 achieving a clean sweep across the board in all our evasion tests. *Fragroute, Whisker, ADMmutate* and even *RPC record fragging* all failed to trick SNS into ignoring valid attacks.

Not only were the fragmented and obfuscated attacks blocked successfully, but all but one of them were decoded accurately as well.

Section 3: Stateful Operation

Out of the box, the SNS 7160 handled 100,000 open connections without tuning. Following a simple tuning operation (a single parameter) the device handled just over 1 million connections (the maximum allowed).

Default operation of the device is to age out old connections when the state tables are full or resources are low. This means that it is technically possible to evade the SNS 7160 once the state tables are full, since it will allow attack traffic from aged-out connections at that point.

Our half-open exploit was not detected when completed, for example, once we exceeded 1 million connections (although the device obviously maintained state on these right up to the 1 million connection mark). This behaviour is not configurable.

Stateless “exploits” are not alerted upon (this is correct behaviour in order to be resistant to *Stick* and *Snot* tools) and mid-flows are not blocked by default. It is not possible to configure the device to block mid-flows.

Section 4: Detection/Blocking Performance Under Load

The SNS 7160 is rated at 1Gbps when deployed in-line, and performance at almost all levels of our load tests was perfect, with 100 per cent of all attacks being detected and blocked under most load conditions.

The one exception was Test 4.2.4 where we determined that there was a limit of 17,000 HTTP connections per second (equating to approximately 850Mbps) meaning the test could not be completed. Beyond this limit, legitimate connections began to fail, but all attack traffic was still blocked successfully.

However, we would happily confirm Symantec’s 1Gbps rating for this device under all normal network conditions.

Section 5: Latency & User Response Times

Basic latency figures were very good for a device of this type at all traffic loads and with all packet sizes, ranging from 155µs with 250Mbps of 256 byte packets, to 259µs with 1Gbps of 1000 byte packets.

Behaviour throughout the tests with no background traffic was very predictable, with minimal increases in latency as traffic levels increased from 250Mbps to 1Gbps across each packet size.

Placing the device under a half load of 500Mbps of HTTP traffic, we noted slightly larger increases of 66 per cent with 256 byte packets (155µs to 258µs), 45 per cent with 550 byte packets (184µs to 267µs), and 32 per cent with 1000 byte packets (222µs to 293µs).

However, all results remained below the “magic figure” of 300 microseconds, our limit for deploying in-line devices in the core of the network. HTTP response times were also very good, meaning the SNS 7160 could be situated anywhere on a Gigabit network, either internally or at the perimeter.

SYN Flood mitigation is a recent addition to this product, and 100Mbps of SYN flood traffic had no significant effect on the SNS 7160. Latency increased by only a few microseconds across all packet sizes (HTTP response times were barely affected), and the SYN Flood was mitigated almost completely.

Section 6: Stability & Reliability

The SNS 7160 performed consistently and completely reliably throughout our tests.

Under eight hours of extended attack (comprising millions of exploits mixed with genuine traffic) it continued to block 100 per cent of attack traffic, whilst passing 100 per cent of legitimate traffic.

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

Section 7: Management Interface

Open ports on the management interface are restricted to TCP/22 (SSH) and TCP/2600 (sensor to console communication as tested, though this can be any TCP high port, configurable or randomised).

The extended ISIC attack against the management interface had no effect on the appliance and its ability to detect and block attacks. No alerts were raised during the attack.

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