BroadWeb NetKeeper NK-3256T V3.6.0

Technical Evaluation

An NSS Group Report

First published January 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, VPN's, Web Application firewalls, multi-function security appliances, cryptographic devices and PKI products.

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

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

Foreword

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

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

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

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

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

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

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

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

Bob Walder

INTRODUCTION

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

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

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

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

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

Intrusion Prevention Systems (IPS)

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

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

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

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

Host IPS (HIPS)

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

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

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

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

Network IPS (NIPS)

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

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

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

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

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

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

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

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

Rate-Based IPS (Attack Mitigator)

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

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

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

Implementation Challenges

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

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

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

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

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

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

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

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

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

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

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

Requirements for effective prevention

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

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

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

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

The NSS Intrusion Prevention Group Test

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

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

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

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

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

Performance

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

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

Detection/Blocking Performance Under Load

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

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

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

Latency & User Response Times

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

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

throughput = window / RTT

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Stability & Reliability

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

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

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

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

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

Security Effectiveness

Detection Accuracy & Breadth

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

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

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

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

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

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

Resistance To Evasion Techniques

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

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

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

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

Stateful Operation

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

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

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

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

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

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

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

Usability

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

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

BROADWEB NETKEEPER NK-3256T V3.6.0

Executive Summary

The BroadWeb NetKeeper series is a family of intrusion detection and prevention appliances designed to detect and prevent attacks across multiple network segments at up to 100Mbps speeds (currently, with more models planned). A range of models are available covering various sizes of installation, and each is licensed based on simultaneous open connections.

The NK-3256T under test here is currently the top of the range, and is designed to support up to 256,000 open connections. A dedicated 1U appliance designed to monitor a single network segment at 100Mbps speeds, the device sports two copper 10/100Mbps ports for in-line operation, and a single 10/100Mbps port for management. Under normal network conditions, we can verify the 100Mbps rating of this device.

We also found the NK-3256T to be very stable and reliable, coping with our extensive reliability tests with ease and without succumbing to most common evasion techniques. SYN flood protection is particularly well implemented for a device of this type.

The GUI has a fairly basic feel to it, and out of the box, policy and alert management are just adequate, whilst reporting is fairly good.

Architecture

The BroadWeb offering supports a three-tier architecture, consisting of the *BEMS Management Server*, the *BEMS Management Client* and the *NetKeeper* appliance.

BEMS Management Server

BEMS (*BroadWeb Extensible Management System*) is a Web-based application that allows multiple users to manage one or more NetKeeper devices concurrently. It can be used from any computer with access to the NetKeeper *BEMS Management Server* via a Web browser.

The BEMS Management Server is installed on a dedicated Windows XP host which is connected to the management network. It logs everything reported from NetKeeper appliances in a MySQL database, and provides the means to configure and manage multiple NetKeeper devices from a single, central console.

Role-based access provides restrictions to individual devices on a per-user basis, allowing separate administrators to be given responsibility for different devices. Each device is treated completely independently, however, bringing up a different management and monitoring window within the BEMS Client. Thus, it is not possible to correlate alerts or create reports that span multiple devices with the current NetKeeper plugin module.

The *BroadWeb Secure Service Team* (BSST) provides a Live Update service allowing installed NetKeeper devices and BEMS Management Servers to connect to a BSST upgrade server automatically for signature and NetKeeper kernel updates.

When a new NetKeeper is added into the BEMS Management Server, it already has the default security policy from the shipped CD-ROM. In addition, during the BEMS Management Server installation process, the scheduled update is configured to download the latest updates every day.

There are no built-in clustering or redundancy capabilities within BEMS Management Server.

BEMS Management Client

The *BEMS Management Client* is a Java-based GUI interface which can be launched from any host on the management network by using a Web browser to connect to BEMS Management Server.

The BEMS Management Client is used to configure and manage NetKeeper appliances, and generate reports.

NetKeeper Appliance

The NetKeeper appliances are available in five models :

- ***NetKeeper NK-3002T -*** *supports up to 2,000 simultaneous connections.*
- ***NetKeeper NK-3008T -*** *supports up to 8,000 simultaneous connections.*
- ***NetKeeper NK-3050T -*** *supports up to 50,000 simultaneous connections.*
- ***NetKeeper NK-3128T -*** *supports up to 128,000 simultaneous connections.*
- ***NetKeeper NK-3256T -*** *supports up to 256,000 simultaneous connections.*

All these devices are based on the same hardware and software platform. The device submitted for testing was the NK-3256T, a dedicated 1U appliance designed to monitor and protect a single network segment at 100Mbps speeds. The NK-3256T runs an optimised, hardened operating system with limited user services to further increase security and performance.

The front panel sports three copper 10/100Mbps ports, one pair for in-line monitoring, and a single port for management. It is also possible to configure the device in SPAN (passive IDS) mode, though only one of the monitoring ports can be used in this mode (we did not verify NetKeeper operation in this mode).

Also on the front panel is a serial console port, providing access to the rich command line capability of the NetKeeper device, and which is used to effect initial configuration before the device can be accessed via the Web-based GUI.

When the device is configured for *In-Line* mode, the administrator can choose to block and alert on designated events. When a protocol anomaly event or an event matching an enabled signature is detected, the offending packet is dropped and the session is cleared from the state table. For TCP traffic, it is also possible to configure the device to Reset Connection, causing a TCP reset to be sent to both sides (client **and** server) of the connection.

This also causes the session to be marked as “bad”, causing subsequent packets for that session to be dropped immediately without further inspection.

It is a one-click operation to switch between *In-Line* mode (blocking) and *Monitor* mode (alert-only) in the management console. In *Monitor* mode, the detection engine 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 management console - it is not necessary to halt network traffic while changing cabling and configuration to switch between in-line alerting and blocking modes.

It is also possible to place the appliance in *Bypass* mode, where traffic is passed directly through the device with no inspection at all (in other words, the device behaves as a simple transparent bridge).

Logging of all alerts is performed directly to the remote BEMS Management Server - there is no local logging facility in the NetKeeper appliance. Thus if communication with the Management Server is interrupted for a significant amount of time alerts will be lost (although the appliance will continue to block malicious traffic as directed by the applied security policy).

There is no built-in redundancy with the current NetKeeper appliances - only a single fixed power supply and fixed fans within each. Despite claims by BroadWeb that NetKeeper features a hardware bypass capability, we could not get this to work in the lab. In our test setup, removing power from the device caused all traffic to be blocked. If fail-open capability is important to you, this feature should be verified in your own network prior to purchase.

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 NetKeeper NK-3256T was tested in in-line mode up to 100Mbps, the rated speed of the device, and performance at all levels of our load tests was impeccable, with 100 per cent of all attacks being detected and blocked under all load conditions. We would happily confirm BroadWeb’s 100Mbps rating for this device.

Basic latency figures were well within acceptable limits for a device of this type at all traffic loads and with all packet sizes, ranging from 80µs with 25Mbps of 256 byte packets, to 218µs with 100Mbps of 1000 byte packets. Behaviour throughout the tests with no background traffic was very predictable.

Placing the device under a half load of 50Mbps of HTTP traffic, we noted acceptable increases in latency ranging from 205µs with 256 byte packets to 323µs with 1000 byte packets. All of these figures are well inside the limits we would expect to see for a 100Mbps device, meaning the NK-3256T could be situated anywhere on a 100Mbps network, either internally or at the perimeter.

The most impressive figures, however, were to be seen in the SYN flood test. 10Mbps (14,800 packets per second) of SYN flood traffic caused minimal increases in latency, and the SYN flood was mitigated completely, thus eliminating any adverse effect on the protected network or target servers. Very impressive performance for a device which does not bill itself as an attack mitigation device.

The NK-3256T performed consistently and completely reliably throughout our tests, continuing to block 100 per cent of all exploits, even when under extended attack. Under eight hours of continuous attack (1 million exploits) the device passed 99.994 per cent of legitimate traffic (blocking an average of 11 out of 200,000 legitimate sessions per run).

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 Policy with every attack signature enabled, disabling just the audit or informational signatures. The SYN flood threshold was adjusted accordingly to differentiate between our “normal” test loads and our attack traffic.

Signature recognition (with blocking disabled) was only just acceptable out of the box (71 per cent), but was increased to a very creditable 97 per cent after the application of a signature pack update which was provided to us in 48 hours. The quality of the new signatures seemed to be as high as the existing ones, and performance of the box was not affected at all by the large update. Blocking performance was identical throughout the tests.

We noted a minimum of “noise”, with very few test cases raising multiple alerts for a single exploit. Performance in our “false negative” tests was reasonable out of the box, though still not perfect following the signature update.

A major concern in deploying an IPS is the blocking of legitimate traffic. The false positive response was perfect following the signature update, despite a couple of problems initially (see *Test Results* section).

The NK-3256T arrives with a default policy with sensible PASS and DROP actions set for appropriate signatures. Having changed this policy, however, there is no way to reset to the recommended settings without completely re-loading the original policy.

Our evasion tests, too, caused problems for NetKeeper initially (see *Test Results* section), but a code update (from 3.5.7 to 3.6.0) rectified this.

Following the update, resistance to known evasion techniques was excellent, with the NK-3256T achieving almost a clean sweep across the board in our evasion tests. *Fragroute, Whisker, ADMmutate* and even *RPC record fragging* all failed to trick NetKeeper into ignoring valid attacks. Not only were the fragmented and obfuscated attacks blocked successfully, but almost all of them were decoded accurately as well with the latest release of code.

Out of the box, the NK-3256T handles 256,000 open connections without tuning, and this is not configurable. Default operation of the device is to reject all new connections when the state tables are full or resources are low.

Stateless “exploits” are not alerted upon (this is correct behaviour in order to be resistant to *Stick* and *Snot* tools), although a BAD_TCP_STATE alert **is** raised to flag the presence of mid-flows, which are also blocked by default. Since enforcement of state depends on the settings of the BAD_TCP_STATE signature, it is possible to alter this default behaviour.

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 NK-3256T is straightforward. Using the serial console and command line interface, it is possible to set the IP address, netmask and default gateway for the management port, and from that point all configuration is usually performed via the Web-based BEMS Client (although it is possible to perform a significant number of configuration tasks from the command line interface (CLI).

The BEMS Management Server software can be installed on any suitable PC with access to the management network. The installation Wizard walks the administrator through a painless process of installing the Java run-time (if necessary), MySQL database server, and the BEMS Management Server software itself all in a single operation. We had issues running BEMS on a Windows 2000 Server - XP is recommended.

Initial configuration of the BEMS system is performed via the *Admin Console*, which is distinct from the *BEMS Management Client* used for day-to-day administration of the NetKeeper appliances.

It appears as though BEMS is designed to manage multiple different types of security devices, since it provides an open framework whose functionality is populated via the installation of plug-in modules. These would normally be installed over the Internet from the BroadWeb update server - we installed the NetKeeper plug-in from CD, which added the ability to manage NetKeeper appliances.

Adding a new NetKeeper appliance is a simple operation using the Admin Console

All that is required is to give it a name, and specify its IP address and the IP address of the parent BEMS Management Server (each device can be associated with a single Management Server only).

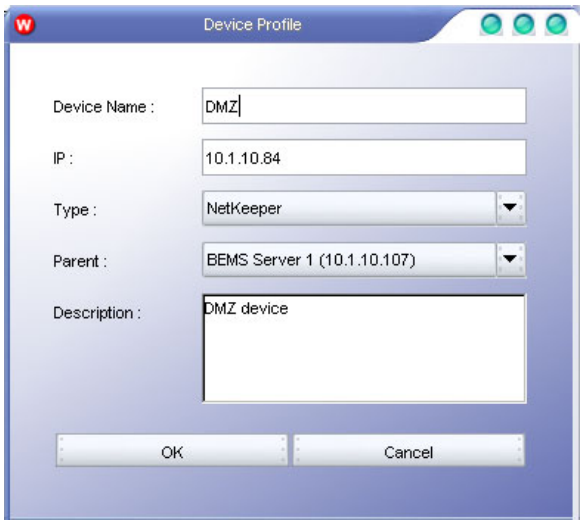

Figure 1 - NetKeeper: Adding devices

Once a device has been added via the Admin Console, it is available for further configuration via the BEMS Management Client. Each NetKeeper is installed initially without a security 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).

Documentation consists of a single user manual provided in PDF format only. It is not particularly detailed, but is adequate for the task, and covers both the CLI and GUI interfaces. However, we feel that more detail would be beneficial - for example, we would like to see more explanation of the function of the various command line parameters and the consequences of altering them.

Configuration

As mentioned previously, administration of the BEMS system is divided across two separate consoles - the *Admin Console* and the *BEMS Management Client*.

The *Admin Console* is used for initial configuration of the system, and the screen is divided into four tabs:

- ***System -*** *Provides real-time view of the BEMS Management Server itself, providing access to CPU utilisation, memory used, logging and communication services (which can be started and stopped from here), and the ability to add and remove plug-in management modules*
- ***User -*** *Manage Users and Groups. This is used to define Users and their privilege levels within the system*
- ***Device -*** *Manage NetKeeper (and other) devices. This is used to define new NetKeeper appliances (and other types of security devices, depending on the plug-in modules loaded) by simply defining the name, IP address, type of device, and BEMS Management Server IP address.*
- ***Log -*** *View system logs, save to file, and filter contents*

User accounts and Groups provide the means to restrict access to portions of the Admin Server and BEMS Management Client, thus allowing the main system administrator to delegate administrative tasks to other users. Privileges are set at a Group level, where it is possible to permit or deny access to the Admin Console, as well as restrict individual functions within the Console (for example, an administrator could be allowed to create new Users, but not modify Groups). When creating a Group, it is also possible to define which NetKeeper devices can be configured, and which can be viewed.

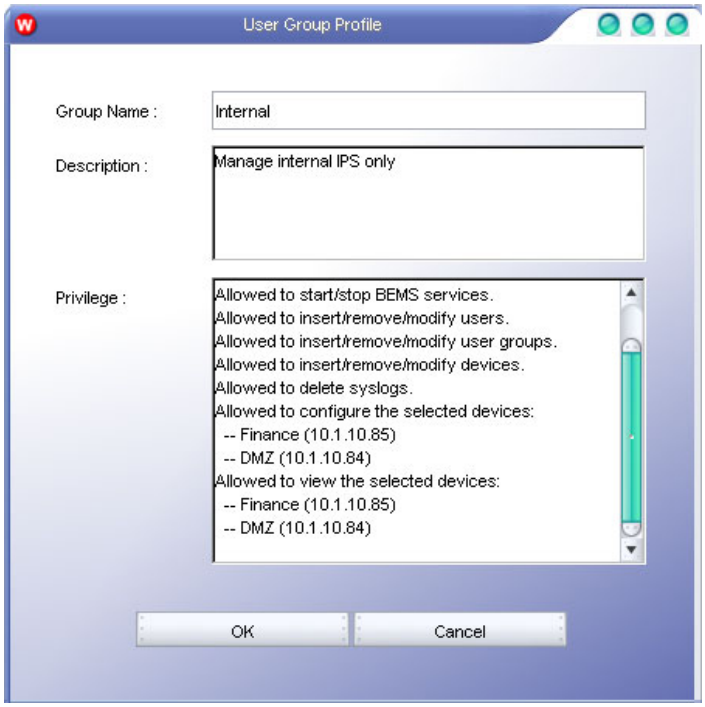

Figure 2 - NetKeeper: Defining user privileges via the use of Groups

When new Users are created, they are allocated to a single Group which thus defines their privileges within the BEMS system. When a user logs in, he is presented with a list of devices which is limited by those he is allowed to view. Within that list, he may be restricted to certain devices on which he is actually allowed to modify the configuration - on others, he may be allowed to view alerts and run reports only.

To initiate the Java-based *BEMS Management Client,* it is necessary to access the home page of the Web server running on the BEMS Management Server. This provides a link to download and run the Client software. The BEMS Management Client contains three main viewing areas:

- ***Device Tree View -*** *All the devices managed by this BEMS Server are displayed in tree format here.*
- ***Device Information -*** *Displays summary information for the device that is selected in the Device Tree View window, including Device Type, Model, Name, IP Address, etc.*
- ***Plug-in Window -*** *This is the main operation window for administrators. When a device is selected in the Device Tree View, the corresponding plug-in program will run in this window, providing the main configuration and management interface.*

The NetKeeper plug-in contains five tabs along the top of the screen:

- ***Real-time Monitor -*** *monitor alerts from NetKeeper devices*
- ***Report Generator -*** *generate reports and queries, and schedule reports for regular runs*
- ***Policy Database -*** *configure security policy*
- ***Device Configuration -*** *configure NetKeeper devices*
- ***System Log -*** *view system log messages*

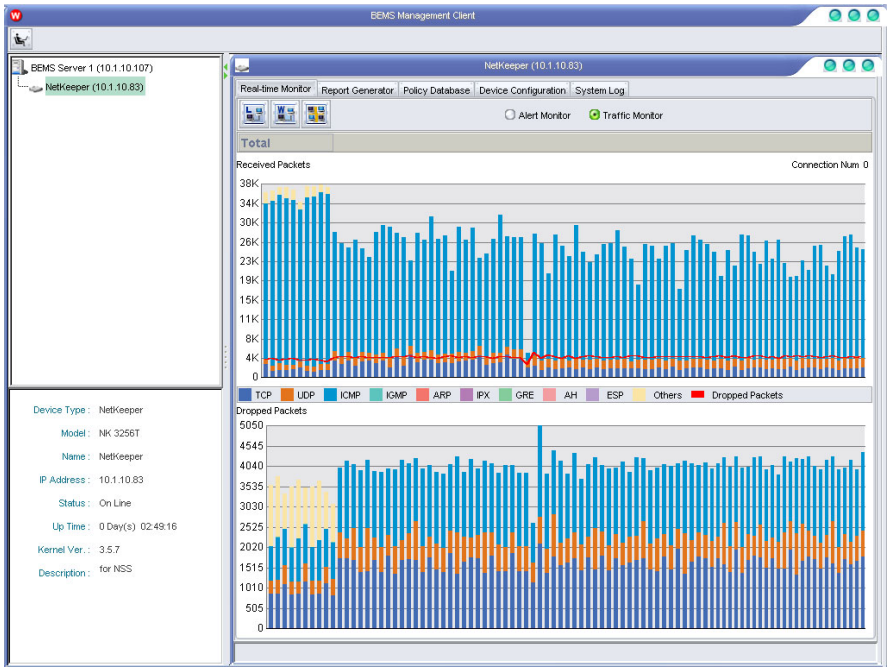

Figure 3 - NetKeeper: Real-time Traffic Monitor

The *Device Configuration* screen allows the administrator to set a number of adjustable parameters, including maximum TCP session idle time, maximum generated log events per second, interval to send log events to policy server (seconds), and operation mode. Operation mode can be one of:

- ***Bypass -*** *In Bypass mode, NetKeeper works like a bridge with all rules and actions disabled. When Bypass mode is enabled, NetKeeper will neither detect nor respond to security events in the network.*
- ***In-line -*** *In this mode, NetKeeper works as a transparent gateway in the network. All traffic passes through NetKeeper and all packet contents are inspected and compared to the signatures applied in the security policy. NetKeeper will respond to illegal activities based on the rules installed in the policy.*
- ***Monitor -*** *In this mode, NetKeeper will log all events, but will not take any countermeasures (reset, drop, etc.). Some administrators may prefer to monitor network traffic in this mode for a period of time before switching to In-line mode, in order to fine tune the security policy and network performance.*
- ***SPAN -*** *In this mode, network traffic is mirrored to NetKeeper via a switch SPAN port. NetKeeper is thus not installed in-line, but works like a passive IDS. Like all IDS systems, NetKeeper can respond to security incidents by resetting attacker's connection and raising corresponding alerts.*

- ***Tap -*** *NetKeeper acts as an in-line device in TAP mode, installed between the network segment to be protected and its uplink. In this mode, NetKeeper works like IDS, resetting connections and raising alerts, but does not actively block malicious traffic. This mode seems somewhat superfluous.*

Policy Database, Real-time Monitor and *Report Generator* tabs are covered in the following sections.

Policy Management

Policy management within BEMS is adequate, but compared with competing products is lacking in some features - for example, deploying a single policy to multiple devices is not as intuitive as it could be.

Each NetKeeper device is effectively stand-alone within BEMS - double clicking on the device in the *Device Tree View* window pulls up a separate instance of the NetKeeper plug-in. The policy for each appliance is stored directly on the appliance rather than in a central database, meaning that each device needs to be configured independently of all the others.

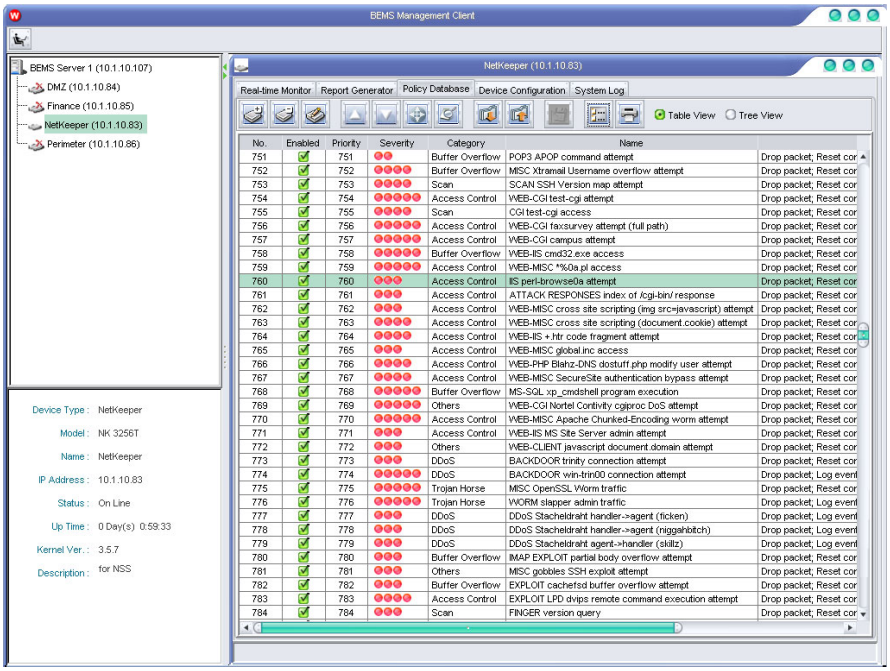

Figure 4 - NetKeeper: Policy Editor (Table View)

Another issue is that with over 2000 rules, an effective way of searching for and selecting individual signatures or groups of signatures is essential. Unfortunately, the basic search method within BEMS is *too* basic to be of real value - after entering a search string, only the first match is returned.

It is then necessary to exit the search dialogue in order to make changes to the signature, which makes it very difficult to search for, say, all IIS-related signatures in order to make bulk changes.

In other respects, however, bulk editing has been made very simple. There are two views of the Policy database - *Table View* and *Tree View*. Table View presents the Policy database in a spreadsheet format, and allows the administrator to select which columns are displayed.

Available columns include enabled/disabled status, severity, category (DDOS, Scan, Buffer Overflow, etc.), name, response actions, origin (built-in or user-defined), scope (source and destination restrictions), and schedule (time periods when the signature is active/inactive). The database can be sorted on any of these columns, making it relatively simple to select groups of signatures using the standard Windows selection keys.

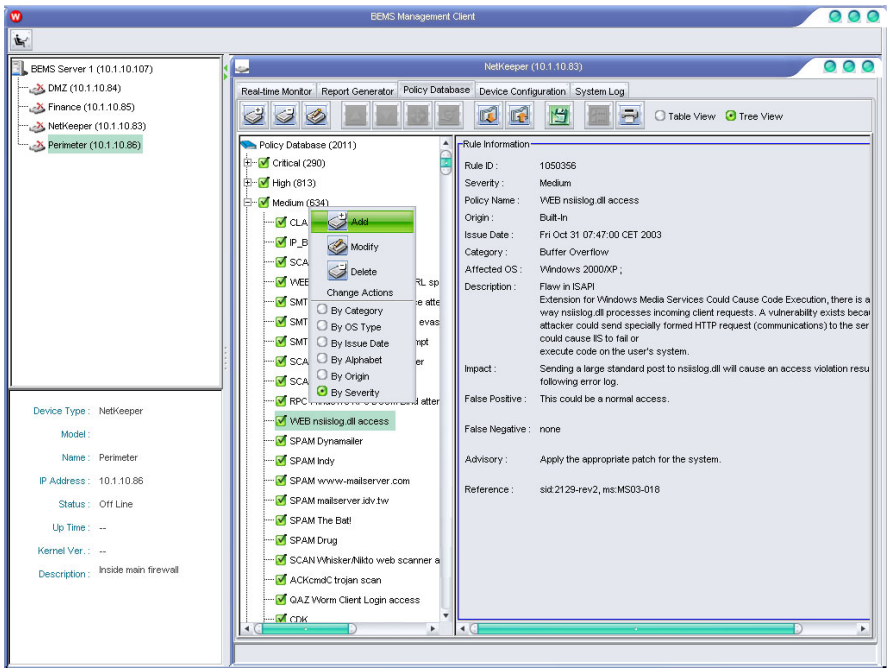

Figure 5 - NetKeeper: Policy Editor (Tree View)

Group selection is made even more straightforward in the Tree View, where the Policy database is represented as a hierarchical tree grouped by *Category, OS Type, Issue Date, Alphabetically, Origin* or *Severity*. Right clicking in the tree allows the grouping to be changed instantly via a drop down menu, and it is possible to select any of the groups once sorted and apply changes to response actions, schedules or scope to the entire group of signatures in a single operation. An entire group of signatures can also be enabled or disabled in the same way, by clicking the check box alongside the group name in the tree.

Naturally, individual signatures can also be selected to be viewed in detail or modified. Changing the Scope parameters allows the administrator to restrict certain signatures to operate only on specific IP addresses or subnets (or to exclude those addresses). Thus it would be possible to remove checking for P2P traffic from addresses on the DMZ, where normal users are not allowed to operate, thus potentially reducing the incidence of false positives. The administrator can define his own set of hosts or groups to be included or excluded in the protection scope.

If a DDoS or scanning signature is selected, there is a DDOS parameter section instead of the Scope parameters. Here it is possible to set the necessary thresholds for such rate-based signatures to operate effectively.

The Schedule parameter allows the administrator to define multiple schedules (corresponding to working days, weekends, and so on) and have certain signatures active or disabled according to those schedules.

This would allow a company to ensure that P2P traffic is denied during working hours, but allowed at other times, for example. A number of Response actions can be configured for each signature:

- ***Drop Packet -*** *The attack packet is dropped and the session details cleared from the state table (subsequent packets for that session will then be handled by the TCP_BAD_STATE signature)*
- ***Reset Connection -*** *TCP resets are sent to the client and server, and the session is entered in a "session blacklist", ensuring that subsequent packets from that same session are dropped immediately without further inspection*
- ***Alert by E-mail -*** *NetKeeper will send an e-mail containing the details of this attack to a list of specified e-mail addresses*
- ***Log without packet -*** *Log the event only without packet content*
- ***Log with whole packet -*** *Log the event with the entire packet which triggered it*
- ***Log with packet header -*** *Log the event with the attack packet header (the first 64 bytes)*
- ***Log with session data -*** *Log the event along with all packets in the session including, and following, the attack packet*

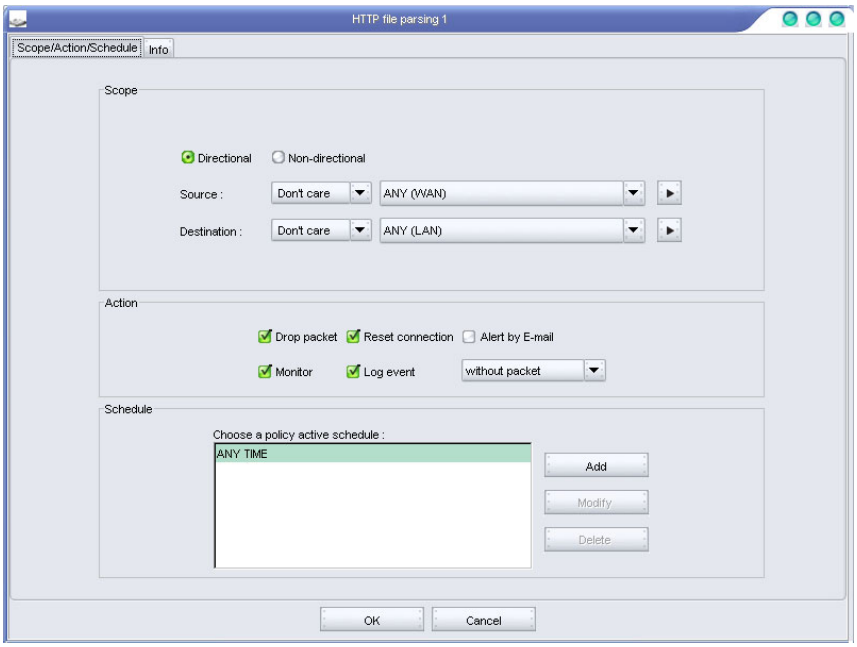

Figure 6 - NetKeeper: Configuring attack responses

BroadWeb applies a priority to each signature (which governs its position in the policy database, much like the rule set on a firewall) so that where a single packet matches different signatures concurrently, the signature with the highest priority is checked first, and the action is determined according to the first rule matched.

Each rule also has a description about the exploit, affected OS, impact, suggested defence actions, and so on. However, this information is often not as detailed as that provided by competing products, and there is no hyperlink to external references via the CVE reference, Bugtraq ID, etc.

It is not possible to modify, copy, or even view the actual contents of the rules in the Policy database, but it is possible to create user-defined rules.

There is no Wizard interface to guide you through this process, and given the wealth of information that can be entered for each signature this is not really an operation suitable for the beginner. For example, in addition to defining basic information such as signature name, category, protocol, and severity, it may also necessary to define various header details such as flags, TTL, header size, sequence numbers, and checksum, as well as the content to match (offset/depth and string to match).

Once all necessary changes have been made to a Policy for a particular NetKeeper device, it can be applied by clicking a button on the toolbar - it would be nice for this to flash red (or for some other highly visible icon to appear) whenever there are pending changes to be applied, since it is easy to forget to apply changes. There is no concept of “recommended settings” in NetKeeper, so it is not possible to revert to BroadWeb’s suggested settings for actions (log, drop, reset, etc.) without re-loading a previously saved Policy completely (which may, of course, destroy other changes you have made).

As mentioned before, if it is required to apply this Policy to several NetKeeper appliances, it is necessary to first save it to disk, then call up each appliance in turn and apply the Policy to each individually.

There is no versioning capability built into NetKeeper’s Policy management, and so it is not possible to track changes made to Policies, and nor is it possible to revert to an earlier version of a Policy should problems be discovered once it has been applied.

Alert Handling

As alerts are raised by NetKeeper devices they are transmitted immediately to the BEMS Management Server where they are written to the central database. At the same time, if the Real-time Monitor tab is active for a particular NetKeeper device, the alerts are displayed on the management console.

This takes the form of a tabular display with one line per alert and the following columns of data:

- *Severity*
- *Date and time of occurrence*
- *Name of attack*
- *Source IP address*
- *Destination IP address*
- *Count - number of times this event was seen within the aggregation interval*
- *Actions - log, drop, e-mail, etc.*

Unfortunately it is not possible to drill down into these events to see detailed packet information. We would also like to see source and destination port added to the above list.

Double-clicking on any of the events listed takes you straight into the tree view of the Policy editor in order to display the detailed exploit/signature information stored within the Policy database.

It is possible to set filters in order to restrict the display to one or more particular Severity settings or Categories.

Unfortunately this does not affect the data already displayed, but only applies to data captured in the Real-time Monitor once the real-time display has been re-activated. Events can also be sorted by any of the columns in the real-time display.

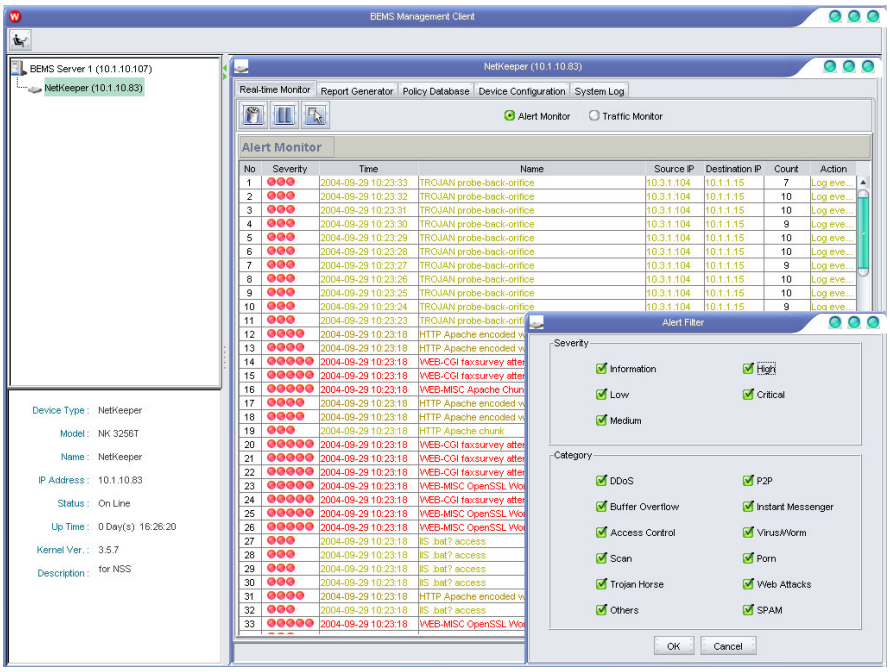

Figure 7 - NetKeeper: Real-time Alert Monitor

In addition to the *Alert Monitor* screen, the real-time display also has a *Traffic Monitor* screen (see *Figure 3*). This is divided into two panes, the upper pane showing the number of network packets received as a constantly-updated graph, and the number of current open connections. The lower pane displays the number of packets that are identified as attacks and discarded by NetKeeper, also as a constantly-updated graph. Different colours are used to represent different protocols.

Reporting and Analysis

Reporting and analysis functions are accessed via a single Report Generator tab which is divided into three main areas:

- ***Query on Demand -*** *Graphical and text displays of attacked hosts, attack types, and severity are available via selection criteria which include source, destination, name and severity.*
- ***Event List -*** *Search by destination IP, type, or source IP, and matching the searching results with a specific time frame; all incidents of attack within that time frame would be listed out.*
- ***Statistics -*** *A statistical analysis of all network attacks is available in the form of a daily report, weekly report and monthly report. A statistical report aiming at a certain type of network attack or a certain host is also available.*

All of the reports can be filtered based on time via a set of drop down menus at the bottom of the Report Generator screen.

The finest resolution is an hour, however - we would like to see this extended to selection down to the nearest minute (important when testing as well as when there is a flooding attack).

Having selected *Query on Demand* it is possible to choose exactly how the report is to be sorted and grouped from a number of drop-down menus, as well as apply specific selection criteria (such as searching for all alerts from a particular source IP). Having selected the required time period, the report is generated - even with millions of records in the database we found report generation to be relatively quick.

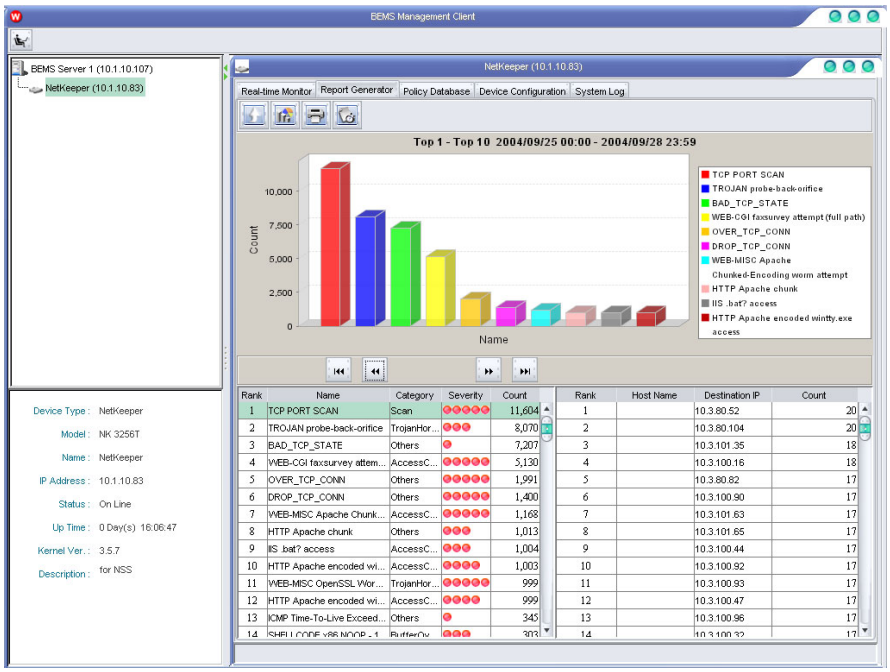

Figure 8 - NetKeeper: Query on Demand

The result is a dynamic on-screen report consisting of three panes. The upper pane contains the main graphical summary, and allows the administrator to switch between bar graphs and pie charts, as well as step through the top 1-10 events, 11-20, 21-30, and so on.

The lower-left pane contains a summary tabular form of the graphical report (for example, the number of exploits seen from a particular source address), and double clicking on any individual entry provides a drill down detailed list of the individual entries in the lower-right pane (the individual exploits seen from that source address).

It is possible to continue drilling down into the report until a list of individual events is produced, at which point - providing the attack response in the Policy included packet capture options - it is possible to drill down further into each event to access the attack packet (plus the rest of the session, depending on the capture option specified).

This is a very powerful and flexible dynamic query tool, and should prove very useful for forensic analysis. Other report options include a simple *Event List*, which lists individual events matching the specified criteria (with the same drill-down option to the attack packets if the capture response was selected in the Policy), and a *Statistical Analysis* report which provides summary totals on a daily, weekly and monthly basis.

Having created the perfect report, it would be nice to be able to save it as a template for recall in the future - but you can't. Report templates of a slightly different kind are supported elsewhere via the *Report Configuration* screen. This is designed to generate a management-oriented summary report which can be saved in HTML or CSV format, automatically e-mailed to a distribution list, and copied to a secure FTP server.

Saving in HTML format creates a number of individual files all linked by a single index page, so it would be possible to publish this directly to a Web server for general access if required.

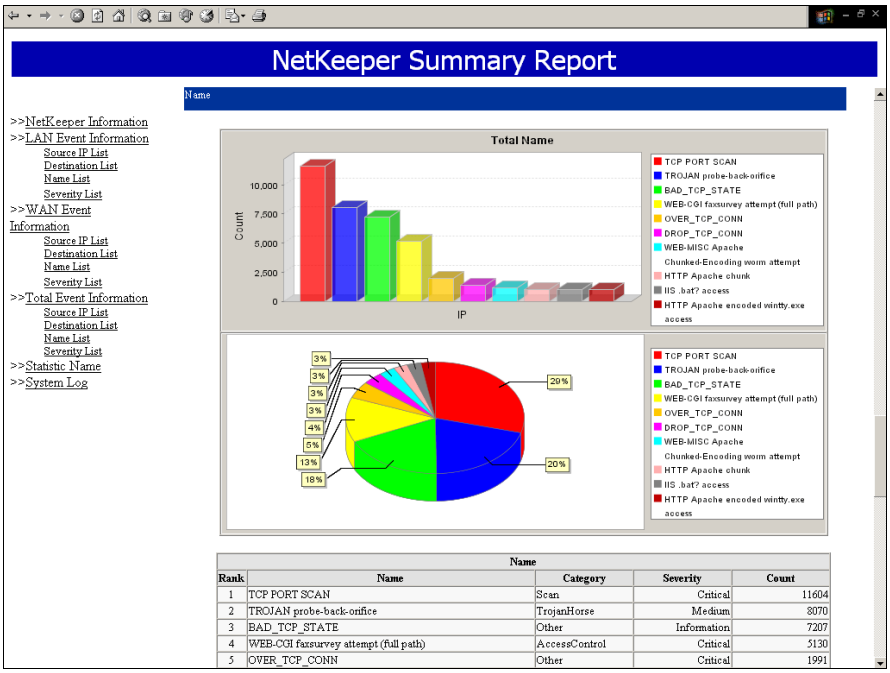

Figure 9 - NetKeeper: HTML Summary Report

The report can be scheduled to run at regular intervals, though scheduling is fairly basic, limited to every *x hours* or every *x days* - but that should be enough for most people.

The finished report is attractive and extensive, containing a number of different graphs and tables generated from the built-in templates. It can easily be added to by creating additional custom templates which generate additional text-based tables which are tagged on to the end of the report.

Whilst of limited use in terms of forensic analysis, this report would be useful as a management tool, perhaps, especially since it can be generated automatically at regular intervals and e-mailed or posted to a secure Web server for immediate access by users who would not otherwise be permitted access to the BEMS Management Client to run reports.

Verdict

Performance

The NetKeeper NK-3256T was tested up to 100Mbps, the rated speed of the device.

Performance at all levels of our load tests was impeccable, with 100 per cent of all attacks being detected and blocked under all load conditions. We would happily confirm BroadWeb's 100Mbps rating for this device.

Basic latency figures were well within acceptable limits for a device of this type at all traffic loads and with all packet sizes, ranging from 80µs with 25Mbps of 256 byte packets, to 218µs with 100Mbps of 1000 byte packets. Behaviour throughout the tests with no background traffic was very predictable. Placing the device under a half load of 50Mbps of HTTP traffic, we noted acceptable increases in latency, and all of the figures reported in our tests are well inside the limits we would expect to see for a 100Mbps device. This means the NK-3256T could be situated anywhere on a 100Mbps network, either internally or at the perimeter.

The most impressive performance, however, was to be seen in the SYN flood test. 10Mbps of SYN flood traffic caused minimal increases in latency, and the SYN flood was mitigated completely, thus eliminating any adverse effect on the protected network or target servers. Very impressive performance for a device which does not bill itself as an attack mitigation device.

The NK-3256T performed consistently and completely reliably throughout our tests, continuing to block 100 per cent of all exploits, even when under extended attack. Under eight hours of continuous attack (1 million exploits) the device passed 99.994 per cent of legitimate traffic (blocking an average of 11 out of 200,000 legitimate sessions per run). We would prefer to see 100 per cent of legitimate traffic passed, but this is a very small percentage of failures under an extreme load of continuous attack 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 only just acceptable out of the box (71 per cent), but was increased to a very creditable 97 per cent after the application of a signature pack update which was provided to us in 48 hours. The quality of the new signatures seemed to be as high as the existing ones, and performance of the box was not affected at all by the large update. Blocking performance was identical throughout the tests.

Performance in our "false negative" tests was reasonable out of the box, though still not perfect following the signature update. Following a signature update, resistance to false positives was also acceptable.

Our evasion tests, too, caused problems for NetKeeper initially (see *Test Results* section), but a code update (from 3.5.7 to 3.6.0) rectified this, following which resistance to known evasion techniques was excellent, with the NK-3256T achieving almost a clean sweep across the board in our evasion tests.

Not only were the fragmented and obfuscated attacks blocked successfully, but almost all of them were decoded accurately as well with the latest release of code.

Although NetKeeper demonstrated some initial problems in the area of false positives and resistance to evasion techniques, the engineers worked very hard to rectify these during the tests. In order to take advantage of these improvements, current users should update to version 3.6.0.

Usability

The Java-based GUI is divided into separate utilities for “super users” and day-to-day administrators, and the role-based access control is effective at ensuring that access to individual NetKeeper devices is restricted on a per-user basis, with each user being allocated different privileges as required.

The day-to-day administrative functions are not as well catered for, however. The policy management and alert handling capabilities are rather too basic, and there is no correlation capability. The management GUI does its job, allowing NetKeeper to function effectively as an IPS device, but policy management and forensic analysis are made more difficult than they should be due to BEMS insisting on managing each NetKeeper device separately. With the market moving at a rapid pace over the last 12 months we do expect to see more in the way of features and functionality - definitely room for improvement.

Reporting and analysis is handled much better, although there are very few built in reports. Instead, there is a very flexible query facility which allows a number of different graphical and text-based reports to be produced. Those reports can be scheduled to run at regular intervals, saved as HTML files, and copied to a secure server or e-mailed to administrators. From any report, it is possible to drill down the summary screens to access more and more detailed event information, eventually accessing the relevant packet contents if they were saved along with the original alert. This is a very powerful facility, and goes some way towards making up for shortcomings in the alert handling. However, in treating every NetKeeper device separately instead of allowing cross-device reporting it does limit its usefulness in larger deployments.

At present, the BEMS systems is lagging behind the competition in terms of features and functionality, but there is a solid base here - in terms of the appliance, the detection software and the multi-tier management system - on which BroadWeb can build.

Contact Details

Company name: BroadWeb Corp.

Internet: http://www.broadweb.com

Address :
222 S. Harbor Blvd. #680 Anaheim,
CA 92805

Tel: +1 714 422 5122

Fax: +1 714 281 1677

E-mail: partner@broadweb.com

Asia-Pacific Address:
3F, No. 24-1, Industry East Rd. IV,
Science Based Industrial Park,
Hsinchu, Taiwan 300, R.O.C.

Tel: +886 3 578 7068

Fax: +886 3 578 7059

E-mail: sales@broadweb.com

APPENDIX A – TEST RESULTS

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

The Test Environment

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

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

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

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

Section 1 – Detection Engine

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

Test 1.1 - Attack Recognition

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

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

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

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

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

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

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

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

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

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

Test 1.2 - Resistance To False Positives

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

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

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

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

Section 2 – Evasion

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

Test 2.1 - Baselines

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

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

Test 2.2 - Packet Fragmentation and Stream Segmentation

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

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

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

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

Test 2.3 - URL Obfuscation

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

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

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

Test 2.4 - Miscellaneous Evasion Techniques

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

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

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

Section 3 – Stateful Operation

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

Test 3.1 - Stateless Attack Replay (Mid-Flows)

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

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

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

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

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

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

Test 3.2 - Simultaneous Open Connections (default settings)

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

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

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

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

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

Test 3.3 - Simultaneous Open Connections (after tuning)

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

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

Section 4 – Detection/Blocking Performance Under Load

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

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

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

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

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

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

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

Test 4.1 - UDP Traffic To Random Valid Ports

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Test 4.4 - Protocol Mix Traffic

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

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

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

Test 4.5 - “Real World” Traffic

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

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

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

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

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

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

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

Section 5 – Latency & User Response Times

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

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

Test 5.1 - Latency

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

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

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

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

Test 5.2 - User Response Times

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

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

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

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

Section 6 – Stability & Reliability

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

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

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

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

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

Section 7 – Management and Configuration

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

Test 7.1 - Management Port

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

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

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

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

BroadWeb NetKeeper NK-3256T V3.6.0 Test Results

Section 1 - Detection Engine

Test 1.1 – Attack Recognition	Attacks	Default ARRD	Default ARRB	Custom ARRD	Custom ARRB
Test 1.1.1 - Backdoors	7	5	5	7	7
Test 1.1.2 - WINS/DNS	3	2	2	3	3
Test 1.1.3 - DOS	10	7	7	10	10
Test 1.1.4 - False negatives (modified exploits)	14	9	9	13	13
Test 1.1.5 - Finger	4	1	1	4	4
Test 1.1.6 - FTP	5	3	3	4	4
Test 1.1.7 - HTTP	43	34	34	42	42
Test 1.1.8 - ICMP	2	2	2	2	2
Test 1.1.9 - Reconnaissance	8	6	6	8	8
Test 1.1.10 - RPC	9	6	6	9	9
Test 1.1.11 - SSH	1	1	1	1	1
Test 1.1.12 - Telnet	1	0	0	1	1
Test 1.1.13 - Database	1	1	1	1	1
Test 1.1.14 - Mail	1	1	1	1	1
Test 1.1.15 - Voice	1	0	0	1	1
Total	**110**	**78 / 110**	**78 / 110**	**107 / 110**	**107 / 110**
		71%	**71%**	**97%**	**97%**

Test 1.2 – Resistance to False Positives	Pass/Fail
Test 1.2.1 - Suspicious FTP traffic	PASS
Test 1.2.2 - HTTP “exploit” using incorrect method	PASS
Test 1.2.3 - Retrieval of Web page containing “suspicious” URLs	PASS
Test 1.2.4 - Simple SMTP QUIT command	PASS
Test 1.2.5 - Normal NetBIOS copy of “suspicious” files	PASS
Test 1.2.6 - Normal NetBIOS traffic	PASS
Test 1.2.7 - POP3 e-mail containing “suspicious” URLs	PASS
Test 1.2.8 - POP3 e-mail with “suspicious” DLL attachment	PASS
Test 1.2.9 - POP3 e-mail with “suspicious” Web page attachment	PASS
Test 1.2.10 - SMTP e-mail transfer containing “suspicious” URLs	PASS
Test 1.2.11 - SMTP e-mail transfer with “suspicious” DLL attachment	PASS
Test 1.2.12 - SMTP e-mail transfer with “suspicious” Web page attachment	PASS
Test 1.2.13 - SNMP V3 packet with invalid parameter	PASS
Test 1.2.14 - Fake DNS /bin/sh buffer overflow	PASS
Test 1.2.15 - Inter-firewall communication traffic	PASS
Test 1.2.16 - Fake SQL Slammer traffic	PASS
Test 1.2.17 - File copy of GIF file (contains bytes which look like NOP sled)	PASS
Total Passed	**17 / 17**[1]

Section 2 - IPS Evasion

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

Test 2.2 – Packet Fragmentation/Stream Segmentation	Detected?	Decoded?	Blocked?
Test 2.2.1 - IP fragmentation - ordered 8 byte fragments	YES	YES	YES
Test 2.2.2 - IP fragmentation - ordered 24 byte fragments	YES	YES	YES
Test 2.2.3 - IP fragmentation - out of order 8 byte fragments	YES	YES	YES
Test 2.2.4 - IP fragmentation - ordered 8 byte fragments, duplicate last packet	YES	YES	YES
Test 2.2.5 - IP fragmentation - out of order 8 byte fragments, duplicate last packet	YES	YES	YES
Test 2.2.6 - IP fragmentation - ordered 8 byte fragments, reorder fragments in reverse	YES	YES	YES
Test 2.2.7 - IP fragmentation - ordered 16 byte fragments, fragment overlap (favour new)	YES	YES	YES
Test 2.2.8 - IP fragmentation - ordered 16 byte fragments, fragment overlap (favour old)	YES	YES	YES
Test 2.2.9 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with invalid TCP checksums	YES	YES	YES
Test 2.2.10 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with null TCP control flags	YES	YES	YES
Test 2.2.11 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with requests to resync sequence nos. mid-stream	YES	YES	YES
Test 2.2.12 - TCP segmentation - ordered 1 byte segments, duplicate last packet	YES	YES	YES
Test 2.2.13 - TCP segmentation - ordered 2 byte segments, segment overlap (favour new)	YES	YES	YES
Test 2.2.14 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with out-of-window sequence numbers	YES	YES	YES
Test 2.2.15 - TCP segmentation - out of order 1 byte segments	YES	YES	YES
Test 2.2.16 - TCP segmentation - out of order 1 byte segments, interleaved duplicate segments with faked retransmits	YES	YES	YES
Test 2.2.17 - TCP segmentation - ordered 1 byte segments, segment overlap (favour new)	YES	YES	YES
Test 2.2.18 - TCP segmentation - out of order 1 byte segments, PAWS elimination (interleaved dup segments with older TCP timestamp options)	YES[2]	NO[2]	YES[2]
Test 2.2.19 - IP fragmentation - out of order 8 byte fragments, interleaved duplicate packets scheduled for later delivery	YES[2]	YES[2]	YES[2]
Test 2.2.20 - TCP segmentation - ordered 16 byte segments, segment overlap (favour new (Unix))	YES	YES	YES
Test 2.2.21 - TCP segmentation - ordered 16 byte segments, segment overlap (favour old (Win32))	YES	YES	YES
Total	**21 / 21[2]**	**20 / 21[2]**	**21 / 21[2]**

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	NO[3]	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**	**8 / 9**	**9 / 9**

Test 2.4 – Miscellaneous Obfuscation Techniques	Detected?	Decoded?	Blocked?
Test 2.4.1 - Altering default ports	NO	NO	NO
Test 2.4.2 - Inserting spaces in FTP command lines	YES	YES	YES
Test 2.4.3 - Inserting non-text Telnet opcodes in FTP data stream	NO	NO	NO
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	**5 / 7**	**5 / 7**	**5 / 7**

Section 3 - Stateful Operation

Test 3.1 – Stateless Attack Replay	Alert?	Blocked?	Pass/Fail
Test 3.1.1 - Stateless Web exploits	NO[4]	YES[4]	PASS
Test 3.1.2 - Stateless FTP exploits	NO[4]	YES[4]	PASS

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

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	FAIL	FAIL

Section 4 - Detection/Blocking Performance Under Load

Test 4.1 – UDP traffic to random valid ports		25Mbps	50Mbps	75Mbps	100Mbps	Max
Test 4.1.1 - 256 byte packet test - max 45,300pps (100Mbps)	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	
Test 4.1.2 - 550 byte packet test - max 22,000pps (100Mbps)	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	
Test 4.1.3 - 1514 byte packet test - max 12,200pps (100Mbps)	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	

Test 4.2 – HTTP "maximum stress" traffic with no transaction delays		25Mbps	50Mbps	75Mbps	100Mbps	Max
Test 4.2.1 - Max 250 connections per second - ave packet size 1000 bytes - max 12,000 packets per second	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	
Test 4.2.2 - Max 500 connections per second - ave packet size 540 bytes - max 22,500 packets per second	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	
Test 4.2.3 - Max 1000 connections per second - ave packet size 440 bytes - max 27,500 packets per second	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	
Test 4.2.4 - Max 2000 connections per second - ave packet size 360 bytes - max 32,000 packets per second	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	

Test 4.3 – HTTP "maximum stress" traffic with transaction delays		25Mbps	50Mbps	75Mbps	100Mbps	Max
Test 4.3.1 - Max 500 connections per second - ave packet size 540 bytes - max 22,500 packets per second - 10 sec delay - max 5,000 open connections	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	
Test 4.3.2 - Max 1000 connections per second - ave packet size 440 bytes - max 27,500 packets per second - 10 sec delay - max 10,000 open connections	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	

Test 4.4 – Protocol mix		25Mbps	50Mbps	75Mbps	100Mbps	Max
Test 4.4.1 - 72% HTTP (540 byte packets) + 20% FTP + 6% UDP (256 byte packets). Max 400 connections per second - ave packet size 540 bytes - max 21,500 packets per second - max 75 open connections	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	

Test 4.5 – Real World traffic		25Mbps	50Mbps	75Mbps	100Mbps	Max
Test 4.5.1 - Pure HTTP (simulated browsing session on NSS Web site). Max 470 connections per second - 2 new users per second - ave packet size 560 bytes - max 21,000 packets per second	Detected	100%	100%	100%	100%	100Mbps
	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 370 connections per second - ave packet size 560 bytes - max 20,500 packets per second - max 150 open connections	Detected	100%	100%	100%	100%	100Mbps
	Blocked	100%	100%	100%	100%	

Section 5 - Latency & User Response Times

Test 5.1 – Latency	Packet Size	25Mbps	50Mbps	75Mbps	100Mbps
Test 5.1.1 Average latency (µs) with no background traffic	256	79.94	81.61	85.30	108.74
	550	134.27	134.47	134.62	136.07
	1000	217.63	217.92	218.02	218.25
Test 5.1.2 Average latency (µs) with background traffic (50Mbps HTTP traffic, max 250 connections per second - ave packet size 540 bytes - max 11,250 packets per second)	256	205.30			
	550	248.99			
	1000	323.26			
Test 5.1.3 Average latency (µs) when under attack (10Mbps SYN flood - 14,800 packets per second)	256	91.92			
	550	145.90			
	1000	232.50			

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 (50Mbps HTTP traffic, max 250 connections per sec - ave packet size 540 bytes - max 11,250 packets per sec)	161397	0	201	2406	206
Test 5.2.2 - Web page response (ms) when under attack (50Mbps HTTP traffic, max 250 connections per sec - ave packet size 540 bytes - max 11,250 packets per sec PLUS 10Mbps SYN flood - 14,800 packets per second	161398	0	201	8425	215

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	99.986%
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. Following signature update - two false positives detected out of the box.
2. Following code update - see below
3. Decoded as “Whisker Masquerade URL”
4. Alert/block depends on setting of BAD_TCP_STATE signature - it is this signature which enforces TCP state, and thus it can be set to drop packets or not, and to alert or not as required

Section 1: Detection Engine

We installed one sensor with the latest signature pack, and created a Policy with every attack signature enabled. We then disabled just the *Severity Level 1* signatures, which are classed as audit or informational signatures. The SYN flood threshold was adjusted accordingly to differentiate between our “normal” test loads and our attack traffic.

Signature recognition (with blocking disabled) was only just acceptable out of the box (71 per cent), but was increased to a very creditable 97 per cent after the application of a signature pack update which was provided to us in 48 hours. The quality of the new signatures seemed to be as high as the existing ones, and performance of the box was not affected at all by the large update. Blocking performance was identical throughout the tests.

We noted a minimum of “noise”, with very few test cases raising multiple alerts for a single exploit. Performance in our “false negative” tests was reasonable out of the box, though still not perfect following the signature update.

A major concern in deploying an IPS is the blocking of legitimate traffic. The NK-3256T’s initial policy out of the box triggered two of our false positive cases, but these were eliminated following the signature update. The NetKeeper family does include protocol decoders (it has many protocol anomaly alerts built in) but also relies heavily on pattern matching, and thus could be prone to false positives based on our observations of the product prior to the signature update.

The NK-3256T arrives with a default policy with sensible PASS and DROP actions set for appropriate signatures. Having changed this policy, however, there is no way to reset to the recommended settings without completely re-loading the original policy.

Section 2: IPS Evasion

Out of the box we noted significant problems with two of our TCP segmentation (fragroute) evasion techniques, and one of the URL obfuscation (Whisker) techniques used in our tests, allowing us to evade the device (and even DOS the device in one case). These issues were fixed to our satisfaction via a code update (from 3.5.7 to 3.6.0), but if you are currently using version 3.5.7 of this product (or earlier) it would be advisable to update as soon as possible.

Following the code update, resistance to known evasion techniques was excellent, with the NK-3256T achieving almost a clean sweep across the board in our evasion tests. *Fragroute, Whisker, ADMmutate* and even *RPC record fragging* all failed to trick NetKeeper into ignoring valid attacks.

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

Section 3: Stateful Operation

Out of the box, the NK-3256T handles 256,000 open connections without tuning, and this is not configurable. The device handled the stated 256,000 connections with ease in our tests.

Default operation of the device is to reject all new connections when the state tables are full or resources are low (this is how the licensing model is enforced) - this means that it is impossible to evade the NetKeeper once the state tables are full, since it will maintain state for the oldest connections throughout the test. However, this also means that once the 256,000 connection limit is reached, legitimate traffic will be blocked as new connections are rejected. 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), although a BAD_TCP_STATE alert **is** raised to flag the presence of mid-flows, which are also blocked by default. Since enforcement of state depends on the settings of the BAD_TCP_STATE signature, it is possible to alter this default behaviour.

Section 4: Detection/Blocking Performance Under Load

Note that the NetKeeper NK-3256T was tested as a 100Mbps IPS device.

Performance at all levels of our load tests was impeccable, with 100 per cent of all attacks being detected and blocked under all load conditions.

We would happily confirm BroadWeb’s 100Mbps rating for this device.

Section 5: Latency & User Response Times

Basic latency figures were well within acceptable limits for a device of this type at all traffic loads and with all packet sizes, ranging from 80µs with 25Mbps of 256 byte packets, to 218µs with 100Mbps of 1000 byte packets.

Behaviour throughout the tests with no background traffic was very predictable. Latency on 256 byte packets increased by 36 per cent as the load was increased from 25Mbps to 100Mbps, and by les than 1 per cent over the same range of test loads using 550 byte and 1000 byte packets.

Placing the device under a half load of 50Mbps of HTTP traffic, we noted latency increases of 156 per cent with 256 byte packets (80µs to 205µs), 86 per cent with 550 byte packets (134µs to 249µs), and 48 per cent with 1000 byte packets (218µs to 323µs). All of these figures are well inside the limits we would expect to see for a 100Mbps device, meaning the NK-3256T could be situated anywhere on a 100Mbps network, either internally or at the perimeter.

The most impressive performance, however, was to be seen in the SYN flood test. 10Mbps (14,800 packets per second) of SYN flood traffic caused minimal increases in latency (ranging from 6 per cent to 15 per cent depending on packet size), and the SYN flood was mitigated completely, thus eliminating any adverse effect on the protected network or target servers.

Very impressive performance for a device which does not bill itself as an attack mitigation device.

Section 6: Stability & Reliability

The NK-3256T 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 99.994 per cent of legitimate traffic (blocking an average of 11 out of 200,000 legitimate sessions per run).

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

TCP ports 8865 and 8867 are used for communication between the management server and the NetKeeper sensors. No open ports are visible to port scanners on the management interface, however.

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

Console response **was** affected during the attack, however, meaning we were unable to apply new policies, etc. Response returned to normal once the attack finished, and there were no residual stability problems.