Sourcefire IS3000 V4.0.2

Technical Evaluation

An NSS Group Report

First published March 2005 (Version 1.0)

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

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

TABLE OF CONTENTS

TABLE OF FIGURES

The NSS Group

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

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

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

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

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

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

Foreword

The NSS Group is pleased to present the results of its third Gigabit IDS Group Test which includes just two brand new products - a further three products failed our stringent testing requirements and thus do not appear in this report.

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

Overall, the suite contains over **700 individual tests**, many of which are run multiple times, to provide the most thorough and complete evaluation of Network IDS products available anywhere today.

We believe that our test methodology will become the *de facto* standard for testing intrusion detection 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 Detection 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 **Gigabit IDS Group Test (Edition 3)** report can be viewed on-line at *www.nss.co.uk/gigabitids*.

Bob Walder

INTRODUCTION

Whenever a company connects its network to the Internet, it opens up a whole can of worms regarding security. As the network grows, it will play host to numerous bugs and security loop holes of which you have never heard - but you can bet intruders have.

Many organisations are recognising the value of a good security policy to define what is and is not allowed in terms of network and Internet access Then they deploy a number of tools to enforce that security policy – usually in the form of a firewall or two.

Firewalls may be billed as commodity items, but the “shrink wrap” element certainly doesn’t extend to their configuration. A detailed knowledge of what a hacker can do and what should and shouldn’t be allowed through the firewall is required before embarking on the configuration adventure, and a slip of the mouse is all it takes to open up a hole big enough for your average hacker to drive the proverbial bus through. The problem is, a badly configured firewall can be worse than no firewall at all, since it will engender a false sense of security.

To protect an organisation completely, therefore, it is necessary to provide a second line of defence, and in order to achieve this, an entire category of software exists in the form of ***Intrusion Detection Systems (IDS).***

When it comes to computer and network security, there are a number of analogies that can be drawn with the “real world”. Such analogies are particularly useful for answering such questions as “*I already have a firewall, why do I need Intrusion Detection Systems as well?*”.

Depending on how you approach the security of your home, for example, you may opt for high quality locks on your doors and windows. That will help to keep intruders out, and could be thought of as the equivalent of the firewall – perimeter defences. It’s nice to feel secure, but the determined burglar can often find ways around these measures. He can always throw a brick through your back window, for instance, and get in that way – or perhaps you simply forget to lock your door one day.

Once he is inside your home he is free to wreak havoc, perhaps making it obvious he has been there by stealing or wrecking things, or perhaps simply taking copies of any keys he finds so he can come and go later at his leisure. Whatever happens, you don’t want your first knowledge of the break-in to be when you return home to the ransacked contents.

That is why many people install a burglar alarm as well. Should the intruder gain access through the perimeter defences, the burglar alarm alerts you or your neighbours to the break in immediately, and provides an additional deterrent to the would-be thieves.

IDS, therefore, are the equivalent of the burglar alarm. To be used alongside firewalls, they are a recognition of the fact that you can never have a 100 per cent secure system. However, should someone be clever enough to breach your perimeter defences, you want to know about it as soon as possible.

It would also be nice to know what they have been up to while they were inside too.

Intrusion Detection and Vulnerability Assessment are becoming increasingly important as the stakes become higher. In the 1980s and early 1990s, denial-of-service (DoS) attacks were infrequent and not considered serious. Today, successful DoS attacks can shut down e-commerce-based organisations like online stockbrokers and retail sites.

Within the IDS market place there are four broad categories of product:

Host IDS (HIDS)

This category, so often overlooked recently in the face of the more "interesting" technology of Network IDS, is seeing something of a resurgence.

This could be due partly to the fact that high speed switched networks are providing a significant obstacle to effective Network IDS implementation, or it could also be that there is a growing realisation that there is more to IDS than detecting suspicious packets on the wire.

This type of product can itself be split into two main categories, with some host-based systems providing elements of both:

"Traditional" Host IDS (HIDS)

Host IDS (HIDS) products employ an agent that resides on each host to be monitored. The agent scrutinises event/system logs, kernel logs, critical system files and other auditable resources looking for unauthorised changes or suspicious patterns of activity. Whenever anything out of the ordinary is noticed, alerts or SNMP traps are raised automatically.

For instance, a HIDS will monitor the Registry for unauthorised access, kernel logs to detect when inappropriate processes are initiated, or logins to take note of when an attempt is made to access an account with an incorrect password. If a login attempt fails too many times within a short time span the system may conclude that someone is trying to gain access illegally and an alarm can be raised.

Traditional HIDS are very good at detecting insider threats and usually provide extensive damage assessment and data forensics. Bear in mind that the term "insider" does not always refer purely to your own employees. It is possible, for example, for an attacker to gain access to internal systems via a legitimate user name and account combination without having to run any exploit that is detectable by a Network IDS product – perhaps gaining access via *social engineering*. At that point, the attacker would have all the rights and privileges associated with that user, and is much harder to detect.

Disadvantages of the HIDS approach are the need for agent deployment on key systems, and the requirement for close attention to audit policy. They can often be the most difficult of all Intrusion Detection Systems to configure.

File Integrity Assessment (FIA)

File Integrity Assessment (FIA) products monitor the state of system and application files, or the Registry.

They do this by making an initial pass of a clean system and storing a condensed “snapshot” of how that system should look, usually in the form of cryptographic “hashes” of the monitored objects. Once this has been done, it is impossible to tamper with either the original objects or the hash values without invalidating the checksum files. At regular intervals, the FIA product makes a fresh pass, recalculating the checksum values and comparing them against those stored previously.

Thus if an intruder - or Trojan Horse - does manage to gain access to the system and make changes to key files, the FIA product will detect this and raise an alert. This makes FIA the perfect technology for assessing the true extent of the damage inflicted by a successful attack.

The downside, of course, is that because the scans are periodic rather than real time, its strength is in forensic analysis after an attack has been perpetrated, and thus it is of little use where real-time alerting is required.

Network IDS (NIDS)

The *Network IDS* (NIDS) monitors traffic on the wire in real time, examining packets in detail in order to detect patterns of misuse – perhaps spotting denial of service attacks or dangerous payload – before the packets reach their destination and do the damage.

They do this by matching one or more packets against a database of known “*attack signatures*”, or performing protocol decodes to detect anomalies, or both. These signature databases are updated regularly by the vendors as new attacks are discovered.

When suspicious activity is noticed, a network based IDS is capable of both raising alerts and terminating the offending connection immediately (as are some host-based IDS). Some will also integrate with your firewall, automatically defining new rules to shut out the attacker in future.

Most of the network-based IDS available to date work in what is known as “*promiscuous mode*”. This means that they examine every packet on the local segment, whether or not those packets are destined for the IDS machine (much like a network monitor, such as Sniffer). Given that they have a lot of work to do in examining every single packet and tracking active sessions, they usually require a dedicated host on which to run due to their heavy use of system resources.

For instance, most attacks are not based on the contents of a single packet, but are made up of several, sometimes sent over a lengthy period of time. This means that the IDS has to store a number of packets in an internal buffer in order to track established sessions and compare *groups* of packets with its attack signature database. This is known as “*maintaining state*”, and allows IDS to compare new packets against its signature database in the context of what has happened previously in a particular session, rather than examining every packet in isolation.

You will also need one Network IDS sensor per segment, since they are unable to see across switches or routers, and some have problems keeping up with heavily loaded Fast Ethernet segments (never mind Gigabit). Clearly, they would also have problems with encrypted traffic.

Network Node IDS (NNIDS)

The Network Node IDS (NNIDS) is a type of "hybrid" IDS agent which overcomes some of the limitations of the network-based IDS.

The NNIDS agent works in a similar manner to the network-based IDS in that it takes network packets and performs protocol analysis and/or compares them against signature database entries. However, this "micro agent" is only concerned with packets targeted at the network node on which it resides. Because it is installed within the protocol stack of the host, it is sometimes referred to as a *Stack-based IDS*.

Rather confusingly, it is also occasionally referred to as "*host-based*", but usually only by those who are looking at the industry purely from a Network IDS viewpoint. For the purposes of this report, Host IDS is concerned with monitoring of log files and behavioural analysis, whereas both Network and Network Node IDS are concerned with TCP analysis – the only difference is that one (NIDS) is running in promiscuous mode whilst the other (NNIDS) is not.

The fact that the NNIDS system is no longer expected to examine every single packet on the wire, however, means that it can be much faster and take less system resources, and this allows it to be installed on existing servers without imposing too much overhead. It also makes it particularly suitable for heavily loaded segments, switched network environments, or VPN implementations with encrypted traffic on the wire – all areas where traditional network-based IDS' have problems.

Obviously it is necessary to install a number of these NNIDS agents – one for every server to be protected – and they will all have to report back to a central console.

Many organisations may opt for a combination of the two – NNIDS on individual servers in switched server farms, and traditional NIDS on less heavily used segments, where a single IDS can protect a large number of hosts.

Intrusion Prevention Systems (IPS)

Most IDS systems tend to be *reactive* rather than *proactive* – that is they often have to wait until something has actually happened before they can raise the alarm.

The *Intrusion Prevention System (IPS)*, however, attempts to be proactive, and is designed to stop intrusions dead, blocking the offending traffic before it does any damage rather than simply raising an alert as, or after, the malicious payload has been delivered.

Of course, the downside with this approach is the potential for introducing a self-inflicted Denial of Service condition. The thorny issue of *false positives* is one that has plagued the IDS industry to date - sometimes it is very difficult to design an attack signature that will alert reliably on every variation of the exploit whilst ensuring that it will not be triggered accidentally by valid traffic. This is bad enough when you are just being pestered by false alarms at your IDS console, but when the IPS system cuts off a potential customer or your CEO from a vital computer system, the consequences can be far more serious.

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.

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)*.

As with a typical firewall, the NIPS has two network interfaces, one designated as *internal* and one as *external*. As packets appear at the internal interface they are passed to the detection engine, at which point the IDS device functions much as any IDS would in determining whether or not the packet being examined poses a threat. However, if it should detect a malicious packet, in addition to raising an alert, it will discard the packet and mark that flow as bad. As the remaining packets that make up that flow arrive at the IPS device, they are discarded immediately.

Legitimate packets are passed through to the internal 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 or packets with obfuscated URLs will be “cleaned up” before being passed to the destination host.

Seems too good to be true, doesn’t it? Why bother with an IDS at all, since the NIPS device does everything an IDS can do as well as provide true prevention capabilities? Well, when something *seems* to good to be true, it usually *is*!

Firstly, there is the aforementioned potential for an in-line device to introduce a Denial of Service condition by accidentally blocking a legitimate packet flow. If the NIPS is at the gateway to your DMZ containing your e-commerce servers, you had better be sure that *none* of your attack signatures are susceptible to false positives.

For this reason, most adopters of this technology would probably run the NIPS with a reduced signature set that contained only those signatures that had a very low or zero propensity for triggering accidentally on benign traffic. They would then run a standard IDS product on the protected network behind the NIPS which utilised a more complete signature set in order to deal with the alerts that were raised by traffic not covered by the NIPS itself.

The second potential problem is that given the amount of work this device has to do, it can act as a serious bottleneck when installed at the gateway to your network. Most NIPS vendors have recognised this and have sought to get around it by using custom hardware, populated with expensive FPGAs and ASICs. At the time of writing, therefore, adopters of this technology tend to require deeper pockets than those buying plain old NIDS products.

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 when they detect an attack (a worrying piece of FUD that we have noticed in some IDS marketing literature recently). That does not count as *prevention* because by the time the IDS has detected it, the attack payload may already been delivered. Certainly the initial packet that caused the alert will have reached its target, and if the payload is contained in a single packet…. game over! On high speed networks, it would also be possible to deliver an entire payload spread across many packets before the TCP reset command took effect. Only an in-line device (or Host IPS, of course) is capable of real *prevention*.

Gigabit IDS

Clearly, host-based IDS in their various forms are not (or *should* not be) affected by the speed of the network on which they are installed. Therefore whenever we talk about Gigabit IDS we are, by definition, focussing on Network IDS with a Gigabit capability.

Where life gets difficult for those tasked with evaluating this technology is that different vendors have different ideas about what constitutes Gigabit IDS. Some products will be *true* Gigabit products, capable of pulling traffic off the wire for analysis at speeds of up to 1000Mbps (or beyond). Others are merely appliances that contain a Gigabit network card, whose main aim is to allow them to cope with 100Mbps or multiple 100Mbps segments easily.

There is nothing inherently wrong with the latter approach providing the marketing message is honest and does not describe the product as a *true* Gigabit appliance. As long as all the customer needs is to be able to handle 100-200Mbps with confidence - and the price is right, of course - then this is a perfectly valid tactic.

Even true wire-speed Gigabit appliances will have problems in certain areas if they are assembled from off-the-shelf components. At the time of writing, not even the best Gigabit network cards on the market are capable of pulling almost 1.5 million packets per second off the wire, never mind analysing that level of traffic. Thus a Gigabit network loaded with small packets (64 bytes) will cause problems for most Gigabit solutions, and the only way around that for the time being is to move towards custom hardware and ASICs.

Administrators need to be aware of the overall performance limitations of any device when deploying on Gigabit networks. As with most Fast Ethernet networks, the average Gigabit subnet is unlikely to see much more than a fraction of its total available bandwidth in use at any given point in time, and so where only 200-400Mbps is being used, the performance of the Gigabit IDS used to monitor it is less of an issue.

However, one tactic being employed in some organisations is to consolidate multiple 100Mbit segments using a Gigabit switch, and copy all the traffic from each segment to a single mirror, or SPAN (Switched Port ANalyser) port. The Gigabit IDS sensor is then attached to this port to monitor all of the traffic across multiple subnets, thus providing a cost-effective solution to monitoring a number of subnets using a single sensor.

Of course, even if the average utilisation of each subnet is only 40-50Mbps, once you mirror 20 of these you are asking your IDS sensor to monitor getting on for a full Gigabit of network traffic (providing your switch is actually *capable* of mirroring that amount of traffic, of course - an entire topic in itself which is beyond the scope of this report). This is when the performance limitations of some so-called Gigabit devices will begin to manifest themselves.

In some respects, detection performance is the least of the problems facing the administrator tasked with deploying these devices. The problem with any Gigabit IDS product is, by its very nature and capabilities, the amount of alert data it is likely to generate. With 1Gbps of traffic passing through the IDS, the number of alerts could reasonably be expected to be ten times that generated by the typical 100Mbps product. How many members of staff would be needed to process, investigate and resolve that number of alerts? How long before the IDS becomes just another device in the corner that is largely ignored thanks to its insistence on overloading the administrator on a daily basis?

More than ever before in the IDS space, centralised management reporting and forensic analysis is key to the success of the Gigabit IDS appliance. A reduction in false positives (through more accurate protocol decodes and signatures) and global pre-filtering of “safe” alerts that can be ignored are both essential. After that comes the ability to consolidate alerts from multiple sensors to a single management console. Far from increasing the load for a single administrator, this consolidation should help in identifying where potential break-ins are disguised by multiple exploits run over multiple subnets and over a long period of time.

Some management solutions will leave the administrator to determine the connection between exploits, providing the tools to “slice and dice” the data in a myriad of different ways in order to achieve this, whilst others will attempt to perform the correlation in an automated fashion (with varying degrees of success). Either way, the ability to create high-level reports of activity over a period of time, reshuffle and resort alerts in different ways, and then drill down to discover the exact trigger that caused the alert in each case in an efficient manner is now essential.

Without effective management capabilities, alert handling, and forensic analysis tools, the Gigabit IDS is just another lump of iron sitting in the machine room equipment rack.

SOURCEFIRE IS3000 V4.0.2

Executive Summary

Sourcefire's 3D product suite has been designed as an integrated real-time network defence infrastructure for identifying and protecting against network threats.

In addition to the usual Network-IDS sensors (various models to handle a wide range of bandwidth requirements up to 1Gbps currently) and a centralised Defense Center management appliance, the 3D product range also includes the unique RNA Sensor. This is a passive scanning device which monitors network flows, identifies new hosts, operating systems and services on the network, provides policy enforcement capabilities and offers additional context for the IDS alerts based on its extensive knowledge of the network.

The IS3000 appliance tested features the tried and tested Snort detection engine running on an Intel-based appliance with a high-performance network driver to provide the ability to monitor up to 1Gbps of traffic. In NSS tests, the IS3000 proved itself capable of handling 1Gps of traffic under normal network conditions. We also found the device to be to be very stable and reliable, coping with our extensive test suite with ease and without succumbing to common evasion techniques.

The Defense Center management system has been well designed to handle management and configuration of large numbers of IDS and RNA sensors across the enterprise. Alert handling, monitoring and reporting are all powerful and flexible, especially when combined with the additional contextual data provided by RNA, which could go a long way to removing one of the biggest perceived "problems" of IDS systems - analyst overload from irrelevant alerts.

Architecture

The *Sourcefire 3D System* (also known simply as *Sourcefire 3D*) is composed of:

- ***Intrusion Sensors**, which perform intrusion detection and are available in models that accommodate a range of network bandwidths*
- ***RNA Sensors**, which perform passive network mapping and vulnerability assessment to augment the IDS*
- ***Defense Center**, which is used to configure devices and aggregate and correlate information from managed Intrusion Sensors and RNA Sensors.*

Sourcefire also offers an *Intrusion Agent* for Snort, the open source Intrusion Detection System. Both Intrusion Sensors and RNA Sensors can be managed directly via a Web-based interface if Defense Center is not required. This report, however, focuses on management via Defense Center.

Intrusion Sensor

Sourcefire produces Intrusion Sensors for every network bandwidth need:

- *The 1U IS500 supports up to 5 Mbps*
- *The 1U IS1000 supports up to 45 Mbps*
- *The 1U IS2000 supports up to 100 Mbps*
- *The 1U IS2100 supports up to 250 Mbps*
- *The 2U IS3000 supports up to 1 Gbps*
- *The 2U-4U IS5800 supports from 2-8 Gbps*

The model submitted for testing was the 1Gbps IS3000. This is a 2U appliance based on standard Intel hardware with dual Xeon processors, dual power supplies (not hot swappable), a large hard drive (for local storage of alert/log data and configuration parameters), 4GB RAM and a two-port Gigabit network card as standard. This can be used to operate in-line as an IPS device (one port in, one port out), monitor two separate subnets/SPAN ports, or recombine the separate connections from a network tap. It can also be augmented with an additional four port card if required.

The IS3000 runs a hardened Linux distribution with custom-written high-performance network drivers. Each Intrusion Sensor has several methods that it uses to look for the broad range of exploits. These methods include packet decoders and preprocessors that report on anomalous traffic that might signal an intrusion attempt. Next, the detection engine uses Snort-based rules to examine the decoded packets for pattern-based attacks.

Capturing and Decoding Packets

Before packets can be inspected by the Intrusion Sensor, the packets must be captured from the network. Sourcefire has eschewed the standard *libpcap* in favour of custom-written network drivers in order to achieve Gigabit performance levels.

In most installations, one of the network interface cards (NIC) on the Intrusion Sensor is designated as the sensing interface. The NIC for the sensing interface is set to promiscuous mode, which means that it receives every packet on the network segment, regardless of whether it is destined for the Intrusion Sensor. Also, by default, the NIC on the Intrusion Sensor runs in stealth mode, which allows the interface to see network traffic without being visible or accessible to other hosts on the network.

As the Intrusion Sensor captures packets, it sends them to the *packet decoder*. The packet decoder takes the packet stream and converts the packet headers and payloads into a format that can be easily used by the preprocessors and the detection engine. Each layer of the TCP/IP stack is decoded in turn, beginning with the data link layer and continuing through the network and transport layers.

Processing Packets

After the packets are decoded through the first three TCP/IP layers, they are sent to preprocessors, which normalise traffic at the application layer and detect protocol anomalies. After the packets have passed through the preprocessors, they are sent to the detection engine.

The detection engine inspects the packet headers and payloads to determine whether they trigger any of the rules. Preprocessors can be enabled or disabled as required to suit a particular environment. For example, one of the preprocessors normalises HTTP traffic.

If the administrator is confident that his network does not include any web servers using Microsoft Internet Information Services (IIS), he can disable the preprocessor option that looks for IIS-specific traffic and thereby reduce system processing overhead (thus increasing performance of the system as a whole).

The detection engine takes three tracks as it inspects packets from the preprocessors:

- *The rule optimiser*
- *The multi-rule search engine*
- *The event selector*

When a policy is applied to the Intrusion Sensor, and before it processes any packets, the rule optimiser reads in all the activated rules and classifies them in subsets based on criteria such as transport layer, application protocol, direction to or from the protected network, and so on.

As packets arrive at the detection engine, the engine selects the appropriate rule subsets to apply to each packet. After the rule subsets are selected, the multi-rule search engine performs three different types of searches:

- *The protocol field search looks for matches in particular fields in an application protocol.*
- *The generic content search looks for ASCII or binary byte matches in the packet payload.*
- *The packet anomaly search looks for packet headers and payloads that, rather than containing specific content, violate well-established protocols.*

After the multi-rule search engine examines the packets, it generates an event for every rule triggered and adds it to an event queue. The event selector prioritises the events in the queue and logs an event to the event database. These are the events that appear in the sensor statistics and event reports.

Generating Events

Packets are evaluated by the packet decoder, the preprocessors, and the detection engine. At each step of the process, a packet could cause Sourcefire 3D to generate an event, which is an indication that the packet or its contents may be a risk to the security of the network or, in the case of an attack that originates from within the network, to the security of an external network.

For example, if the packet decoder receives an IP packet that is less than 20 bytes (which is the size of an IP datagram without any options or payload), the decoder interprets this as anomalous traffic and generates an event.

Similarly, at the preprocessing step, if the IP defragmentation preprocessor encounters a series of overlapping IP fragments, the preprocessor interprets this as a possible attack and generates an event. The same kind of response occurs within the detection engine, with most rules written so that they also generate events when triggered by packet or stream content.

Each event in the database includes two types of information about the potential attack. The first is called an event header, and contains information about the event name and classification, the source and destination IP addresses and ports, the process that generated the event, and the date and time of the event. The second is the packet log, and includes a copy of the decoded packet header and packet payload.

The Intrusion Sensor has features for viewing all events, filtering them according to any number of criteria, creating incidents to gather information about events that seem related and warrant further investigation, and producing reports on event statistics.

Preprocessors

The detection engine maintains a configuration file that details how, and in what order, the system executes preprocessors, as well as other configuration information. The order in which preprocessors are configured in this file specifies the order in which they act on packets. Sourcefire appliances simplify this by providing a way to configure preprocessors through the user interface. The default policy configuration sets the preprocessors to perform IP transfer layer protocol decoding first, then perform data normalization as needed.

This approach provides the following benefits:

- *The Intrusion Sensor can generate an intrusion event against fragmented IP datagrams that cannot be defragmented, and then stop inspecting those packets.*
- *The Intrusion Sensor can generate an event against TCP packets whose state cannot be validated, and then stop inspecting those packets.*
- *Only packets that can be appropriately tested by rules are normalised, optimizing performance by dropping TCP packets that cannot be reassembled and are not part of a valid TCP session.*

Sourcefire Defense Center

The *Defense Center* is the centrepiece of the Sourcefire 3D System. Contrasted with RNA Sensors and Intrusion Sensors, which generate events based on the traffic they see, the Defense Center is a central management point that can manage both types of sensors and automatically aggregate the events generated by them. The Defense Center then correlates the different events in a way that helps the administrator understand the vulnerabilities on his network and the attacks against the hosts on it.

Event correlation occurs when the Defense Center manages one or more Intrusion Sensors and one or more RNA Sensors on the same network segment. In this configuration, the Defense Center qualifies each intrusion event from an Intrusion Sensor based on whether the RNA Sensor on the same segment detected a vulnerable service or open port on the host which is the target of the attack.

If the targeted host is not vulnerable, the *impact* of the attack is low (and the *Impact Flag* of the event is modified accordingly). However, if the targeted host is vulnerable, then the impact is high.

The event views on the Defense Center also tightly integrate events generated by Intrusion Sensors and events from RNA Sensors. For example, the table view of intrusion events, which lists important information about each event generated by Intrusion Sensors, includes a hyperlink next to each internal host. Selecting this link displays a host profile generated by the RNA Sensor for that host.

The host profile lists the information about the host including the IP address, the MAC address, the operating system and vendor, the protocols and services in use, and the known vulnerabilities associated with the services and operating system. Similarly, as a network discovery event is viewed on the Defense Center, a link is provided to view all the intrusion events associated with that data flow.

If it is necessary to monitor a high bandwidth network segment, the Defense Center's built-in load balancing can be used to divide the network traffic over two or more Intrusion Sensors. Although the Sourcefire product line includes Intrusion Sensors rated up to 8Gbps, it is possible to monitor even higher bandwidth networks without the need to deploy third-party load-balancing solutions.

The built-in high availability feature also allows the designation of redundant Defense Centers to manage groups of Intrusion Sensors and RNA Sensors (each Intrusion or RNA Sensor can be configured to transmit alerts to two Defense Center appliances). Event data streams from managed sensors to both Defense Centers, and certain configuration elements are also maintained on both Defense Centers. If the primary Defense Center fails, it is still possible to monitor the network for RNA events, intrusions, and policy violations without interruption using the secondary Defense Center.

RNA Sensor

RNA Sensors are appliance-based sensors that analyze traffic passively on the network segment where they are installed, and then build a network map showing the topology of that network segment. The network map can be used quickly to view information about:

- *Hosts running on the network*
- *Devices that bridge the network to other networks*
- *Services running on the network*

RNA Sensors also gather information about new hosts and services that appear on the network as well as changes to the existing network infrastructure. This makes Sourcefire 3D (with RNA) perfect for policy enforcement and change management, as well as intrusion detection.

RNA Sensors are delivered with a vulnerability database. This database, updated regularly, correlates known vulnerabilities with the operating systems and services running on the network and helps prioritise the steps needed to strengthen the network's security. Once patches are applied that remediate the vulnerabilities, the administrator can mark specific hosts as no longer vulnerable.

RNA Sensors are also able to collect flow data, which allows them to track sessions for hosts on a monitored network. This flow data can be used to detect anomalous traffic within the network and between a network and the outside world.

A feature called *policy and response* on RNA Sensors can be used to monitor compliance with corporate security policies. For example, if the security policy forbids the use of wireless access points, a compliance policy can be created that automatically alerts the administrator if a new router appears on the network.

Key components of RNA are also available as a software-only installation for Intrusion Sensors and Red Hat Linux servers.

Performance

The aim of this section is to verify that the sensor is capable of detecting and logging 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 sensor can sustain before it begins to drop packets/miss alerts.

The Sourcefire IS3000 was tested up to 1Gbps, the rated speed of the appliance. Performance under all but the most extreme load conditions was impeccable, with 100 per cent of all attacks being detected under all load conditions except the 10,000cps and 20,000cps HTTP tests (see *Test Results* section of this report).

In particular, the device turned in a flawless performance in all of our “real world” tests, and we would thus have no problem in rating the Sourcefire IS3000 as a true 1Gbps device under all normal network conditions.

The Sourcefire IS3000 performed consistently and mostly reliably throughout our tests. Exposing the sensor interface to an extended run of ISIC-generated traffic had no adverse effect, and the device continued to detect and log all other exploits throughout and following the ISIC attack. There were no residual stability problems.

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

Security Effectiveness

We installed one IS3000 sensor with the latest signature pack reporting to a single Defense Center server. We used a modified version of the default policy, which had **all** attack signatures enabled apart from a small number of server-to-client signatures which are disabled by default for performance reasons. We also enabled some key audit-only signatures.

Signature recognition was acceptable out of the box (79 per cent), and was increased to 91 per cent after the application of a signature pack update which was provided to us in under 48 hours.

We noted a few test cases raised multiple alerts for a single exploit. Our “false negative” (modified exploit) cases also posed some problems, even following the signature update, indicating that many of the Snort signatures are still detecting the *exploit* rather than the underlying *vulnerability*. However, it should be noted that recent improvements in the Snort rule “language” and efforts by Sourcefire’s signature-writing team have done a lot to improve this area.

A major concern in deploying an IDS is the raising of false alarms. We noted two false positive alerts from our test suite, both of which were fixed following the signature update. We also noted some “noise” from our standard network traffic traces (which contain nothing but basic NetBIOS and DNS traffic). As with most devices of this type, we would recommend deploying in a live network to determine how it responds to live traffic before purchasing.

Resistance to known evasion techniques was excellent following a code update to rectify some issues with FTP traffic normalisation. Once these had been fixed, Sourcefire collected a clean sheet across the board in our evasion tests. *Fragroute, Whisker, ADMmutate, running exploits on non-standard ports* and even *RPC record fragging* all failed to trick Sourcefire into ignoring valid attacks.

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

Out of the box, the Sourcefire IS3000 handled 1 million open connections easily, and the device appeared to be immune to stateless attack replay tools such as *Stick* and *Snot*.

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 Entire Sourcefire 3D system is very straightforward since each device is provided as a complete turnkey appliance, and each appliance can be managed directly via a Web-based interface. In a full Sourcefire 3D installation, however, once the Sensors have been installed they are managed from the Defense Center rather than directly.

Initial installation is restricted to entering the IP address of the management interface of each device, should the default IP addresses be unsuitable. This is achieved either by changing a host on the management network to match the same default IP address range as the Sourcefire devices and then using the Web interface, or by logging in to the console of each appliance and changing the IP address via the CLI.

Unfortunately, if the latter method is preferred then it is necessary to log in as *root* and enter the appropriate Linux commands to change the IP address and select the default route. Whilst this is not a difficult task, and is well documented in the *Installation Guide*, we believe that it should not be necessary (or even possible) to login as *root* on the sensor, let alone have the administrator be expected to issue Unix commands.

Whilst *root* access may occasionally be required for serious troubleshooting or recovery, this should be tightly controlled.

Instead, it would be better to have a separate *admin* or *service* account with a simple text-based menu covering all the most common admin functions such as setting network parameters, date and time, restarting the Snort daemon, shutting down the sensor, rebooting the sensor, and so on.

Once the IP details have been configured it is necessary to enable communications to the Defense Center (which includes an authentication process). Once the devices are communicating, it is necessary to apply license files and configure the Sensor devices to be managed within the Defense Center.

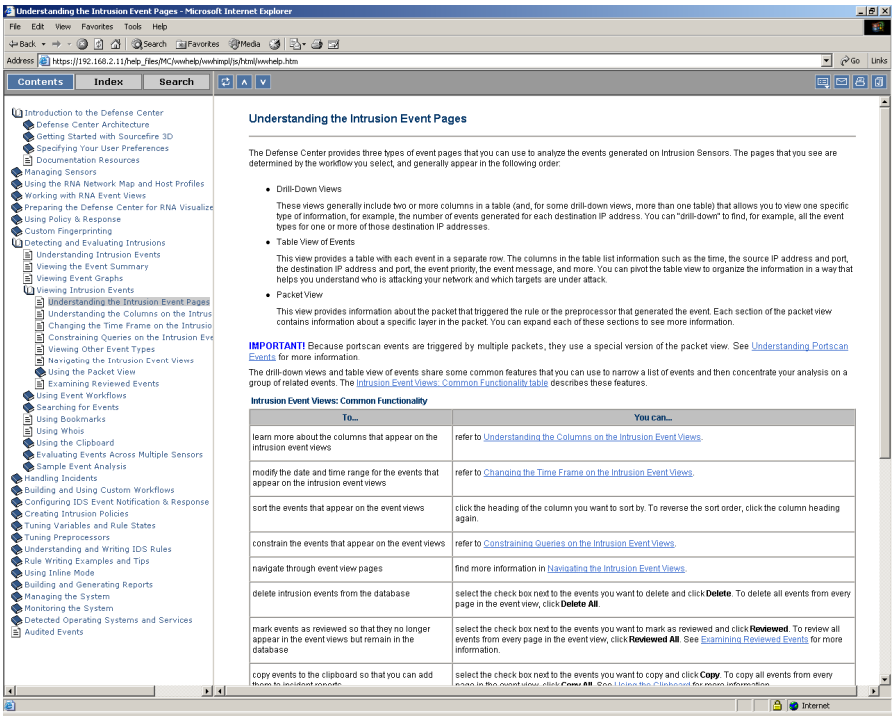

Figure 1 - Sourcefire 3D: The Help system

Once this has been done, it is possible to manage the sensor configurations and view aggregated sensor data from the Defense Center's user interface. The Defense Center allows the administrator to group sensors in order to more easily apply policies and install updates on multiple sensors in a single operation.

The documentation set is provided in electronic format (PDF files) only, and each appliance (Intrusion Sensor, RNA Sensor and Defense Center) has its own *Installation Guide* and *User Guide*.

The documentation is outstanding, amongst the best we have ever seen for *any* product in our labs. Apart from providing extensive instruction on all aspects of deploying and managing Sourcefire products, it also contains extensive background information on the technology and the concepts behind it.

This includes stateful inspection, stream reassembly, understanding stream-based attacks, understanding state-related exploits, packet reassembly, understanding fragmentation attacks, and so on. There is also an extensive and extremely thorough section on understanding and writing Snort-based rules.

In other words, not only does the documentation cover everything you need to know about the product, but it also acts as a useful source of IDS/IPS-related reference material.

In addition, all of the extensive documentation is also made available as an on-line, context-sensitive help system. Wherever you are in the Sourcefire 3D management system, a click on the nearest *Help* icon will bring you straight to the relevant part of the documentation.

Configuration

The Web-based UI is clean and fast, the latter being helped particularly by the fact that it is pure Web-based technology with not a line of Java code in sight. This also means that it is possible to access the management GUI from any permitted host on the management network (access can be restricted to certain hosts) without the need to install any local code or client software.

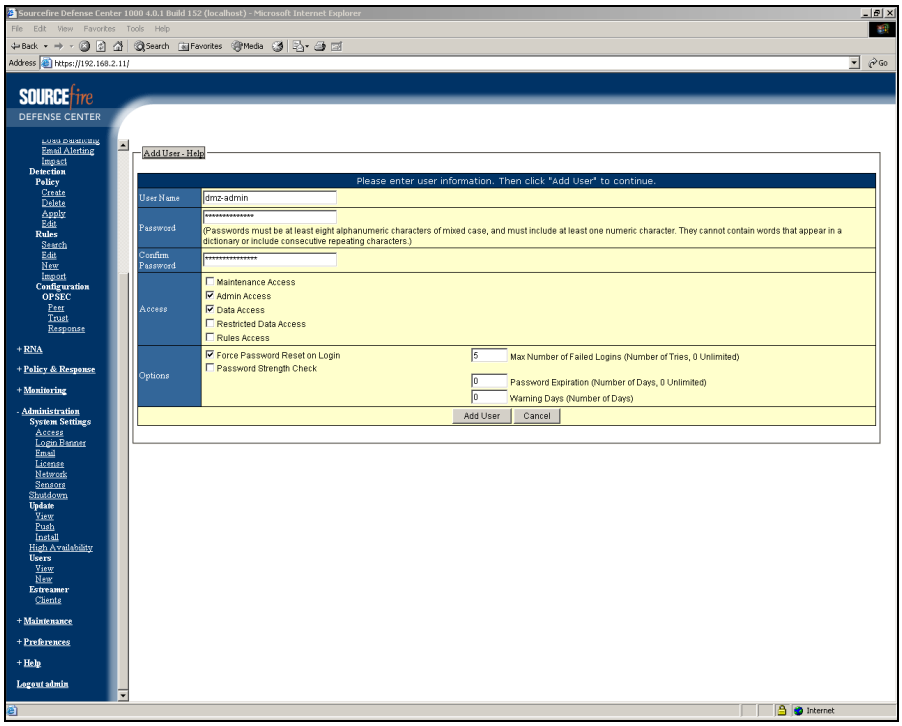

Figure 2 - Sourcefire 3D: User account maintenance

After logging in, the features that are accessible are controlled by the privileges granted to the user account. The administrator is presented with a dual-pane display: the left-hand pane contains a hierarchical menu containing options for *Intrusion Sensor, RNA, Policy & Response, Monitoring, Administration, Maintenance, Preferences* and *Help*. The right hand pane contains various data-entry windows which obviously depend on which menu option is selected.

When the system was installed, the user who performed the installation created a single administrative user account and password. That account can then be used to create additional user accounts which should be used on a day to day basis.

Certain user preferences can be specified for each account, including passwords, time zone settings, and event viewing preferences such as number of rows to display per page, default workflows to use in various parts of the UI, default auto-refresh interval (or disable auto-refresh), and so on.

The following user privileges are available:

- ***Maintenance access -*** *allows users to access monitoring functions (including host statistics, performance data, audit logs, system logs, and generating and viewing reports) and maintenance functions (including task scheduling, backing up and restoring the system, and configuring DHCP cache settings, proxy settings, and time synchronization properties).*
- ***Data access -*** *allows users to view and analyze events, generate event reports, and create and edit incident reports.*
- ***Restricted data access -*** *allows users to view and analyze events, copy events to the clipboard, and create incident reports for specific subnets. Restricted data users can have rules access, but cannot have data, maintenance, or admin access.*
- ***Rule access -*** *allows users to add and modify rules, configure preprocessors, create and modify variables, and create and apply intrusion detection policies.*
- ***Admin access -*** *allows users to set up the Intrusion Sensor's network configuration, create and modify user accounts, control which IP addresses can access the Intrusion Sensor, and configure network and database settings. Administrative users also have data, rule, and maintenance access. Users without admin access are restricted from accessing administrative features, and the navigation menu differs in appearance for each type of user.*

From the remaining configuration menus, it is possible to define which IP addresses can access the Sensors, specify how many events the intrusion event database can store (when the maximum number of events is reached, the oldest events in the archive are deleted to maintain the event limit), and backup and restore system configuration files.

Backup files can be saved to the Sensor or to the local host, and it is possible to upload backup files to fully recover a Sensor to its saved configuration. Event and packet data can be saved at the same time as configuration data if required.

Rule updates can be downloaded manually (outside the GUI) and applied via the GUI, or can be downloaded and applied entirely via the GUI. New rules can be added to the Rules database in their default state or always disabled.

All of the critical management tasks can be scheduled to run at regular intervals, or as one-off jobs. It is possible to run backups, apply policies, generate reports, download rules and updates, and install rules and updates. Creating scheduled jobs is extremely easy using the calendar-format interface.

After adding scheduled tasks, they can be viewed and monitored via the same intuitive calendar interface, which provides a list of all scheduled tasks and a timeline.

Policy Management

Intrusion policies provide a variety of components including:

- *Preprocessors that process traffic so that the engine can inspect it*
- *Rules that inspect the protocol header values, payload content, and certain packet size characteristics*
- *Tools that allow the administrator to control how often events are logged and displayed.*

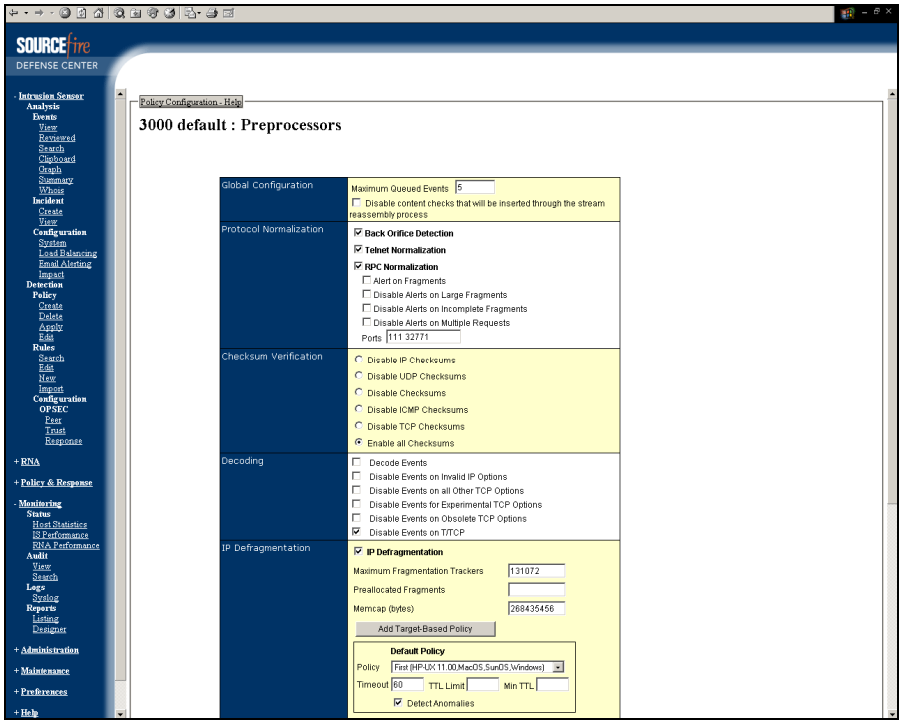

Figure 3 - Sourcefire 3D: Policy Editor (preprocessors)

Sourcefire 3D provides a default intrusion policy for each model of the sensor, and we found the default policy to be very usable, with almost everything enabled barring:

- *Chat signatures*
- *Shellcode signatures*
- *Peer-to-Peer signatures*
- *Policy signatures*
- *Information signatures*
- *Client-to-server signatures (a small number which could affect performance in high-bandwidth deployments)*

Whereas the default policy may be adequate for many, it is obviously possible for the administrator to create custom policies with a tuned rule set and preprocessor configuration. This provides the means to improve both the IDS performance and the administrator's ability to respond effectively to the events generated by an Intrusion Sensor.

There is a very useful *Policy Wizard* which leads the administrator through each step, allowing him to:

- *Name the policy*

- *Configure preprocessor settings*
- *Set event thresholds and suppress specified events*
- *Set variable values and create new variables*
- *Set rule states to enable, disable (or for IPS policies, drop)*

In addition to defining variables, setting rule states, and configuring preprocessors, it is possible to specify intrusion event notification limits, define external intrusion event notification, and configure external responses to intrusion events within each intrusion detection policy (such as reconfiguring a firewall).

Some analysts prefer not to be inundated by multiple instances of the same intrusion event, but want to control how often they are notified of a given intrusion event occurrence. In some cases, for example, the analyst may not care about an event until it has occurred a certain number of times. Thresholds can be set to limit the number of alerts raised from a single attack, and it is also possible to suppress certain alerts based on the source or target (for example, suppressing port scan alerts when coming from a particular host which is used to perform internal port scans on a regular basis).

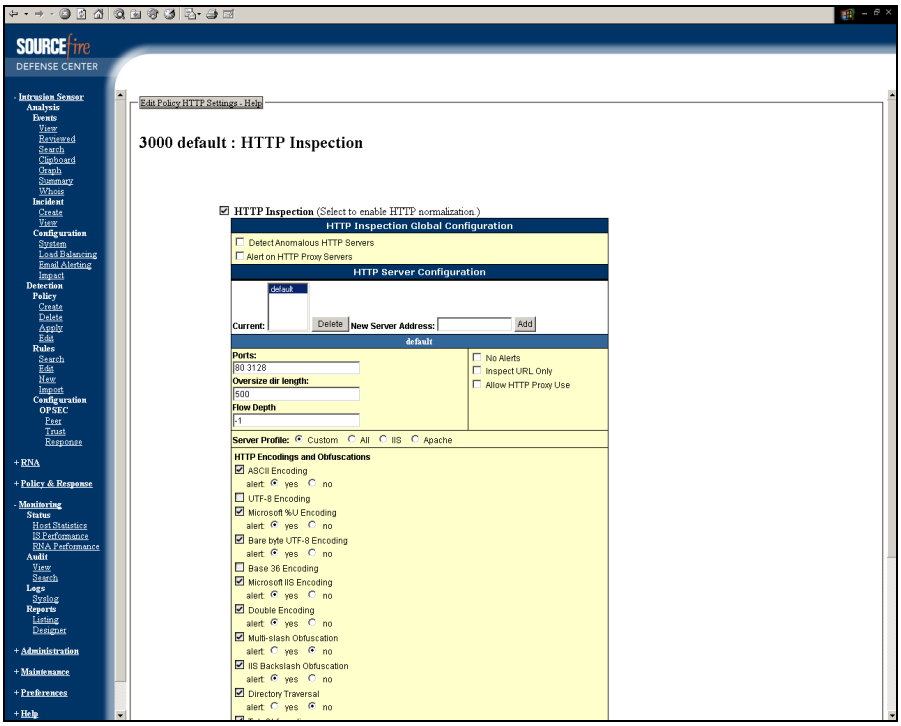

Figure 4 - Sourcefire 3D: Configuring HTTP Inspection preprocessor

The system can be configured to send notification of intrusion events to systems that are external to the Sourcefire 3D, such as syslog servers, SNMP trap servers, and specified e-mail addresses. These can be set per policy only in the *Policy Editor*, but can be overridden on a per-signature basis in the *Policy & Response* module.

Sourcefire 3D categorises rules into groups based on functional area. The default settings include *Category, Platform, Priority, Local, SANS Top 20,* and *Suggested Inline Rules*, and it is not possible to create custom groups.

On the *Rule State* page (accessed from the *New Policy Wizard* or Policy Editor), it is possible to deactivate or activate rules individually, or by rule category. For example, if there are no IIS web servers on the network, the administrator might choose to deactivate the entire *web-iis* category instead of deactivating individual rules within the category.

A *disable/enable all* check box makes this a simple operation. Note, however, that there is no search facility within the Policy Editor, and thus it is not possible to search for all rules which have Apache in the description and bulk enable/disable them in a single operation. There is a comprehensive search facility outside of the Policy Editor which allows searching on various parameters such as *Message, Protocol, Signature IDS, Source/Destination Port, Keywords*, and so on. However, this simply returns a list of matching signatures which can then be edited/duplicated one by one, and is not really intended to aid policy management.

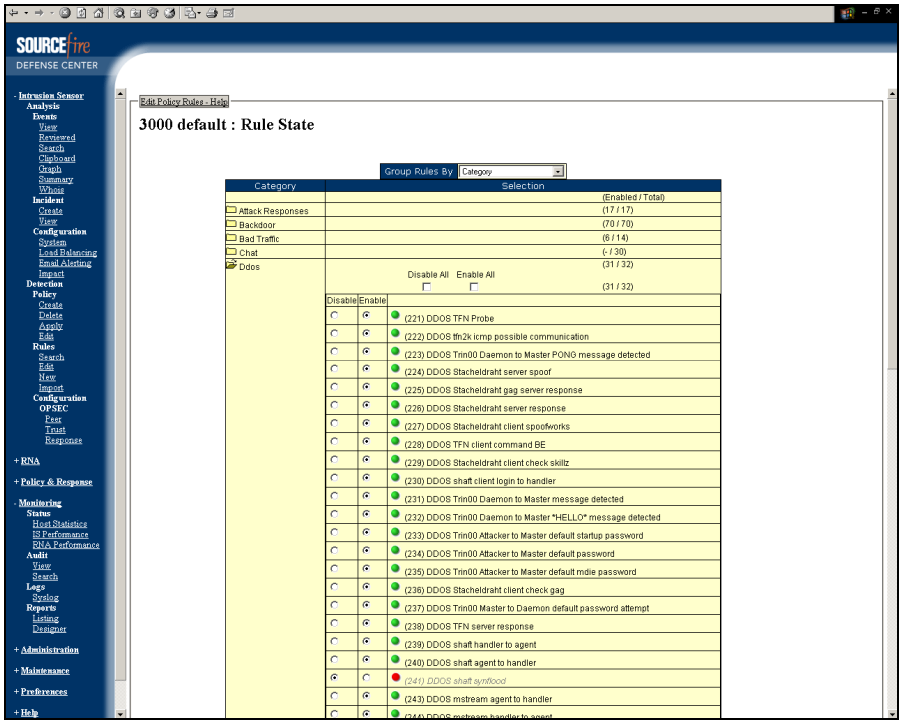

Figure 5 - Sourcefire 3D: Enable/disable signatures

Existing rules can be modified, but the results are always saved as new *Local* rules (with a new SID) and the original rule is disabled. The original rule will never be reactivated by a Sourcefire update, preventing potentially nasty “clashes” where two different rules perform similar operations.

Sourcefire has attempted to wrap as much of the rule editing procedure in a usable GUI as possible, and for simple rules it has succeeded. All of the rule header information - such as *Signature ID, Message, Classification, Action, Flow Direction, Source/Destination Port and IP Address, Protocol*, etc. can be entered via simple text boxes and drop-down menus.

However, the Snort rule “language” has become very feature rich, extensive and free format in recent years, and so each of the required detection options (*content, byte jump, byte test, pcre*, and so on) must be entered “long-hand” in one or more *Detection Option* boxes.

Any number of these can be added from a drop-down menu at the bottom of the screen, and the *Rule Editor* uses these to build a standard Snort-format rule when the Policy is finally saved. Extensive documentation is available to assist with the rule-writing process.

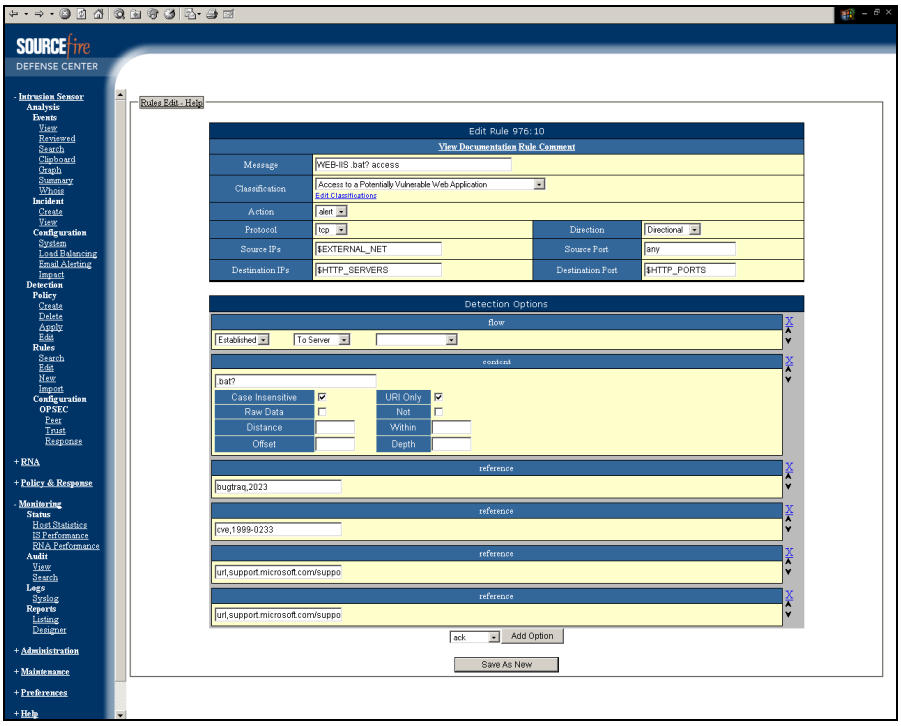

Figure 6 - Sourcefire 3D: Editing rules

Once the policy is saved to the database, it can be applied to one or more Intrusion Sensors. This is a simple operation, where the administrator selects a policy from the list at the top of the screen, and then selects one or more Intrusion Sensors (or groups of Sensors) at the bottom. Once the *Apply* button is clicked, the job is added to the *Task Queue* to be performed in the background. The progress of the job can be monitored via the Task Queue screen.

There is no means to automatically re-apply a policy following modification. However, where a changed policy is currently in use, it will show that it has been changed on the Apply Policy screen against the sensor. The most efficient way is to group sensors logically according to the type of policy they will require - i.e. DMZ sensors have a DMZ policy - and then a single policy can be applied to an entire group of sensors in a single operation.

When a policy is applied, the contents of the policy database are written to each Intrusion Sensor in the usual *snort.conf* and associated rules files, before the Snort engine is restarted automatically to activate the policy.

Alert Handling

Intrusion events are records of the network traffic that violates the intrusion detection policy on the Intrusion Sensor monitoring a specific network segment. These events can be generated by:

- *A link layer decoder such as the Ethernet II decoder*
- *A network layer decoder such as the IP decoder*

- *A transport layer decoder such as the TCP decoder*
- *An application layer decoder or preprocessor such as the HTTP Inspect preprocessor*
- *The rule engine*

One of the unique features of the Sourcefire Event Viewer is the use of *Workflows*. These provide a pre-defined means of navigating and drilling-down through the event data, starting with high-level groupings, for example (perhaps by event name), and then allowing drill-down via source/destination port, individual event details, and down to packet contents

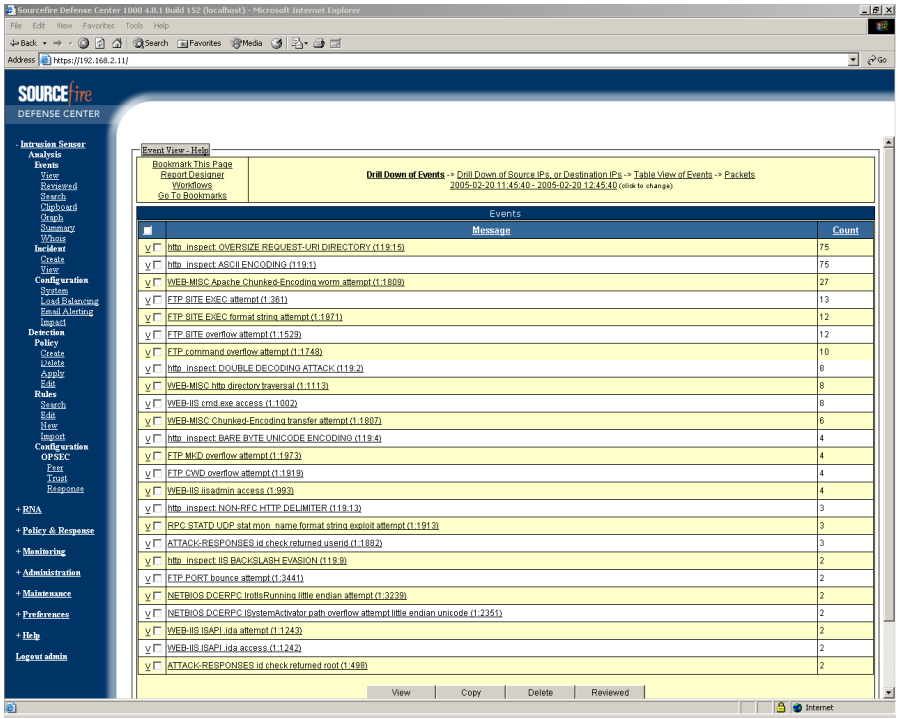

Figure 7 - Sourcefire 3D: Workflows determine the flow of event handling

The Event Viewer presents a list of default Workflows:

- ***Destination Port -*** *begins with a page showing the destination ports associated with the intrusion events, followed by a page showing the event types that were generated. Next is a tabular view of event information, called the table view of events, followed by a packet view that shows the decoded contents of the packets associated with each event.*
- ***Event-Specific -*** *begins with a page showing the event types that were generated. Next comes a page with two tables, one listing the source IP addresses associated with the events, the other showing the destination IP addresses associated with the events. The last pages in the workflow are the table view of events and the packet view.*
- ***Events to Destinations -*** *begins with a page of paired event types and destination ports that can be used to investigate which types of events are directed towards specific ports. The last pages in the workflow are the table view of events, and the packet view.*
- ***IP-Specific -*** *begins with a page showing two tables, one each for the source and destination IP addresses that are associated with the events.*

The next page shows the event types that were generated. The last pages in the workflow are the table view of events and the packet view.

- ***Source Port -*** *begins with a page showing the source ports associated with the intrusion events, followed by a page showing the event types that were generated. The last pages in the workflow are the table view of events and the packet view.*
- ***Source and Destination -*** *begins with a page showing the source and destination IP addresses for each event, followed by a page showing the event types that were generated. The last pages in the workflow are the table view of events and the packet view.*

Each of these Workflows steps the analyst through a series of pages to help him narrow down a list of intrusion events to be evaluated. Custom Workflows can be created, allowing the analyst almost unlimited variations on how to process event data, and it is very easy to switch between different pre-defined and custom workflows as required. Once the analyst determines which Workflow is best suited to his day-to-day analysis needs, he can select a default Workflow.

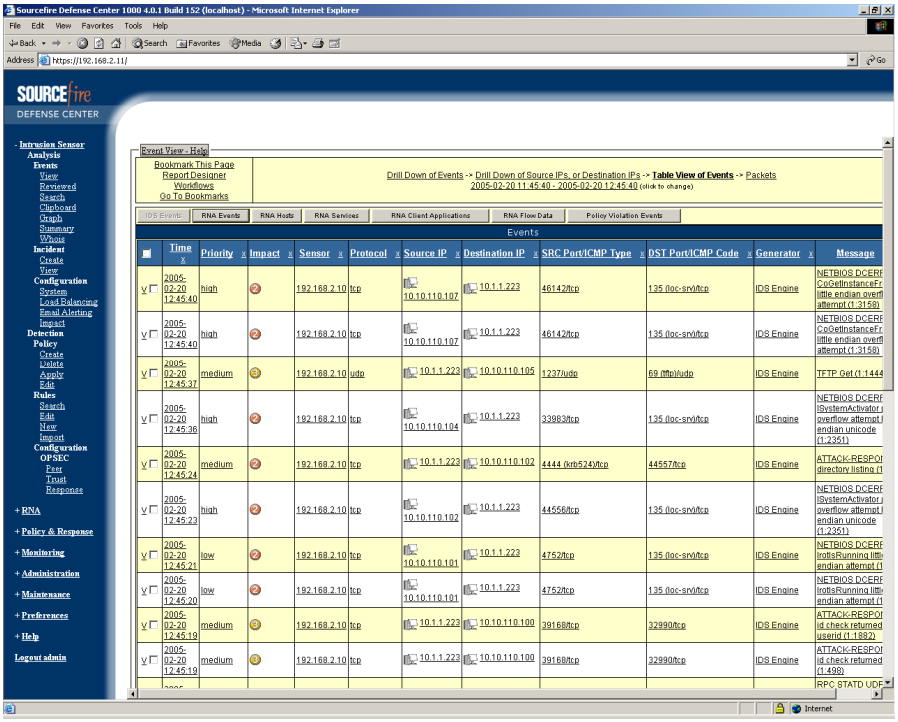

Figure 8 - Sourcefire 3D: Table View of Events

There is also a huge amount of on-the-fly customisation which can be performed on each default view, providing incredible flexibility in how event data is processed. Each individual event includes the following data:

- *Date and time the event was generated*
- *Event priority*
- *Event impact - when RNA Sensors are deployed, the event data is correlated with the known vulnerabilities for the victim in order to elevate or reduce the level of the Impact Flag automatically*
- *Protocol of the packet that caused the event*
- *IP address and port of the sender*
- *IP address and port of the target*
- *Brief description of the event*

It is also possible to view the packet that was captured with the event, and each of the rows associated with the packet can be expanded to display additional information. This includes both detailed packet contents and the associated event information, which comprises the rule which was triggered, and various hyperlinks which allow the analyst to view rule documentation, edit the rule directly, disable the rule in the current policy, and disable the rule across all policies.

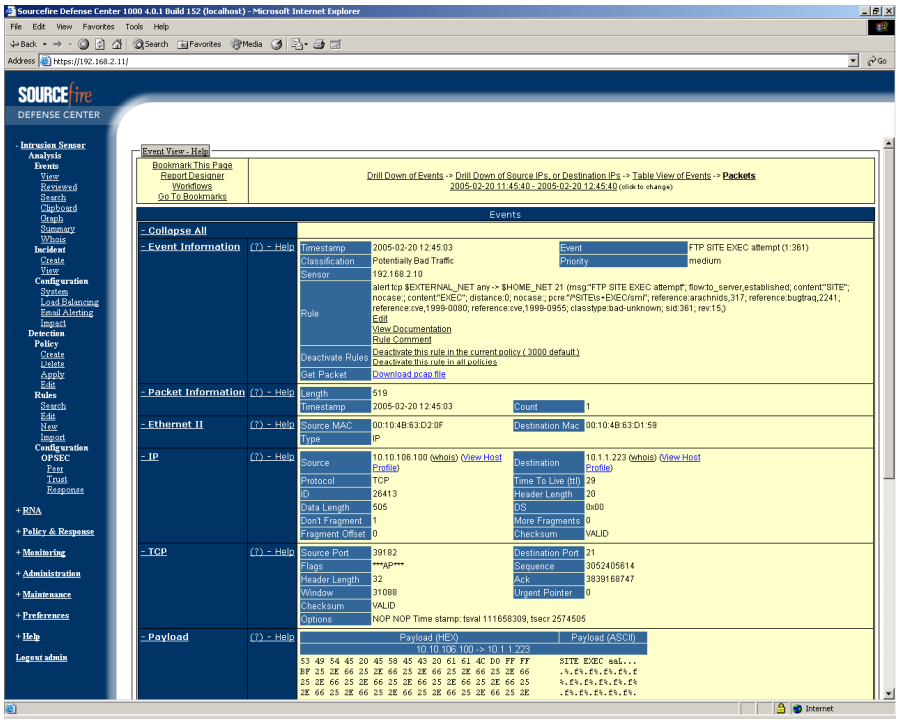

Figure 9 - Sourcefire 3D: Viewing packet details

Where RNA Sensors are deployed, hyperlinks are also available to view the Host Profile of the source and destination IP hosts, providing a detailed overview of the host configuration, active services and known vulnerabilities for that host (more on *Host Profiles* in the *RNA* section of this report).

From the table view of event details, it is possible for the analyst to further summarise the contents by eliminating one or more columns of data. This has the effect of grouping entries together by the item of data removed, producing new summary counts for each line in the table.

For example, from the lowest level detail table, the analyst can click on the *Remove* button for the *Time* column which effectively eliminates each individual alert and groups them together by name, providing a total for each alert within the selected time interval. Or he could eliminate the *Destination IP* address column to provide a grouping and totals by Destination IP.

Another way to drill down is to select an individual piece of data within any of the columns. For example, clicking on a particular *Source IP* will filter all the data on that item, showing only those entries for that particular Source IP. Clicking on a *level 1* entry in the *Impact Flag* field will show only those alerts with the highest priority *Impact Flag* - where RNA is used, this is the perfect way to determine which alerts need to be addressed most urgently.

Columns can be re-instated at the click of a button, and wherever navigation becomes a little too complicated, the pure Web-based nature of the GUI allows the browser Back button to be used to retrace steps through the displays.

At any point, a particularly interesting view can be bookmarked for use again by the analyst, allowing him to return quickly to a specific location and time in an event analysis. Bookmarks retain information about the following:

- *Which workflow is being used*
- *Which part of the workflow is currently being viewed*
- *Current page number within the workflow*
- *Time range selected*

Bookmarks available to all users with unrestricted data access, meaning that should one analyst uncover a set of events that require more in-depth analysis, he can easily create a bookmark and turn over the investigation to another person.

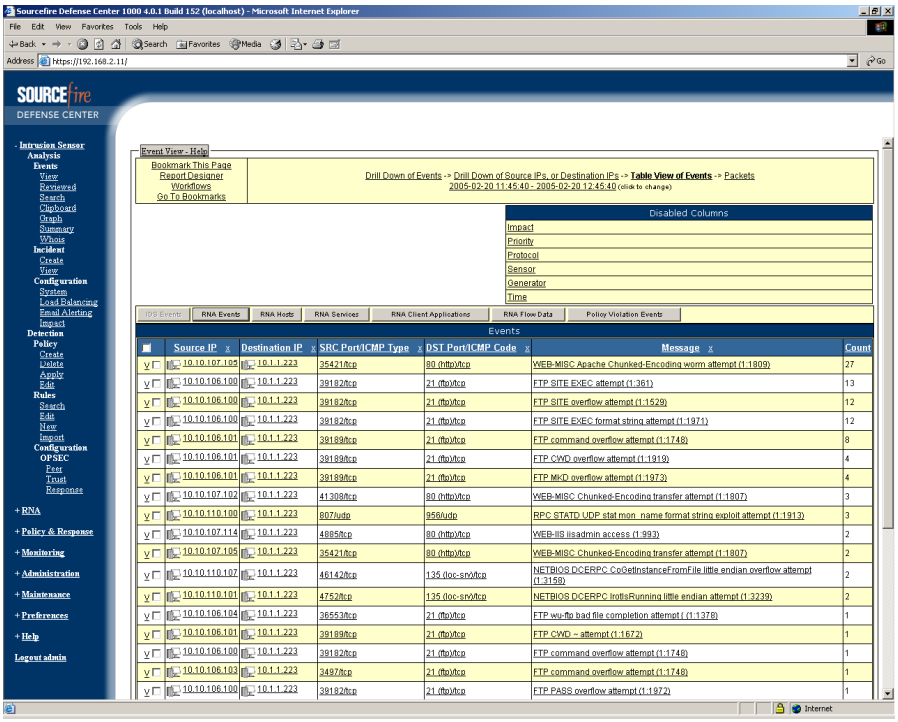

Figure 10 - Sourcefire 3D: Removing columns to summarise events

The analyst can also select and copy a range of events to the clipboard. He can generate a report for the events on the clipboard (up to a maximum of 100) just as he would from any of the event views.

In addition to building search criteria on the fly by drilling down into event data, the analyst can also use the *Intrusion Event Search Criteria* option to build customised search criteria from scratch. He can use a single search criterion like source IP address, or he can combine several criteria to narrow his search results. Commonly used search criteria can be saved to be used again and again without having to re-define the search (custom search criteria are tied to specific user accounts).

The following criteria can be used:

- ***Priority -*** *Returns events based on the value of the priority keyword in rules. Valid values are high, medium, and low.*
- ***Protocol -*** *Returns events based on the protocol of the packet.*
- ***Source IP -*** *Returns events based on the IP address of the sending host.*
- ***Destination IP -*** *Returns events based on the IP address of the receiving host.*
- ***Source/ Destination IP -*** *Returns events that have the IP address as either the source, the destination, or both.*
- ***SRC Port/ ICMP Type -*** *Returns events based on the source port or ICMP type*
- ***DST Port/ ICMP Code -*** *Returns events based on the destination port or ICMP code*
- ***Message -*** *Returns events based on all or part of the event message. It is also possible to search by generator ID and SID.*
- ***Reviewed -*** *Returns events based on whether the event has been marked as reviewed.*
- ***Impact -*** *For Intrusion Sensors in inline mode, returns events based on whether the packet was dropped as part of the event. Otherwise, the Impact value is based on the severity of the event coupled with the likelihood of that event succeeding (based on a vulnerability analysis by RNA)*
- ***Generator -*** *Returns events based on the component that generated the event.*
- ***Classification -*** *Returns events based on the classification to which the rule belongs.*

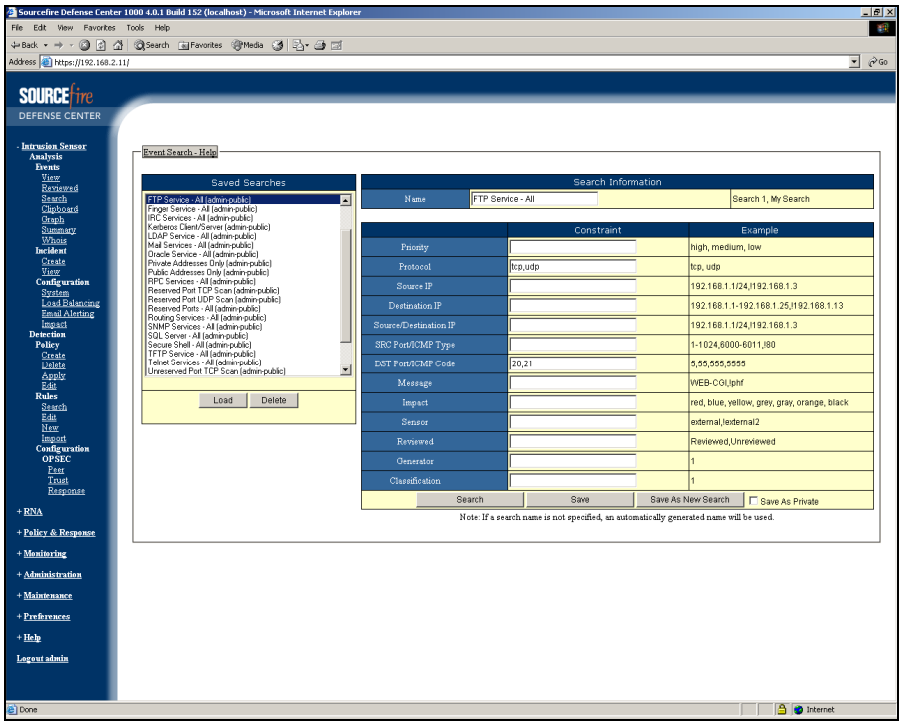

Figure 11 - Sourcefire 3D: Search filters for alert queries

Events which have been examined and determined to be of little or no interest or importance can be marked as *Reviewed*.

Events that are marked as *Reviewed* remain in the event database, but no longer appear in the event pages or as part of the event summary statistics. It is still possible, however, to examine events that were previously marked as *Reviewed*, as well as reinstate them to the normal event views by marking them as *Unreviewed*.

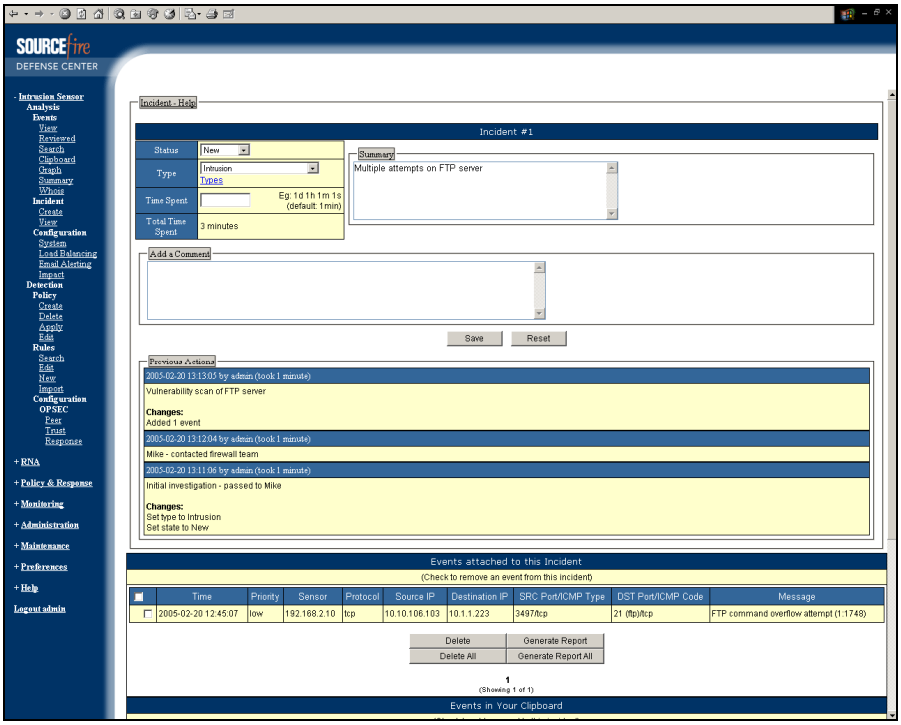

Figure 12 - Sourcefire 3D: Incident handling

As event analysts review the events and determine whether they are important in the context of their network, separate events (with their associated packet data) can be manually correlated and annotated into *Incidents*. These Incidents can be monitored, tracked and updated via the *Incident Handling* menu options.

For example, an incident can be used as a repository for notes about any activity that an analyst might take outside of Sourcefire 3D to mitigate the effects of the attack (perhaps noting that the infected hosts have been quarantined, that patches have been applied, and so on). Sourcefire 3D also supports an *incident life cycle*, allowing analysts to change an incident's status as they progress through their response to an attack.

Reporting and Analysis

The Sourcefire system is geared primarily towards incident handling and forensic analysis, and thus the usual high-level management-oriented graphical reports that can be found in other products have been omitted (trend reporting will be added in a future release).

Instead, Sourcefire 3D provides a flexible reporting system that can be used to generate a variety of event reports showing both intrusion events and audit events. We have already mentioned that reports can be built from the data that appears on one of the event views (a number of pre-defined Workflows provides an easy way to generate reports initially), generating defined reports against any subset of events in an event view or Workflow.

The parameters that were used to generate the current list of events are automatically reflected on the *Report Designer* page, following which they can be modified, if necessary, including the time range for the events included in the report and workflow pages used display the events. The finished report can be formatted as PDF, HTML, or as comma-separated values (CSV).

It is also possible to build a report profile from scratch on the *Report Designer* page. This can be saved for later use to generate the report on demand, or can be scheduled to run automatically through the scheduling facility. The analyst can define a variety of attributes for any report, including adding a company logo, defining the set of events that appear, and selecting the amount of detail contained in the report.

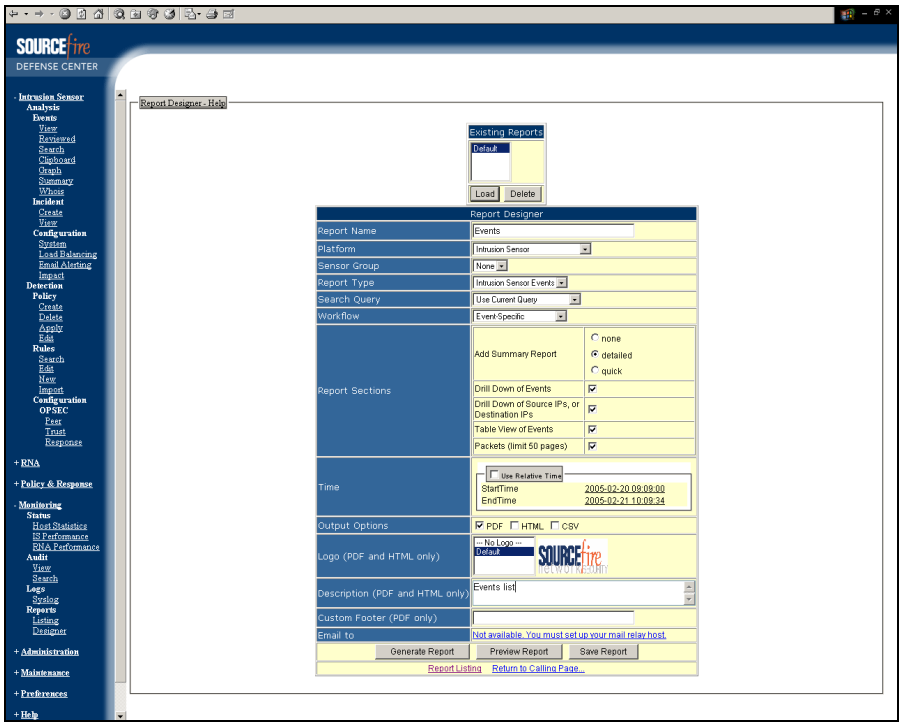

Figure 13 - Sourcefire 3D: Report Designer

Sourcefire 3D provides summary reports (*Quick Summary* or *Detail Summary*) for IDS event data which can be appended to any custom report by selecting the appropriate radio button in the report profile. Quick Summary reports can include up to 100 million events, and contain the following:

- *Pie chart showing the percentage of events in each event type (which maps to the rule category for the rule that generated the event)*
- *10 most active and 10 least active events*
- *Graph showing the number of events over time*
- *Pie charts showing the percentage of events by protocol (for example, TCP, UDP, or ICMP) and event classification (which maps to the value for the classtype keyword in the rule that generated the event)*
- *Tables listing the 50 most and least active events*
- *Tables listing the 50 most active source and destination ports*
- *Tables listing the 25 most active source and destination IP addresses and IP address combinations.*

Detailed Summary Reports can include up to 250 thousand events and contain the following:

- *Pie chart showing the percentage of events in each event type (which maps to the rule category for the rule that generated the event)*
- *10 most active and 10 least active events*
- *Graph showing the number of events over time*
- *Pie charts showing the percentage of events by protocol (for example, TCP, UDP, or ICMP) and event classification (which maps to the value for the classtype keyword in the rule that generated the event)*
- *Tables listing the 50 most and least active events*
- *Tables listing the 50 most active source and destination ports*
- *Tables listing the most active events for each of the 25 most active destination ports*
- *Tables listing the 25 most active source and destination IP addresses as well as the 25 most active source and destination IP combinations*
- *Tables listing the most active events for each of the 25 most active destination IP addresses*
- *Tables listing the most active events for the 25 most active source/ destination IP address combinations*

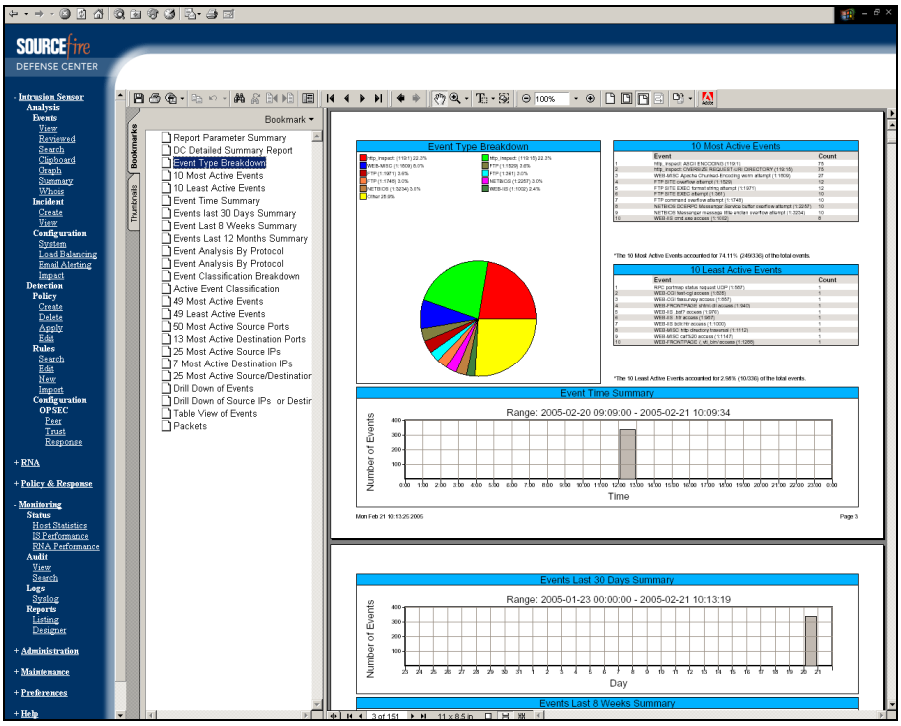

Figure 14 - Sourcefire 3D: Typical report

As well as the ability to preview reports on-screen, all reports are saved to the Defense Center system, and the *Report* page lists the previously generated reports. Each report is listed with the report name as defined in the report profile, plus the date and time the report was generated, and each one can be viewed on demand or downloaded to a local host.

Monitoring system processes, disk space, and performance is an important part of the daily administration of the Intrusion Sensor, and the following monitoring functions are provided:

- ***View Host Statistics -*** *view host statistics such as system uptime, disk and memory usage, and system processes.*
- ***Monitor System Status and Disk Space Usage -*** *view basic event and disk partition information.*
- ***View System Process Status -*** *view basic process status.*
- ***View Intrusion Sensor Performance Statistics -*** *view Intrusion Sensor performance statistics and generate graphs based on these statistics*
- ***Manage Audit Records -*** *view and manage system audit information*
- ***View the System Log -*** *view system status messages*

The Intrusion Sensor *Performance Statistics* page allows the administrator to generate graphs that display performance statistics for the Intrusion Sensor over a specific period of time. Graphs can be generated to display number of alerts per second, number of megabits per second, average number of bytes per packet, and the percent of packets dropped from the system. These graphs can show statistics for the last hour, last day, last week, or last month of operation.

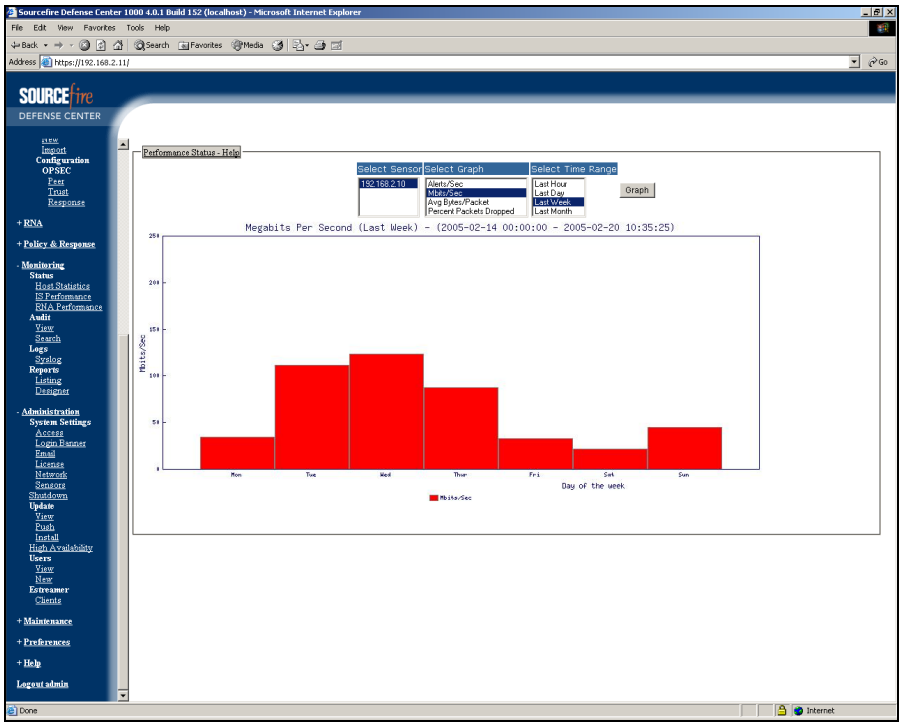

Figure 15 - Sourcefire 3D: Monitoring performance statistics

The Intrusion Sensor also logs read-only auditing information for user activity. Audit logs are presented in a standard event view that allows the administrator to view, sort, and constrain on audit log messages based on any item in the audit view.

RNA

As with intrusion detection security policies, the administrator can define RNA configuration policies that can be applied to multiple managed RNA Sensors via the Defense Center.

These are much more straightforward than Intrusion Detection policies, since they simply define which IP addresses or address ranges should be monitored, and what limits should be placed on the amount of data collected.

Once the policy has been applied, the RNA Sensors passively collect traffic travelling over the network, decoding data and then comparing it to established operating system and service fingerprints. They also collect flow data for all network sessions that involve at least one monitored host. This information is collected in RNA network discovery events that are transmitted to the Defense Center, and all discovered hosts and services reported by the events are mapped into a detailed representation of the network that can be viewed from the Defense Center interface. This is called the *Network Map*.

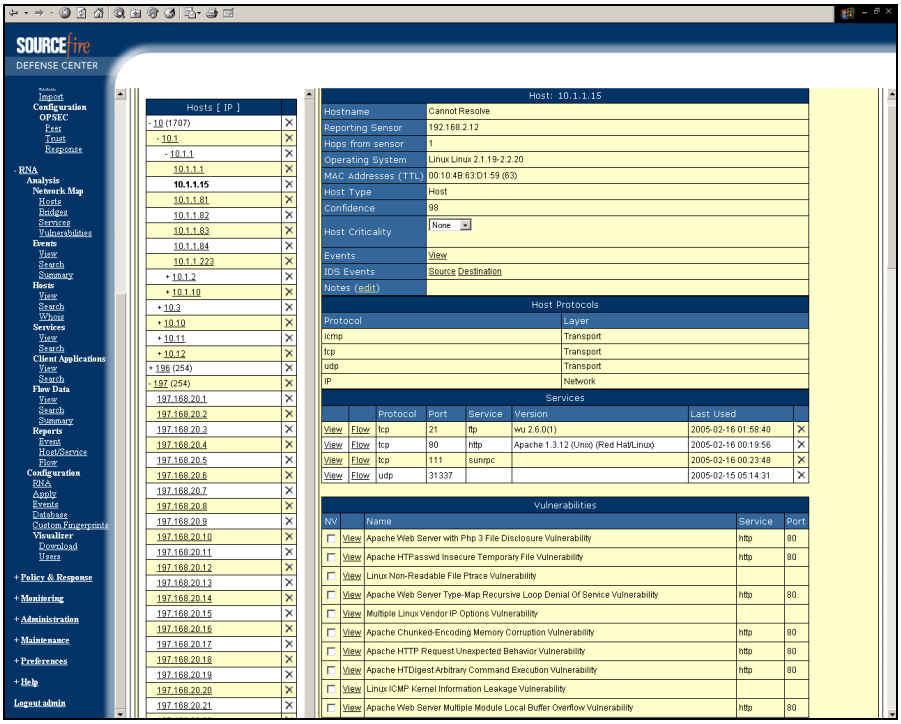

Figure 16 - Sourcefire 3D: RNA Network Map

The Network Map allows the analyst to view the network topology in terms of the hosts that reside on the network, the bridges on the network (which may include hubs, routers, or switches), the services running on the network, and the vulnerabilities on the network.

It is possible to delete entire subnets, specific hosts, service categories, specific services, or deactivate vulnerabilities from the network map in order to ignore them or watch for changes at a later date. Deleted services will be re-added to the network map if RNA detects a change in the service (for example, if an Apache web server changes versions).

Deactivated vulnerabilities will be re-activated when the operating system or service mapped to the deactivated vulnerability changes. Deleted hosts will be re-added to the network map if RNA is restarted and it detects host activity again. If there are specific subnets or hosts that should be permanently excluded from the network map, it is possible to exclude the subnet or IP address in the configuration policy for the RNA Sensor.

The network map is split into the following components:

- ***Hosts map -*** *displays network devices (such as computers, printers, and so on) detected on the network.*
- ***Bridges map -*** *displays bridges, switches, or routers detected on the network.*
- ***Services map -*** *displays all services detected on the network.*
- ***Vulnerabilities map -*** *displays all vulnerabilities detected on the network, organised by RNA ID, BugTraq ID, CVE ID, or Snort ID (SID).*

The Defense Center gathers data from all managed RNA Sensors and correlates the data into a composite Network Map. If multiple RNA Sensors generate events for the same host or service, the Defense Center combines the information from each sensor into a composite representation of that host or service.

From the Network Map, the analyst can view a list of hosts detected by RNA, organised by subnet, as well as viewing a detailed host profile for any listed host. Host profiles contain detailed information about hosts, running services, and vulnerabilities for specific hosts, and allow the analyst to modify host criticality, add notes associated with the host, and re-categorise vulnerabilities.

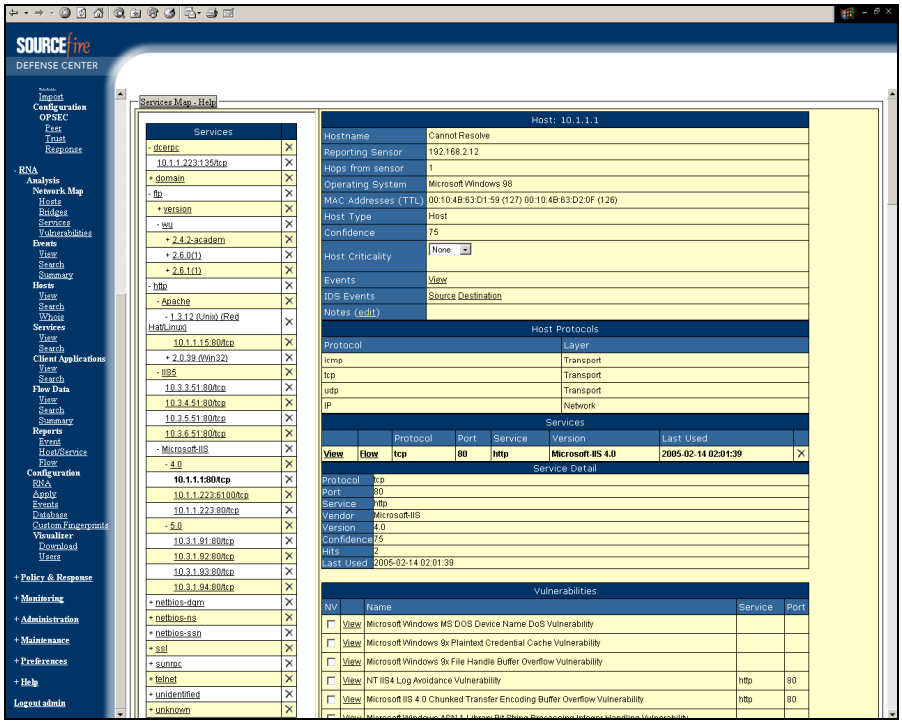

Figure 17 - Sourcefire 3D: Host Profile (via Service View)

Each Host Profile contains information such as:

- ***Hostname -*** *displays the fully-qualified domain name of the host, providing a valid DNS server has been added on the Network page. If DNS is not enabled, the host's IP address appears.*
- ***Reporting Sensor -*** *displays the name of the managed RNA Sensor that detected the host.*

- ***Hops from Sensor -*** *indicates the number of network hops that exist between the RNA Sensor that detected the host and the host itself, providing information about the physical location of the host.*
- ***Operating System -*** *displays the operating system that RNA detects in use on the host. If this field is blank, RNA has not yet identified an operating system. If this field is listed as "unknown," RNA is unable to identify the host.*
- ***MAC Addresses (TTL) -*** *displays the host's detected MAC address or addresses, with the current time-to-live (TTL) in parentheses.*
- ***VLAN Tag -*** *shows the Virtual LAN (VLAN) information associated with a host, if applicable. VLAN information helps the Defense Center generate more accurate network maps. RNA detects 802.1q VLAN tags and displays the following information for each:*
    - ***VID*** *- displays the VLAN ID, which indicates the VLAN of which that host is a member.*
    - ***Type*** *- displays the type of encapsulated packet containing the VLAN tag, which can be Ethernet or Token Ring.*
    - ***Priority*** *- displays the priority included in the VLAN tag.*
- ***Host Type -*** *indicates whether the network device is a bridge, router, or host:*
    - ***Bridges*** *can be bridges or switches that communicate using Cisco Discovery Protocol (CDP) or Spanning Tree Protocol (STP). If the bridge communicates using CDP, the bridge may have an IP address. If the bridge communicates using STP, it may only have a MAC address.*
    - ***Routers*** *have been detected as communicating using CDP or have otherwise been identified as a router. For example, if multiple hosts appear to be using the same MAC address, that MAC address is often identified as the router to which the multiple hosts are attached.*
    - ***Hosts*** *are network devices that have been identified but do not meet the criteria to be categorised as a bridge or a router.*
- ***Confidence -*** *displays the percentage of confidence (between 0% and 100%) that RNA has in the operating system identification of a specific host. For example, if RNA detects a Microsoft Windows 2000 Server running Apache for Windows and other Windows-related services, the confidence field will display a high percentage.*
- ***Host Criticality -*** *provides a way for the analyst to designate the business criticality of any given host. For example, the main mail server probably has a higher criticality to the business than a test server used by the Quality Assurance department. These host criticality values can then be used when searching for hosts or when creating and deploying compliance policies using Policy and Response.*
- ***Events -*** *provides a View link that allows the analyst to view all network discovery events associated with the host in the current time range.*
- ***IDS Events -*** *provides a link that allows the analyst to quickly access all intrusion events related to the host.*
- ***Notes -*** *provides a place to record information about the host which can be viewed by other analysts. For example, if there is a computer on the network that has an older, un-patched version of an operating system for testing purposes, the analyst may want to use the notes box to indicate that the system is intentionally un-patched.*

The analyst can click the *View* hyperlink next to any service, vulnerability or flow to access more information.

The *Bridges* view of the Network Map provides a complete view of the network bridges and routers that connect one segment of the network to another.

The *Services* view of the Network Map provides a complete view of the services running on the network (including unidentified services), the vendor and version of each service, and the hosts running each service.

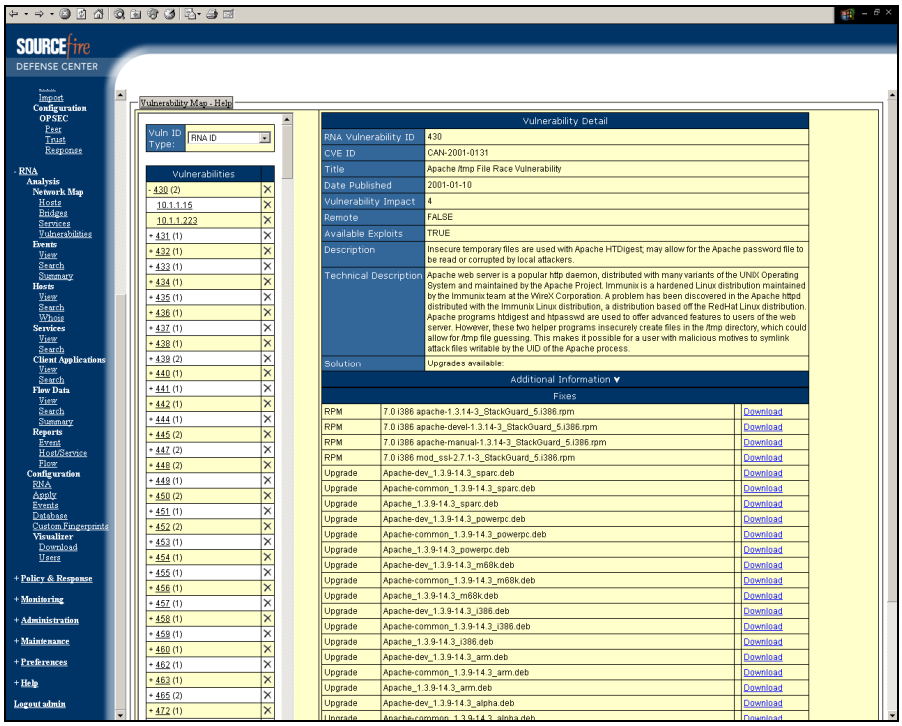

Figure 18 - Sourcefire 3D: Viewing vulnerability details

The *Vulnerabilities* view of the network map provides a view all of the vulnerabilities that RNA has detected on the network, organised by RNA ID, ArachNIDS ID, BugTraq ID, CVE ID, Nessus ID, or Snort ID. The vulnerabilities are arranged by identification number, with affected hosts listed beneath each vulnerability.

From the Vulnerabilities view it is possible to select a specific vulnerability to view its details, or select any host beneath a vulnerability to view the host's detailed profile, which allows the analyst to better evaluate the threat posed by the vulnerability on a specific affected host. The analyst can also deactivate vulnerabilities from the Network Map, meaning that they will no longer be used for intrusion impact correlation for any hosts that are currently vulnerable. Deactivated vulnerabilities appear in the *Not Applicable* vulnerability list for affected host profiles and can be re-activated at any time by moving them back into the active vulnerabilities list.

From each different view, the analyst can access the host profile of each host that is detailed within the view, and RNA events can be queried, viewed, bookmarked, copied to clipboard, drilled down via Workflows, and handled in the same way as normal IDS Events

RNA is ideal for policy enforcement and change management, since the RNA Sensor generates events that communicate the details of any change in a monitored network segment. New events are generated for newly discovered network assets, and change events for any change in previously identified network assets,

The first time a managed RNA Sensor runs, it generates new events for each host and any TCP or UDP services discovered running on that host. In addition, the RNA Sensor generates new events for each network and transport protocol running on each discovered host. After the initial network mapping is complete, the RNA Sensor continuously records network changes by generating change events. Change events are generated whenever the configuration of a previously discovered host or service changes, thus preventing users from adding unauthorised laptops or Wireless Access Points to a monitored network unnoticed.

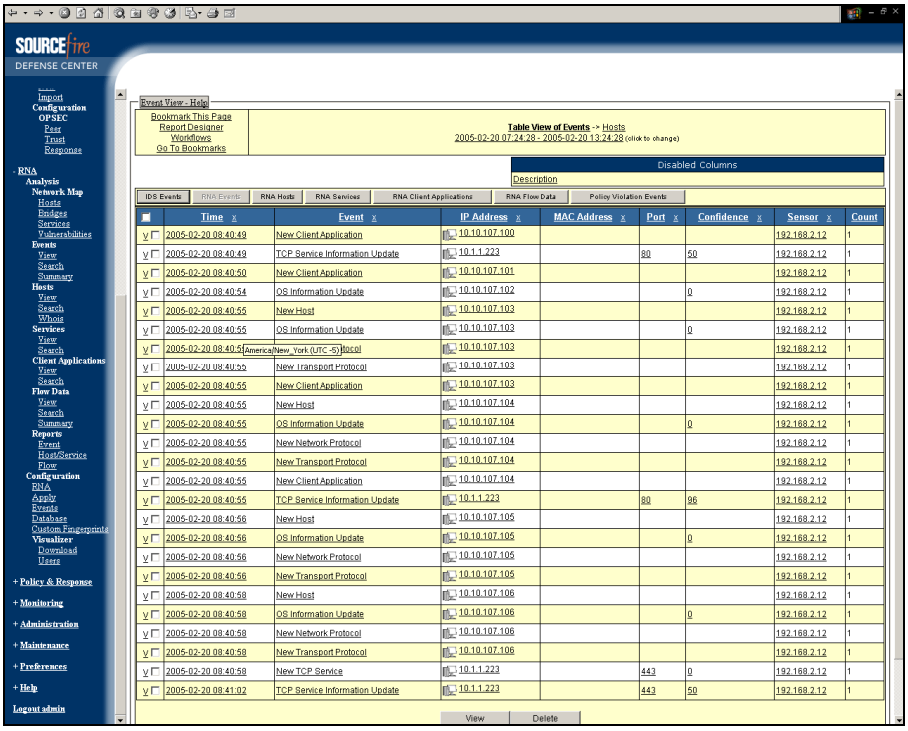

Figure 19 - Sourcefire 3D: Viewing RNA Events

Network discovery events are logged to the Defense Center and are displayed on the RNA Event View page. They are also used within compliance policies and can be configured to trigger *syslog*, *SNMP*, and *e-mail* notifications.

When a monitored host connects to another host, RNA can, in many cases, determine what application was used. RNA detects the use of many mail and web clients including Microsoft Internet Explorer, Microsoft Outlook and Outlook Express, Lotus Notes, Mozilla Thunderbird, Ximian Evolution, and more. For each client application event that is generated, RNA logs the IP address that used the application, the product, the version, and the number of times its use was detected. The analyst can use standard event views to display client application events and can use the search facility to quickly search for specific applications, if necessary.

RNA Sensors are also continuously monitoring traffic generated by each host that they are configured to detect. When an RNA Sensor detects that a connection between a monitored host and any other host is terminated, it generates a flow event. Flow events include information about the collected traffic, including the first packet of the transaction, the last packet of the transaction, source IP address and port, destination IP address and port, the number of packets and bytes sent and received by the monitored host, and the client application and URL involved in the transaction, if applicable.

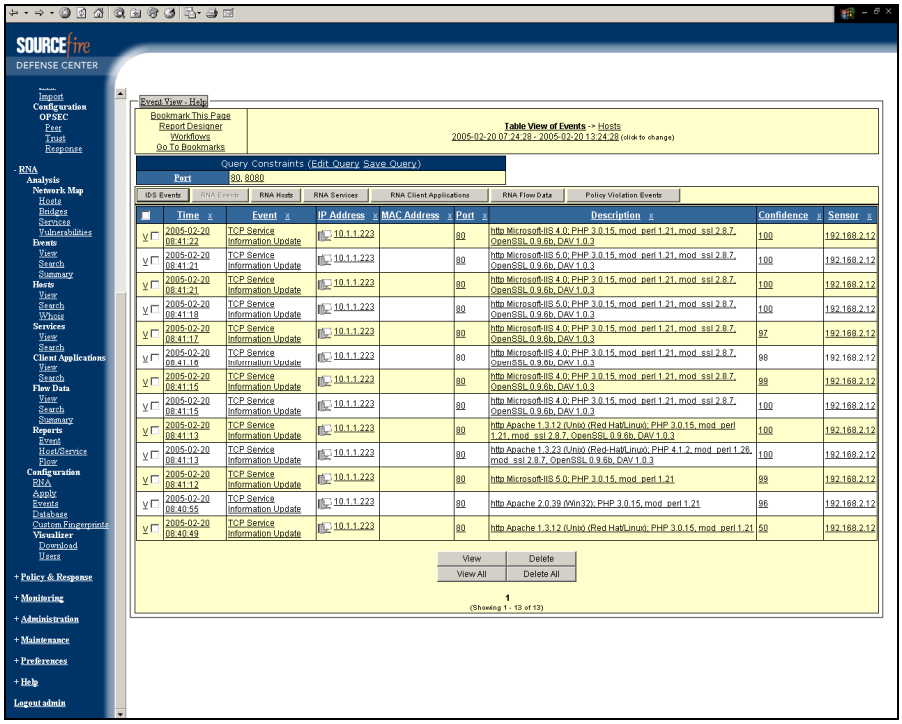

Figure 20 - Sourcefire 3D: RNA Flows

The analyst can search for flow data specific to a source or destination IP address, port, or protocol, and can create, save, and re-use flow data searches. In addition, RNA provides a number of default flow data searches that serve as examples (they can be duplicated and/or modified) and can provide quick access to important information about the flow data traversing the network. Default searches include:

- ***Possible Database Access search**, which searches for TCP traffic flows to ports that are commonly used by database servers.*
- ***Standard HTTP search**, which searches for TCP traffic flows to standard web server ports.*
- ***Standard Mail search**, which searches for TCP traffic flows to standard mail server ports.*
- ***Standard SSL search**, which searches for TCP traffic flows to port 443 (standard SSL port).*
- ***Unauthorised SMTP**, which searches for traffic to the standard SMTP port (25) on hosts that are not authorised SMTP servers from hosts inside the network.*

The *Flow Summary* page provides a list of the most active hosts on the monitored network segment organised by initiator and responder.

Finally, one of the most valuable uses for RNA is to correlate Intrusion Events with RNA data in order to more accurately define the *Impact Flag* for an Intrusion Event. For example, if the Impact Flag for a particular event is set to level 3 initially, but Defense Center determines from RNA data that the target host is running a service that is vulnerable to the exploit used, the Impact Flag is modified to a higher value accordingly.

However, should an Intrusion Sensor discover an attempted IIS exploit on the wire, but RNA has determined that the target host is a Linux box running Apache, the Impact Flag is lowered. If the target host is not running a Web server at all, the Impact Flag may be lowered even further.

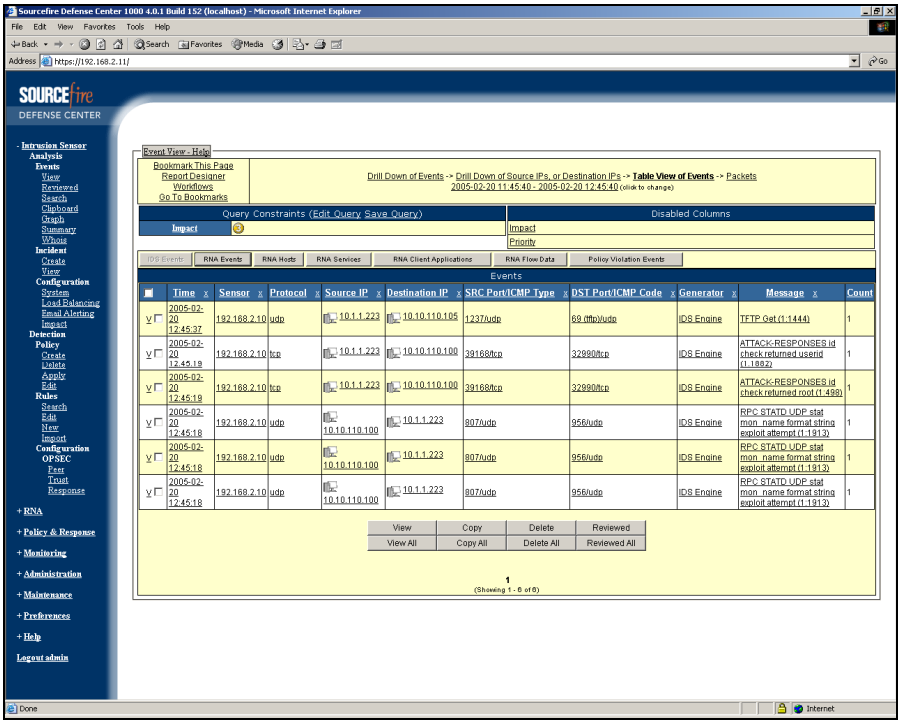

Figure 21 - Sourcefire 3D: Viewing alerts by Impact Flag

Thus, RNA adds valuable context to IDS intrusion events based on actual network knowledge, and therefore allows the administrator to prioritise his activities. Where RNA Sensors are running on a network, for example, the administrator can be sure that all alerts with an Impact Flag value of 1 (the highest) should be dealt with immediately, whereas those with a value of 5 can be left for a more leisurely analysis.

When combined with the *Host Criticality* field in particular, Sourcefire 3D is able to very accurately determine those hosts at the greatest risk.

Verdict

Performance

Performance under all but the most extreme load conditions was impeccable, with 100 per cent of all attacks being detected under all load conditions except the 10,000cps and 20,000cps HTTP tests (see *Test Results* section of this report).

In particular, the device turned in a flawless performance in all of our “real world” tests, and we would thus have no problem in rating the Sourcefire IS3000 as a true 1Gbps device under all normal network conditions.

The Sourcefire IS3000 performed consistently and mostly reliably throughout our tests. Exposing the sensor interface to an extended run of ISIC-generated traffic had no adverse effect, and the device continued to detect and log all other exploits throughout and following the ISIC attack.

Overall, performance was excellent under normal network conditions.

Security Effectiveness

Signature recognition was acceptable out of the box (79 per cent), and was increased to 91 per cent after the application of a signature pack update which was provided to us in under 48 hours.

The device did show some signs of weakness in our false negative and false positive tests, indicating that some signatures are detecting the *exploit* rather than the underlying *vulnerability*. However, it should be noted that recent improvements in the Snort rule “language” and efforts by Sourcefire’s signature-writing team have done a lot to improve this area.

A code update was required to rectify some issues with FTP traffic normalisation, following which the IS3000 collected a clean sheet across the board in our evasion tests. *Fragroute, Whisker, ADMmutate, running exploits on non-standard ports* and even *RPC record fragging* all failed to trick Sourcefire into ignoring valid attacks. In addition, all but one of them was decoded accurately.

Out of the box, the Sourcefire IS3000 handled 1 million open connections easily, and the device appeared to be immune to stateless attack replay tools such as *Stick* and *Snot*.

Usability

Although navigating the Sourcefire 3D Defense Center interface can be daunting initially, once the initial learning curve has been mastered the UI provides one of the most powerful and flexible we have seen - particularly when it comes to forensic analysis and reporting.

The *Workflow* concept is an excellent idea, allowing the analyst to determine exactly how he would like to process events and drill down to get more detail. At the same time, when at the more detailed levels, the ability to group events by any column of data on the fly allows the analyst to re-summarise the events in almost any way imaginable. At any point during his analysis, he is then able to generate bookmarks that can be shared with others, or custom reports for his own use.

The ability to mark events as *reviewed* or group them together and annotate them as *Incidents* adds a further dimension to the Sourcefire 3D system.

Sensor and policy management are also extremely straightforward, and very scalable. Policies can be applied to single or multiple sensors at the click of a mouse, and it is always easy to determine which policies have been modified and need to be re-applied.

The “crown jewels” of Sourcefire 3D come in the shape of RNA which provides much needed context to intrusion alerts via the use of extensive passive scanning and vulnerability analysis. This allows Defense Center to accurately determine the impact on a particular host of a particular vulnerability and set the *Impact Flag* accordingly. This should greatly enhance the analyst’s ability to focus on exactly those alerts which pose the biggest security concerns on his network, and this should go a long way to removing one of the biggest perceived “problems” with IDS systems - analyst overload from too many irrelevant alerts.

Throughout our testing, this feature worked impeccably, and is a very compelling reason on its own to purchase this product.

Contact Details

Company name: Sourcefire, Inc.

E-mail: sales@sourcefire.com

Internet: www.sourcefire.com

Address:
9212 Berger Road,
Suite 200,
Columbia,
MD 21046

Tel: +1 800 917 4134 or +1 410 290 1616
Fax: +1 410 290 0024

APPENDIX A – TEST RESULTS

The aim of this procedure (based on V3.0 of the NSS Group IDS Testing Methodology) is to provide a thorough test of all the main components of a Gigabit IDS device in a controlled and repeatable manner, and in the most "real world" environment which it is possible to simulate in a test lab.

The Test Environment

The network is 100/1000Mbit Ethernet with CAT 5e cabling and Cisco Catalyst 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. There is no firewall protecting the target network.

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.

All "normal" network traffic, background load traffic and exploit traffic crossing the switches is mirrored to **two** SPAN ports on the Catalyst 6503 (the same traffic is mirrored to both simultaneously). The sensor's detection interface is connected to one SPAN port, whilst an Adtech network monitoring device monitors the same mirrored traffic via the second SPAN port to ensure that the total amount of traffic never exceeds 1Gbps (which would invalidate the test run).

The sensor is bound to the Gigabit network interface in "stealth mode" wherever that is supported (i.e. no IP address) and a separate interface is used to connect the sensor 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 logging 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 product, this test attempts to demonstrate how accurately the sensor detects and logs a wide range of common exploits, port scans, and Denial of Service attempts. All exploits are run with no load on the network and no IP fragmentation.

Our attack suite contains over 100 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.

The "**default**" *Attack Recognition Rating* (ARR) is expressed as a percentage of detected 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**" ARR is determined. This demonstrates how effective the vendor is at responding to a requirement for new or updated signatures.

Both the *default* and *custom* ARR 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.

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 device attains a “PASS” for each test case if it does **not** raise an alert. 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.

- ***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 logging 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 a number of common basic attacks (our baseline suite) in their normal state, with no evasion techniques applied.

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

Test 2.2 - Packet Fragmentation and Stream Segmentation

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

- ***Test 2.2.1 -*** *IP fragmentation - ordered 8 byte fragments*
- ***Test 2.2.2 -*** *IP fragmentation - ordered 24 byte fragments*
- ***Test 2.2.3 -*** *IP fragmentation - out of order 8 byte fragments*
- ***Test 2.2.4 -*** *IP fragmentation - ordered 8 byte fragments, duplicate last packet*
- ***Test 2.2.5 -*** *IP fragmentation - out of order 8 byte fragments, duplicate last packet*
- ***Test 2.2.6 -*** *IP fragmentation - ordered 8 byte fragments, reorder fragments in reverse*
- ***Test 2.2.7 -*** *IP fragmentation - ordered 16 byte fragments, fragment overlap (favour new)*
- ***Test 2.2.8 -*** *IP fragmentation - ordered 16 byte fragments, fragment overlap (favour old)*
- ***Test 2.2.9 -*** *TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with invalid TCP checksums*
- ***Test 2.2.10 -*** *TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with null TCP control flags*
- ***Test 2.2.11 -*** *TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with requests to resync sequence numbers mid-stream*
- ***Test 2.2.12 -*** *TCP segmentation - ordered 1 byte segments, duplicate last packet*

- ***Test 2.2.13 -*** *TCP segmentation - ordered 2 byte segments, segment overlap (favour new)*
- ***Test 2.2.14 -*** *TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with out-of-window sequence numbers*
- ***Test 2.2.15 -*** *TCP segmentation - out of order 1 byte segments*
- ***Test 2.2.16 -*** *TCP segmentation - out of order 1 byte segments, interleaved duplicate segments with faked retransmits*
- ***Test 2.2.17 -*** *TCP segmentation - ordered 1 byte segments, segment overlap (favour new)*
- ***Test 2.2.18 -*** *TCP segmentation - out of order 1 byte segments, PAWS elimination (interleaved dup segs with older TCP timestamp options)*
- ***Test 2.2.19 -*** *IP fragmentation - out of order 8 byte fragments, interleaved duplicate packets scheduled for later delivery*
- ***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 detected and an alert raised in **any** form, and (ii) 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 detected and an alert raised in **any** form, and (ii) 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 detected and an alert raised in **any** form, and (ii) 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 across the network 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).

- ***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 log new exploits when the 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" network) then opens various numbers of TCP sessions from 10,000 to 1,000,000 (one million) with the Spirent Reflector (on the "internal" network) and the sensor will be expected to track each of these legitimate sessions. 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 device 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 freshly-launched exploits once all the connections are open.

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 logging 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 - 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 3.3 - 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 - State Preservation:*** *As Test 3.2.3 following tuning*

Section 4 – Detection Performance Under Load

The aim of this section is to verify that the sensor is capable of detecting and logging 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 log** suspicious traffic - no session-termination techniques are employed (i.e. RST packets from the sensor).

Our "attacker" host launches a fixed number of exploits at a target host on the subnet being monitored by the sensor. The Adtech network monitor is configured to monitor the *same* traffic on a second 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 across the network 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 Detection Rate* (ADR) at each background load is expressed as a percentage of the number of exploits detected by the sensor against the number verified by the Adtech network monitor and target host.

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 across the network protected by the sensor. 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 throughout each test. Multiple tests are run and averages taken where necessary.

This traffic does not attempt to simulate any form of “real world” network condition, and the aim of this test is purely to determine the raw packet processing capability of the sensor, and its effectiveness at passing “useless” packets quickly in order to pass potential attack packets to the detection engine.

- ***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 logging 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% detection 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 – Stability & Reliability

These tests attempt to verify the stability of the device under test under various extreme conditions.

- ***Test 5.1.1 - 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 to the external network, and the ISIC target is located on the internal network protected by the sensor. ISIC traffic is transmitted across the network 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 logging exploits throughout the test to attain a PASS.*

Section 6 – 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 6.1 - 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 6.1.1 - Open ports:*** *We will scan the open ports and active services on the management interface and report on known vulnerabilities.*
- ***Test 6.1.2 - 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 sensor, and that interface is also the target. ISIC traffic is transmitted to the management interface of 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 (a) operational and capable of detecting and logging exploits, and (b) capable of communicating in both directions with the management server/console throughout the test to attain a PASS.*
- ***Test 6.1.3 -*** *We note whether the ISIC attacks themselves are detected by the sensor even though targeted at the management port.*

Sourcefire IS3000 V4.0.2 Test Results

Section 1 - Detection Engine

Test 1.1 – Attack Recognition	Attacks	Default ARR	Custom ARR
Test 1.1.1 - Backdoors	7	6	6
Test 1.1.2 - WINS/DNS	3	2	3
Test 1.1.3 - DOS	10	6	10
Test 1.1.4 - False negatives (modified exploits)	14	9	10
Test 1.1.5 - Finger	4	3	4
Test 1.1.6 - FTP	5	4	5
Test 1.1.7 - HTTP	43	37[1]	39[1]
Test 1.1.8 - ICMP	2	2	2
Test 1.1.9 - Reconnaissance	8	8	8
Test 1.1.10 - RPC	9	6	9
Test 1.1.11 - SSH	1	1	1
Test 1.1.12 - Telnet	1	1	1
Test 1.1.13 - Database	1	1	1
Test 1.1.14 - Mail	1	1	1
Test 1.1.15 - Voice	1	0	0
Total	**110**	**87 / 110[1]**	**100 / 110[1]**
		79%[1]	**91%[1]**

Test 1.2 – Resistance to False Positives	Default	Custom
Test 1.2.1 - Suspicious FTP traffic	PASS	PASS
Test 1.2.2 - HTTP “exploit” using incorrect method	FAIL	PASS
Test 1.2.3 - Retrieval of Web page containing “suspicious” URLs	PASS	PASS
Test 1.2.4 - Simple SMTP QUIT command	PASS	PASS
Test 1.2.5 - Normal NetBIOS copy of “suspicious” files	PASS	PASS
Test 1.2.6 - Normal NetBIOS traffic	PASS	PASS
Test 1.2.7 - POP3 e-mail containing “suspicious” URLs	PASS	PASS
Test 1.2.8 - POP3 e-mail with “suspicious” DLL attachment	PASS	PASS
Test 1.2.9 - POP3 e-mail with “suspicious” Web page attachment	PASS	PASS
Test 1.2.10 - SMTP e-mail transfer containing “suspicious” URLs	PASS	PASS
Test 1.2.11 - SMTP e-mail transfer with “suspicious” DLL attachment	PASS	PASS
Test 1.2.12 - SMTP e-mail transfer with “suspicious” Web page attachment	PASS	PASS
Test 1.2.13 - SNMP V3 packet with invalid parameter	PASS	PASS
Test 1.2.14 - Fake DNS /bin/sh buffer overflow	PASS	PASS
Test 1.2.15 - Inter-firewall communication traffic	PASS	PASS
Test 1.2.16 - Fake SQL Slammer traffic	FAIL	PASS
Test 1.2.17 - File copy of GIF file (contains bytes which look like NOP sled)	PASS	PASS
Total Passed	**15 / 17**	**17 / 17**

Section 2 - IPS Evasion

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

Test 2.2 – Packet Fragmentation/Stream Segmentation	Detected?	Decoded?
Test 2.2.1 - IP fragmentation - ordered 8 byte fragments	YES	YES
Test 2.2.2 - IP fragmentation - ordered 24 byte fragments	YES	YES
Test 2.2.3 - IP fragmentation - out of order 8 byte fragments	YES	YES
Test 2.2.4 - IP fragmentation - ordered 8 byte fragments, duplicate last packet	YES	YES

Test 2.2.5 - IP fragmentation - out of order 8 byte fragments, duplicate last packet	YES	YES
Test 2.2.6 - IP fragmentation - ordered 8 byte fragments, reorder fragments in reverse	YES	YES
Test 2.2.7 - IP fragmentation - ordered 16 byte fragments, fragment overlap (favour new)	YES	YES
Test 2.2.8 - IP fragmentation - ordered 16 byte fragments, fragment overlap (favour old)	YES	YES
Test 2.2.9 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with invalid TCP checksums	YES	YES
Test 2.2.10 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with null TCP control flags	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
Test 2.2.12 - TCP segmentation - ordered 1 byte segments, duplicate last packet	YES	YES
Test 2.2.13 - TCP segmentation - ordered 2 byte segments, segment overlap (favour new)	YES	YES
Test 2.2.14 - TCP segmentation - ordered 1 byte segments, interleaved duplicate segments with out-of-window sequence numbers	YES	YES
Test 2.2.15 - TCP segmentation - out of order 1 byte segments	YES	YES
Test 2.2.16 - TCP segmentation - out of order 1 byte segments, interleaved duplicate segments with faked retransmits	YES	YES
Test 2.2.17 - TCP segmentation - ordered 1 byte segments, segment overlap (favour new)	YES	NO
Test 2.2.18 - TCP segmentation - out of order 1 byte segments, PAWS elimination (interleaved dup segments with older TCP timestamp options)	YES	YES
Test 2.2.19 - IP fragmentation - out of order 8 byte fragments, interleaved duplicate packets scheduled for later delivery	YES	YES
Test 2.2.20 - TCP segmentation - ordered 16 byte segments, segment overlap (favour new (Unix))	YES	YES
Total	**20 / 20**	**19 / 20**

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

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

Section 3 - Stateful Operation

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

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 - State Preservation	PASS	PASS	PASS	PASS	PASS	PASS	PASS

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

Section 4 - Detection/Blocking Performance Under Load

Test 4.1 – UDP traffic to random valid ports	250Mbps	500Mbps	750Mbps	1Gbps	Max
Test 4.1.1 - 256 byte packet test - max 453,000pps	100%	100%	96%	74%	715Mbps
Test 4.1.2 - 550 byte packet test - max 220,000pps	100%	100%	100%	100%	1Gbps
Test 4.1.3 - 1000 byte packet test - max 122,000pps	100%	100%	100%	100%	1Gbps

Test 4.2 – HTTP "maximum stress" traffic with no transaction delays	250Mbps	500Mbps	750Mbps	1Gbps	Max
Test 4.2.1 - Max 2500 connections per second - ave packet size 1000 bytes - max 120,000 packets per second	100%	100%	100%	100%	1Gbps
Test 4.2.2 - Max 5000 connections per second - ave packet size 540 bytes - max 225,000 packets per second	100%	100%	100%	100%	1Gbps
Test 4.2.3 - Max 10000 connections per second - ave packet size 440 bytes - max 275,000 packets per second	100%	100%	100%	96%	900Mbps
Test 4.2.4 - Max 20000 connections per second - ave packet size 360 bytes - max 320,000 packets per second	100%	100%	70%	N/A[4]	650Mbps

Test 4.3 – HTTP "maximum stress" traffic with transaction delays	250Mbps	500Mbps	750Mbps	1Gbps	Max
Test 4.3.1 - Max 5000 connections per second - ave packet size 540 bytes - max 225,000 packets per second - 10 sec delay - max 50,000 open connections	100%	100%	100%	100%	1Gbps
Test 4.3.2 - Max 10000 connections per second - ave packet size 440 bytes - max 275,000 packets per second - 10 sec delay - max 100,000 open connections	100%	100%	100%	N/A[4]	825Mbps

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

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

Section 5 - Stability & Reliability

Test ID	Result
Test 5.1.1 - ISIC/ESIC/TCPSIC/UDPSIC/ICMPSIC	PASS

Section 6 - Management Interface

Test ID	Result
Test 6.1.1 - Open Ports	PASS
Test 6.1.2 - ISIC/ESIC/TCPSIC/UDPSIC/ICMPSIC	PASS
Test 6.1.3 - ISIC attacks detected against management interface?	NO

Notes:

1. Signatures for a further 3 exploits were available, but these were for server-client exploits and were disabled for performance reasons (would have achieved 94% total recognition)
2. These failed out of the box and required a code update to pass. Problem has been fixed in current shipping version
3. Alerts on mid-flow pick-ups (configurable - can enforce state/3-way handshake)
4. Not possible to run this test - erratic results - limit is 13000cps
5. Not possible to run this test - erratic results - limit is 16000cps

Section 1: Detection Engine

We installed one IS3000 sensor with the latest signature pack reporting to a single Defense Center server. We used a modified version of the default policy, which had **all** attack signatures enabled apart from a small number of server-to-client signatures which are disabled by default for performance reasons. We also enabled some key audit-only signatures. Apart from the few performance-related (server-client) signatures which Sourcefire recommends remain disabled in high-performance environments, the default policy also omits all chat, shellcode, P2P, policy enforcement and informational signatures.

Signature recognition was acceptable out of the box (79 per cent), and was increased to 91 per cent after the application of a signature pack update which was provided to us in under 48 hours.

We noted a few test cases raised multiple alerts for a single exploit. Our “false negative” (modified exploit) cases also posed some problems, even following the signature update, indicating that many of the Snort signatures are still detecting the exploit rather than the underlying vulnerability. However, it should be noted that recent improvements in the Snort rule “language” and efforts by Sourcefire’s signature-writing team have done a lot to improve this area.

A major concern in deploying an IDS is the raising of false alarms. We noted two false positive alerts from our test suite, both of which were fixed following the signature update. We also noted some “noise” from our standard network traffic traces (which contain nothing but basic NetBIOS and DNS traffic). As with most devices of this type, we would recommend deploying in a live network to determine how it responds to live traffic before purchasing.

Section 2: IPS Evasion

Resistance to known evasion techniques was excellent following a code update to rectify some issues with FTP traffic normalisation. Once these had been fixed, Sourcefire collected a clean sheet across the board in our evasion tests. *Fragroute, Whisker, ADMmutate, running exploits on non-standard ports* and even *RPC record fragging* all failed to trick Sourcefire into ignoring valid attacks.

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

Section 3: Stateful Operation

Out of the box, the Sourcefire IS3000 handled 1 million open connections easily. When resources are low, old sessions are aged out as required - this behaviour is not configurable.

The IS3000 appeared to be immune to stateless attack replay tools such as *Stick* and *Snot*.

Section 4: Detection/Blocking Performance Under Load

The Sourcefire IS3000 was tested up to 1Gbps, the rated speed of the appliance.

Performance under all but the most extreme load conditions was impeccable, with only the 10,000cps and 20,000cps HTTP tests proving too much for it to handle (the maximum possible being around 9000cps and 13000cps respectively before packet loss occurred). This appears to be as a result of an occasional context-switching problem when under excessive load which can peg one CPU at 100 per cent and cause loss of packets until it recovers.

However, the IS3000 turned in a flawless performance in all of our “real world” tests, and we would thus have no problem in rating the Sourcefire IS3000 as a true 1Gbps device under all normal network conditions.

Section 7: Stability & Reliability

The Sourcefire IS3000 performed consistently and mostly reliably throughout our tests.

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

We did manage to cause some performance problems with the database during this test by disabling the default alert threshold causing the generation of several million alerts. Once the database high-water mark of 1 million was reached, constant “pruning” of database entries (almost as fast as they were being added), caused loss of management console response. However, after increasing the high-water mark to 4 million and reinstating the default alert threshold we had no more problems. Overall, it should be noted that we found database performance to be excellent, even when under heavy load.

Section 6: Management Interface

Open ports on the management interface are restricted to 443 for HTTPS traffic (required for the web-based UI), 8300 and 8303 for internal management traffic with the Defense Center, 8301 for event traffic to the Defense Center, and 22 for SSH (for command line management).

Access control rules can be put in place to ensure that connections are made from known management server(s) only. Once in place, port scans failed completely from any other PC on the management network.

The extended ISIC attack against the management interface had no effect on the appliance and its ability to detect and log attacks, and there was only a slight delay in communicating with the management server. However, no alerts were raised at any time to inform the administrator that the management port was under attack.

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