

## **Intrusion Detection: Forensic Computing Insights arising from a Case Study on SNORT**

*Vlasti Broucek, Paul Turner*

*School of Information Systems, University of Tasmania*

### **About the authors**

*Vlasti Broucek, MSc has been working in the computer industry since 1985 in various research and industry positions. Currently, he is a researcher at the School of Information Systems, University of Tasmania and a network administrator at the School of Psychology, University of Tasmania, Australia. His current research interest is in Forensic Computing – Legal and Technical Issues. He has an extensive knowledge of systems administration on a range of systems (Novell Netware, Microsoft Windows and UNIX flavours), computer security, mathematics and Expert Systems. He has an MSc degree from the Czech Technical University in Prague where he worked with Professor Vladimír Mařík, DrSc and currently is pursuing a PhD at the University of Tasmania under leadership of Paul Turner, PhD and Professor Chris Keen, PhD. His contribution to the field of Forensic Computing was recognised this year by being selected as a chair of Professional Clinic at EICAR2003 conference and co-chair of IT Law track at the same.*

*Mailing address: School of Information Systems, Private Bag 87, Hobart TAS 7001, Australia; Phone: +61-3-62262346; Fax: +61-3-62262883; E-mail: Vlasti.Broucek@utas.edu.au*

*Prior to joining the School of Information Systems, Dr. Paul Turner (Senior Research fellow) was a research fellow at CRID (Computer, Telecommunications and Law Research Institute) in Belgium where he worked on a variety of European Commission contracts in the field of electronic commerce, telecommunications and intellectual property rights. Paul has also worked as an independent information and telecommunications consultant in a number of countries in Europe and was for three years editor of the London-based monthly publication Telecommunications Regulation Review. Paul's strong research focus in the field of electronic commerce has continued in his work as senior research fellow at the University of Tasmania. Paul was also for two years Research Manager for the Tasmanian Electronic Commerce Centre ([www.tecc.com.au](http://www.tecc.com.au)). In Paul's work for the TECC he was responsible for coordinating research at basic, applied and strategic levels across a range of industry sectors with a focus on small to medium sized enterprises and their electronic business practices. This research assisted the TECC in refining the advice, products and services it continues to deliver to Tasmanian businesses.*

*Mailing address: School of Information Systems, Private Bag 87, Hobart TAS 7001, Australia; Phone: +61-3-62266240; Fax: +61-3-62266211; E-mail: Paul.Turner@utas.edu.au*

continued on page 2

## **Descriptors**

*Alert, False-Positive, Forensic Computing, Intrusion Detection, Monitoring, Privacy, SNORT*

**Reference** to this paper should be made as follows:

Broucek, V. & Turner, P. (2003). Intrusion Detection: Forensic Computing Insights Arising from a Case Study on SNORT. In U.E. Gattiker (Ed.), EICAR Conference Best Paper Proceedings (ISBN: 87-987271-2-5) 19 pages. Copenhagen: EICAR.

## **Intrusion Detection: Forensic Computing Insights arising from a Case Study on SNORT.**

### **Abstract**

*As the dangers of hacking and cyber-warfare for network security become a reality, the need to be able to generate legally admissible evidence of criminal or other illegal on-line behaviours has become increasingly important. While technical systems providing intrusion detection and network monitoring are constantly being improved, the security they provide is never absolute.*

*As a result, when assessing the value and nature of the data these systems produce, it becomes critical to be aware of a number of factors: these systems themselves are susceptible to attack and/or evasion (Arona, Bruschi, & Rosti, 1999; Handley, Paxson, & Kreibich, 2001; Ptacek & Newsham, 1998); these systems may only collect a partial data set; and, these data sets may themselves be flawed, erroneous or already have been tampered with (Broucek & Turner, 2002b). Additionally, the issue of privacy and data protection is emerging as a central debate in forensic computing research (Broucek & Turner, 2002b).*

*In this context, this paper provides a detailed case study on the use of the SNORT intrusion detection system (IDS) on a university department World Wide Web (WWW) server. The case study is analysed and discussed using a forensic computing perspective. This perspective considers the nature of the intrusion detection and network monitoring security provided and evaluates the system in terms of its evidence acquisition ("forensic") capabilities, the legal admissibility of the digital evidence generated and privacy implications of intrusion detection systems and network monitoring.*

### **Introduction**

In the "information age" alongside exciting opportunities for activities such as e-business and e-learning, there is an awareness of the serious risks posed by malware, hacktivism and information-warfare. A fundamental cause of many of these risks is the variety of ways that individuals and/or groups can now utilise digital technologies to engage in inappropriate, criminal or other illegal online activities. While it can be assumed that much of this online behaviour continues to go undetected, when it is identified the need for evidence and proof becomes of paramount importance. Although technical advances in the ability of systems to detect intrusions, denial of services attacks, other forms of attacks and to enhance network monitoring and maintenance are well documented and subject to constant research and development, they are rarely evaluated from this "forensic computing" perspective. Forensic computing having previously been defined as "the process of identifying, preserving, analysing and presenting digital evidence in a manner that is legally acceptable" (McKemmish, 1999). Although these advanced systems provide network administrators with the tools to treat some of the symptoms of such illegal and/or inappropriate on-line behaviours, they are not designed with the need to track, trace and generate legally admissible evidence about these behaviours. Additionally, there is a

growing awareness that most computer security systems can easily be tainted and their evidence contaminated. As an example, it is well-known that there are numerous hacks for *wtmpx*, *utmpx* and other built-in system log files that are often used as a source of information on system events (Farmer & Venema, 1993) and papers have been published describing techniques used to avoid Intrusion Detection Systems (Handley et al., 2001; Ptacek & Newsham, 1998).

This paper provides a case study on the network traffic data collected by SNORT (Roesch, 1999, 2001a, 2001b) network intrusion detection system (NIDS) on a University department World Wide Web server over a two-month period. At this point, it is necessary to stipulate that this case study is based on the data collected by running SNORT as a part of "normal" business operations, although some adjustments have been made to ensure consistent data. Further, it is not the intention of this paper to evaluate the SNORT IDS. No comparisons are being made concerning the relative merits of SNORT against any other free or commercially available IDS systems and all the conclusions made in this paper would apply to IDS in general. The case study is analysed and discussed from a forensic computing perspective. This perspective considers the nature of the intrusion detection and network monitoring security provided and evaluates the system in terms of its evidence acquisition ("forensic") capabilities and the legal admissibility of the digital evidence generated. It also discusses privacy implications that the deployment of IDS creates and illustrates them with examples from the collected data. The SNORT system was selected primarily because of its easy availability (i.e. being free, open source software) and its widespread use in Universities, public institutions and many large commercial organisations.

This paper is divided into three parts. Part one describes the hardware and software set-up used in the two-month trial. Part two provides details of the basic statistics on the data including the number of alerts (false positive alarms and attacks) during the two-month period and diary of significant events during collection of the data. Part three presents an analysis and discussion of the case study data and concludes this paper by summarising key forensic and privacy insights arising from the case study.

## Part One: Set Up

The experiment was conducted on a production server operated by one of the departments at the University. This server is currently used for providing web services and non-anonymous file transfer protocol (ftp) access for updating departmental antivirus software clients. It is running on dedicated hardware unit.

### Hardware

The server is based on an "off-the-shelf" Intel Celeron 300 computer with 256MB RAM and 3GB IDE hard drive. The server is currently connected to 100 Mbit/sec port on a Cisco 2900 series switch that is connected to the University backbone. The server is located inside the University's network that contains more than 10,000 registered hosts. This network is protected against unauthorised access from the Internet by Cisco PIX firewalls.

During the experiment access to the server from the Internet was initially possible only on TCP/IP port 80 (http protocol) and after 30 days access was also possible on port 443 (https protocol). The server is not located in the demilitarised zone (DMZ) and as a result the computers connected to the internal University network have full access to the server. Although this is not an ideal situation because of an associated increase in security risks, it is very typical of University departments' WWW servers.

## Software

The server is using the Sun Solaris 8 operating system. This operating system was completely patched on 31 July 2002 and the case study experiment was conducted during the period 1 August to 30 September 2002. To maintain the integrity of the collected data sets, no further OS and application software patching was done during the period of the experiment.

In relation to the software set-up used during the experiment standard security precautions based on AusCERT recommendations (Australian Computer Emergency Response Team, 2001a, 2001b; Smith & Indulska, 2001) were made prior the data collection. Among these TCP Wrappers (Venema, 1992) were installed and configured to limit access to services run on the server. Non-essential services, for example, telnet and tftp, were disabled. The web server was for the first 30 days of the experiment running Apache 1.3.26 http daemon ("Apache HTTP Server," 2002) with Microsoft FrontPage 2002 server extensions. The MS FrontPage extensions are used for web authoring as well as for collecting data from students using built in form tools. Subsequently, the server was extended to support encrypted SSL connections using mod\_ssl version 2.8.10 ("mod\_ssl," 2002) and openssl version 0.9.6g ("openssl," 2002). The ftp server is using the standard ftp daemon provided in Solaris 8, with TCP Wrappers limiting access to this server from two particular class B sub-networks within the University. Anonymous access to ftp server is not installed at all. This ftp server provides updates for McAfee antivirus software used in the department. An in-house developed Perl based program contacts McAfee's master site twice daily and checks for the availability of software updates. If the updates are available, it downloads them and sends e-mail to the administrator. All departmental IBM-PC hosts (around 150) running various versions of Microsoft Windows are set up to contact this server every morning between the hours of 06:00 and 08:00, to check for the availability of update files and download them. The only additional access to the server is through ssh. The server runs ssh daemon version 3.2 ("ssh," 2002) and access to ssh daemon is again limited using TCP Wrappers.

Three "name based" (see Apache documentation at <http://httpd.apache.org/doc/>) virtual web servers are running on the server. Two of these servers are accessible from the Internet on port 80. After thirty days of the experiment one of these two Internet-connected virtual servers was also made accessible on port 443 using Secure Socket Layer (SSL) for encryption. The third virtual server uses ".htaccess" method to limit access from within the University networks only. However, being a "name based" server and sharing numerical TCP/IP addresses with other two virtual servers, it is in some sense "visible" to the Internet.

Although SNORT (Roesch, 1999, 2001a, 2001b) is typically used as a network intrusion detection system (NIDS), it has been set up and installed directly on the web-server in this department. There are several reasons for this including that the primary focus was in monitoring only this particular server. After several months of previous experiments with two different versions of SNORT on both UNIX and MS Windows platforms, version 1.8.7 was installed on 31 July 2002 on this server and started with a default, up-to-date rule set on 1 August 2002. The only modifications made to SNORT's default configuration files were to define "home" and "external" networks, and to instruct SNORT to collect all traffic in "tcpdump" format. Due to the fact that the University's environment can be considered "hostile", the "external" network was set up as "any network" with the exception of two "trusted" hosts maintained and used solely by the authors of the paper.

In analysing the traffic collected, two common free (under GNU licence) tools – tcpdump ("tcpdump/libpcap," 2002) and Ethereal ("ethereal," 2002) programs were used in addition to SNORT's own outputs. These tools were used on both UNIX based and Windows 2000 based workstations to eliminate possible bugs. Results received on both platforms were identical; however, small but from forensic computing perspective significant differences were observed between the two tools. In particular, the time resolutions used and the different amounts of detail provided in the outputs.

## Part Two: The Experiment

The following basic statistics were produced over the two-month experimental trial. The first alert was triggered at 00:33:43 EST (GMT+10) on 1 August 2002 and the last alert of the trial was triggered at 18:30:19 EST on 30 September 2002. There were 2669 alerts triggered during the trial coming from 301 different sources with 41 different signatures. After taking out sources of false positive alarms within the University network, 1457 alarms remained. Out of these, 540 (more than 33%) were found to come from domains within Denmark. The hosts triggering the alerts were mainly using ADSL connections (as their FQDN suggests) and there is a well-founded suspicion that many of these attacks were launched automatically from infected machines and the owners/users of these computers were not aware of what was happening on their hosts. A further 408 alarms were from hosts with TCP/IP addresses without domain name service (DNS) entries. These hosts' TCP/IP addresses did not resolve to host names, hence it can be presumed that these hosts were not properly registered with their Internet Services Provider (ISP), or ISP did not provide correctly DNS services.

One particular form of attack was most prevalent. It was the exploitation of the Microsoft IIS server via the "cmd.exe" weakness. Of the total, 592 (22.18%) alerts were of this form of attack. Again, most of these came from hosts based in Denmark. Other forms of attacks that were noted included 83 (3.11%) attempts using "WEB-FRONTPAGE autor.exe" access, 60 (2.25%) attempts using "WEB-IIS ISAPI .ida" and 46 (1.72%) using "WEB-IIS CodeRed v2 root.exe" accesses. (Readers are encouraged to visit computer security related sites to find out more about these attacks. Some examples of good sites to visit are <http://www.snort.org/>, <http://www.whitehats.com/>, <http://www.uscert.org.au/>, <http://www.sans.org/> and many more.) These statistics clearly illustrate that the majority of attempted attacks tried to exploit known weaknesses of the Microsoft IIS server. This is in-

spite of the fact that the server being used was running Apache web server software that would be very easy to query before trying to make an attack. Those attackers trying to exploit these weaknesses consequently only succeeded in leaving data about themselves. This leads to the preliminary conclusion that many of these attacks were either initiated by so called "script kiddies" or launched automatically by infected computers.

## Diary of the Experiment

### Day one and two

The first day of running SNORT on the server illustrated that Intrusion Detection Systems present many problems as reliable sources of intrusion alarms. To ensure they operate at their optimal levels requires highly trained network professionals and considerable efforts in fine-tuning the system. In the first several hours of being on-line SNORT produced a large amount of alarms due to regular network traffic because its default rule set triggered low level alarms on many types of such traffic. The source hosts for these alarms were from all around the University network but mainly from the central Information Technology Services (ITS) department. This department provides a range of services that would regularly give reasons for alarms to be triggered including DNS, BOOTP/DHCP and SNMP network management.

In responding to these alarms it was necessary to adjust the system configuration files as well as the rule sets by making decisions on whether the hosts identified as the sources of these alarms should be treated as "trusted" hosts or not. As a result of this fine-tuning ITS hosts were treated as "trusted" and this immediately reduced the number of false positive alarms. However, in the opinion of the authors this is not an ideal situation when fine-tuning IDS. Although the majority of these hosts were dedicated hosts managed by well-trained administrators, some of these administrators may have a lack of understanding of several security issues related to services run on their hosts. This in-turn may lead to undetected security breaches.

One finding came from an analysis of regular alerts triggered by one of the central hosts broadcasting "rusers" requests. Because "r" commands are considered one of the least secure commands in the UNIX world, alarms concerning these activities were raised by both TCP Wrappers and SNORT. It has been found that these requests were generated by the CiscoWorks suite used by the University for managing its Cisco based network. This suite regularly tries to do "rusers" queries on every host known to it. It is surprising that Cisco is still using something that has been considered to be insecure and a source of information leaking.

Further sources of alerts were triggered by a variety of other hosts around the University. One alarm was triggered by hosts trying connection on UDP port 161 (SNMP) using a default "public" string. This was of particular interest due to recent recognition of the vulnerability in SNMP protocol implementations. Investigation of these alarms discovered that they were triggered by HP JetAdmin and HP WebAdmin suites installed on these hosts. These suites can be set up to look for HP printers in regular intervals and then to check each printer in their databases regularly. Again, a decision had to be made as to

whether these hosts should be “trusted” or not. It was found that it is impossible to trust these hosts because in some instances they were not dedicated servers, for example, an MS Window NT 4 workstation used by a postgraduate student and an MS Windows 2000 Server that was not being maintained by an IT professional. This particular W2K server was running a default installation that had not been patched for several months and was as a result vulnerable to several security issues current at the time.

Day two brought more false positive alarms as well as alerts indicating attempted attacks exploring the web server’s vulnerabilities from the Internet. These attempts mainly focused on known vulnerabilities in Microsoft IIS server. These attempts were futile, because of the Apache software running on the server. In one particular case a host based in Denmark, continued to attempt gaining unauthorised access to the web server three times a day over several days.

The first alarm of the second day was triggered by NOP86 rule on a connection from a host with a non-functioning reverse DNS lookup. Furthermore, this alarm was triggered on a connection initiated by the server that runs IDS. Further investigation revealed that it was an ftp connection to Network Associates site initiated from the server for McAfee antivirus software updates - these updates having been found and downloaded by the server. The question as to why this activity triggered an alarm is surprisingly simple. One of the virus signatures in the zipped file exactly matches the NOP86 attack signature. This gave rise to a dilemma - was it possible to declare the McAfee server to be a “trusted” host? The decision was made to do so, however, the alarm was triggered again in the afternoon during a second update triggered by a different host. Unfortunately, Network Associates provides (quite understandably) updates from a server farm where TCP/IP numbers can be different every time. The question remains why McAfee does not have a properly set up reverse DNS lookup. At the time of writing this paper, Network Associate’s master site for McAfee antivirus software ftp.nai.com resolves to 161.69.201.237 and 161.69.201.238. However, reverse lookup on both addresses does not resolve back.

Similar problems arising from incorrect reverse DNS lookup were observed later in relation to one visiting web crawler/spider. In this case, the research section of one of the Blue Chip IT companies running the crawler was contacted and the issue was resolved very quickly. The crawler was based behind the company’s firewall and was being assigned a dynamic TCP/IP address without a proper DNS entry.

During the second and subsequent days more false positive alerts were triggered. Many of them remain unexplained as yet – for example why Netware 5.1 servers quite regularly trigger “Possible evasive RST” alerts; why filling in an MS FrontPage form on the web server and submitting it triggers several MS FrontPage vulnerability alerts. One possible explanation of this last point may be that SNORT’s “default” rule set is not detailed enough to recognise this form of FP extensions usage as legitimate and for reasons stated before it was impossible to consider departmental computers in the labs as trusted.

### **Day three to thirty**

For the rest of the month data were collected and no further adjustments were made to the rule sets or SNORT's configuration to ensure comparable data was collected under the same rule set and configuration. It was also decided not to update the rule sets or configuration during the remainder of the experiment for any newly discovered security threats. Such a decision would be totally unacceptable in a commercial environment; however, in the context of this trial it was acceptable because the departmental web server does not hold any commercially valuable data and mechanisms are in place to ensure that data and server can be restored quickly without any losses.

One bug in SNORT version 1.8.7 was identified during this period. Its analysis and implications are discussed further in this paper.

### **Day thirty one**

The server was extended to provide for the secure socket layer (SSL) capability, providing the duplicate information from one of the virtual servers on port 443 (https) in addition to standard port 80 (http). The reason for this was to prove that signature based intrusion detection systems would not be able to detect any attacks conducted over SSL. An internal host was used to try to exploit several known security issues. As expected and quite understandably, none of the attacks were detected by SNORT. However, it was possible to observe them in the web server's log files.

### **Day thirty two to sixty**

Monitoring continued in the same fashion again without any updates or further tuning of the system and further data were collected. Several hosts on the Internet discovered that the server was running also on port 443 and several attempts on that port were recorded in the log files of the server. However, as expected, none of these attempts were detected by SNORT. Indeed, again as expected, even detailed analysis of traffic data using ethereal and tcpdump did not provide any more information than that there was encrypted traffic flowing between two hosts.

## **Part Three: Analysis and Discussion**

One of the most significant findings from this two-month experiment was a bug in the reporting/alerting part of the SNORT system. While the system correctly raised alarms on one particular rule set, the report it produced was not correct (Appendix A shows the alert message and the analysis of this bug). From a network administrators' or security officers' point of view, this bug is not very significant because they would be able to identify a potential problem and then examine the captured network packets using tcpdump and/or ethereal tools to provide the correct information. However, from a forensic computing perspective this bug could hamper the admissibility and/or validity of evidence derived from the captured data. The discrepancy between text of the alert and real data (illustrated in figures 1-5, Appendix A), supports the view that IDS' are problematic as tools for the collection of forensic computing data sets (Broucek & Turner, 2002a, 2002b; Sommer, 1998, 1999). This bug was fixed in SNORT version 1.9.0, however another bug, this time

in formatting alert output files was discovered. The new bug prevents some of the tools developed for SNORT log files manipulation from correct operation (eg SnortSnarf).

The two months of monitoring the server provided large amounts of data. As a matter of fact, due to storage limitations full "tcpdump" format log file could not be collected for the whole period. The experiment has illustrated that rule sets and configuration files have to be highly customised and continuously fine-tuned. Decisions about "trusted hosts" have to be made often on the basis of partial knowledge. It has further been shown that many hosts that should potentially be granted "trusted status" due to the service provided cannot be granted such a status due to the nature of other problems, for example with DNS lookups. The data reveals that attempted attacks were made both from inside the University Network and the Internet. The server was probed not only on ports that have been made available and known (the server is advertised on University's website), but also on ports that were not readily "visible" to users of the University network or the Internet. On ports 80 and 443, probes mainly attempted to exploit known vulnerabilities in Microsoft IIS server. The persistence of these repeated attempts suggests that many of these attacks were automated and the scripts not written very well. It is questionable why a "would be attacker" risks being discovered by repeating these attacks on a system that is not susceptible to such attacks. This is especially the case given that it is possible for them to identify first the server software and then make decisions about whether to attempt an attack. While external attacks from the Internet were only experienced on TCP ports 80 and 443 due to firewall restrictions, internal attempts were much broader. Again, some of the attacks were completely futile for the attacker but provided valuable data for a forensic investigator wishing to identify the location or intention of the attacker.

Using the IDS system can help to protect the web server against malicious attacks. However, the amount of false positive alarms, fine tuning and alert monitoring required definitely proves that properly trained security personnel/system administrator needs to be available to monitor such a system continuously. It has proved useful to collect all the traffic in the "tcpdump" format. This is particularly the case because it opens up the possibility of the data later being examined by other tools, in this case ethereal and tcpdump. Clearly, it is very hard, if not impossible, to rule out all of the false positive alarms based only on information obtained from alert file produced by SNORT. However, with careful analysis of the whole connection it is possible to prove beyond a reasonable doubt that the traffic is either malicious or false positive alarm. However, even such data would have only limited value in prosecuting potential attackers as the information provided in the "tcpdump" format file is limited to the TCP/IP packet load. Additional information needs to be obtained from other sources – log files, route tracing etc.

As an example of this point (see Appendix A), one particular repeating alert in the data collected during the experiment highlights that such data does not provide sufficient information for tracking back and tracing the attacker or source of the alarms. In this case, an attack exploiting vulnerabilities of NetBIOS name service and Domain Name Service on UDP ports 137 and 53 appeared to be initiated inside the University network. This triggered "ICMP Destination Unreachable" alerts. It appeared that the "ICMP Destination Unreachable" packets were being sent to our network due to a host on our network attempting this particular attack. The TCP/IP address of the "offending" host pointed to a

Hewlett-Packard printer located on our network. However, checking the firewall log files and firewall set up proved that the attack was not initiated inside of the University's network. The conclusion reached was that the attacker, somewhere on the Internet, was using a "decoy" technique and we were unlucky enough to be in the "decoy" address space. When the attacker tried to attack the site that had port 53 blocked, our network was receiving a broadcast ICMP message.

To be completely sure that there is not something going wrong, the host supposedly causing the messages that generated the alarms was shut down. The alerts continued to be triggered in nearly regular 12 hours intervals. Administrators of the router sending the ICMP messages were contacted and asked for help in locating the attacker. While they did not respond, the messages stopped coming after several days, even with our host back on-line. At the time when this paper was written, the messages started to come yet again, from a slightly different router, however at the same domain. Furthermore, as described previously and analysed in Appendix A, this particular alert message suffered from a bug in the used version of SNORT.

This experiment also confirms that SNORT NIDS is not capable of protecting against attacks conducted over the secure socket layer (SSL). The only way of detecting these attacks is to examine carefully server log files. The encrypted traffic simply does not provide patterns that could be used for creating rules for attacks run through SSL. These results suggest that it might be highly desirable to develop a module for Apache server software that would provide "application based" intrusion detection. With the availability of rule sets (either from SNORT or any other public domain source), application programming interface (API) and in particular the greater modularity of Apache version 2.x this should be a relatively straightforward module to develop.

Finally, these findings support calls being made by governments and law enforcement agencies for continuous monitoring of network traffic. However, to protect individuals' privacy it is necessary to develop techniques for this monitoring that will not interfere with the individual's rights.

## **Conclusion: Forensic Insight**

By adopting a forensic computing perspective this paper has confirmed concerns raised about the suitability of Intrusion Detection Systems as sources for evidence acquisition (Sommer, 1998, 1999). Specifically, two main concerns previously raised by (Broucek & Turner, 2002b) have been validated by this case study:

- IDS systems may collect only a partial data set – it has been shown that data collected by SNORT were not sufficient to track and trace attackers and that the data collected in the case of encrypted communication using SSL had no value in this regard;
- The data sets collected may be flawed, erroneous or already have been tampered with – it has been shown that even with significant fine-tuning SNORT produced numerous false positive alerts and/or wrongly identified the source of attacks.

More significantly, the case study has highlighted that from a legal perspective IDS, the data they produce and how that data is analysed pose numerous challenges for those interested in evidence acquisition that produces legally admissible evidence. In particular, even where data has been captured the process of its technical analysis may invalidate it in terms of legal admissibility by “tampering” with the evidence. It is then imperative that in approaching the collection of forensic evidence that there is awareness of some key principles: minimise handling of the original data set; account for any change; comply with the rules of evidence; and, do not exceed your knowledge (McKemmish, 1999).

The case study has also highlighted that the “collect everything” approach is highly desirable but has severe limitations and implications:

- Appendix B provides an analysis of one, minor FTP session. It shows how much information investigators with access to such data can obtain from it. Clear text transactions are easily visible and, source and destination hosts are easily identifiable. If SNORT was deployed in its usual configuration as a network IDS, it would collect heaps of data that could clearly violate the privacy rights of academics, students and other network and Internet users.
- More generally the log files are huge in size even where SNORT is monitoring only one rather minor and uninteresting web server. All alerts were collected into one ‘flat’ file and several binary “tcpdump” files were also collected. The longest time period collected into one file was about 193 hours – during which 664848 packets were captured and the size of the file was 157MB. To read this file into Ethereal with only MAC and transport name resolutions switched on took 3 minutes and 50 seconds on a Pentium IV 1.7GHz system running RedHat Linux 8.0. It took even longer on Sun Ultra 5 running Solaris 8 that was used for these analyses. Requests for filtering or “following TCP stream” took even longer. This clearly demonstrates that in a case of serious deployment, large storage space is needed and that an effective log rotation system has to be developed to allow quick and timely analysis.

Finally, the experiment confirms that Intrusion Detection Systems play important role in protecting Information Systems infrastructure. However, to be effective they require attention of highly trained security personnel/system administrators in:

- regular monitoring and analysing alerts and other log files and acting upon them,
- continuous fine-tuning the configuration files and rule sets for the ever changing IS environment, and
- regular updating the rule sets to accommodate newly discovered threats.

## Appendix A

This appendix provides an analysis of one particular alarm that revealed a bug in SNORT version 1.8.7. The same method was used to analyse all other interesting packets during the course of the trial. The real TCP/IP addresses have been replaced with addresses in form of “XXX.XXX.XXX.XXX” where they were in “dotted” form and in form of “XX XX XX XX” where they were in hexadecimal form. MAC addresses were left intact. It is believed it would be nearly impossible to identify involved hosts from their MAC addresses.

```
[**] [1:485:2] ICMP Destination Unreachable (Communication Administratively Prohibited) [**]
[Classification: Misc activity] [Priority: 3]
08/17-05:15:22.029533 XXX.XXX.XXX.XXX -> YYY.YYY.YYY.YYY
ICMP TTL:242 TOS:0x0 ID:54572 IpLen:20 DgmLen:56
Type:3 Code:13 DESTINATION UNREACHABLE: ADMINISTRATIVELY PROHIBITED,
PACKET FILTERED_
** ORIGINAL DATAGRAM DUMP:
YYY.YYY.YYY.YYY:0 -> ZZZ.ZZZ.ZZZ.ZZZ:0
UDP TTL:126 TOS:0x0 ID:7936 IpLen:20 DgmLen:71
Len: 51
** END OF DUMP
```

Figure 1. Alarm Raised by SNORT

Figure 1 shows the alarm that was raised on 17 August 2002 at 05:15:22.029533. This alert was most probably triggered by a host on the Internet using “decoy” techniques during scanning for vulnerabilities. However, interestingly SNORT claims that there was communication from port zero to port zero in the original datagram dump part. This is clearly erroneous. To find out what was really happening, tcpdump and ethereal were used to view the actually captured packet. Ethereal’s GUI interface unfortunately does not provide for easy copying and pasting of output to documents, hence its less user-friendly “command line based” brother called tethereal was used for dumps presented in this paper.

First the packets were printed in hexadecimal/ascii format. This is the simplest possible format to print, however its analysis requires significant knowledge of packet structure, theory of TCP/IP networking and all the protocols involved. Both tools provide simple decoding of the packet together with the hex/ascii dump. Also note time resolution differences between figures 1-5.

```
Frame 226 2002-08-17 05:15:22.0295 XXX.XXX.XXX.XXX -> YYY.YYY.YYY.YYY ICMP Destination unreachable
0000 00 30 c1 0a 82 6b 00 04 9b 2f e3 fc 08 00 45 00 .0...k.../....E.
0010 00 38 d5 2c 00 00 f2 01 ca 48 XX XX XX XX YY YY .8.,.....H.n....
0020 YY YY 03 0d 55 6e 00 00 00 00 45 00 00 47 1f 00 #...Un....E..G..
0030 00 00 7e 11 e1 7f 83 d9 23 06 ZZ ZZ ZZ ZZ 00 89 ..~.....#..=....
0040 00 35 00 33 a6 93 93 a6 4b 71 .5.3....Kq
```

Figure 2. Hexadecimal output using tethereal

Figure 2 represents the output from ethereal. Bytes at addresses 0x003E and 0x003F represent the source port of the original datagram (0x0089 = 137 - NetBIOS Name Service) and bytes 0x0040 and 0x0041 destination port of original datagram (0x0035 = 53 - Domain Name Server).

```
05:15:22.029533 0:4:9b:2f:e3:fc 0:30:c1:a:82:6b 0800 74: XXX.XXX.XXX.XXX > YYY.YYY.YYY.YYY: icmp:
host ZZZ.ZZZ.ZZZ.ZZZ unreachable - admin prohibited filter
```

```

0x0000      4500 0038 d52c 0000 f201 ca48 XXXX XXXX E..8.,.....H.n..
0x0010      YYYY YYYY 030d 556e 0000 0000 4500 0047 ..#...Un....E..G
0x0020      1f00 0000 7e11 e17f 83d9 2306 ZZZZ ZZZZ ....~.....#..=..
0x0030      0089 0035 0033 a693 93a6 4b71          ...5.3....Kq

```

Figure 3. Hexadecimal output using tcpdump

Similarly, Figure 3 is an output from tcpdump. It outputs slightly differently by taking out of the hexadecimal output the link-level header. To see it, “-e” option was used and it is displayed at the top. In this case the source and destination ports are at bytes 0x0030-0x0031 and 0x0032-0x0033 respectively. What is significant is that the numbers are the same with both tools, 137 and 53 respectively, although SNORT was reporting both as 0. To confirm this direct, “hands-on” analysis, more advanced options were used to decode the packets to obtain the more details directly from the tools. Ethereal and tcpdump produced again same results for source and destination ports of original datagram reported back with ICMP message.

```

Frame 226 (74 bytes on wire, 74 bytes captured)
  Arrival Time: Aug 17, 2002 05:15:22.029533000
  Time delta from previous packet: 695.795558000 seconds
  Time relative to first packet: 231551.206677000 seconds
  Frame Number: 226
  Packet Length: 74 bytes
  Capture Length: 74 bytes
Ethernet II, Src: 00:04:9b:2f:e3:fc, Dst: 00:30:c1:0a:82:6b
  Destination: 00:30:c1:0a:82:6b (HEWLETT-_0a:82:6b)
  Source: 00:04:9b:2f:e3:fc (Cisco_2f:e3:fc)
  Type: IP (0x0800)
  Trailer: 93A64B71
Internet Protocol, Src Addr: XXX.XXX.XXX.XXX (XXX.XXX.XXX.XXX), Dst Addr: YYY.YYY.YYY.YYY
(YYY.YYY.YYY.YYY)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    0000 00.. = Differentiated Services Codepoint: Default (0x00)
    .... ..0. = ECN-Capable Transport (ECT): 0
    .... ...0 = ECN-CE: 0
  Total Length: 56
  Identification: 0xd52c
  Flags: 0x00
    .0.. = Don't fragment: Not set
    ..0. = More fragments: Not set
  Fragment offset: 0
  Time to live: 242
  Protocol: ICMP (0x01)
  Header checksum: 0xca48 (correct)
  Source: XXX.XXX.XXX.XXX (XXX.XXX.XXX.XXX)
  Destination: YYY.YYY.YYY.YYY (YYY.YYY.YYY.YYY)
Internet Control Message Protocol
  Type: 3 (Destination unreachable)
  Code: 13 (Communication administratively filtered)
  Checksum: 0x556e (correct)
  Internet Protocol, Src Addr: YYY.YYY.YYY.YYY (YYY.YYY.YYY.YYY), Dst Addr: ZZZ.ZZZ.ZZZ.ZZZ
(ZZZ.ZZZ.ZZZ.ZZZ)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    0000 00.. = Differentiated Services Codepoint: Default (0x00)
    .... ..0. = ECN-Capable Transport (ECT): 0
    .... ...0 = ECN-CE: 0
  Total Length: 71
  Identification: 0x1f00
  Flags: 0x00
    .0.. = Don't fragment: Not set
    ..0. = More fragments: Not set
  Fragment offset: 0

```

```
Time to live: 126
Protocol: UDP (0x11)
Header checksum: 0xe17f (correct)
Source: YYY.YYY.YYY.YYY (YYY.YYY.YYY.YYY)
Destination: ZZZ.ZZZ.ZZZ.ZZZ (ZZZ.ZZZ.ZZZ.ZZZ)
User Datagram Protocol, Src Port: netbios-ns (137), Dst Port: domain (53)
Source port: netbios-ns (137)
Destination port: domain (53)
Length: 51
Checksum: 0xa693
```

Figure 4. Verbose output using tethereal

Output from ethereal in much more readable format is seen in Figure 4. This is arguably most detailed view of a packet one can get with tools easily/freely available.

```
05:15:22.029533 XXX.XXX.XXX.XXX > YYY.YYY.YYY.YYY: icmp: host ZZZ.ZZZ.ZZZ.ZZZ unreachable - admin
prohibited filter for YYY.YYY.YYY.YYY.137 > ZZZ.ZZZ.ZZZ.ZZZ.53: 37798 updataA+$ [b2&3=0x4b71]
[1537a] [1579q] [513n] [11013au][|domain] (ttl 126, id 7936, len 71) (ttl 242, id 54572, len 56)
```

Figure 5. Most verbose output using tcpdump

The most “verbose” or detailed output from tcpdump is in Figure 5. It doesn’t give as many details as ethereal, but all necessary details are clear.

## Appendix B

This appendix illustrates privacy implications arising from an analysis using ethereal on a full “tcpdump” format log file.

```

220 XXX.XXX.XXX.XXX FTP server (SunOS 5.8) ready.
USER antivir
331 Password required for antivir.
PASS McAfee
230 User antivir logged in.
TYPE A
200 Type set to A.
PORT YYY,YYY,YYY,YYY,4,23
200 PORT command successful.
LIST
150 ASCII data connection for /bin/ls (YYY.YYY.YYY.YYY,1047) (0 bytes).
226 ASCII Transfer complete.
TYPE I
200 Type set to I.
PORT YYY,YYY,YYY,YYY,4,24
200 PORT command successful.
SIZE update.ini
500 'SIZE update.ini': command not understood.
RETR update.ini
150 Binary data connection for update.ini (144.6.34.23,1048) (710 bytes).
226 Binary Transfer complete.
TYPE I
200 Type set to I.
PORT YYY,YYY,YYY,YYY,4,25
200 PORT command successful.
SIZE delta.ini
500 'SIZE delta.ini': command not understood.
RETR delta.ini
150 Binary data connection for delta.ini (YYY.YYY.YYY.YYY,1049) (1234 bytes).
226 Binary Transfer complete.
TYPE I
200 Type set to I.
PORT YYY,YYY,YYY,YYY,4,26
200 PORT command successful.
SIZE 42174218.upd
500 'SIZE 42174218.upd': command not understood.
RETR 42174218.upd
150 Binary data connection for 42174218.upd (YYY.YYY.YYY.YYY,1050) (118475 bytes).
226 Binary Transfer complete.
421 Timeout (900 seconds): closing control connection.
    
```

Figure 6. FTP session analysis using ethereal

Figure 6 is “ethereal” output of one particular ftp session obtained from tcpdump data. It clearly shows that in a case of ftp connection all traffic flows unencrypted and is clearly visible to any person with access to the files. The same case would apply for SMTP, POP and many other protocols regularly used on networks. Bold lines show user/password negotiation between two hosts on the network. Real addresses of the hosts were again replaced.

```

total 16808
drwxr-xr-x  2 antivir  staff      1536 Aug 16 09:00 .
drwxr-xr-x  5 root     other       512 Feb  2  2002 ..
-rw-----  1 antivir  staff      2674 Aug 14 09:36 .history
-rw-r--r--  1 antivir  staff       322 Jan 29  2002 .tcshrc
-rw-r--r--  1 antivir  staff     10171 May  5 02:04 42004201.upd
-rw-r--r--  1 antivir  staff    128208 May 10 02:04 42014202.upd
-rw-r--r--  1 antivir  staff   116751 May 16 18:36 42024203.upd
-rw-r--r--  1 antivir  staff   102355 May 23 08:15 42034204.upd
-rw-r--r--  1 antivir  staff   113660 May 31 02:04 42044205.upd
    
```

-rw-r--r--	1	antivir	staff	110231	Jun	7	02:10	42054206.upd
-rw-r--r--	1	antivir	staff	99892	Jun	15	02:06	42064207.upd
-rw-r--r--	1	antivir	staff	122500	Jun	21	02:13	42074208.upd
-rw-r--r--	1	antivir	staff	108621	Jun	28	02:08	42084209.upd
-rw-r--r--	1	antivir	staff	114676	Jul	4	14:16	42094210.upd
-rw-r--r--	1	antivir	staff	106037	Jul	11	10:18	42104211.upd
-rw-r--r--	1	antivir	staff	39037	Jul	16	14:17	42114212.upd
-rw-r--r--	1	antivir	staff	109103	Jul	18	15:05	42124213.upd
-rw-r--r--	1	antivir	staff	113554	Jul	25	03:18	42134214.upd
-rw-r--r--	1	antivir	staff	121067	Aug	1	08:31	42144215.upd
-rw-r--r--	1	antivir	staff	45615	Aug	3	15:04	42154216.upd
-rw-r--r--	1	antivir	staff	127155	Aug	8	15:09	42164217.upd
-rw-r--r--	1	antivir	staff	118475	Aug	15	15:13	42174218.upd
-rwx-----	1	antivir	staff	2362	Jan	29	2002	antivirload
-rwx-----	1	antivir	staff	781	May	22	12:10	checkupdate
-rw-r--r--	1	antivir	staff	2237545	Aug	15	15:08	dat-4218.zip
-rw-r--r--	1	antivir	staff	1234	Aug	15	15:13	delta.ini
-rw-r--r--	1	antivir	staff	4349267	Aug	15	15:13	sdat4218.exe
-rw-r--r--	1	antivir	staff	710	Aug	15	15:13	update.ini

Figure 7. FTP session analysis using ethereal

Figure 7 shows that the above privacy concerns apply not only to communication between the two hosts, but also to transferred data. The table shows what data have been transferred between the two hosts and subsequently captured by IDS after the command LIST was issued (underlined in Figure 6)

## References

- Apache HTTP Server. (Version 1.3.26)(2002).
- Arona, A., Bruschi, D., & Rosti, E. (1999, 6-10 December 1999). Adding availability to log services of untrusted machines. Paper presented at the 15th Annual Computer Security Applications Conference (ACSAC'99), Phoenix, AZ, USA.
- Australian Computer Emergency Response Team. (2001a, 8 October 2001). UNIX Security Checklist v2.0 [On-line]. Available: [http://www.uscert.org.au/Information/Auscert\\_info/Papers/usc20.html](http://www.uscert.org.au/Information/Auscert_info/Papers/usc20.html). Last access: 2001, November 19.
- Australian Computer Emergency Response Team. (2001b, 8 October 2001). UNIX Security Checklist v2.0 - The Essentials [On-line]. Available: [http://www.uscert.org.au/Information/Auscert\\_info/Papers/usc20\\_essentials.html](http://www.uscert.org.au/Information/Auscert_info/Papers/usc20_essentials.html). Last access: 2001, November 19.
- Broucek, V., & Turner, P. (2002a, 27-31 May 2002). Bridging the Divide: Rising Awareness of Forensic Issues amongst Systems Administrators. Paper presented at the 3rd International System Administration and Networking Conference, Maastricht, The Netherlands.
- Broucek, V., & Turner, P. (2002b, 8-11 June 2002). Risks and Solutions to problems arising from illegal or Inappropriate On-line Behaviours: Two Core Debates within Forensic Computing. Paper presented at the EICAR2002 Conference, Berlin, Germany.
- ethereal. (Version 0.9.7)(2002).
- Farmer, D., & Venema, W. (1993). Improving the Security of Your Site by Breaking Into it [On-line]. Available: <http://www.fish.com/security/admin-guide-to-cracking.html>. Last access: 2001, January 12.
- Handley, M., Paxson, V., & Kreibich, C. (2001). Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics. Paper presented at the 10th USENIX Security Symposium, Washington, DC, USA.
- McKemmish, R. (1999). What is Forensic Computing. Trends and Issues in Crime and Criminal Justice(118).
- mod\_ssl. (Version 2.8.10)(2002).
- openssl. (Version 0.9.6g)(2002).
- Ptacek, T. H., & Newsham, T. N. (1998). Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection [On-line]. Available: <http://www.snort.org/docs/idspaper/>. Last access: 2001, November 11.
- Roesch, M. (1999, November 7-12). Snort - Lightweight Intrusion Detection for Networks. Paper presented at the 13th Systems Administration Conference - LISA '99, Seattle, WA.
- Roesch, M. (2001a). Snort 1.8.7 [man pages].
- Roesch, M. (2001b, July 4, 2002). Snort Users Manual - Snort Release: 1.8.7 [On-line]. Available: <http://www.snort.org>. Last access: 2002, July 27.
- Smith, D., & Indulska, J. (2001). Enhancing Security of Unix Systems [On-line]. Available: [http://www.uscert.org.au/Information/Auscert\\_info/Papers/Enhancing\\_Security\\_of\\_Unix\\_Systems.html](http://www.uscert.org.au/Information/Auscert_info/Papers/Enhancing_Security_of_Unix_Systems.html). Last access: 2001, November 17.

- Sommer, P. (1998, 14-16 September 1998). Intrusion Detection Systems as Evidence. Paper presented at the Recent Advances in Intrusion Detection - RAID'98, Louvain-la-Neuve, Belgium.
- Sommer, P. (1999). Intrusion Detection Systems as Evidence. Computer Networks, 31(23-24), 2477-2487.
- ssh. (Version 3.2)(2002).
- tcpdump/libpcap. (Version 3.7.1)(2002).
- Venema, W. (1992, September 1992). TCP WRAPPER: Network monitoring, access control, and booby traps. Paper presented at the 3rd UNIX Security Symposium, Baltimore, USA.