Using Intrusion Detection Systems with a Firewall: Evaluation on DARPA 99 Dataset

H. Günes Kayacık, A. Nur Zincir-Heywood
Dalhousie University,
Faculty of Computer Science,
6050 University Avenue, Halifax, Nova Scotia. B3H 1W5
{kayacik,zincir}@cs.dal.ca

***Abstract*—In this paper, two open-source network intrusion detection systems –*Snort* and *Pakemon*– are combined with *Cisco IOS Firewall* intrusion detection features to increase detection of attacks. Evaluation of the systems is performed on DARPA 99 Intrusion Detection dataset. Individual and combined performance is characterized using multiple performance metrics. Results show that different tools perform well under different attack categories; hence demonstrating the benefit of deploying intrusion detection systems working together with a firewall.**

***Index terms*—Security Management, Intrusion Detection Systems, Case Study, Open Source Software**

I. Introduction

Security management plays an important role in today's management tasks. Defensive information operations, and intrusion detection systems are primarily designed to protect the availability, confidentiality and integrity of critical network information systems. These operations protect computer networks against denial-of-service attacks, unauthorized disclosure of information, and the modification or destruction of data. The automated detection and immediate reporting of these events are required in order to provide a timely response to attacks [1]. The two main classes of intrusion detection systems (IDS) are those that analyze network traffic and those that analyze operating system audit trails. In all of these approaches however, the amount of audit data is extensive, thus incurring large processing overheads. A balance therefore exists between the use of resources, the accuracy and timeliness of intrusion detection information. Thus, the authors of this paper believe that the selection and deployment of the IDS represents an increasingly important decision for any organization. Detecting or blocking attacks are not within the responsibilities of a firewall. Basically, firewalls are used to block certain types of traffic to improve the security. Therefore, more dynamic defense systems like intrusion detection systems should be deployed to detect attacks, which firewalls cannot see or detect. Some reasons for using firewalls with intrusion detection systems are [2]: (a) IDS double-checks mis-configured firewalls; (b) IDS catches the attacks, which firewall allowed to pass through; (c) IDS catches insider attacks which firewall never sees. The objective of this work is to determine the similarities and differences of these tools and find the cumulative benefit of using them together.

The remainder of the paper is organized as follows. Security management tools to be evaluated are introduced in section II. Details of the test set up and procedures are provided in section III. Results on DARPA 99 dataset are given in section IV and conclusions are drawn in section V.

II. Security Management Tools

Within this context, the term "security management tool" is used to imply any software or hardware, which improves the defense mechanism of a network system. In this work, we concentrate on three such security management tools.

A. Cisco IOS Firewall

Cisco IOS provides a cost effective way to deploy a firewall with intrusion detection capabilities. In addition to the firewall features, *Cisco IOS* Firewall has 59 built-in, static signatures to detect common attacks and misuse attempts. IDS process on the firewall inspects packet headers for intrusion detection by using those 59 signatures. In some cases routers may examine the whole packet and maintain the state information for the connection. Signatures fall into two categories: compound and atomic. There is no traffic dependent memory requirement for atomic signatures because they do not involve connection state. For compound signatures memory is allocated to inspect the state of the connection [3]. Upon attack detection, firewall can be configured to log the incident, drop the packet or reset the connection. The purpose of the intrusion detection component – on which we focused in this work – is to detect basic attacks on firewall without consuming resources, which should be used for routing, and forward the filtered traffic to the IDS in order to be inspected in more detail.

B. Pakemon IDS

"*Pakemon* has been developed to share IDS components based on the open source model" [4]. *Pakemon* is an open source experimental IDS, which aims to detect evasion methods such as fragmentation, disorder, duplication, overlap, insertion, and de-synchronization at the IP or TCP layer. Intrusion detection systems that perform monitoring at the packet level will not be able to see the intrusion data in the same way that final destination of a packet experiences. Hence, *Pakemon* processes captured packets like

a Linux node by reassembling IP packets and reconstructing the TCP streams. This was an important feature to provide especially in the light of earlier versions of *Snort*, which lacked such a facility. *Pakemons*'s signature structure is simpler than other IDS (such as S*nort*), where this simplicity is both strength, and weakness. That is to say, it takes time for IDS organizations to release new signature files. Meanwhile, as the signatures of new attacks are revealed, it is much easier to add them to the lightweight IDS signature databases such as *Pakemon* [4, 5].

C. Snort IDS

Snort is one of the best-known lightweight IDSs, which focuses on performance, flexibility and simplicity. It is an open-source intrusion detection system that is now in quite widespread use [5]. It can detect various attacks and probes including instances of buffer overflows, stealth port scans, common gateway interface attacks, and service message block system probes [5]. Hence, it is an example of active intrusion detection systems that detects possible intrusions or access violations while they are occurring [6]. Later versions of *Snort* provide IP de-fragmentation and TCP assembly to detect the attacks, or be it at the expense of having to view the whole attack data. *Snort* is lighter than commercial IDSs but it provides more features than any other IDS evaluated in this study. Although not as straightforward as the *Pakemon* system, flexible rule writing is supported in *Snort*.

III. TEST SET UP AND PROCEDURES

The test set up of this work consists of the following components: DARPA 1999 data set, traffic re-player and three security management tools under evaluation.

A. Data Set Characteristics

As mentioned above, for benchmarking purposes use is made of the DARPA 1999 Intrusion Detection Evaluation data set [7]. This represents Tcpdump and audit data generated over five weeks of simulated network traffic in a hypothetical military local area network (LAN). This simulated network consists of five victim machines, which are the targets of attacks in the evaluation (Solaris 2.5.1, Sun OS 4.1.4, Linux Red Hat 5.0 Kernel 2.0.32, Windows NT 4.0 Build 1381 SP1 and Windows 98), one sniffer, inside attackers and virtual hosts. On the other hand, outside hosts include a sniffer, outside attackers and virtual hosts. Inside and outside virtual hosts are used to spoof different IP addresses. Data collected for evaluation on this test bed includes Tcpdump data from inside and outside sniffers, Solaris Basic Security Module (BSM) audit data, Windows NT audit event logs, nightly listings of all files on the victim machines and nightly dumps of security related files on all victim machines over a 5 week period.

This work concentrates on the traffic data collected by inside and outside sniffers on week-4. The reason we chose week-4 is that the first three weeks of the data set was designed for training the data driven learning systems in the original competition, hence not applicable to this work, whereas weeks 4 and 5 represented the test data. In this case for, reasons of expediency, we concentrate on the 2.5GB of data present in week 4 data set (week 5 is even larger and beyond computing resources available). The data used for testing (week 4) therefore either represented a normal connection or one of the 55 different attack types [8]. There are 80 attacks in week-4 data set, where all attacks fell into one of the five following categories:

- *Denial of Service:* Attacker tries to prevent legitimate users from using a service.
- *Remote to Local:* Attacker does not have an account on the victim machine, hence tries to gain local access.
- *User to Root:* Attacker has local access to the victim machine and tries to gain super-user privileges.
- *Probe:* Attacker tries to gather information on the target host.
- *Data:* Attacker performs some action, which is prohibited by the security policy.

B. Traffic Re-player and Security Management Tools

In order to evaluate the three open source security management tools based on the 1999 DARPA data set, an environment was necessary where test data could be re-run from the 4th week for the 3 target security management tools. To this end, the *TCPReplay* utility provided by SourceForge.net is used to replay packets to a live network that were previously captured with the *tcpdump* program [9]. In effect, *TCPReplay* attempts to match the timing of the original traffic, optionally speeding it up or slowing it down [9]. In addition, *TCPReplay* supports multiple network interfaces allowing replayed packets to be injected into different points on a network based on the source address [9].

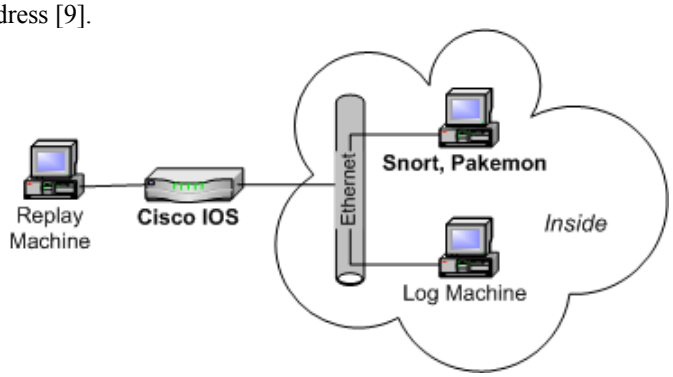

Figure 1. Network diagram of the benchmarking environment

Intrusion detection benchmarking environment shown in figure 1 consists of one Pentium 200 machine, two Pentium 133 machines all with 32 MB memory and a Cisco 3600 router with IOS version 12. Cisco router is configured to log alerts to the *syslog* service of the log machine. One of the Pentium 133 machine is designated as Intrusion Detection (ID) server (on which *Pakemon* and *Snort* runs and listens the Ethernet in promiscuous mode) and the other is designated as the log machine, which logs the alerts *Cisco IOS* sends. Pentium 200 machine is designated to *TCPReplay,* where this is responsible for replaying the

recorded traffic. Filtering attacks before it is inspected by firewall's intrusion detection component and the intrusion detection server is not desirable in this work. Therefore packet filtering is disabled throughout the benchmarking experiments. Router is configured to inspect the packets for intrusions.

Linux Mandrake 8.1 is installed on all machines as the operating system including all the necessary libraries (such as *libpcap*, *libnet*, *libnids* etc.). It should be noted that *Pakemon* and *Snort* are used with their default configurations. Moreover, the latest signature files available are used for both intrusion detection systems. On the other hand, the data set is replayed with 1Mbps speed because of the hardware limitations of the ID server (Pakemon/Snort server, Figure 1). It took approximately 2 hours to replay one-day of traffic.

C. Evaluation Procedure

It should be noted that log or alert files of the tools that are evaluated contain different types of entries including different amounts of information about the events that occurred on the network. Each entry is a packet/message that contains information about an event from a specific IP address (destination IP and ports). However, an individual attack might contain more than one entry and many TCP sessions. Therefore, different scripts are developed in order to filter out the required information from different types of entries in the log files of *Snort*, *Pakemon* and *Cisco IOS*. We configured *Pakemon* to record everything in system log and dump the packets to another file, whereas *Snort* is configured to record intrusion attempts in directories. *Cisco IOS* is configured to use system log service of a Linux Machine. Thus, our scripts run on these files for *Snort*, *Pakemon* and *Cisco IOS*.

TABLE-1
SUMMARY OF THE CONFIDENCE LEVELS

	CL1	CL2	CL3	CL4
Source and Attacker IP match	Yes	Yes	Yes	Yes
Destination and Victim IP match	Yes	Yes	Yes	No
Source and Attacker port match	Yes	No	No	No
Destination and Victim port match	Yes	Yes	No	No

Basically, the scripts extract the IP and port information from the log files, and compare them to the ones in the attack identification list, which holds the true attack information in the DARPA data set [8]. Thus, the tools are compared against the true attacks that occurred in the 4th week of the simulation, where there were 80 attack instances. The comparison of the attack identification list and the log file entries is performed based on source (attacker) and destination (victim) IP addresses and ports. Information about the source or destination is extracted from the IDS log files, whereas information about the attacker or victim is extracted from the attack identification list. In other words, we compare attacker information in the identification list with the source information in the log files and victim information in the identification list with destination information in the log files. However, since most entries do not include all the required information (in the case of *Pakemon*, a global port-scan entry in a log file usually includes only the source IP), it becomes difficult to match the relevant fields. Therefore, four confidence levels (CL) are defined for determining the degree of match in order to detect different attacks, table-1. A log entry – attack match is most confident if it is a CL1 match, whereas it is least confident if it is a CL4 match.

IV. RESULTS

As indicated before, scripts match attacks with log entries. If there is a match, scripts output attack ID, attack name, attack category from attack identification list and match confidence level. Table 2 summarizes the detection rate of each tool on different categories on the 4th week of traffic generated for DARPA 1999 evaluation.

TABLE-2
NUMBER OF DETECTED ATTACK INSTANCES IN DIFFERENT CATEGORIES COMPARED WITH THE TOTAL NUMBERS IN 4TH WEEK

	U2R	R2L	DoS	Probe	Data	Total
Snort	5	18	5	4	3	35
Pakemon	1	17	6	2	3	29
Cisco IOS	1	7	5	4	0	17
Week 4	8	37	16	15	4	80

When the performances of these tools are compared based on different categories, we see that performance of *Snort* and *Pakemon* share similar detection counts over different attack categories. To actually determine which tool performs better two more parameters are taken into consideration: (1) number of false alarms and (2) Total number of entries i.e. the number of entries that it takes to be parsed by a network administrator to detect those attacks.

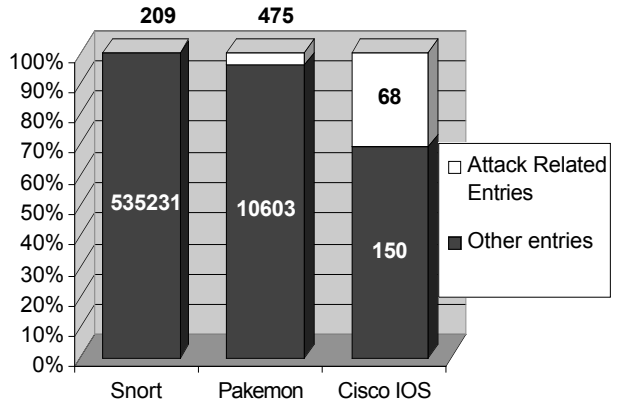

Figure 2. Log file analysis in terms of number of entries.

Figure 2 shows the number of attack related entries in the corresponding log files and their percentage. The reason the number of entries is so high for both *Snort* and *Pakemon* is that both IDSs usually log attack entries more than once. This in return increases the size of the log files requiring analysis by network administrators. Occurrence of non-attack entries, in other terms false alarm rate, is very high in both of the intrusion detection systems. In both cases it is very costly to examine all log entries. Although

Cisco IOS detects fewer attacks, it has low false alarm rate and small log file size, which are the significant advantages over the two IDS.

TABLE-3
DETECTION CONFIDENCE LEVELS FOR ECH TOOL

	CL1	CL2	CL3	CL4
Snort	0	6	29	0
Pakemon	0	5	21	1
Cisco IOS	0	0	17	0

As shown in table 3, most of the attacks are detected with the third confidence level. *Cisco IOS* always detects with third confidence level because it does not log port information whereas *Pakemon* and *Snort* do in some cases.

Among 59 signatures documented in Cisco IOS documentation [3], only 5 signatures are triggered by the test data. Distribution of the 5 signatures over attack related entries is shown in table 4. Signature IDs and names are as follows:

- **1102 Impossible IP Packet:** This signature is triggered if the source and destination addresses are the same.
- **2000 ICMP Echo Reply:** This signature is triggered if the ICMP message is "echo reply".
- **2001 ICMP Host Unreachable:** This signature is triggered if the ICMP message is "Host Unreachable".
- **3042 TCP-FIN bit with no ACK in flags:** This signature is triggered if FIN bit is set but ACK is not set in a packet.
- **3050 Half-open SYN attack:** This signature is triggered if a connection is improperly initiated to a well-known TCP port such as FTP, Telnet, HTTP or E-Mail.

TABLE-4
DISTRIBUTION OF TRIGGERED CISCO IOS SIGNATURES AMONG ATTACK RELATED ENTRIES

	U2R	R2L	DoS	Probe	Data
Sig. 1102	0	0	1	0	0
Sig. 2000	0	0	4	2	0
Sig. 2001	3	14	0	19	0
Sig. 3042	0	0	0	1	0
Sig. 3050	1	16	3	4	0

The only instance of signature 1102 is at *DoS* category –which is expected – because it is triggered by the *land* attack. *Land* is a denial of service attack, which involves packets with the same source and destination addresses. Signature 2001, which produced majority of the attack related alerts, is triggered mostly by *R2L* and *Probe*. We believe this is natural since ICMP messages can be used to probe a host or launch a remote attack. In figure 3, each tool is represented as a set, which contains detected attacks. The regions that intersect show the attacks detected by more than one tool. Each element in the figure 3 represents a detected attack with the format: Attack Name (Number of detected instances in that region / Total instances in Week 4) – Attack Category.

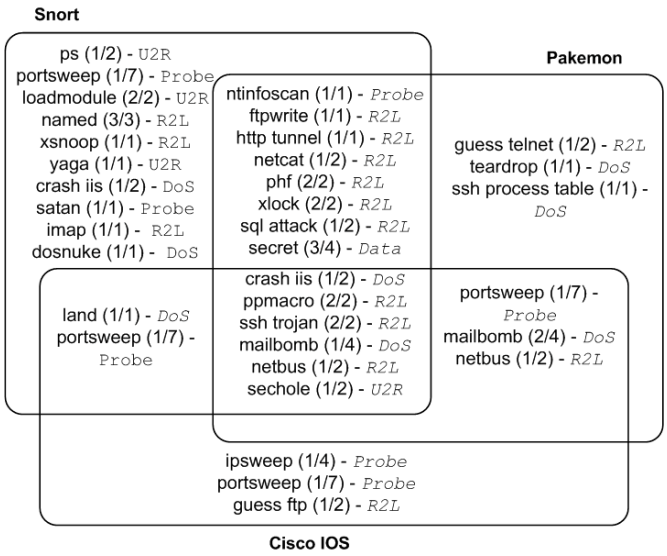

Figure 3. Analysis of detected attacks

Crash IIS (Day 2), *Power Point Macro* (Day 3 and 4), *Mail Bomb* (Day 3), *SSH Trojan* (Day 4 and 5), *Netbus* (Day 5) and *Windows Security Hole* (Day 5) attacks are detected by all three security management tools. To the total defense system formed by three tools, *Snort* contributes 13 (16.3%) attacks (upper left region), *Pakemon* contributes 3 (3.8%) attacks (upper right region) and *Cisco IOS* contributes 3 (3.8%) attacks. Mutually detected attacks are not counted in the net contribution because even one system is taken out of the defense mechanism, remaining systems will still be able to detect them. By using *Snort*, *Pakemon* and *Cisco IOS* together, 45 attacks are detected whereas individual performances are 35, 27, and 17 for *Snort*, *Pakemon* and *Cisco IOS* respectively. Figure 4 visually summarizes the performance of each tool individually and combined together.

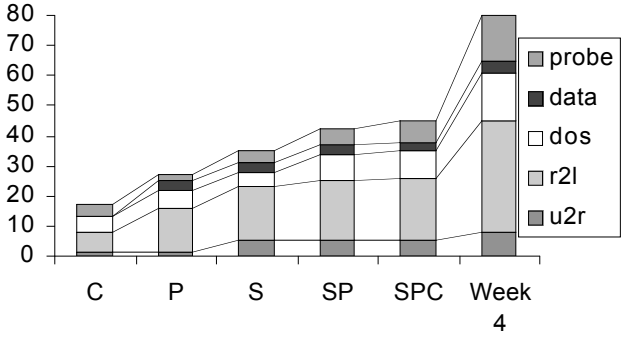

Figure 4. Performance of the evaluated tools (S: *Snort*, P: *Pakemon*, C: *Cisco IOS*)

V. CONCLUSION

The work presented here is of a case study nature, but we believe sufficient to warrant continued development. In particular, we demonstrated a benchmark evaluation of three security management tools. The results show that none of the tools could detect all the attacks. *Snort* detected ~44%, *Pakemon* detected ~34% and *Cisco IOS* detected ~21%.

However, figure 4 shows that when we combine all three tools we can get ~56% detection rate. In terms of false alarm rates, *Snort* and *Pakemon* performed poorly with ~99% and ~95% false alarm rates respectively, whereas *Cisco IOS* is significantly better with ~68% false alarm rate. Results also show that *Cisco IOS* performs as good as other systems on denial of service attacks and probes, therefore it is possible to filter those kind of intrusions at the firewall level, which in turn decrease the attack traffic passing to the IDSs.

ACKNOWLEDGEMENTS

This research was conducted in Telecom Applications Research Alliance Laboratory with Cisco products. The authors gratefully acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada for the second author's Discovery Grant.

REFERENCES

[1] Bass T., "Intrusion Detection Systems and Multisensor Data Fusion", Communications of the ACM, Vol. 43, No. 4, pp 99-105, April, 2000.

[2] IDS Frequently Asked Questions, The Intrusion Detection System Group, http://www.intrusion-detection-system-group.co.uk/faq.htm#firewall

[3] Cisco IOS Firewall Intrusion Detection System Documentation, http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120t/120t5/iosfw2/ios_ids.htm

[4] Takeda K., Takefuji Y., "Pakemon – A Rule Based Network Intrusion Detection System", International Journal of Knowledge-Based Intelligent Engineering Systems, Vol. 5, No. 4, pp 240-246, October 2001.

[5] Roesch M., "Snort – Lightweight Intrusion Detection for Networks", 13th Systems Administration Conference, Proceedings of LISA 1999.

[6] The CERT®/CC, http://www.cert.org/security-improvement/implementations/i042.07.html

[7] MIT Lincoln Laboratory, DARPA Intrusion Detection Evaluation Data Sets, http://www.ll.mit.edu/IST/ideval/data/data_index.html

[8] DARPA 99 Intrusion Detection Data Set Attack Documentation, http://www.ll.mit.edu/IST/ideval/docs/1999/attackDB.html

[9] TCPReplay traffic replay utility, http://tcpreplay.sourceforge.net/