EURASIP Journal on Information Security
a SpringerOpen Journal

RESEARCH Open Access

Intrusion detection model based on selective packet sampling

Ezzat G Bakhoum

Abstract

Recent experimental work by Androulidakis and Papavassiliou (IET Commun 2(3):399, 2008; IEEE Netw 23(1):6, 2009) has shown that it is possible to maintain a high level of network security while selectively inspecting packets for the existence of intrusive activity, thereby resulting in a minimal amount of processing overhead. In this paper, a statistical approach for the modeling of network intrusions as Markov processes is introduced. The theoretical findings presented here confirm the earlier experimental results of Androulidakis and Papavassiliou. A common notion about network intrusion detection systems is that every packet arriving into a network must be inspected in order to prevent intrusions. This investigation, together with the earlier experimental results, disproves that notion. Additional experimental testing of a corporate local area network is reported.

Keywords: Network Intrusion, Intrusion Detection System, IP Packets, Markov Process, Birth and Death Model

1. Introduction

Network intrusion detection systems (IDS) perform a vital role in protecting networks connected to the World Wide Web from malicious attacks. Traditionally, IDS software products such as SNORT [1], SecureNet [2], and Hogwash [3] work by monitoring traffic at the network choke-point, where every incoming IP packet is analyzed for suspicious patterns that may indicate hostile activity. Because those software systems must match packets against thousands of known ominous patterns, they must work extremely fast. Under heavy traffic, however, the IDS is usually forced to drop packets so that the IDS itself will not become the bottleneck of the network, of course at the risk of allowing an attack to go undetected. Because of this deficiency, host-based IDS solutions have been introduced [4,5]. Host-based IDS products run on a server rather than at the network gateway. Unfortunately, however, host-based solutions can slow down the server considerably under heavy traffic conditions. Because of the inherent limitations of all software solutions, hardware solutions were finally introduced. The state-of-the-art hardware solution is a field programmable gate array (FPGA) that performs the same IDS function at substantially higher speeds [6,7]. There are serious other problems, however, to contend with when hardware solutions are implemented [7].

The purpose of this paper is to introduce an analytic and statistical model for the process of network intrusion and to demonstrate that the common notion of the necessity of having the content of every IP packet inspected is flawed. In the past, numerous research articles that addressed the problem of network intrusion modeling have appeared in the literature [8-19]. Kephart and White [8,9] published the first analytical work on the modeling of the propagation of viruses and worms. More recently, Wang and Wang [10], guided by the analysis of Kephart and White, recognized that the problem of network intrusion can be modeled after the popular "birth and death" epidemiological model. Wang and Wang (WW), however, did not develop such a model analytically, as the problem is mathematically challenging. Very recently, an important experimental discovery was made by Androulidakis and Papavassiliou (AP) [11,12], when they demonstrated experimentally that the selective inspection of packets for the purpose of detecting network intrusion can be as effective as the full inspection of all packets. In this paper, it will be demonstrated that the seemingly unrelated discoveries of WW and AP do in fact stem from the same mathematical origin. More specifically, the WW hypothesis that the process of network intrusion can be modeled

Correspondence: ebakhoum@uwf.edu
Department of Electrical and Computer Engineering, University of West Florida 11000 University Parkway, Pensacola, FL, 32514, USA

after the "birth and death" epidemiological model will be developed analytically for the first time. The results are surprising and essentially confirm the experimental findings of AP. The main conclusion is that it is possible to selectively inspect packets from only certain packet flows, thereby eliminating the speed bottleneck problem and the necessity to drop packets at high bit rates, while simultaneously maintaining a high degree of network security.

Actual testing by the author that involved a corporate local area network has confirmed the theoretical findings. Additional testing of an optimized SNORT software package–in combination with a traffic generator and an Agilent network analyzer–has further confirmed the theoretical findings. The implication of these theoretical and experimental results to the structure and the design of future IDS will be quite substantial.

2. Statistical model of network intrusion

The analysis that will be now developed is based on the observation that the birth and death model of network intrusion that was advocated by Wang and Wang is a class of Markov processes [20,21]. By applying Markov chain analysis to the process of network intrusion, a statistical formula that relates the probability of a network being compromised to the probability of occurrence of intrusion will be obtained. In the following sections, it will be demonstrated that it is possible to selectively inspect packets arriving into a network while maintaining a high degree of security at the same time, as long as such inspection is performed in accordance with the statistical formula.

Consider an Intranet (such as a corporate LAN) that is connected to the World Wide Web and protected by means of a network intrusion detection system, as shown in Figure 1. We shall assume that at any given time the Intranet has a total of n processes, of which i processes may be intrusive or hostile. We shall characterize the network as being in state S_i when i hostile processes are running, state S_{i-1} when i - 1 of such processes are running, etc. State S_0 will be therefore the "clean" state, where no intrusive activity exists. The first step toward modeling the process of intrusion is to realize that the different states of the network can be regarded as a set of mutually exclusive and collectively exhaustive states. Furthermore, the transition of the network from the present state to a different state is a function only of the present state and the probability of transition to the next state. These characteristics are the characteristics of Markov processes [22,23]. Wang and Wang [10] recognized that the process of network intrusion can be modeled after the popular "birth and death" epidemiological model (a class of Markov processes). Indeed, the birth and death model has been applied in the past in a number of other engineering problems of similar characteristics [20,21]. The model is based on the assumption that it is equally likely for a system to make a transition from state S_{i-1} to state S_i or from state S_i to state S_{i-1}. To understand this fact, consider Figure 1. The i hostile processes that are running on the network may initiate a new hostile process, so that the number of such processes becomes i + 1 (even without the occurrence of external activity; e.g., an infected host on the network that attempts to infect other hosts). Alternatively, a hostile process may be terminated and the number therefore drops to i - 1. In other words, it is equally likely for a new hostile process to be started (born) on the network or for an existing hostile process to be terminated (die) and hence the name "birth and death" model. This is the model that will be now adopted in the present analysis. It is important to point out that if the probability of the transition ($S_i \rightarrow S_{i-1}$) is in fact not equal to the probability of the transition ($S_{i-1} \rightarrow$

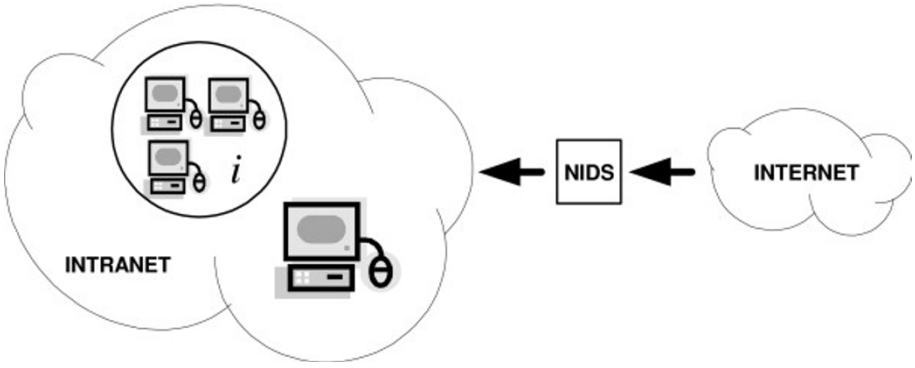

Figure 1 A network intrusion detection system inspects packets coming to a protected Intranet from the unprotected Internet. The Intranet may have a total of *n* processes, of which *i* processes are intrusive or hostile. The hostile activity may originate externally or internally from within the Intranet.

S_i), the analysis presented here will not be altered, but an additional constant of proportionality will simply appear in the final equation. It is finally important to point out that no assumption, explicit or implicit, should be made about this generally complex problem other than what is specifically described in the analysis below. We shall now define four variables, let:

- b be the birth rate (or initiation rate) of new processes on the network at any given time;
- d be the death rate (or termination rate) of processes;
- P_i be the probability that the network is in state S_i;
- P_H be the probability that any new process started on the network be a hostile process. (This probability is an independent variable that strongly depends on the circumstances. The numerical value of this probability will be calculated as described further in Section 3).

Following the basic assumption of the equality of the two transitions ($S_i \rightarrow S_{i\text{-}1}$) and ($S_{i\text{-}1} \rightarrow S_i$), this equality can be written in terms of the above variables as follows:

$$\frac{i}{n} d\, P_i = b\, P_H\, P_{i-1} \tag{1}$$

where the ratio i/n is the ratio of the number of hostile processes to the total number of processes running on the network. Now, let $\alpha = b/d$ be the "workload growth rate" at any given time. The above equation can therefore be written as

$$P_i = \frac{n}{i} \alpha\, P_H\, P_{i-1} \tag{2}$$

from which we must conclude that n

$$P_1 = \frac{n}{1} \alpha\, P_H\, P_0 \tag{3}$$

and

$$\begin{aligned} P_2 &= \frac{1}{2} n\, \alpha\, P_H\, P_1 \\ &= \frac{1}{2} (n\alpha P_H)^2 P_0 \end{aligned} \tag{4}$$

(note that n is a general number that specifies the total number of processes at any given state. It is therefore not appropriate to use $n + 1$ instead of n in the last equation). Now P_3 will be given by

$$\begin{aligned} P_3 &= \frac{1}{3} n\, \alpha\, P_H\, P_2 \\ &= \frac{1}{3} \cdot \frac{1}{2} (n\alpha P_H)^3 P_0 \end{aligned} \tag{5}$$

Hence, for any state S_i,

$$P_i = \frac{1}{i!} (n\alpha P_H)^i P_0 \tag{6}$$

Equation (6) can be alternatively and more rigorously proven using an inductive argument. Assuming that Eq. (6) is valid for any value of i, if we replace i by $i + 1$, we have

$$P_{i+1} = \frac{1}{(i+1)!} (n\alpha P_H)^{i+1} P_0 \tag{7}$$

By similarly replacing i by $i + 1$ in Eq. (2), we have

$$P_{i+1} = \frac{n}{i+1} \alpha\, P_H\, P_i = \frac{n}{i+1} \alpha\, P_H \frac{1}{i!} (n\alpha P_H)^i P_0 = \frac{1}{(i+1)!} (n\alpha P_H)^{i+1} P_0 \tag{8}$$

which is exactly the same as Eq. (7). Hence, Eq. (6) is indeed valid for any value of i.

Given a maximum number $i_{\max}$ of possible hostile processes, where $i_{\max} < n$, and given that the set of all "infected" states, together with the clean state, are collectively exhaustive, we have the identity

$$\sum_{i=0}^{i_{\max}} P_i = 1 \tag{9}$$

From Eqs. (6) and (9), we now have

$$P_0 \sum_{i=0}^{i_{\max}} \frac{1}{i!} (n\alpha P_H)^i = 1 \tag{10}$$

Since $i_{\max}$ would normally be a large number, the summation approximates the infinite series

$$\sum_{i=0}^{\infty} \frac{x^i}{i!} = e^x \tag{11}$$

Equation (10) can therefore be written as

$$P_0\, e^{n\alpha P_H} = 1 \tag{12}$$

or

$$P_0 = e^{-n\alpha P_H} \tag{13}$$

The above equation gives the probability that the network is in an intrusion-free state, P_0, as a function of the probability that any new process attempted on the network might be a hostile process, P_H. A plot of this equation is shown in Figure 2 for different values of n, where the workload growth rate a is assumed to be equal to 1 (stable workload).

The number n in Eq. (13) and in the figure is the number of processes initiated by users and does not include such things as system processes or background tasks, since the probability of occurrence of intrusion is

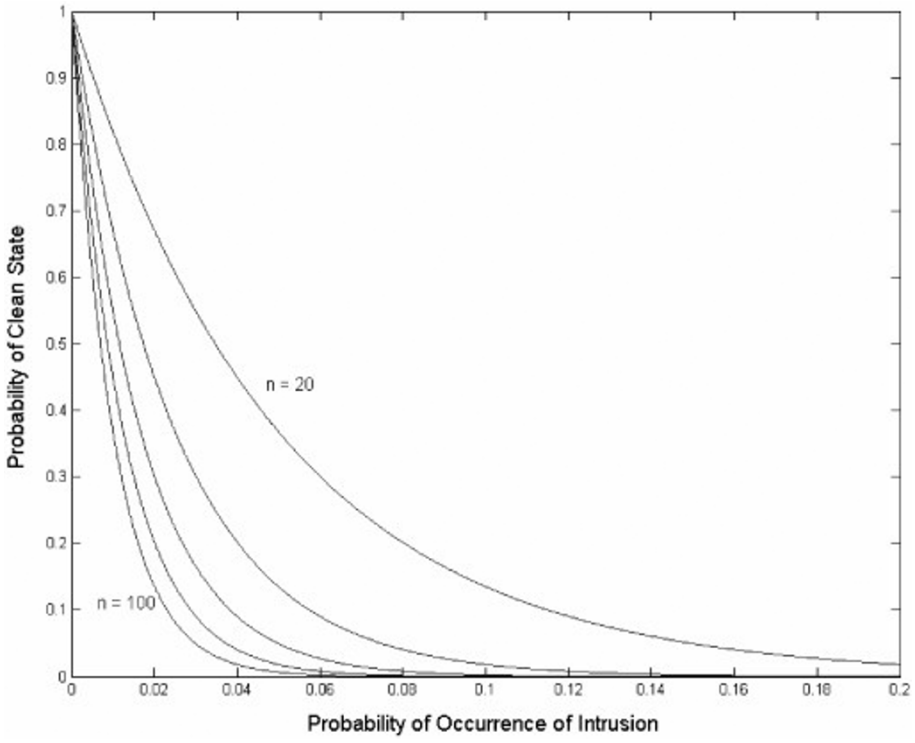

Figure 2 Probability that network is in clean state (P_0) versus the probability of occurrence of intrusion (P_H), for values of n ranging from 20 to 100.

associated with user processes only. The inverse exponential relationship of Eq. (13) is remarkable because, as is well known, it can be practically approximated by a linear function that drops to zero at a specific threshold. For the n = 100 curve, for example, it can be easily concluded that the inspection of all the incoming packets will be inevitable if the probability of occurrence of intrusion is larger than about 2%, as the probability of a clean state is practically zero at all points past that threshold. For P_H < 2%, the probability of a clean state is substantial, and, as will be demonstrated in the following section, it is possible to inspect packets selectively under such conditions without sacrificing security. This is clearly a better alternative to the strategy of "inspect all packets, or drop packets randomly" that is currently being implemented in IDS software solutions.

3. Optimization of network intrusion detection systems

In view of the above, two questions must now be answered: first, how the probability P_H will be determined at any given time; and secondly, if P_H is below the threshold determined by Eq. (13), then what kind of packet sampling strategy must be used to ensure that intrusion would still be detected if it occurs.[a] In 2008, Androulidakis and Papavassiliou [11,12] demonstrated experimentally for the first time that under certain conditions, the selective inspection of packets for the purpose of detecting network intrusion can be as effective as the full inspection of all packets. We shall now demonstrate that the Androulidakis-Papavassiliou criterion corresponds with the conclusions reached above.

3.1. The connection between P_H and the Androulidakis-Papavassiliou criterion

Androulidakis and Papavassiliou suggested that by selectively inspecting packets and calculating the Shannon entropy for the packets selected, a number that is indicative of the likelihood of occurrence of intrusion is obtained. The Shannon entropy H is defined as

$$H = \sum_{k=1}^{N} P_k \log_2 \frac{1}{P_k} \tag{14}$$

where N is the number of packets inspected and P_k is the probability of occurrence of message k within the stream of packets selected. The "messages" of concern here are the following: the source IP address, the destination IP address, the source port, the destination port, and the protocol. According to [12]: "Entropy measures the randomness of a data set. High entropy values signify a dispersed probability density function, while low

entropy values indicate a more concentrated distribution. For example, an anomaly such as an infected host on the Internet that tries to infect other hosts (worm propagation) results in a decrease of the entropy of the source IP address. The infected machine produces a disproportionately large number of packets, causing the same source IP address to dominate in the distribution of source IP addresses". The entropy according to Eq. (14) has a maximum value of $\log_2 N$. In [12], the entropy was "normalized" by dividing the expression by $\log_2 N$, so that it ranges from 0 to 1, that is

$$H(\text{normalized}) = \frac{\sum_{k=1}^{N} P_k \log_2 1/P_k}{\log_2 N} \quad (15)$$

While the normalized entropy takes values in the range (0,1), it is actually an inverse measure of the probability of occurrence of intrusion. As indicated above, an intrusion attempt would actually cause the value of the entropy of the source IP address to decrease. A value of H that is close to 0 indicates a probability of occurrence of intrusion that is close to 1 and vice versa. Hence, the probability of occurrence of intrusion P_H can be defined in terms of H as follows:

$$P_H = 1 - H(\text{normalized}) = 1 - \frac{\sum_{k=1}^{N} P_k \log_2 1/P_k}{\log_2 N} \quad (16)$$

To test the theoretical predictions made in this paper,[b] a number of tests that involved the network of a local corporation were conducted. The first test was a test to determine whether P_H as defined by Eq. (16) correlates with the results shown in Figure 2. In the test, the LAN was subjected to traffic that emulates the propagation of the Slammer worm [24] (additional details about the tests are discussed in the following sections). The malicious traffic consisted of a single UDP packet per destination IP address, where the destination IP address was chosen randomly. The source port of each UDP packet was also chosen randomly, ranging from 1 to 65,535. The packets arriving at the LAN were inspected selectively, and the value of P_H was calculated from Eq. (16) at regular time intervals. The number of user processes n running on the LAN was purposely maintained at a constant value of approximately 100. The result is shown in Figure 3.

As can be seen from the graph, the calculated value of P_H increased from less than 1% to more than 90% for a time duration of approximately 1 min during which the attack was simulated. This corresponds very well with the data shown in Figure 2. Clearly, the formula in Eq. (16) for calculating the instantaneous value of P_H correlates with the analysis of the previous section.

3.2. The principle of selective packet inspection

As suggested by Eq. (13) and Figure 2, full inspections of all the packets should be implemented by the IDS if the value of P_H is above a calculated threshold, and selective inspection is conceivably possible if P_H is below the threshold. Under heavy traffic conditions and

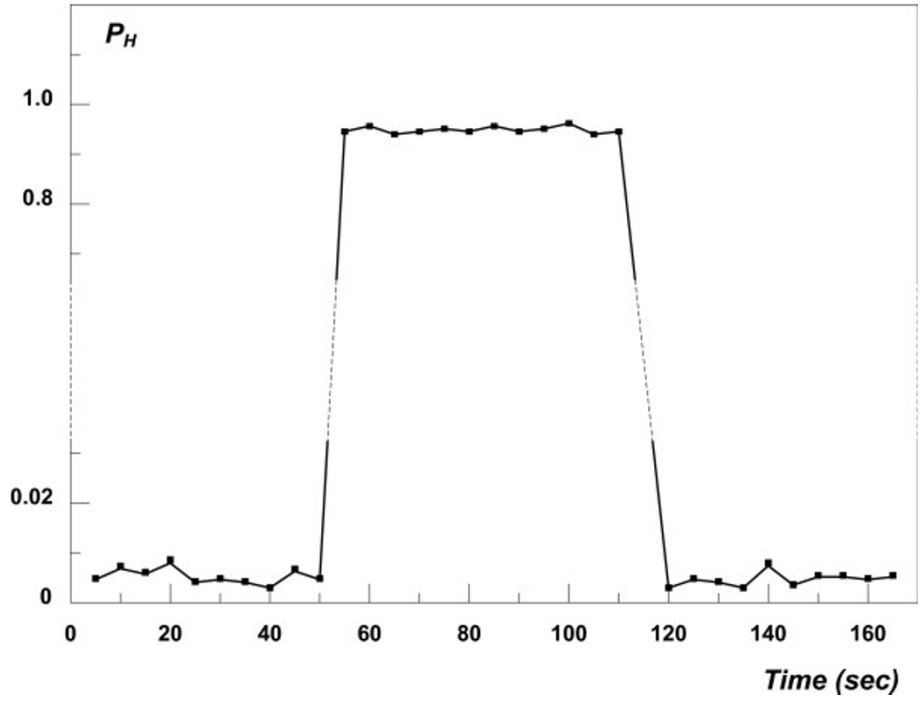

Figure 3 Value of P_H as calculated from Eq. (16) for the source IP address and/or the destination port (worm propagation test).

low probability of occurrence of intrusion, such a solution is obviously very desirable. We shall now answer the important question of what kind of packet sampling strategy must be used to ensure that intrusion would still be detected if it occurs. A number of studies have differentiated between packet-based sampling and flow-based sampling [25-27]. In packet-based sampling, packets are selected from the global traffic using a pre-specified method. In flow-based sampling, packets are first classified into flows. A "flow" is defined as a set of packets that have in common the following packet header fields: source IP address, destination IP address, source port, destination port, and protocol. The published studies, particularly the studies by Barford et al. [28] and Sridharan et al. [29], have showed that small flows (flows that consist of 1-4 packets) are usually the source of most network attacks. Androulidakis and Papavassiliou have in fact advocated and demonstrated the success of the selective inspection of packets from small flows in their experimental investigation. According to that approach, flows that consist of 1-4 packets are fully inspected, and larger flows are inspected with a sampling frequency that is inversely proportional to their size (see ref. [11]). We now give a rigorous proof that such a technique for the selective inspection of packets guarantees that intrusion will be detected if it occurs:

Lemma: If the selective inspection of packets with a sampling probability that favors small flows is implemented, the probability of detecting a network intrusion is approximately equal to 1.

Proof: Assume that the probability of any packet selected being a malicious packet is P. If a total number of N packets are selected, the probability of detecting at least one malicious packet will be given by the Bernoulli statistical trial probability [30]

$$P_{\text{detection}} = \sum_{r=1}^{N} \binom{N}{r} (P)^r (1-P)^{N-r} \tag{17}$$

For sufficiently large N (e.g., $N > 100$) and sufficiently large P (e.g., $P > 0.01$, or 1%), the above summation is approximately equal to 1. If packets are selected predominantly from small flows, P is guaranteed to be substantially higher than 1% (port scan, for instance, is only one packet). ■

To summarize the above conclusions, a modern, efficient IDS should selectively inspect packets such that small flows (flows that consist of 1-4 packets) are fully inspected, and larger flows are inspected with a frequency that is inversely proportional to their size. The probability of occurrence of intrusion P_H should be calculated in real time by using Eq. (16). For calculating P_H, only the packet headers need to be inspected (see the discussion in the previous section) and the probabilities of occurrence of the source/destination IP address, the source/destination port, and/or the protocol must be calculated and used in Eq. (16). If at any time P_H exceeds a suitable threshold that is calculated from Eq. (13), the IDS must switch immediately to the full inspection of the content of all the packet traffic and quarantine any packets that are found to be malicious.

3.3. Testing of the proposed IDS approach

The local area network of a small local corporation of 50 employees was used to test the IDS approach suggested above. The experimental setup is shown in Figure 4.

As shown, malicious traffic was generated from a Linux machine on which two different packet-generation programs were installed: IDSWakeup [31] and D-ITG [32]. These programs make use of the powerful kernel of Linux to generate packets at speeds of up to one Gigabit per second. The main purpose of IDSwakeup is to generate false intrusive attacks that mimic well-known ones (e.g., Denial of Service (DoS) attacks, port scan, and worm propagation), in order to determine how the IDS detects and responds to those attacks. D-ITG (which stands for Distributed Internet Traffic Generator), on the other hand, is a simple but very versatile packet generator that can generate packets of different sizes and different inter-departure times. The packet-generation machine is equipped with a 3 GHz Pentium 4 processor, 4 GB of RAM, and a 1 Gb/s network interface card. The malicious traffic generated was merged with regular Internet traffic through a Cisco router and directed to the corporate LAN, as shown. A simple IDS software solution was developed for implementing the inspection strategy described above. The code was developed in Matlab and converted to C (for brevity, the details of the code will not be discussed here). Essentially, the code inspects the headers of the packets in small flows (flows that are 1-4 packets in length). The headers of packets in larger flows are inspected with a frequency that is inversely proportional to the size of the flow, as described in the previous section. After 100 packets are selected, the code computes P_H from Eq. (16), for 3 different attack scenarios: DoS, port scan, and worm propagation. If P_H is found to have exceeded a suitable threshold that is calculated from Eq. (13), the code immediately moves to full inspection mode, where the actual contents of the packets selected and all subsequent packets are inspected for the presence of well-known patterns [11,28,29]. Any packets that are found to be malicious are quarantined.[c] Throughout each test conducted, the number of user processes n running on the LAN was purposely maintained at a constant value (according to the theory in Section 2, the higher the value of n the lower the threshold that must be used).

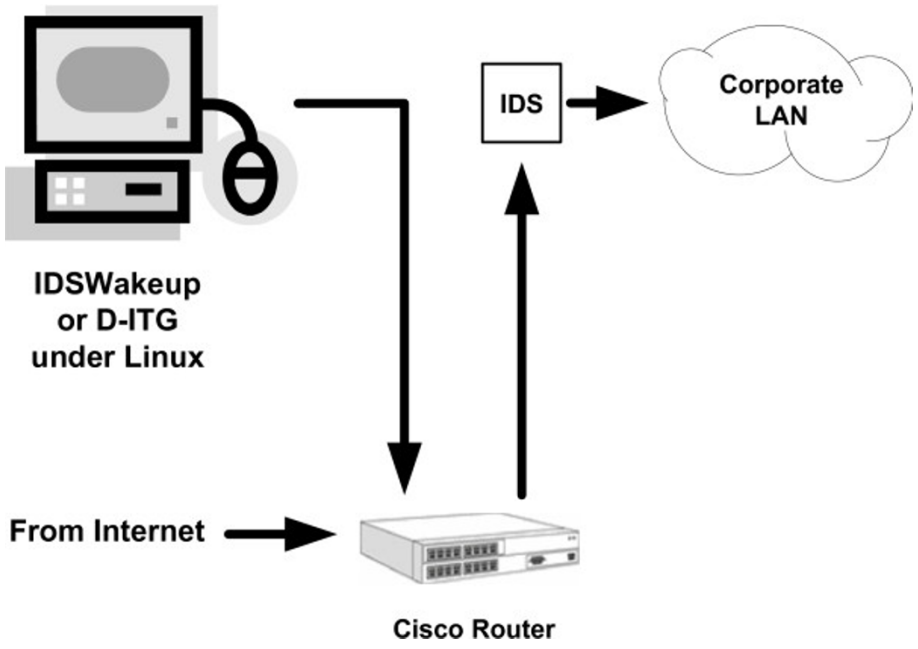

Figure 4 Setup for testing the proposed optimized IDS approach.

The first objective of the testing was to determine the number of malicious packets that managed to slip through the IDS when P_H was below the calculated threshold. Figures 5 and 6 show the results that were obtained for an average number of user processes n = 20 and 100, respectively. As Figure 5 shows, the maximum percentage of hostile packets that slipped through the IDS in the first case was slightly over 0.1% (i.e., one in every 1,000 hostile packets managed to slip through undetected). The results were very similar for the 3 types of attacks: DoS, port scan, and worm propagation. In this test, an extremely small percentage of the global flow was made hostile, instead of actually launching an outright intrusive attack. This percentage was then increased gradually, which helped increase the calculated value of P_H, as the graph shows. Finally, when an

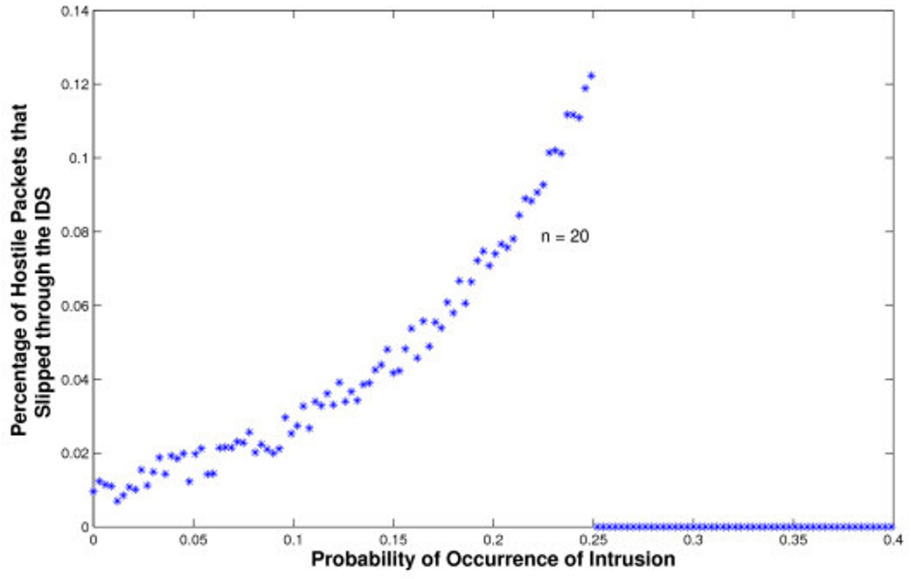

Figure 5 The percentage of hostile packets that "slipped" through the IDS as a function of P_H, for n = 20.

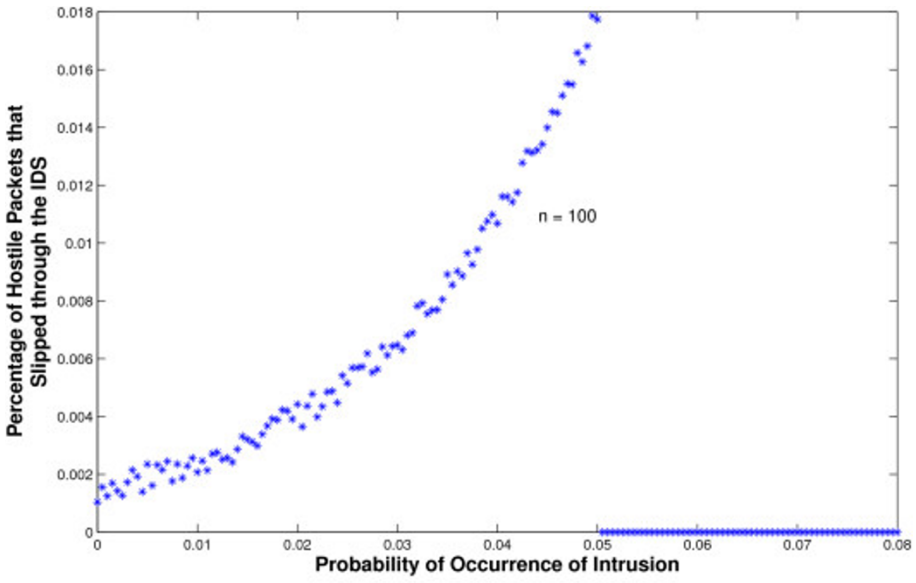

Figure 6 The percentage of hostile packets that "slipped" through the IDS as a function of P_H, for n = 100.

outright intrusive attack is launched, the value of P_H increases substantially above the threshold (which was chosen to be 0.25 for the n = 20 case and 0.05 for the n = 100 case). As the graphs in Figures 5 and 6 show, all the malicious packets were indeed detected and quarantined as P_H exceeded the calculated threshold. It is important to note here that there is essentially no difference between the data in Figures 5 and 6. The value of n was irrelevant, as the graphs clearly show, since all the intrusive activity in these tests originated from an external source (i.e., no intrusive activity originated from within the LAN).

Figure 7 shows a histogram of the number of hostile packets per source IP address (i.e., per hostile user) that slipped through the IDS (a total of 100 source IP addresses were generated in the test). As the histogram shows, the maximum number of hostile packets for a single user that slipped through the IDS was three. It is to be added that those 3 hostile packets were non-contiguous packets. As is well known, three hostile packets for a single user cannot initiate any serious intrusive process on a network [33]. The above results clearly demonstrate that the selective inspection approach is a highly effective alternative to the common technique of blindly inspecting all Internet traffic. Obviously, after obtaining a rough estimate of the important threshold of P_H from Eq. (13), the value of the threshold can be fine-tuned to meet a more lenient or a more restrictive IDS policy. As demonstrated here, a rough estimate based on Eq. (13) does in fact result in very good performance. Finally, it should be mentioned that the link speed used in the tests described above was quite low (10 Mbps), as the IDS software was not configured to drop any packets on the basis of traffic intensity. This kind of test was performed with the SNORT software package and is described further below.

3.4. Testing of an optimized SNORT[d] software package

Snort is a very popular open-source IDS software solution [1]. Snort optimally runs under the operating system FreeBSD (an open-source variant of Unix). An optimal setup for testing the performance of Snort has been suggested in a number of references [34,35]. This setup is shown in Figure 8, and it is the configuration that was chosen for the present analysis.

Essentially, the traffic-generation workstation that was used in the previous test was augmented with an Agilent J3446E LAN Advisor to monitor the traffic between the workstation and the machine running Snort. The LAN Advisor includes the optional J2901A Gigabit Advisor. The machine running Snort is a workstation that is similar to the traffic-generation workstation and is equipped with a 1 Gb/s Network Interface Card. The purpose of using the Agilent equipment was to measure the packet speed, analyze the packet headers and to pinpoint compatibility problems on the Gigabit link between the Linux machine and the FreeBSD machine. Snort was first tested in the regular mode, where 100% of the traffic is fully inspected. It is well known that any software IDS drops packets at an increasing rate as the speed of the packets on the link increases. Snort has a reporting feature that provides the percentage of the packets dropped during any given period of time. Figure 9

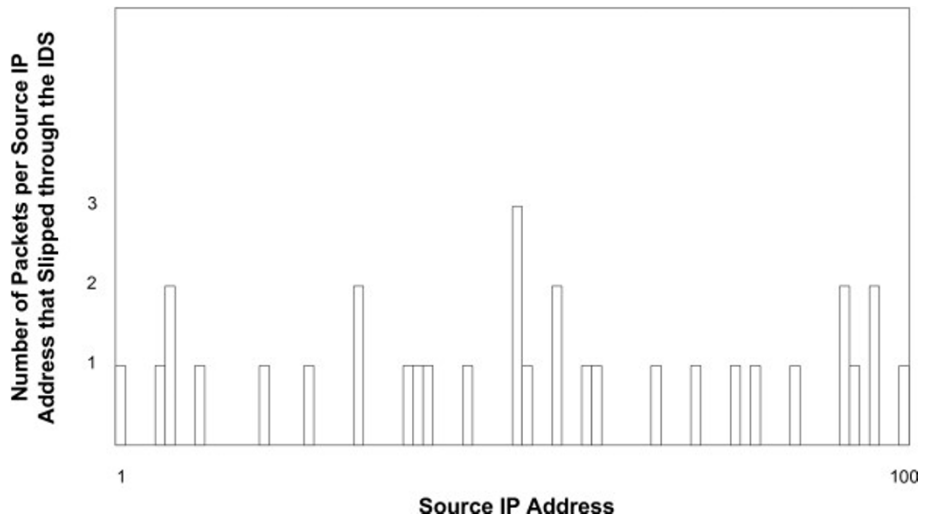

Figure 7 Histogram of the number of packets per hostile user that managed to slip through the IDS undetected.

shows the percentage of the packets dropped as the link speed was increased from 10 Mbps up to 1 Gb/s.

As the figure shows, the percentage of packets dropped was essentially negligible for bit rates up to 100 Mbps. That percentage grows considerably and reaches almost 90% at the full link speed of 1 Gb/s. These results correlate with results published previously by other authors [36,37]. The source code of Snort was subsequently modified so that packets can be selectively inspected according to the procedure described in the previous section (this task is not difficult since Snort is written in C). By inspecting packet samples predominantly from small flows, Snort did not drop any packets, even at the full link speed of 1 Gb/s. Figure 10 shows smoothed, best-fit plots, of the percentage of hostile packets that slipped through Snort as a function of P_H, as the link speed was varied. P_H was calculated from Eq. (16).

As Figure 10 shows, for a link speed of 10 Mbps, the percentage of hostile packets that slipped through Snort was essentially the same as the percentage shown in Figure 5. The percentage increases slightly at higher link speeds and reaches a maximum of about 0.2% (or 2 packets for every 1,000 malicious packets) at a link speed of 1 Gb/s. As the results clearly show, the effect of the link speed on this intrusion detection approach is essentially negligible.

Figure 11 shows the percentage utilization of CPU and memory, before and after the code enhancement, at the full link speed of 1 Gb/s.

4. Conclusion

The analysis of network intrusions as Markov chains disproves the common notion that it is necessary to fully inspect every packet entering a network in order to ensure security. The results shown here fully support the experimental results that were published recently by Androulidakis and Papavassiliou [11,12]. The analysis, together with the testing data, demonstrates that it is sufficient to inspect only a small number of packets sampled predominantly from small flows, as long as the probability of occurrence of intrusion P_H is below a critical threshold

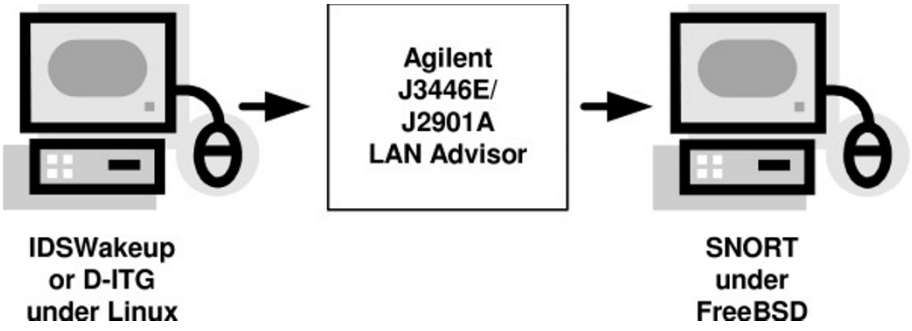

Figure 8 Setup for testing an optimized Snort IDS software package.

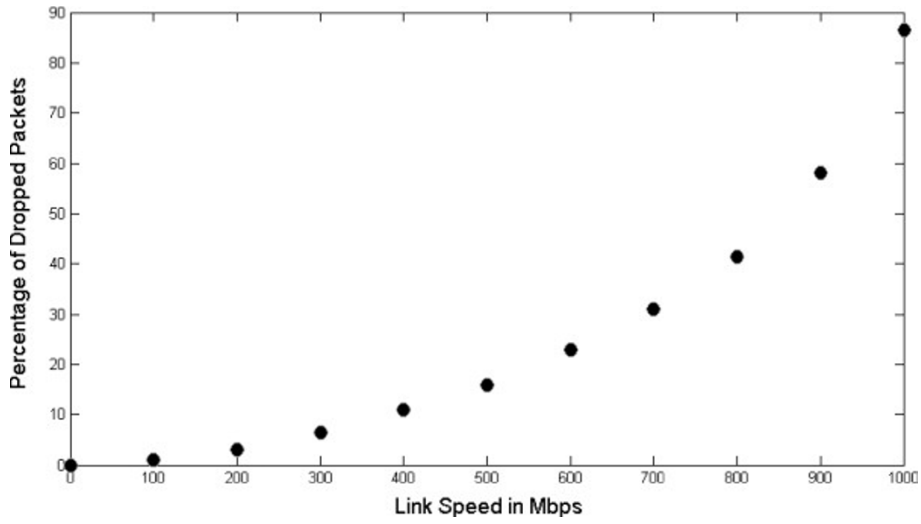

Figure 9 Test of Snort in the regular mode (100% inspection). Snort drops packets at an increasing rate as the link speed is increased.

that is determined from Eq. (13) and calculated in real time from Eq. (16). The implications of the research presented here for software IDS solutions such as SNORT are substantial, as the selective inspection of packets allows the IDS to handle high speed links without dropping any packets. Hence, it is essentially possible for most of the time to eliminate the speed bottleneck problem without compromising security.

EndNotes

[a]Here, it is important to point out that a "process", as defined in the previous section, can be started with one or more packets. The procedure for calculating P_H, however, will be based on the direct inspection of packets.

[b]It can be argued that the relationship between P_H and H(normalized) should be a proportionality relationship, not an exact equality as shown in Eq. (16). However, the

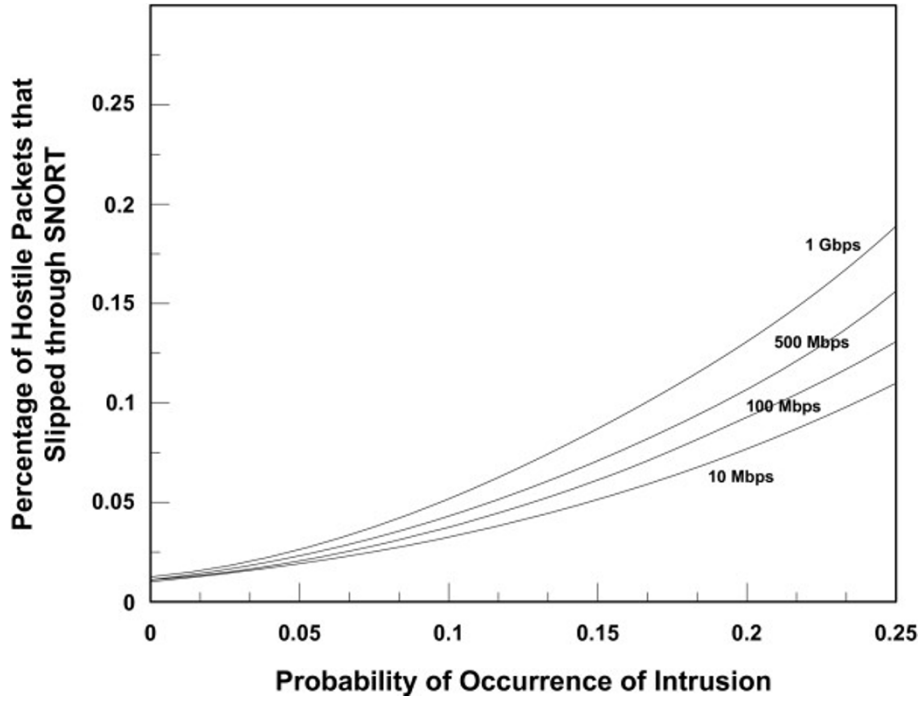

Figure 10 The percentage of hostile packets that slipped through Snort as a function of P_H, for link speeds ranging from 10 Mbps to 1 Gb/s.

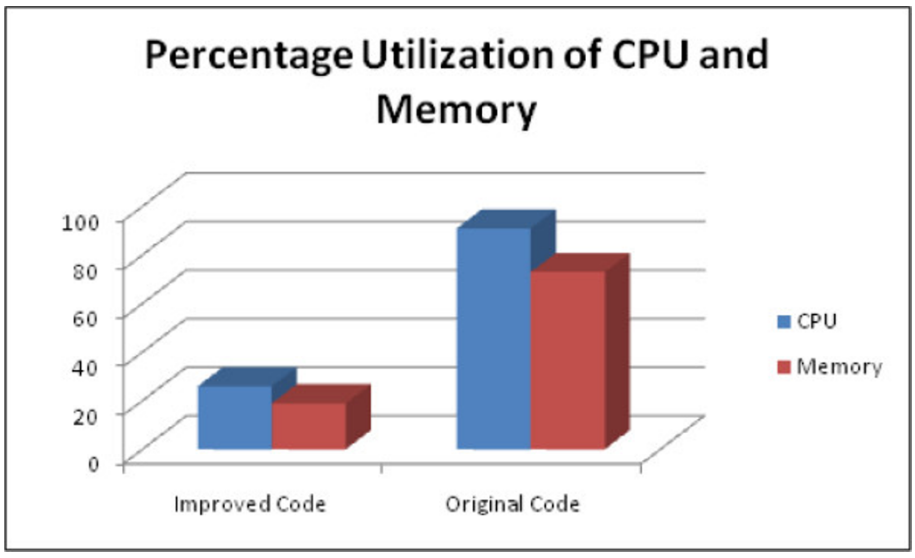

Figure 11 Percentage utilization of CPU and memory by SNORT, before and after the code enhancement, at the full link speed of 1 Gb/s.

objective of this work is to obtain a reasonable estimate for the likelihood of the occurrence of intrusion, not to seek idealized, precise mathematical relationships. In reality, due to the nature of the problem, the mathematical framework presented here is not meant to be highly precise, but it can be made sufficiently precise with the inclusion of experimental data.

[c]It is to be pointed out that DoS attacks can be identified only from the packet headers.

[d]SNORT is a registered trademark of Sourcefire, Inc.

Competing interests

The authors declare that they have no competing interests.

Received: 8 January 2011 Accepted: 19 September 2011
Published: 19 September 2011

References

1. Sourcefire, Inc., Snort: The Open Source Network Intrusion Detection System. http://www.snort.org (2007)
2. Secutrain, Inc., SecureNet Pro: Protection Against Internet Security Threats. http://www.intrusion.com (2007)
3. Hogwash Intrusion Detection System. http://hogwash.sourceforge.net/ (2007)
4. Symantec, Inc., Symantec Host IDS: Scalable Intrusion Detection and Prevention Solution for Critical Servers. http://www.symantec.com (2007)
5. Checkpoint Ltd., IPS1: Robust and Accurate Intrusion Prevention. http://www.checkpoint.com (2007)
6. N Weaver, V Paxson, JM Gonzalez, The shunt: an FPGA-based accelerator for network intrusion prevention. Proc. 15th Ann. ACM Intl Symp. Field-Programmable Gate Arrays (FPGA 07). 292 (2007)
7. WJ Hwang, HC Roan, YN Shih, CT DanLo, CM Ou, FPGA-based ROM-free network intrusion detection using shift-or circuit. J Embedded Comput. **3**(2):99 (2009)
8. JO Kephart, SR White, Directed graph epidemiological models of computer viruses. Proceedings of the 1991 IEEE Computer Society Symposium on Research in Security and Privacy. 343 (1991)
9. JO Kephart, SR White, Measuring and modeling computer virus prevalence. Proceedings of the 1993 IEEE Computer Society Symposium on Research in Security and Privacy. 2 (1993)
10. Y Wang, C Wang, Modeling the effects of timing parameters on virus propagation. Proceedings of the 2003 ACM Workshop on Rapid Malcode. 61 (2003)
11. G Androulidakis, S Papavassiliou, Improving network anomaly detection via selective flow-based sampling. IET Commun. **2**(3):399 (2008). doi:10.1049/iet-com:20070231
12. G Androulidakis, V Chatzigiannakis, S Papavassiliou, Network anomaly detection and classification via opportunistic sampling. IEEE Netw. **23**(9):6 (2009)
13. G Vert, DA Frincke, JC McConnell, A visual mathematical model for intrusion detection. Proceedings of the 21st NIST-NCSC National Information Systems Security Conference. 1 (1998)
14. Z Zhang, J Li, C Manikopoulos, J Jorgenson, J Ucles, A hierarchical anomaly network intrusion detection system using neural network classification. Proceedings of the 2nd Annual IEEE Systems, Man, Cybernetics Information Assurance Workshop (IAW 2001). 6 (2001)
15. M Kodialam, TV Lakshman, Detecting network intrusions via sampling: a game theoretic approach. INFOCOM–22nd Annual Joint Conference of the IEEE Computer and Communications Societies. 1880 (2003)
16. H Song, JW Lockwood, Multi-pattern signature matching for hardware network intrusion detection systems. GLOBECOM–IEEE Global Telecommunications Conference. 5 (2005)
17. D Subhadrabandhu, S Sarkar, F Anjum, A framework for misuse detection in ad hoc networks–Part I. IEEE J Sel Areas Commun. **24**(2):274 (2006)
18. D Subhadrabandhu, S Sarkar, F Anjum, A framework for misuse detection in ad hoc networks–Part II. IEEE J Sel Areas Commun. **24**(2):290 (2006)
19. S Jin, DS Yeung, X Wang, Network intrusion detection in covariance feature space. Pattern Recogn. **40**(8):2185 (2007). doi:10.1016/j.patcog.2006.12.010
20. CH Sauer, KM Chandy, Computer Systems Performance Modeling. (Prentice Hall, Englewood Cliffs, NJ, 1981)
21. H Kobayashi, Modeling and Analysis: an Introduction to System Performance Evaluation Methodology. (Addison Wesley, Reading, MA, 1978)
22. FM Reza, An Introduction to Information Theory. (Dover, New York, NY, 1994)
23. TM Cover, JA Thomas, Elements of Information Theory. (Wiley, New York, NY, 1999)
24. D Moore., *et al*, Inside the slammer worm. IEEE Sec Privacy. **1**(4):33 (2003). doi:10.1109/MSECP.2003.1219056
25. N Hohn, D Veitch, Inverting sampled traffic. IEEE/ACM Trans Netw. **14**(1):68 (2006)

26. J Mai., *et al*, Impact of packet sampling on portscan detection. IEEE J Sel Areas Commun. **24**(12):2285 (2006)
27. J Mai., *et al*, Is sampled data sufficient for anomaly detection. Internet Measurement Conf., Rio de Janeiro, Brazil. 165 (2006)
28. P Barford, D Plonka, Characteristics of network traffic flow anomalies. Proceedings of the 1st ACM SIGCOMM Internet Measurement Wksp., San Francisco, CA. 69 (2001)
29. A Sridharan, T Ye, S Bhattacharyya, Connectionless Port Scan Detection on the Backbone. IEEE IPCCC Malware Wksp., Phoenix, Az. 1 (2006)
30. PZ Peebles, Probability, Random Variables, and Random Signal Principles. (McGraw Hill, New York, NY, 1993)
31. IDS Wakeup: A collection of tools for testing network intrusion detection systems. http://www.hsc.fr/ressources/outils/idswakeup/index.html.en (2007)
32. A Botta, A Dainotti, A Pescape, Multi-Protocol and Multi-Platform Traffic Generation and Measurement. IEEE INFOCOM, Anchorage, Alaska. 12http://www.grid.unina.it/software/ITG/ (2007)
33. K Lan, A Hussain, D Dutta, Effect of malicious traffic on the network. Proceeding of Passive and Active Measurement Workshop (PAM). 1 (2003)
34. J Koziol, Intrusion Detection with SNORT. (Pearson Education, Upper Saddle River, NJ, 2003)
35. K Cox, C Greg, Managing Security with SNORT and IDS Tools. (O'Reilly Media, Sebastopol, CA, 2004)
36. W Lee, JB Cabrera, A Thomas, N Balwalli, S Saluja, Y Zhang, Performance adaptation in real-time intrusion detection systems. Proceedings of the Fifth International Symposium on Recent Advances in Intrusion Detection (RAID 2002), Lecture Notes in Computer Science, Zurich, Switzerland. (2002)
37. L Schaelicke, T Slabach, B Moore, C Freeland, Characterizing the performance of network intrusion detection sensors. Proceedings of the Sixth International Symposium on Recent Advances in Intrusion Detection (RAID 2003), Lecture Notes in Computer Science, Berlin-Heidelberg-New York. (2003)

doi:10.1186/1687-417X-2011-2
Cite this article as: Bakhoum: **Intrusion detection model based on selective packet sampling.** *EURASIP Journal on Information Security* 2011 **2011**:2.