Embedded Markov Process based Model for Performance Analysis of Intrusion Detection and Prevention Systems

Khalid Alsubhi*, Mohamed Faten Zhani*, Raouf Boutaba*†
*David R. Cheriton School of Computer Science, University of Waterloo, Ontario, Canada
†Division of IT Convergence Engineering, POSTECH, Pohang, KB 790784, Korea
email:{kaalsubh, mfzhani, rboutaba}@uwaterloo.ca

***Abstract*—Intrusion Detection and/or Prevention Systems (IDPSs) are now a crucial defensive measure to defend against attacks intended to breach the security and operation of enterprise information systems. The IDPS configuration can, however, have a negative impact on network performance in terms of end-to-end delay and packet loss. This paper proposes an analytical queuing model based on the embedded Markov chain which analyzes the performance of the IDPS and evaluates its impact on performance. Through extensive simulations, we validate the proposed model and the numerical equations that estimate various performance metrics. Our results show that this model can be leveraged to assess and set up an effective configuration for the IDPS, achieving simultaneously the trade-off between security enforcement levels on one side and network Quality of Service (QoS) requirements on the other.**

***Index Terms*—Security Performance Evaluation, Intrusion Detection and Prevention Systems, Markov Chain Modeling.**

I. Introduction

In addition to other utilities, Intrusion Detection and/or Prevention Systems are a vital defensive measure against a range of malicious exploits [1], [2]. The necessity of defending against a range of attacks is paramount for the use of any security technology is. Avoiding unnecessary performance degradation in a network when extensive security measures are used is another important consideration. This necessitates a combination of security at one end of the spectrum and efficiency and speed at the other end, and this combination is not generally well served by current IDPSs.

Intrusion Detection Systems (IDSs) examine packets sent over networks and raise warnings when malicious content is discovered. Intrusion Prevention Systems (IPSs), however, do have the additional ability to defend against such attacks. IDSs satisfy performance requirements, but their defensive abilities are less than optimal. Somewhat mitigating this, IPSs do protect networks through the rejection of packets that correspond to known malicious patterns, but as attacks increase, network performance can be affected. Combining the best of both utilities, Intrusion Detection and/or Prevention Systems (IDPSs) both detect malicious activity and block the most harmful ones.

The design of an IDPS poses many challenges as to how to identify attack signatures, how to improve the detection engine and how to manage the system. A large body of work has addressed these issues; however, little work has studied the impact of deploying an IDPS on network performance in terms of processing delay, throughput, and packet loss [3], [4], [5], [6]. Ironically, while the IDPS can efficiently detect and prevent many attacks that can compromise network performance, it may itself cause the performance degradation. For instance, the processing delay of the IDPS can increase significantly as the size of the signature database grows. In addition, the IDPS may not be able to cope with the increase in the amounts of traffic, mainly because of the limited resources in terms of CPU and memory (depending on whether the IPDS is installed in a server or a stand-alone system). This particular case leads to an increase in queue length, resulting in higher waiting times for packets, and eventually many losses. Thus, the IDPS becomes the network bottleneck, and its configuration is no longer appropriate. To decrease queuing delay, processing time and packet loss inside the IDPS, the configuration should be adjusted such that the complexity of the detection engine is reduced. In other words, we decrease the attack coverage in order to provide better networking performance. As a consequence, the operator faces a trade-off between security enforcement levels on one hand and Quality of Service (QoS) requirements on the other. Thus, it is crucial to study the impact of different IDPS configurations on network performance and select the one that achieves at the same time security objectives and QoS requirements.

This paper aims to analyze the performance of the IDPS for different configurations and under different traffic characteristics. Different from existing works on IDPS performance analysis, we develop an analytical model for the system based on embedded Markov chain, which can allow the prediction of the impact on network performance. We then leverage the model to provide mathematical derivations of key performance metrics: namely the throughput, queuing delay, system utilization, and packet loss at the IDPS level. We define many configurations for the IDPS that reflect security enforcement levels and we study their effect on the network performance metrics under different traffic intensities. The analytical model is then validated through extensive simulations. In fact, our model allows security administrators to strike a balance be-

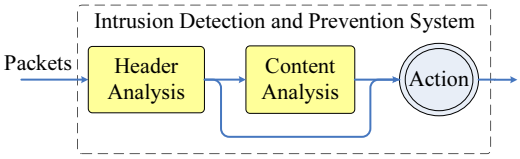


Fig. 1: Analysis Tasks for IDPS

tween security level and network performance.

The remainder of the paper is organized as follows. We proceed with an overview of related work in Section II. Section III presents our analytical model for the IDPS based on a finite queuing system. In Section IV, we provide the performance evaluation of the IDPS analysis model. Finally, we draw our conclusions and follow up with potential future work in Section V.

II. Related Work

In the literature, a plethora of research has addressed the trade-off between the security level and IDPS resource consumption (*i.e.*, CPU and memory) [7], [4], [8], [9]. Many studies showed that the IDPS heavily relies on deep packet inspection and is indeed a performance bottleneck [10], [11]. For instance, Dreger et al. study the trade-off between security level and resource consumption [8], [9]. Lee et al. [12] put forward a method to determine the performance of an IDPS through quantifying the benefits and drawbacks of each detection rule. In order to reach the best possible configuration for an overloaded IDS, they propose an algorithm with heuristics. However, they do not rely on any analytical model, thus it is hard to predict in advance the resulting performance for a given configuration.

On the other hand, the impact of the IDPS on network performance has received less attention. Setting the right configuration parameters for the IDPS while avoiding the drawbacks on the quality of service is still a challenging problem. Hess et al. [13] try to mitigate the impact of IPS services on the end-to-end delay. They propose an architecture that allows the running of an overlay network of IPSs by means of programmable routers. However, such a framework cannot be easily deployed as it requires programmability which is not a common feature in commercial off-the-shelf routers. Salah et al. [7] derived an analytical queuing model based on the embedded Markov chain in order to analyze the performance of rule-based firewalls. This model can not be applied in the context of IDPSs, however, because they have different processing stages.

To the best of our knowledge, none of the existing work has proposed queuing-based modelling for IDPS systems with two stages, *i.e.*, header and content analysis. This paper derives an embedded Markov chain model, and thereby makes the setting of the IDPS configuration possible analytically. The security administrator is then able to select the appropriate configuration that achieves the trade-off between security and network performance.

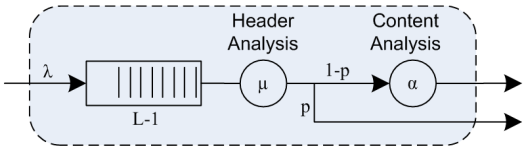


Fig. 2: Tandem Queue Model with Blocking

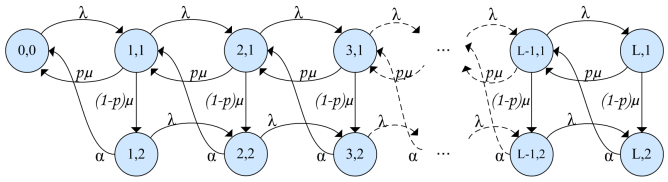


Fig. 3: State Transition Diagram for Finite Queuing Model

III. Analytical Model

In this work, we consider an architecture similar to the one adopted by Snort, the widely-used IDPS [14]. As shown in Figure 1, the IDPS consists mainly of two stages, namely the header analysis and the content analysis. The header analysis stage examines packets for many types of malicious activity. For instance, this first analysis can detect attacks that exploit fragmentation vulnerabilities, such as the Ping of Death attack where attackers use many small fragmented ICMP packets, such that their assembling results in a huge packet that exceeds the maximum allowable size for an IP datagram [14]. If a particular packet is identified as belonging to such an attack, there is no need to do any further analysis. Thus, the first stage processing result allows the IDPS to decide to either release the packet to the network without any further analysis or to forward it to the content analysis for further checking. The content analysis stage is usually a rule-checking engine that uses a signature database to determine if the examined packet belongs to any malicious traffic. Generally, this second stage is more time-consuming than the first one (*e.g.,* four times higher as reported in [15]).

In the following, we present a finite queuing model for this architecture and we use it to analyze the performance of the IDPS.

A. Finite Queuing Model

In order to model the IDPS, we propose a two-stage embedded Markov chain [16], [17], [18], [19] as shown in Figure 2. The model consists of a FIFO queue and two servers. The first and second servers correspond respectively to the header analysis and content analysis stages.We assume that packets arrive at the IDPS according to a Poisson process with an arrival rate λ. We assume a fixed packet size in order to simply the analysis. Packets join a finite queue of a maximum size $L-1$ (such as $L \in \mathbb{N}^*$). Only one packet at a time is passed to the header analysis stage that has an average service time $1/\mu$. When the header analysis is completed, the packet either leaves the system with an *early decision probability* p or moves to the second stage, with a probability $(1-p)$. The content header process has an average service

time $1/\alpha$. Service times for both stages are exponentially distributed.Packets are considered lost only when the buffer becomes full and, they cannot join the IDPS system. In order to make the analytical solution feasible, we consider Poisson arrivals, exponential services, and fixed packet sizes. This applies only for certain types of traffic as reported in [20]. Considering other distributions and variable packet sizes is part of our future work.

On the other hand, we assume that the header analysis stage can not accept any new packet if the content analysis process is already busy. Thus, the execution of the two stages is mutually exclusive; *i.e.,* if one of the stages is running, the other is idle. This assumption is realistic for two reasons. The first reason is that the CPU is usually executing one task at a time. The second reason is that if we allow the pre-processor stage to accept packets while the rule-checking stage is busy, we need to introduce another queue at the second server. However, this situation can result in out-of-order packets, as service times of the two stages are different and some packets can directly leave the system from the first stage. In practice, the service time of packet header analysis is much lower than that of the content checking ($\alpha < \mu$), since the underlying idea behind the header analysis stage is to avoid detailed rule checking if no anomaly is detected.

We represent the behavior of the multi-stage service queuing system by a finite queuing model based on the embedded Markov chain process with a state space $S = \{(n, m), 0 < n \leq L, m = \{1, 2\}\}$, where n denotes the number of packets in the system and m denotes the stage which the IDPS is performing. In particular, in the first stage ($m = 1$), the IDPS is performing the packet processing task, and when $m = 2$, the IDPS is performing the rule-checking process. The queuing system has a buffer size of $L - 1$. State $(0, 0)$ represents the special case when the IDPS is idle.

From the state transition diagram depicted in Figure 3, we can infer steady-state equations. Thus, for the initial states (0,0), (1,1), and (1,2) we have:

$$\begin{aligned}
0 &= -\lambda q_{0,0} + p\mu q_{1,1} + \alpha q_{1,2} && \text{at state } (0,0)\\
0 &= -(\lambda+\mu) q_{1,1} + \lambda q_{0,0} + p\mu q_{2,1} + \alpha q_{2,2} && \text{at state } (1,1)\\
0 &= -(\lambda+\alpha) q_{1,2} + (1-p)\mu q_{1,1} && \text{at state } (1,2)
\end{aligned}$$

For intermediate states $(n, 1)$ and $(n, 2)$, where $n \in [2, L-1]$, we have:

$$\begin{aligned}
0 &= -(\lambda+\mu) q_{\text{n,1}} + \lambda q_{\text{n-1,1}} + p\,\mu\, q_{n+1,1} + \alpha q_{\text{n+1,2}} && \text{at state } (n,1)\\
0 &= -(\lambda+\alpha) q_{\text{n,2}} + \lambda q_{\text{n-1,2}} + (1-p)\mu q_{\text{n,1}} && \text{at state } (n,2)
\end{aligned}$$

At the boundary states $(L, 1)$ and $(L, 2)$, steady-state equations are expressed as follows:

$$\begin{aligned}
0 &= -\mu q_{\text{L,1}} + \lambda q_{\text{L-1,1}} && \text{at state } (L,1)\\
0 &= -\alpha q_{\text{L,2}} + \lambda q_{\text{L-1,2}} + (1-p)\mu q_{\text{L,1}} && \text{at state } (L,2)
\end{aligned}$$

Based on these equations, we would like to express $q_{\text{n,1}}$ and $q_{\text{n,2}}$ in terms of $q_{0,0}$. It is straightforward to calculate the probabilities of the initial and boundary states. Thus, from state equations $(0, 0)$ and $(1, 2)$, the probabilities $q_{1,1}$ and $q_{1,2}$ can be expressed in terms of $q_{0,0}$ as follows:

$$\begin{aligned}
q_{1,1} &= \frac{(\alpha+\lambda)\lambda}{(\alpha+p)\mu} q_{0,0}\\
q_{1,2} &= \frac{\lambda - p\lambda}{\alpha + p\lambda} q_{0,0}
\end{aligned}$$

In order to calculate the state probabilities $q_{\text{n,1}}$ and $q_{\text{n,2}}$ in terms of $q_{0,0}$ where $n \in [1, L-1]$, we define $(w_{\text{n,1}})_{n\in[1,L-1]}$ and $(w_{\text{n,2}})_{n\in[1,L-1]}$ such as:

$$\begin{cases}
w_{0,1} = w_{0,2} = 1 & \\
w_{1,1} = \frac{(\alpha+\lambda)\lambda}{(\alpha+p)\mu} & \\
w_{1,2} = \frac{\lambda - p\lambda}{\alpha + p\lambda} & \\
w_{\text{n,1}} = \frac{(\alpha+\lambda)(\lambda+\mu)}{(p\lambda+\alpha)\mu} w_{\text{n-1,1}} - \frac{(\lambda+\alpha)}{(\lambda p+\alpha)} w_{\text{n-2,1}} & 2 \leq n < L\\
\qquad - \frac{(\lambda\alpha)}{(\lambda p+\alpha)\mu} w_{\text{n-1,2}} & \\
w_{\text{n,2}} = \frac{\lambda}{\lambda+\alpha} w_{\text{n-1,2}} + \frac{(1-p)\mu}{\lambda+\alpha} w_{\text{n,1}} & 2 \leq n < L
\end{cases} \tag{1}$$

Using state equations $(n, 1)$ and $(n, 2)$ along with the definition of $(w_{\text{n,1}})_{n\in[1,L-1]}$ and $(w_{\text{n,2}})_{n\in[1,L-1]}$, we can express $q_{\text{n,1}}$ and $q_{\text{n,2}}$ in terms of $q_{0,0}$ as follows:

$$\begin{aligned}
q_{\text{n,1}} &= w_{n,1} q_{0,0}\\
q_{\text{n,2}} &= w_{n,2} q_{0,0}
\end{aligned} \tag{2}$$

Furthermore, the probabilities at the boundaries are calculated in terms of $q_{0,0}$ based on equations $(L, 1)$, $(L, 2)$ and (1) as follows:

$$q_{\text{L,1}} = \begin{cases} \left(\frac{\lambda}{\mu}\right) w_{\text{L-1,1}}\, q_{0,0} & L > 1\\ \left(\frac{\lambda}{\mu}\right) q_{0,0} & L = 1 \end{cases}$$

$$q_{\text{L,2}} = \begin{cases} \frac{\lambda}{\alpha}(w_{\text{L-1,2}} + (1-p) w_{\text{L-1,1}}) q_{0,0} & L > 1\\ \left(\frac{(1-p)\lambda}{\alpha}\right) q_{0,0} & L = 1 \end{cases}$$

The next step is to determine $q_{0,0}$. To this purpose, we use the normalization condition, which is expressed as follows:

$$q_{0,0} + \sum_{n=1}^{L} (q_{\text{n,1}} + q_{\text{n,2}}) = 1 \tag{3}$$

Using equation (2), we obtain:

$$\begin{aligned}
& q_{0,0} + \sum_{n=1}^{L} (w_{\text{n,1}} + w_{\text{n,2}}) q_{0,0} = 1\\
\Rightarrow & q_{0,0} = \frac{1}{1 + \sum_{n=1}^{L} (w_{\text{n,1}} + w_{\text{n,2}})}
\end{aligned} \tag{4}$$

Based on equations (1), (2) and (4), it is possible to calculate the steady-state probabilities, and then we are able to determine different performance metrics of the system at the steady state. The next subsection discusses those metrics and their equations.

B. Performance metrics

In this subsection, we identify the metrics that should be measured at the IDPS system level, and have an impact on the network performance. Particularly, end-to-end delay and packet loss ratio can be directly affected respectively by the average time spent in the IDPS per packet and the packet loss ratio at the IDPS level. Furthermore, other important metrics are also considered in our study such as the mean system throughput, the average number of packets in the system, and packet average waiting delay. In the following, we provide the equation of each of those metrics in function of the steady-state probabilities [17].

The average of the IDPS throughput γ is the average number of packets (per second) leaving the IDPS system, either from the first stage or the second one, and regardless of the decision of the IDPS with respect to packets. It is expressed as follows:

$$\gamma = p\mu \sum_{n=1}^{L} q_{\text{n,1}} + \alpha \sum_{n=1}^{L} q_{\text{n,2}} \tag{5}$$

The packet loss probability q_{lost} is the probability of being in the state $(L, 1)$ or $(L, 2)$. This means that the queue is full, and as a consequence incoming packets will not be admitted. It is given by:

$$q_{\text{lost}} = q_{\text{L,1}} + q_{\text{L,2}} \tag{6}$$

The average number of packets $\bar{X}$ in the system can be expressed as follows:

$$\bar{X} = \sum_{n=1}^{L} n(q_{\text{n,1}} + q_{\text{n,2}}) \tag{7}$$

The average time that a packet spends in the system W_s is then expressed using $\bar{X}$ and γ as:

$$W_s = \frac{\bar{X}}{\gamma} \tag{8}$$

The average service time of the two stages denoted W_a is given by:

$$W_a = \frac{1}{\mu} + \frac{1-p}{\alpha}$$

The average time spent by a packet in the queue W_q can be measured as follows:

$$W_q = W_s - W_a \tag{9}$$

Based on those equations, we compute the analytical values of all the performance metrics. The next section is dedicated to performance evaluation and comparison with simulations results.

TABLE I: Security enforcement levels and their corresponding processing time

Security level	Service time	
	Stage one($1/\mu$)	Stage two($1/\alpha$)
1	$0.5\,\boldsymbol{\mu s}$	$4\,\boldsymbol{\mu s}$
2	$0.5\,\boldsymbol{\mu s}$	$8\,\boldsymbol{\mu s}$
3	$0.5\,\boldsymbol{\mu s}$	$12\,\boldsymbol{\mu s}$
4	$5\,\boldsymbol{\mu s}$	$12\,\boldsymbol{\mu s}$

IV. Performance Evaluation and Results

This section has two objectives. First, we aim to validate our analytical model through extensive simulations. Second, we would like to analyze the effect of different security levels on network performance. In the following, we describe the settings of the different security enforcement levels, and then analyze the effect of the packet arrival rates, the queue size and the early decision probability p on the studied performance metrics. For all experiments, we provide the results determined from the analytical model and from simulations.

A. Settings

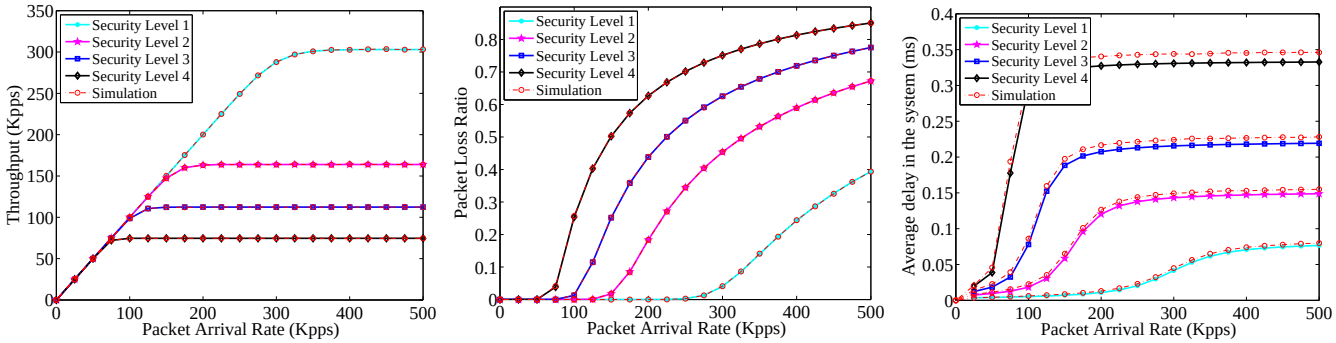
For the sake of our experiment, we defined four configurations that reflect different levels of the detection capabilities of the IDPS, and Table I illustrates the selected configuration for each. While setting those values, we consider that the average service time at stage 2 is an increasing function of the number of selected rules. This is motivated by our previous work, where we provide an analytical model that relates the service time to the number of rules [6]. It was also in other works using realistic data [21].

At the first three levels, we fixed the processing time at the first stage. As the security level is increased, the IDPS enlarges the selection of checking rules to improve the security coverage, and therefore the server processing time at stage 2 increases. For instance, level 1 corresponds to the minimum detection level where a small set of rules is checked at stage 2, whereas levels 2 and 3 increase the size of the rules database to provide better detection of malicious traffic. In addition, security level 4 is a particular case where both stages have to check a larger number of rules in order to increase the protection capability of the IDPS especially at stage 1.

B. Results

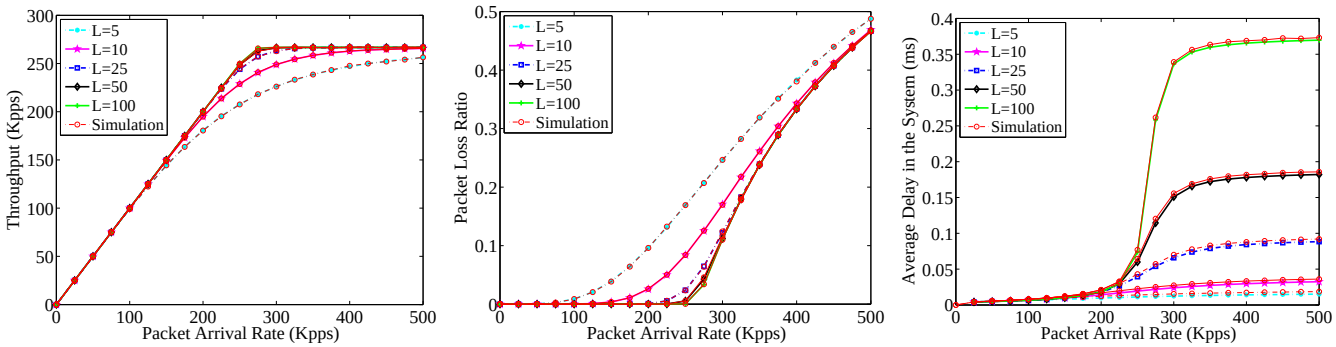
We conducted several experiments in order to validate the analytical results. We implemented discrete-event simulation of a finite queueing for the system using Matlab [22]. Every simulation goes through independent sub-runs with different initial seeds (after having discarded the transient part), and it is terminated when the confidence interval of 95% is constructed [23].

In our first experiment, we evaluate the accuracy of the proposed queuing model for a fixed queue size ($L = 25$), a fixed early decision probability ($p = 0.3$) and for different security enforcement levels. The results are depicted in Figure 4. Each figure shows a performance metric calculated, at every arrival rate, based on the analytical model compared



(a) Throughput (b) Packet loss ratio (c) Average time spent in the system per packet

Fig. 4: Performance metrics versus packet arrival rate for different Security Enforcement Levels ($L = 25$, $p = 0.3$).



(a) Throughput (b) Packet loss ratio (c) Average time spent in the system per packet

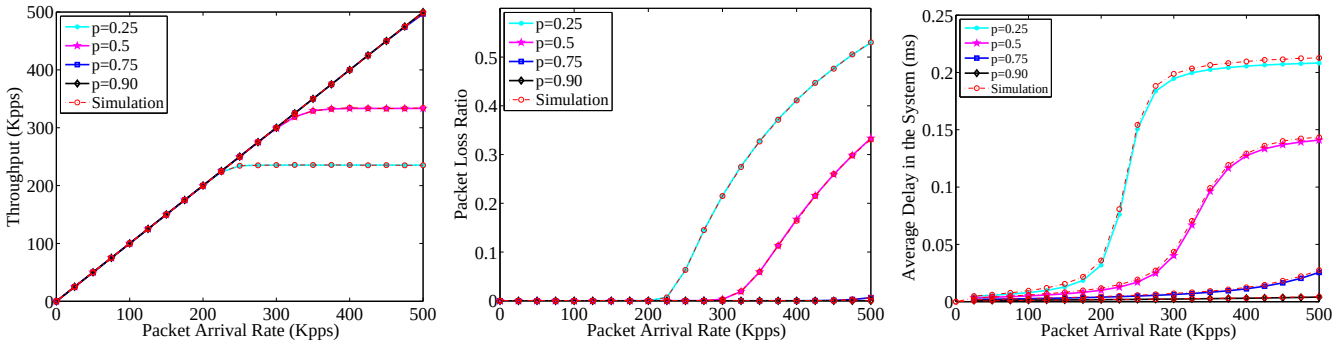
Fig. 5: Performance metrics versus packet arrival rate for different queue size values (*Security Level* 1, $p = 0.3$).

with its corresponding value as computed by simulation. The first observation is that the values obtained by simulations are almost identical to the analytical ones (notice that in all figures, circles, which represent values obtained by simulations, are overlapping with the points which represent the values obtained using the analytical model).

On the other hand, the figures show the effect of the security level on the performance metrics. As the security level is set higher, we can observe that the throughput is decreasing (Fig. 4a) while the packet loss ratio (Fig. 4b) and the average time spent in the system per packet (Fig. 4c) are getting higher. This shows clearly that improving security coverage is at the expense of performance. The degradation of performance is due to the extra analysis carried by the IDPS in order to cover more checking rules.

The second set of experiments was conducted to evaluate the effect of the queue size (L). Therefore, we fixed other parameters like the security level (set to 1) and probability (set to 0.3). Figure 5 shows the different metrics. It shows that although increasing the size of the queue can slightly improve the throughput (Fig. 5a), the packet loss ratio does not change so much when the traffic rate gets higher (Fig. 6b). More importantly, a high queue size can cause a significant delay per packet, as shown in Figure 5c. We can leverage such results practically in order to configure the queue size of an IDPS that uses security level 1 and a probability of $p = 0.3$. As a practical example, assuming that to achieve the QoS target, we require a certain performance at the IDPS level, for example, a packet loss ratio of no more than 0.1, a processing time that does not exceed 0.03 ms, and a throughput higher than 225 Kpps. Knowing that the incoming traffic is fluctuating between 200 and 300 Kpps, we can infer from the figures 5a, 6b and 5c that setting the queue size to 10 packets achieves the required QoS.

The final set of experiments investigates the effect of the early decision probability (p) on the different performance metrics (Fig. 6). From Figure 6a, it can be seen that for a high packet arrival rate, a high value of p can increase the IDPS throughput. This can be explained by the fact that the header analysis is able to take the decision upon the receival of the packet without the need for a content analysis process. Thus, the throughput increases while the average packet delay in the system is reduced. As a consequence, there are fewer packets in the queue and the packet loss ratio is almost zero, as shown in Fig. 6b. On the other hand, a small value of p can result in more packets being directed to the second stage, and thereby incurring a higher time spent in the system per packet. In this case, there are more packets waiting in the queue and the loss ratio can easily increase (Fig. 6b). These experiments



(a) Throughput (b) Packet loss ratio (c) Average time spent in the system per packet

Fig. 6: Performance metrics versus packet arrival rate for different early decision propability (*Security Level* 1, $L = 25$).

demonstrate that if the header analysis can efficiently take a decision on the malicious traffic, and thereby avoiding the second stage, it can significantly reduce packet loss as well as the average time spent in the system. Furthermore, these results can also help to choose the appropriate p value that can make the IPDS satisfy QoS requirements. However, this depends on whether the administrator can control the complexity and the accuracy of the header analysis stage.

V. Conclusion

Although intrusion detection systems can shield the network from various attacks and malicious traffic, they can have drawbacks. That is, they can introduce significant delay and packet loss due to their large processing stages and eventually their inappropriate configuration. In this paper, we have tried to address this particular problem by evaluating the impact of such systems on the key performance metrics. To this end, we modelled the IDPS as an analytical queuing model based on embedded Markov chain. We also performed extensive simulations that demonstrated the accuracy of the model. In addition, we analyzed through the results the effect of the different configuration parameters of the IDPS on network performance. This study provides concrete examples of how to tune those parameters in order to control the impact on network performance. As such, this model not only allows the analysis of various performance metrics at the IPDS level, but it can be considered a valuable tool in setting up an appropriate configuration able to strike a balance between a high security enforcement level and network performance objectives.

Our future work is to leverage this model along with a feedback control-theoretic approach to allow dynamic adjustment of the IDPS configuration. Hence, it will be able to cope with network traffic dynamics while achieving the security-performance trade-off.

References

[1] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems(idps)," *National Institute of Standards and Technology (NIST)*, no. CSRC special publication SP 800-94, Feb 2007.

[2] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, "An overview of IP flow-based intrusion detection," *IEEE Communications Surveys Tutorials*, vol. 12, no. 3, pp. 343 –356, 2010.

[3] H. Debar, M. Dacier, and A. Wespi, "Towards a taxonomy of intrusion-detection systems," *Computer Networks*, vol. 31, no. 8, 1999.

[4] S. M. Bellovin and R. Bush, "Configuration management and security," *IEEE J.Sel. A. Commun.(JSAC)*, vol. 27, no. 3, 2009.

[5] L. Schaelicke, T. Slabach, B. Moore, and C. Freeland, "Characterizing the Performance of Network Intrusion Detection Sensors," *International Symposium on Recent Advances in Intrusion Detection(RAID)*.

[6] K. Alsubhi, N. Bouabdallah, and R. Boutaba, "Performance analysis in intrusion detection and prevention systems," in *IFIP/IEEE Integrated Network Management Symposium (IM)*, 2011.

[7] K. Salah, K. Elbadawi, and R. Boutaba, "Performance modeling and analysis of network firewalls," *IEEE Transactions on Network and Service Management*, vol. 9, no. 1, pp. 12–21, 2012.

[8] H. Dreger, A. Feldmann, V. Paxson, and R. a. Sommer, "Predicting the resource consumption of network intrusion detection systems," in *Recent Advances in Intrusion Detection*. Springer, 2008.

[9] H. Dreger, A. Feldmann, V. Paxson, and R. Sommer, "Operational experiences with high-volume network intrusion detection," in *Proceedings of the 11th ACM CCS*, 2004.

[10] D. Schuff and V. Pai, "Design alternatives for a high-performance self-securing ethernet network interface," in *IEEE International Parallel and Distributed Processing Symposium, IPDPS 2007*.

[11] C. Wu, J. Yin, Z. Cai, E. Zhu, and J. Chen, "A hybrid parallel signature matching model for network security applications using simd GPU," in *Advanced Parallel Processing Technologies*. Springer, 2009.

[12] W. Lee, J. Cabrera, A. Thomas, N. Balwalli, S. Saluja, and Y. Zhang, "Performance adaptation in real-time intrusion detection systems," in *Recent Advances in Intrusion Detection*. Springer, RAID, 2002.

[13] A. Hess, H. Geerdes, and R. Wessäly, "Intelligent distribution of intrusion prevention services on programmable routers," in *Proc. of IEEE INFOCOM, Barcelona, Spain*, 2006.

[14] M. Roesch, "Snort - lightweight intrusion detection for networks," in *LISA '99: Proceedings of the 13th USENIX conference on System administration*. Berkeley, CA, USA: USENIX Association, 1999.

[15] J. Cabrera, J. Gosar, W. Lee, and R. Mehra, "On the statistical distribution of processing times in network intrusion detection," in *IEEE Conference on Decision and Control (CDC)*, 2004.

[16] D. Gross and C. Harris, "Fundamentals of queueing theory. 1998," *ISBN: 0-471-17083-6*, pp. 244–247.

[17] H. Takagi, *Queueing analysis*. North-Holland Amsterdam, 1991.

[18] L. Kleinrock, "Queueing systems. volume 1: Theory," 1975.

[19] M. Neuts, *Matrix-geometric solutions in stochastic models: an algorithmic approach*. Dover Pubns, 1981.

[20] M. J. Karam and F. A. Tobagi, "Analysis of delay and delay jitter of voice traffic in the Internet," *Computer Networks*, vol. 40, no. 6, pp. 711–726, Dec. 2002.

[21] A. Tongaonkar, S. Vasudevan, and R. Sekar, "Fast packet classification for snort by native compilation of rules," in *Proceedings of the conference on Large installation system administration conference (LISA)*, 2008, pp. 159–165.

[22] MATLAB, *version 7.12.0 (R2011a)*. The MathWorks Inc., 2011.

[23] I. Mitrani, *Simulation techniques for discrete event systems*. Cambridge Univ Pr, 1982.