This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

Rule Mode Selection in Intrusion Detection and Prevention Systems

Khalid Alsubhi⋆, Yassir Alhazmi†, Nizar Bouabdallah‡, Raouf Boutaba⋆§
⋆David R. Cheriton School of Computer Science, University of Waterloo, Ontario, Canada
† Electrical and Computer Engineering, University of Waterloo, Ontario, Canada
‡INRIA, Campus universitaire de Beaulieu; 35042 Rennes Cedex, France
§Division of IT Convergence Engineering, POSTECH, Republic of South Korea
E-Mail:{kaalsubh,yalhazmi,rboutaba}@uwaterloo.ca; nizar.bouabdallah@inria.fr

***Abstract*—Protection and performance are the major requirements for any Intrusion Detection and/or Prevention System (IDPS). Existing IDPSs do not seem to provide a satisfactory method of achieving these two conflicting goals. Intrusion Detection Systems (IDSs) fulfill the network performance requirement but exhibit poor protection under successive attacks. On the other hand, Intrusion Prevention Systems (IPSs) can protect the network by dropping the malicious packets that match any attacking pattern; however, this can have a negative impact on network performance in terms of delay as the attacking patterns increase. This results in a tradeoff between security enforcement levels on one hand and the performance and usability of an enterprise information system on the other.**

This paper aims to study the impact of security enforcement levels on the performance and usability of an enterprise information system. We propose a rule mode selection optimization technique that aims to determine an appropriate IDPS configuration set in order to maximize the security enforcement levels while avoiding any unnecessary network performance degradation. Simulation was conducted to validate our proposed technique. The results demonstrate that it is desirable to strike a balance between system security and network performance.

***Index Terms*—Security Performance Evaluation, Security Management, Security Configuration.**

I. Introduction

Intrusion Detection and/or Prevention Systems (IDPSs) are a significant mechanism in defending against various attacks which may interfere with security and the proper operation of an enterprise information system [1]. These systems can be anomaly-based or signature-based . IDPSs based on signatures, such as SNORT [2] and BRO [3], are the most common and work with the foreknowledge of attack signatures to help distinguish between traffic which is malicious and that which is benign. IDPSs based on anomalies are different, since they learn the regular behavior of a system and then note when unusual behavior is detected. IDPSs can be host-based or network-based, and can function in either distributed or centralized clusters to provide better recognition of hostile traffic in a distributed networked system.

A primary requirement for the deployment of any security technology is the need to protect against a range of attacks. Another need relates to avoiding any needless performance degradation in the network when maximum safety measures are applied. This requires a balance between security on one hand, and speed and functionality on the other [4]. Current IDPSs do not tend to provide an adequate means of achieving these two contradictory needs. Network-based Intrusion Detection Systems (NIDSs) scrutinize copies of the packets sent over the network and raise flags whenever hostile content is found. In comparison to this method, Network-based Intrusion Prevention Systems (NIPSs) have the extra capacity of defending against the attacks. IDSs realize network performance requirements but display poor defense capabilities, as attacks succeed. On the other hand, IPSs can shield networks by rejecting packets that match any hostile pattern, but this can negatively impact network performance as malicious attacks increase.

Many IDPS systems have been proposed, but their designs and controls for efficient attack detection and prevention combined with efficient resource usage has always been difficult [5], [6]. Evaluating IDPS performance for a particular security arrangement is a key step for improving their real-time ability [7], [8]. Another concern relates to the impact of safety measures on the performance and usability of an enterprise information system. This paper aims to study the impact of security enforcement levels on the performance and usability of an enterprise information system. We propose a rule mode selection optimization technique that aims to determine an appropriate IDPS configuration set in order to maximize the security enforcement levels while avoiding any unnecessary network performance degradation. The proposed method demonstrates that the application of various sets of rules categories and configuration parameters affects service time as well as system security. As a result, it is advantageous to find a balance between security and performance to acquire a satisfactory technique that does not cause computational time to suffer. Simulation was conducted to validate our proposed technique.

The paper proceeds with an overview of related work in Section II, then presents a background of the rule-checking process in Section III. Section IV presents the rule mode selection optimization technique. In Section V, we present the IDPS performance analysis related to rules mode selection problem. Finally, Section VII concludes the paper and anticipates the nature of future work.

II. Related Work

Research has revealed that the IDPS rule-checking process is a performance bottleneck [9], [10]. Consequently, researchers have focused on discovering solutions and algorithms, either software or hardware, to improve the performance of this process. However, there has been little research to look at the issue of dynamic adaptation to balance system performance and security.

Lee et al. [11] put forward a method to determine the performance of an IDS through quantifying the benefits and drawbacks of detection rules. Their goal was to establish the best possible configuration for an overloaded IDS to prevent the dropping of information under resource constraints, and to elicit adjustment to existing conditions. Their work is comparable to ours in that it measures the service time of different IDS configuration sets in order to establish the best one. Nevertheless, defining the cost and benefit metrics accurately is not easy, and varies from one environment to another. Moreover, in view of the preventive capacity of an IDPS, the analysis offered by Lee et al. seems insufficient. This is due to violation of the stringent QoS constraint in terms of end-to-end delay attributable to the prevention services.

The authors of [12] seek to convert an IDS system into an IPS by putting forward a policy management for firewall devices incorporated with intrusion prevention capabilities. They offer an attack response matrix template which maps intrusion types to traffic enforcement responses. Their application is, however, only at the design stage and no firm implementation or policy parameters have been given. Also, they do not reflect on the performance aspect but only on how to convert an IDS into an IPS using policies. Consequently, a balance between performance, in term of delay, and prevention ought to be considered when IDPS is applied. In [9], [10], the authors aim to fine-tune the trade-off between security level versus resource consumption; nevertheless, the impact of IDPS configuration on average service time has not been conducted. Thus, our study goes a step further by studying the impact of IDPS configuration on system performance.

III. Background and Problem Description

In this section, we explain the operation of current intrusion detection and prevention systems. In general, IDPSs carry out a number of analysis in order to discover hostile traffic. `SNORT`, for example, performs various tasks including data decoding, preprocessing, rule checking, and action implementation. We are focused on the rule-checking process together with the action related to each rule. We assume that the appraisal of these filtered rules and the non-content-based rules are applied successively. Once a rule matches a packet, the related action will be performed.

A rule can be either detective or preventive. The action of a detective rule is `alert` and that of a preventive rule is `drop`. The detective rule is aimed at inspecting a copy of a packet transmitted over the network, generating an alert when a hostile pattern exists. Clearly, this passive inspection mode has no impact on network performance, as it checks only for malicious activity, while genuine traffic is delivered successfully. However, since by its nature it is a passive system, this inspection mode provides poor protection. Unlike the former, the preventive rule is designed to be in-line, so that a packet will be dropped if it carries a hostile pattern. This mode can meet security requirements, but can have an adverse impact on performance, especially as malicious patterns increase. We assume that the preventive rules are applied first, so that no packet can enter the network until it is checked by all applicable rules.

A. *Definitions and Preliminaries*

IDPSs are sent out with a large quantity of rules. The security administrator is responsible for including and excluding rules, in accordance with the particular needs of the protected network environment. For example, SNORT allows the enabling/disabling of rule libraries or individual rules via a set of configuration files. We let $\mathcal{R} = \{r_1, r_2, \ldots, r_N\}$ denote the set of a fixed number of rules included in the IDPS with cardinality $|\mathcal{R}| = N$. Furthermore, the security administrator can denote the form of the rules as being either detective or preventive. To categorize the rule as to which group it belongs to, we define a binary vector $\mathcal{G} = \{g_1, g_2, \ldots, g_N\}$ that indicates whether a rule is detective or preventive (i.e., detection mode if $g_k = 0$, prevention mode if $g_k = 1$, where $k = 1, 2, \ldots, N$). This binary vector is defined as corresponding to rules vector $\mathcal{R}$ with $\mathcal{N}$ rules.

Each rule r_k has a processing time t_k. We consider only the time that it takes a rule to process an actual packet. Obviously, a detective rule that merely examines a copy of traffic is assumed to need no processing time on the actual traffic. The processing time t_k will be considered only if the rule r_k is in a preventive mode ($g_k = 1$).

Each rule $r_k \in \mathcal{R}$ is account for only one type of malicious event. We let $\mathcal{A} = \{a_1, a_2, \ldots a_N\}$ be the set of different attacks covered by the IDPS, assuming that each attack is independent of the others. Since the IDPS that we are considering in this case is a signature-based IDPS, the treatment of it does not incorporate the detection/prevention of the "zero day" attacks.

We denote E as an arriving event or flow. The event E is malicious with attack of type k where $k \in \{0, 1, \ldots, N\}$ and is denoted as $\mathrm{E} \leftarrow a_k$. Note that an event contains at most only one type of maliciousness. We denote by $\mathrm{E} \leftarrow a_0$ a benign event which does not contain any malicious content with regards to the different rules' restrictions R_i ($i = 1, 2, \ldots, N$).

A rule r_i announces event E as malicious with regards to attack type a_i is defined as $\mathrm{E} \overset{r_i}{\leftarrow} a_i$. Similarly, we define $\mathrm{E} \overset{r_i}{\leftarrow} a_0$ to indicate that the event E is announced as normal when no rule r_i reports the presence of attack a_i in it for all $i = 1, 2, \ldots, N$. The probability that rule r_k triggers an arriving event E as malicious, given that it is malicious with regards to attack type a_k is defined by: $\mathbb{P}\mathrm{rob}\{\mathrm{E} \overset{r_i}{\leftarrow} a_k \mid \mathrm{E} \leftarrow a_k\}$ which is equal to the true positive probability TP_k =1-FN_k. FN_k represents the false negative rate of rule r_k when miss-announcing a malicious event that contains an attack of type a_k. We let FP_k=$\mathbb{P}\mathrm{rob}\{\mathrm{E} \overset{r_i}{\leftarrow} a_k \mid \mathrm{E} \leftarrow a_0\}$ be the false positive rate of rule r_k, that is, the probability that rule r_k triggers an arriving event E as malicious, given that it is not malicious with regards to rule r_k.

We denote P_M as the probability of maliciousness that categorizes an arriving event E to be malicious. This probability

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

can be used to estimate future attacks. We denote by H(k) the vector indicating the proportion of malicious event of type i among all the malicious events for all i=1,,N. Clearly, the sum of this vector is equal 1 ($\sum H(i) = 1, i = 1, \ldots, N$).

IV. Optimization of IDPS Rule Mode Selection

In this section we address the problem of determining the appropriate IDPS configuration set necessary to balance network security and performance. As explained before, IDPS preventive rules have the capability of blocking attacks once they have been matched. However, this induces a negative impact on network performance in terms of delay, especially when the number of preventive rules increases. Therefore, the main concern is to find the appropriate preventive rule set that maximizes security enforcement levels while avoiding any unnecessary performance degradation in terms of delay. We assume that the security administrator excludes the rules that are suppose to be strictly in preventive or detective modes. The optimal solution for the rule mode selection technique (RMST) problem is the one that can maximize the prevention level and minimize system delay. Hence, the RMS problem is considered as NP-complete due to multiobjective goals with maximal minimal matching.

A. Rule Mode Selection Problem

The rule mode selection problem is formulated as follows: Given a set of IDPS rules, find a legitimate preventive rule subset that maximizes the level of security, subject to the delay constraint. In our study, we assume a sequential rule checking process where each event passes through a sequence of rules until a decision is made. For an IDPS with N rules associated with weight w_i that resembles the dominance of rule r_i in the expected value metric, we can then formalize the RMST problem as an binary integer program (BIP) as follows:

$$\begin{aligned} &\max \sum_{i=1}^{N} w_i x_i \\ &\text{s.t. (a)} \ \sum_{i=1}^{N} t_i x_i \leq D_{max} \\ &\quad \text{(b)} \ x_i \in \{0,1\}, i \in \{i = 1, \ldots, N\}. \end{aligned} \tag{1}$$

where x_i is a binary variable such that $x_i = 1$ if rule r_i is a preventive rule and $x_i = 0$if rule r_i is a detective rule. The rule weight computation is explained in more detail later in this section. Inequality (a) provides an upper boundary on the expected response time. Since the expected response time for an event entering the system is proportional to the number of preventive rules, an event has to be served at a rate faster than the arrival rate in order to preserve the stability of the system. The delay constraint is thus translated into an upper boundary D_{max} on the number of preventive rules as the mean of the inter-arrival rate.

The RMS problem can be mapped to the 0-1 knapsack problem [13] in order to dispose of the complexity in max-min structure. In 0-1 knapsack problem, we are given n items, each associated with a value and weight; the objective is to select a set of items that maximize the total value where the total weight is less than or equal to a given value W. The RMS problem is similar to the 0-1 knapsack problem, where the weight of the rule w_i is similar to the item's value and the upper limit response time is similar to the maximum allowed weight. Given that the 0-1 knapsack problem was proven to be also NP-complete [13], the NP-completeness of the RMS problem can be proven.

Algorithm 1 OptimizeRuleSelection: rule_set, D_{max}

$D_{sum} \leftarrow 0$
$pool\_list \leftarrow empty$
for $r_i : i = 0$ to N **do**
  if $t(r_i) + D_{sum} \leq D_{max}$ **then**
    $prevention\_list \leftarrow r_i$
    $D_{sum} \leftarrow D_{sum} + t(r_i)$
  else if $prevention\_list$ is not empty **then**
    for $r_j : j = 0$ to N **do**
      if ($r_i \in prevention\_list$; $w_j \geq w_i$; $t_j \leq t_i$) **then**
        remove r_i from $prevention\_list$;
        $prevention\_list \leftarrow r_j$;
        $D_{sum} \leftarrow D_{sum} + t(r_j)$ - $t(r_j)$
        $pool\_list \leftarrow r_i$
      end if
    end for
    if $D_{sum} < D_{max}$ **then**
      Add valid rules from $pool\_list$ to $prevention\_list$
    end if
  end if
end for
Return $prevention\_list$

B. RMS Technique

The optimal solution of the RMS problem can be guaranteed to be obtained when performing an exhaustive search in the solution space. However, the brute force method becomes computationally impractical when the number of rules is large. The use of an approximate heuristic solution allows us to obtain a reasonably good solution in polynomial time without searching the entire solution space. Using optimization techniques Branch & Bound and Branch & Cut for solving RMS problems is not practical when the rules number is high due to the 2^N search space. However, the results obtained from a simple selection method, Greedy Algorithm, can meet the computational time limit, but it suffers from the system security requirements. Hence, the proposed rules mode selection technique (RMST) can obtain a solution for the RMS problem in polynomial processing time, and the system security level is in good agreement with the results obtained from the optimization techniques. RMST is expressed in Algorithm 1.

C. Rule Weight Computation

In this section, we describe the technique of computing rule weight with the intention of capturing the importance of every rule in the IDPS. We assign a weight to each rule in the rule configuration set that reflects its value in protecting the network by the IDPS. two factors can be used to calculate the rule weight: (1) *potential damage* that can be prevented by a true detection, and (2) *operational loss* which is incurred due to the false detection.

Potential damage $D(r_i)$: the damage prevented by rule r_i can be measured by using the severity of an attack and the accuracy of rule r_i. The attack severity measures the risk level posed by a particular attack. We let $Sev(r_i)$ denote the severity score for rule r_i that is responsible for attack a_i. There are several knowledge base sources which provide severity scores for known attacks, including MITRE-CVE, NIST-NVD, Secunia, as well as software developer specific severity score databases. For example, the FileZilla unspecified format string vulnerability has been reported in NIST-NVD to be scored as 7.5 out of 10, where SNORT includes a rule accountable for this attack with multiple score references. The potential damage $D(r_i)$ can be expressed as follows:

$$D(r_i) = (1 - FN_{r_i}) \times Sev_{r_i} \tag{2}$$

Operational loss $L(r_i)$: The operational loss incurred by rule r_i can be measured using the cost associated by the response triggered by false detection of rule r_i. For example, cost of blocking legitimate traffic or analyzing false alarm. This cost can be measured/estimated from business mission as follow.

$$L(r_i) = FP_{r_i} \times Cost_{r_i} \tag{3}$$

The rule weight $w(r_i)$ can be measured using the above factors as follows:

$$w(r_i) = \big(\alpha \times D(r_i) - (1-\alpha) \times L(r_i)\big) \times H(r_i) \tag{4}$$

where α is a configurable variable indicating how much of the rule weight should rely on the potential damage and operational loss. The factors used to calculate the rule weight can be obtained during the site-specific risk analysis. We are trying to benefit from information gathered by security administrators during the site-specific risk analysis process about activities that were encountered in the past. Among these are false positive rate, false negative rate, and H, and P_M.

V. Performance Analysis

In this section we illustrate the means of calculating the impact of vector $\mathcal{G}$ on the resultant security of an enterprise information system and on the average response time to scrutinize an event. Once an event occurs, it goes through a sequence of detection and/or prevention rules consistent with the existing configuration of the IDPS represented by vector $\mathcal{G}$. The process ends if the event is dropped by a preventive rule or reported by a detective rule as a hostile event. In case an event is normal, the process terminates when all rules are checked.

A. Average Response Time

Here we evaluate the typical response time of an IDPS. It is the time needed by the IDPS with a rule configuration $\mathcal{G}$ to effectively decide whether an arriving event is accepted as a normal event or is reported/rejected with the existence of an attack. We define $B(i)$ as the blocking probability of rule $B(i)$. It is the probability of identifying an event as hostile by a preventing rule r_i, $\forall\ i = 1,\ldots,N$. The blocking probability of rule r_i is defined by:

$$B(i) = (\mathrm{B}_{\text{mal}}(i) + B_{\text{safe}}(i)) \times \mathcal{G}(i) \tag{5}$$

where B_{mal} is the probability of announcing an event as malicious by rule r_i which depends on the probability that the event is malicious and on the probability of accepting the event as normal by all the rules previously checked. The term B_{safe} the probability of announcing an event as hostile by rule r_i given that the event is benign and all previously evaluated rules r_j (*i.e.*, $j < i$) mark the event as harmless.

$$\mathrm{B}_{\text{mal}}(i) = \sum_{k=1}^{N} \mathrm{PB}_{\text{mal}}(k,i) \times \mathrm{PE}_{\text{mal}}(k,i) \tag{6}$$

PB_{mal} represents the case when the IDPS announces the event as hostile by rule r_k given that the event arrives to rule r_i and is malicious. In this case, the probability that the IDPS correctly announces the event as hostile or mistakenly classifies it as such is calculated as follows:

$$\mathrm{PB}_{\text{mal}}(k,i) = \begin{cases} 1 - \mathrm{FN}_i & \text{if } k = i \\ \mathrm{FP}_i & \text{if } k \neq i \end{cases} \tag{7}$$

PE_{mal} represents the likelihood that the IDPS accepts the event as normal by all rules r_j, $j = 1,\ldots,i-1$, ahead of the current evaluated rule r_i where the event E is hostile. PE_{mal} can be calculated as follows:

$$\mathrm{PE}_{\text{mal}}(k,i) = \begin{cases} H(k)P_M & \text{if } i = 1 \\ \prod_{j=1}^{i-1} \Big(1 - (1-\mathrm{FN}_j)\mathcal{G}(j)\Big)\mathrm{H}(k)P_M & \text{if } k \neq j \\ \prod_{j=1}^{i-1} \Big(1 - \mathrm{FP}_j\mathcal{G}(j)\Big)\mathrm{H}(k)P_M & \text{if } k = j \end{cases} \tag{8}$$

The first term is for the case when the existing evaluated rule r_i is the first one (i=1), where no rule has been checked so far. As well, PE_{mal} encountered the cases when there is at least one rule r_j that has been checked before rule r_i; that is, r_i is not the first rule to be evaluated (*i.e.*, $i > 1$)

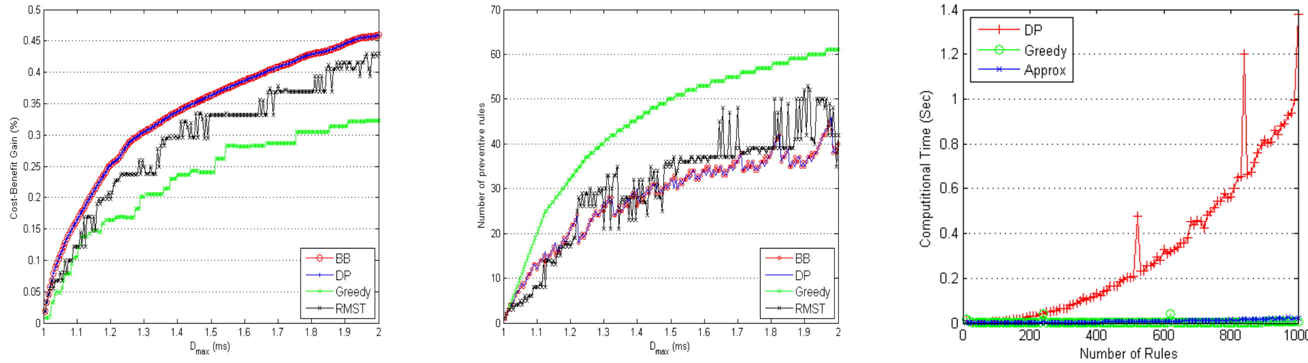
Now let us reflect on the situation when the incident is normal in Eq. 5. We are interested in the likelihood of announcing an event as malicious by rule r_i given that the event is benign and that all previously evaluated rules r_j (*i.e.*, $j < i$) mark the event as safe. B_{safe} can be calculated as follows:

$$B_{\text{safe}}(i) = \mathrm{FP}_i \times \left\{ \begin{array}{ll} 1 - P_M & \text{if } i = 1 \\ \prod_{j=1}^{i-1} \Big(1 - \mathrm{FP}_j\mathcal{G}(j)\Big)(1-P_M) & \text{if } i > 1 \end{array} \right\} \tag{9}$$

Finally, the average response time can be measured as follows:

$$ART = \left[\sum_{i=1}^{N} B(i) \sum_{k=1}^{N} T(k)G(k)\right] + \Big(1 - \sum_{i=1}^{N} B(i)\Big) \times \sum_{i=1}^{N} T(i)G(i) \tag{10}$$

This is the time needed by an IDPS with a given configuration set to completely serve an incoming event. The accuracy of the rules and their modes (detective or preventive) play a key role in determining the response time of an IDPS.



(a) Gain percentage vs D_{max} (b) Number of selected rules vs D_{max} (c) Computational time

Fig. 1. Selected results of the maximum weight and number of preventive rules using different methods

VI. Performance Evaluation and Results

In this section, we evaluate the efficiency of the RMST. A number of simulation experiments were conducted to solve the RMS problem. We also present the optimal solution produced by solving the BIP model of the RMS problem using matlab [14].

In the first experiment, we examine the accuracy of our technique in selecting the preventive rules subset. We analyze the impact of varying the maximum delay constraint (D_{max}) on the resulting security and on the preventive rules selection. In this scenario, the total number of rules is 200. The weight and processing time of the rules are assigned based on zipf distribution [15] which is inspired by [16].

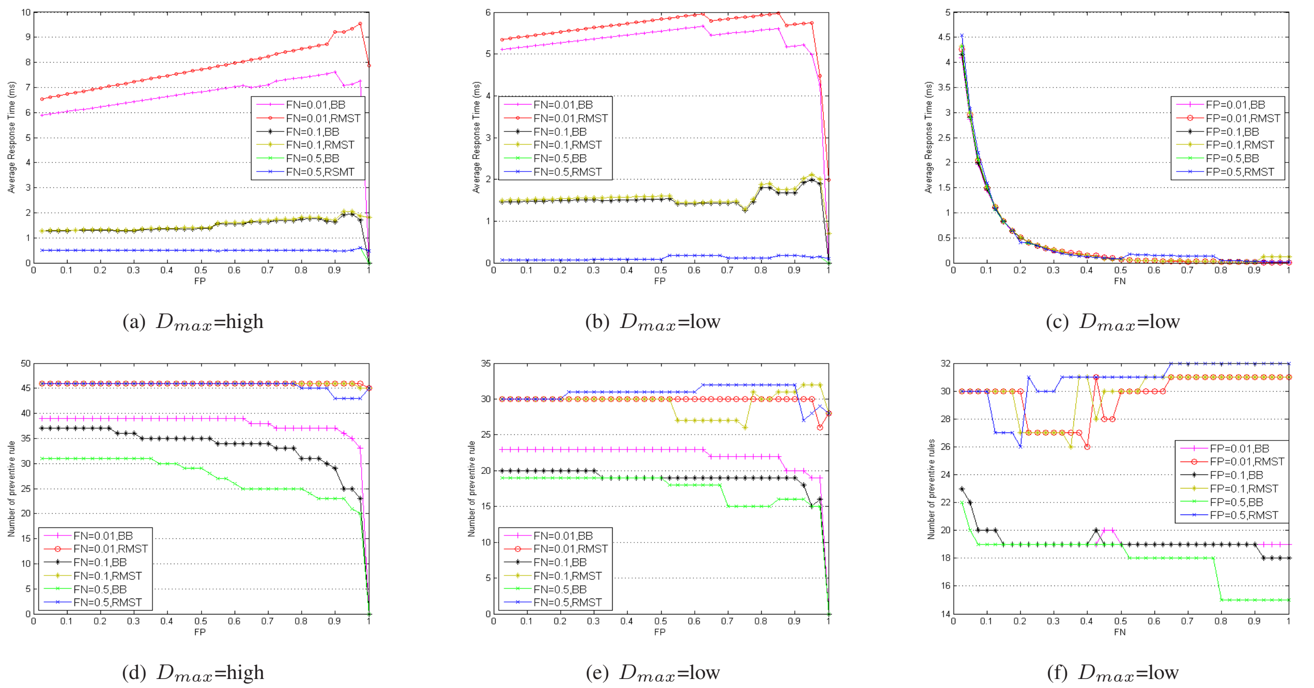
In order to validate RMST, other methods have been used for solving the RMS problem. Three techniques beside RMST are tested and all techniques' results are compared in Figure 1(a). A simple solution for the RMS problem can be obtained by a greedy algorithm, where rules are sorted by their processing time in decreasing order and chosen sequentially until the maximum allowed delay is achieved. However, the cost-benefit gain of the greedy algorithm is not desirable in terms of system security performance. Although the gain obtained using branch and bound (BB) and dynamic programming (DP) for the RMS problem is better than the proposed technique, when the number of rules increases the computational time of (BB) or (DP) is not applicable in our application, whereas the proposed RMST solves the problem in polynomial time, and the obtained rule set has an acceptable security level. The qualitative and quantitative comparison of selected rules is shown in Figure 1(b). The comparison shows that the number of preventive rules selected in (BB) or (DP) is lower than in others; however, the cost-benefit gain is maximum. On the other hand, the quality of the rules selected by the greedy algorithm is the lowest in terms of the cost-benefit gain, so the number of the preventive rules is high in order to satisfy the system security performance, whereas RMST selects prevention rules fairly with a reasonable number of preventive rules and cost-benefit gain.

The scale of the IDPS system affects the required computational time to find an optimal solution. In other words, 2^N combinations have to be computed in order to obtain the optimal solution. When the rules number is high, the search space is very large; therefore, the computational time needed is high, too. Figure 1(c) shows the relationship between the number of rules and the computational time. The Branch and Bound method takes a longer time so it is not included in Figure 1(c). The greedy algorithm and RMST do not suffer from system scalability. While the number of preventive rules and the computational time is related exponentially in Dynamic Programming (DP), that limits applying DP to a low number of IDPS rules.

The second set of experiments studies the impact of the accuracy of the rules set in terms of FP and FN on the average response time and the total number of preventives rules selected by BB and RMST techniques. The average response time is measured by the performance analysis model presented in Section V for any configuration set chosen by BB and RMST. The total number of rules is chosen to be 100 in this experiment. Also, we set the probability of maliciousness to be $P_M = 0.5$ and $\alpha = 0.7$.

Figures 2(a), 2(b), and 2(c), plot the average response time as a function of increasing both the false positive (FP) and false negative (FN) rates. Figures 2(d), 2(e), and 2(f) show the number of preventive rules chosen by BB and RMST in corresponding to the configuration of the previous three subfigures.

Figure 2(a) presents the results when D_{max} is relatively high. We can see that the average response time decreases with an increase in the FN for all FP values. We can see that the average response time is longer when the IDPS becomes accurate in terms of the FN rate, no matter what the FP rates are. The number of preventive rules selected by BB and RMST for this case is presented in Figures 2(d). We can see that the BB technique adapts its selection criteria according to the change of the accuracy values while the RMST remains the same. This happens because assigning a high value to the D_{max} constraint is similar as if we are relaxing it. Figure 2(b) and 2(e) illustrate the impact of the accuracy parameters when the D_{max} is set to be low. The figures share similar results to the previous case, except that the gap between BB and RMST in average response time is reduced. This is because the low value of the D_{max} restriction makes the BB adapt its selection criteria. Overall, with a high FN factor, the average response time of both techniques are similar, although the BB has better average response time when the FN factor is



(a) D_{max}=high (b) D_{max}=low (c) D_{max}=low

(d) D_{max}=high (e) D_{max}=low (f) D_{max}=low

Fig. 2. Selected results for impact of detection rates on average response time and preventive rule selection techniques

very low. In other words, FN factor affects the optimization results, and thus system performance will be affected. Finally, 2(c) and 2(e) show the deep relation between FN factor and system performance in terms of number of preventive rules and average response time.

VII. Conclusion

We propose a rule mode selection optimization technique that aims to determine an appropriate IDPS configuration set in order to maximize security enforcement levels while avoiding any unnecessary network performance degradation. The results demonstrate that it is desirable to strike a balance between system security and network performance. With a small number of IDPS rules, the optimization techniques are more preferable to get better results; however, when the number of IDPS rules is large, the optimization techniques are not applicable due to 2^N search space. Ongoing work is considering the investigation of attack graphs and attack statistical relationships, as well as learning mechanisms. The intent is to determine an appropriate IDPS configuration that will balance network security and performance. We also plan to validate our analysis using real IDPS systems such as `SNORT` and `BRO`.

Acknowledgment

This work was supported in part by the Natural Science and Engineering Council of Canada (NSERC) under its Discovery program, and in part by the WCU (World Class University) program through the Korea Science and Engineering Foundation funded by the Ministry of Education, Science and Technology (Project No. R31-2008-000-10100-0).

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] 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.

[3] V. Paxson, "Bro: a system for detecting network intruders in real-time," in *SSYM'98: Proceedings of the 7th conference on USENIX Security Symposium, 1998*. Berkeley, CA, USA: USENIX Association, 1998.

[4] K. Alsubhi, I. Aib, J. François, and R. Boutaba, "Policy-based security configuration management application to intrusion detection and prevention," in *IEEE conference on Communications (ICC)*, 2009.

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

[6] S. Bellovin and R. Bush, "Configuration management and security," *JSAC*, vol. 27, no. 3, 2009.

[7] L. Schaelicke, T. Slabach, B. Moore, and C. Freeland, "Characterizing the Performance of Network Intrusion Detection Sensors," *Recent Advances in Intrusion Detection: 6th International Symposium, RAID 2003, Pittsburgh, PA, Usa, September 8-10, 2003*.

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

[9] 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.

[10] 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.

[11] 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.

[12] Y. Chen, Y. Yang, and I. WatchGuard Technologies, "Policy management for network-based intrusion detection and prevention," in *NOMS 2004*.

[13] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

[14] www.mathworks.com.

[15] G. K. Zipf, *Human Behavior and the Principle of Least Effort*. Addison-Wesley, Reading MA (USA), 1949.

[16] J. Cabrera, J. Gosar, W. Lee, and R. Mehra, "On the statistical distribution of processing times in network intrusion detection," in *CDC*, 2004.