Performance Security Trade-off of Network Intrusion Detection and Prevention Systems.

Conference Paper · September 2016

CITATIONS
0

READS
2,313

6 authors, including:

Rashid Munir
University of Bradford
6 PUBLICATIONS **94** CITATIONS
SEE PROFILE

Botan Ahmed
University of Sulaimani
2 PUBLICATIONS **2** CITATIONS
SEE PROFILE

Hamad Al-Mohannadi
University of Bradford
9 PUBLICATIONS **148** CITATIONS
SEE PROFILE

Muhammad Rafiq
COMSATS Institute of Information Technology, Wah Campus, Wah Cantt
69 PUBLICATIONS **711** CITATIONS
SEE PROFILE

Performance Security Trade-off of Network Intrusion Detection and Prevention Systems

Rashid Munir*, Botan Ahmed*, Hamad Al-Mohannadi*, M Rafiq Mufti*, Anitta Patience Namanya*, and Irfan Awan*

Abstract

Security cyber threats are increasing with most companies being overwhelm by the complexity attached to prevention against attacks. Network Intrusion detection and prevention systems (NIDPS) are now a stable in any enterprise network with the purpose of filtering through the network traffic and sniffing for malicious traffic. Given the amount of traffic generated on enterprise networks nowadays, any NIDPS is sure to go through a big number of packets that a need arises for a performance- security trade-off. On any given day, based on the rules used in the NIDPS, the number of alerts it generates are in thousands. This can be quite overwhelming to security analysts who analyse them to understand the cyber threat landscape. Although it is true the more alerts, the higher the probability of detecting malicious traffic, it is also true that alerts require the traffic to go through many rules which can be quite a performance hindrance. This is the paradox plagued by the cyber security community currently. In this paper, we examine 2 scenarios to evaluate the performance security trade-off for the purpose of propose ways of improving the performance while minimising the impact on the security purpose for the NIDPS.

Keywords—network security; risk assessment; network intrusion detection system; risk assessment; performance evaluation

1 Introduction

There is increasing concern amongst computer security communities about the rising number of malicious attacks. Two major components which are mostly used to counter security threats are firewalls and NIDPS. Despite the protective mechanism, a firewall does not provide full protection against data leakage, hidden and multithread attacks. In order to protect from these attacks and to make more secure network against unwanted malicious traffic, sophisticated IDS are required. These tools analyze the traffic in depth and decide whether the traffic is friendly (normal) or hostile. NIDPS have proved to be quite effective and with the increasing number of threats, majority of all network communities have been investigating how to produce a more effective NIDPS. The main issue with the current NIDPS is that it loses packets in case of high traffic volumes. This affects the assessment of security risk level of any organization. As, packet loss is one of the most significant problems faced today by the networking teams; therefore to increase the security of a network, there is a need to evaluate the performance of existing NIDPS [1, 2]. To deal with this issue, a paper on performance evaluation comparison of most commonly used NIDPS Snort with newly developed

* School of Electrical Engineering and Computer Science, University of Bradford. r.munir@student.bradford.ac.uk

NIDPS Suricata has been produced by [3] and [4] which conclude that Snort performs better as compare to Suricata despite being its multithreaded features. Assessing the security risk level of malicious traffic running through the network is also another major problem which needs special attention. Currently deployed NIDPS shows the qualitative risk level of attacks in terms of high, medium, low and very low priority. These fuzzy values are not enough to estimate the security risk level of a network. There must be a quantitative metric which shows the security risk level produced by attacks based on absolute value. The third and the most important issue faced by security communities is a performance security trade-off on NIDPS. When number of rules in NIDPS increases, security increases but at the same time its performance decreases and vice versa. It is important to find the balance between the performance and the security [2], [5-7].
This paper is divided into two scenarios. Scenario-1 describes how to assess the quantitative security risk level of NIDPS based on the selected traffic speed used for generating traffic against Snort. Scenario-2 talks about performance security trade-off of NIDPS Snort. These experiments will help any network administrator to: Quantitatively measure the security risk level of a network by creating NIDPS security metric based on best traffic speed used for analyzing network traffic, achieve the balance between performance in term of CPU usage and security in terms of number of rules defined in Snort rules database.

2 Overview of Snort

Snort is well-known name in the information security community [8]. It is capable of performing packet logging and real-time traffic analysis on the network [1]. It inspects packet header and performs protocol analysis, monitors a range of network threats by using content/ signature matching algorithms, logs the traffic on the network and generates alerts against malicious events [9], [10]. Its architecture is composed of four components: packet sniffing, pre-processor, detection engine and output device, as shown in Figure 1. Snort relies on libpcap and winpcap libraries for packet acquisition. Libpcap and winpcap are specific platforms used to receive traffic from the network. Packet acquisition monitors packet arriving time and calculates its total length, and checks interface link type on which the packet was captured.

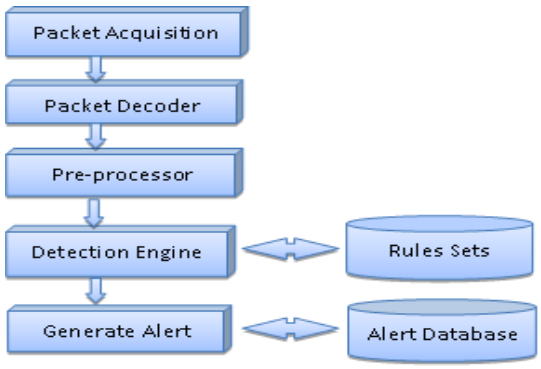

Figure 1. Snort architecture.

Snort processes a single packet at a time. If it has to detect a vast volume of traffic on the network then it will take a little longer to handle that traffic. Buffer overflow is one of the major limitations of Snort, which is the cause of packet drops. As packet decoder, it examine the link layer and try to find out which layer is used for packet fetching.

With the help of pcap files, Snort analyses the traffic on each link layer and if it finds any malicious packet, it begins to make a queue for that traffic. At TCP packet decoding, Snort makes the structure of packets and then sends the packets towards the protocol decoder [11]. After decoding, Snort forwards these packets towards pre-processor which performs various operations like examining protocols and their behavior for anomaly-based detection. All the information going through the pre-processor must pass through the protocol decoder. The three most important pre-processors in Snort are packet defragmentation, stateful inspection session and application layer pre-processor [11]. A pre-processor is basically a program used to normalize the raw packets and check them against anomaly-based behavior (plug-in), for example HTTP plug-in. The Plug-in provides facility to manage the application at traffic flow time. It also avoids unwanted traffic processing which causes an overload on the network. The detection engine is one of the key components of NIDPS and of Snort, its basic purpose is to get data or packets from the pre-processor and match the pattern of the received packet against the database of a certain set of rules. If the pattern matches the predefined rules, it generates alerts against that malicious packet and stops working on that particular packet. Otherwise, Snort treats the packet as normal traffic and does not generate any alert against it. The detection engine generates alerts or logs depending upon the rule.

3 Test Bench

The tests are performed on a real network composed of high performance PCs used to generate malicious traffic for obtaining alerts from NIDPS Snort.

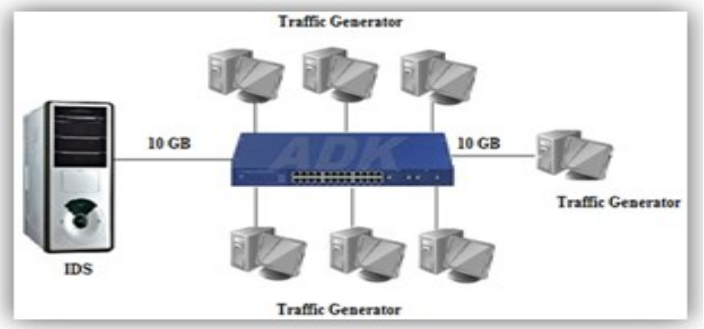

Figure 2. Test bench

Machine Type	Hardware specifications	Tools used
Windows SP2	Dell Precision, T3400,Intel Quad-core,Q6600,2GB Ram, 1Gbps network card	LAN Traffic Generator
FreeBSD Linux 2.6	Dell Precision, T3400,Intel Quad-core, Q6600,2GB Ram ,10Gbps network card	Suricata, Snort Bandwidth monitor
ESXi Server	Dell Precision, T3400,Intel Quad-core, Q6600,4GB Ram, 1Gbps network card (for monitoring server), 10Gb for IDS	VMware ESXi Hypervisor, Linux 2.6, Suricata, Snort Bandwidth monitor
Attacker	Dell Precision, T3400,Intel Quad-core, Q6600,2GB Ram , 1Gbps network card	Backtrack Linux Metasploit 3 Framework
Network Switch	ProCurve series 2900	

Table 1. Network Components Specifications

All the computers are connected via the ProCurve Series 2900 Switch using a 1.0 Gigabit cable and two 10 Gigabit cables as shown in Figure 2. The port connecting the

IDS to the network on the switch is setup as a spanning port, while the other port is connected to high performance PC to generate more malicious traffic as required. LAN traffic V2 Enhanced tool is deployed on all traffic generating machines for generating traffic as shown in Figure 2. The hardware specifications used are shown in Table 1

4 Scenario Setups

NIDPS can be implemented in different ways either inside or outside the local network of organization. The decision is made based on how a network administrator of the organization desires the security mechanisms to take place. Regardless of how the sensor is placed, NIDPS provides a significant view into traffic crossing the network. We created different scenarios in order to study the alert metrics and later the performance factors.

4.1 Scenario-1

In order to create NIDPS metrics, it is necessary for network administrator to determine: is NIDPS working as intended; is NIDPS alarming on the correct events, to mention but a few. Current NIDPS metrics, such as Snort have capability to classify the alerts using common vulnerability scoring system (CVSS) based on the impact of attacks in terms of High, Medium, Low and Very low.
To assess the security risk level of a network, these fuzzy measures are not sufficient. Therefore, there is a need of new NIDPS security metrics that should state the status of a network in terms of absolute value based on the alerts generated by NIDPS quarterly, fortnightly or every day. To tackle this issue, Scenario-1 is implemented based on the results achieved from [3], which is the selection of best NIDPS in terms of its packed handling capabilities. Snort NIDPS is used to assess the security risk level for the designed test bench shown in Figure 2 by analyzing the number of alerts generated at different traffic speed ranging from 250Mbps – 2.0Gbps. 8,000 rules were applied to analyze the traffic against malicious events. Metasploit penetration testing tool was elected to generate malicious traffic from different nodes running Linux OS to the targeted NIDPS.

4.1.1 Attack Detection Rate (Alerts)

According to the network components specification presented in Table 1, specific amount of attacks have been generated in high speeds network, which shows the behavior of Snort in terms of attack detection rate based on different traffic speed observed from the test bench.

Traffic speed per second (ps)	Packets analyzed (%)	Packet dropped (%)	Alerts generated by Snort (%)
250 Mb	100	0	100
500 Mb	100	0	100
750 Mb	100	0	100
1.0 Gb	100	0	100
1.5 Gb	100	0	100
2.0 Gb	100	13.6	99.7

Table 2. Snort Behavior at Different Traffic Speed

Table 2 shows that Snort detected all the attacks until the speed reached 1.5 Gbps. There were only 0.3% packets which were not analyzed by Snort at high speed therefore Snort didn't generate any alert against them. All the alerts generated by Snort till speed 1.5 Gbps are then classified using CVSS based on the impact of the attacks in terms of high, medium, low and very low as shown in Figure 3.

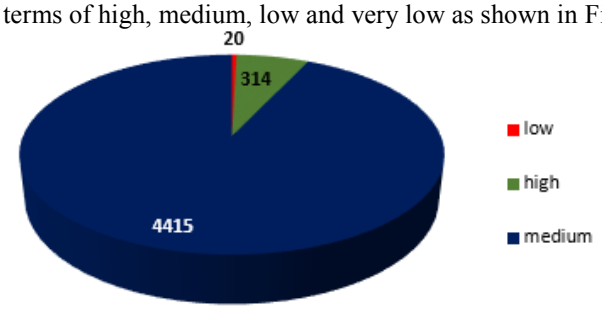

Figure 3. Total number of alerts based on attacks' impact

No	Class type	Description	No. of alerts received	Priority
A1	Attempted-admin	Attempted Administrator Privilege	229	High
A2	Attempted-user	Attempted User Privilege Gain	3	High
A3	Inappropriate-content	Inappropriate content was detected	0	-
A4	Policy-violation	Potential Corporate Privacy Violation	0	-
A5	Shellcode-detect	Executable code was detected	22	High
A6	Successful-admin	Successful Administrator Privilege Gain	0	-
A7	Successful-user	Successful User Privilege Gain	0	-
A8	Trojan-activity	A Network Trojan was detected	2	High
A9	Unsuccessful-user	Unsuccessful User Privilege Gain	0	-
A10	Web-app attack	Web Application attack	58	High
A11	Attempted-dos	Attempted Denial of Service	141	Medium
A12	Attempted-recon	Attempted Information Leak	443	Medium
A13	Bad-unknown	Potentially Bad Traffic	0	-
A14	Default-login attempt	Attempt to login by a default username and password	1	Medium
A15	denial-of-service	Detection of a Denial of Service Attack	0	-
A16	Misc-attack	Misc Attack	6	Medium
A17	Non-standard-protocol	Detection of a non-standard protocol or event	1607	Medium
A18	rpc-portmap-decode	Decode of an RPC Query	0	-
A19	Successful-dos	Denial of Service	0	-
A20	Successful-recon large scale	Large Scale Information Leak	0	-
A21	Successful-recon limited	Information Leak	2206	Medium
A22	Suspicious-filename-detect	A suspicious filename was detected	0	-

A23	Suspicious-login	Attempted login using a suspicious username detect	0	-
A24	System-call-detect	A system call was detected	0	-
A25	Unusual-client-port	A client was using an unusual port	0	-
A26	Web-app activity	Access to a potentially vulnerable web application	18	Medium
A27	Icmp-event	Generic ICMP event	0	-
A28	Misc-activity	Misc Activity	20	Low
A29	Network-scan	Denial of a network Scan	0	-
A30	Not-suspicious	Not Suspicious Traffic	0	-
A31	Prot-command decode	Generic Protocol Command Decode	0	-
A32	String-detect	A suspicious string was detected	0	-
A33	Unknown	Unknown Traffic	0	-
A34	Tcp-connection	A Tcp Connection was detected	0	-

Table 3. Classification of Attacks defined in Snort Rule

4.1.2 NIDPS Security Metrics Calculation

Section- 1

This section describes the method for calculating the security level (SL) based on different types of attack occurrence. As, it is obvious from Table 3, the attacks are classified into 34 different categories on the basis of class types derived from the Snort rules. Assuming A_i, where $i \in z$ and $1 \leq i \leq 34$ is the i^{th} attack generated against the network, and $SL\ (A_i)$ is the security level of a network based on type of attack, and is determined by the following expression:

$$SL\ (A_i) = \frac{\propto_i}{\sum a} \quad (1)$$

Where $\propto_i$ denotes the number of alerts generated against A_i and $\sum a$ is the total number of received alerts against all A_i. Now using Table 3, $SL(.)$ are evaluated as:

$$SL\ (A_1) = \frac{\propto_i}{\sum a} = \frac{229}{4756} = 4.8\% \quad (2)$$

Similarly, $SL\ (A_2), SL\ (A_3), SL\ (A_4), SL\ (A_5), \ldots SL\ (A_{34})$ are estimated, and are listed in Table 4.

$\boldsymbol{SL\ (A_i)}$ **%**				
A_1 = 4.8%	A_2 = .06%	A_3 = 0	A_4= 0	A_5 = 0.5%
A_6 = 0	A_7 = 0	A_8 = .04%	A_9 = 0	A_{10}= 1.2%
A_{11} = 3%	A_{12} = 9.3%	A_{13}= 0	A_{14} = .02%	A_{15} = 0
A_{16} = 0.13%	A_{17} = 34%	A_{18} = 0	A_{19} = 0	A_{20} = 0
A_{21}= 46.4%	A_{22} = 0	A_{23} = 0	A_{24} = 0	A_{25} = 0
A_{26} = 0.4%	A_{27}= 0	A_{28} = 0.4%	A_{29} = 0	A_{30} = 0
$A_{31} = 0$	$A_{32} = 0$	A_{33}= 0	$A_{34} = 0$	

Table 4. The Security Level of a Network Based on Different Types of Attack Occurrences

The security level of a network due to attack A_1 is 4.8% as given by (2), which indicates that the evaluated network is 95.2% secure against it. Now consider the worst case, the security level is highest, i.e. 46.4% with respect to attack A_{21} as clear from Table 4. It means, the network is just 53.6% secure and sufficient countermeasures are therefore to be required to mitigate the potential risks from it.

Section-2

This section describes the method for calculating the *SL* based on the impact of attacks. For simplicity, let $high$, $medium$, low and $verylow$ impacts of attack are represented by H, M, L and V respectively. It is important to note that these impacts or alerts are generated by Snort. As shown in Figure 3, $\sum \mathrm{H} = 314, \sum \mathrm{M} = 4422, \sum \mathrm{L} = 20$, and $\sum \mathrm{V} = 0$, as no alert is generated of this category. Suppose, $\sum H + \sum M + \sum L + \sum V = S$. Now, SL of a network based on impact is determined by the following expression:

$$SL\ (I_m) = \frac{\sum I_m}{S} \tag{3}$$

Where I_m denotes the particular impact category, $\sum I_m$ total alerts of I_m category, and S represents the total alerts of all impact categories. Now, plugging in the above values, the security levels of network against $high$, $medium$, low and $verylow$ impacts of attack are as follows:

$$SL(H) = \frac{\sum H}{S} = \frac{314}{4756} = 6\% \tag{4}$$

$$SL(M) = \frac{\sum M}{S} = \frac{4422}{4756} = 93\% \tag{5}$$

$$SL(L) = \frac{\sum L}{S} = \frac{20}{4756} = 0.42\% \tag{6}$$

$$SL(V) = \frac{\sum V}{S} = 0\% \quad \because \sum V = 0 \tag{7}$$

From (4)-(7), it is clear that security level of a network with respect to $high$ impact is 6%, which implies that network is 94% secure enough against this impact. The security level against $medium$ impact of attack is 93%, which shows that network is just 7% secure and therefore, appropriate countermeasures need to be employed to secure it. The security level due to low impact is 0.42%, which indicates that network is 99% secure. However, it is not almost fully secure. Now, with respect to $verylow$ impact of attack, the security level is zero. It means that the network is fully secure and, consequently, there are no threats of having $verylow$ impact level.

4.2 Scenario-2

Scenario-2 was designed to determine the balance between performance and security based on Snort rules and its CPU usage. Snort has the capability to classify all known attacks based on their level of impact. Using the same test bench shown in Figure 2 and the number of rules used in scenario 1, CPU usage was calculated for each rule. By conducting this experiment, it was observed that by enabling all the rules in NIDPS, performance decreases. Due to this decrease in performance, most of the packets (which have critical and moderate impact on network must be analyzing by NIDPS) are unanalyzed by Snort NIDPS [2], [9], [11]. Therefore, there was need to investigate if disabling low priority rules from Snort rules database would significantly increase the performance of the NIDPS. This leads to increase in security in terms of detecting high and medium priority attacks.

4.2.1 Performance- Security tradeoff analysis.

The following is the detailed analysis of the experimentation results on performance security tradeoff of Snort NIDPS. Figure 4 depicts the total time consumed by each priority (Critical, severe and moderate) signatures defined in Snort rules database.

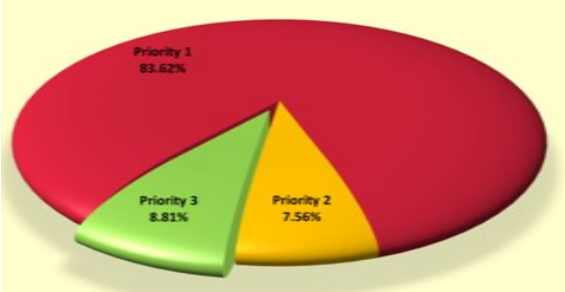

Figure 4. Total processing time consumed by each priority of rules defined in Snort

From the results achieved, most of the CPU processing time is consumed on those Snort's rules which are used in detecting critical attacks (priority 1). The utilized processing time is 83.62%, where as 7.56% and 8.81% are the proportion of total CPU time utilized in processing of priority 2 and priority 3 rules respectively.

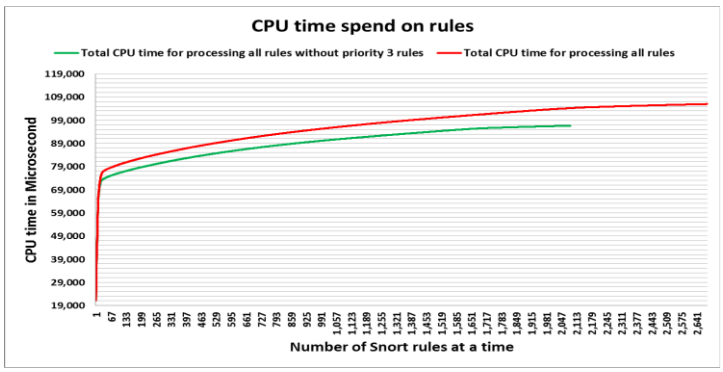

Figure 5. The impact of excluding priority 3 rules on each individual Snort Rule.

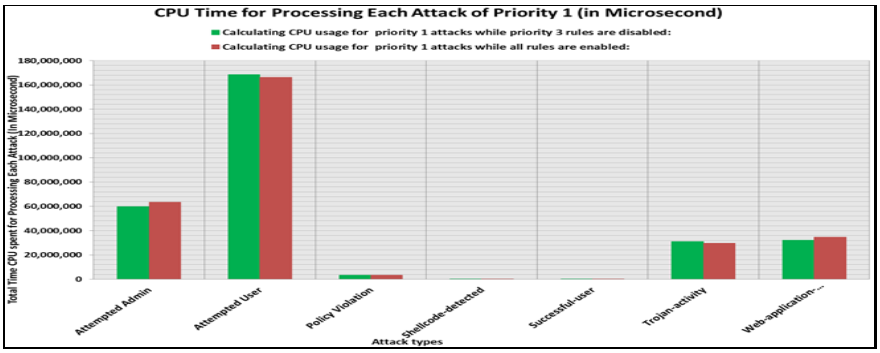

Figure 6. The effect of excluding priority 3 rules on performance (CPU usage) of each priority 1 rule.

When all the rules are enabled, the total processing time utilized is 259,319,731 micro seconds. While by disabling priority 3 rules, the total processing time utilized by priority 1 and priority 2 rules reaches 185,234,078 micro seconds. This implies that by disabling priority 3 rules, the performance of CPU increased by 74,085,653 micro seconds. Figure 6 shows the CPU processing time of each priority 1 rule.

	Processing time of each priority 1 rule after excluding priority 3 rules	**Processing time of each priority 1 rule when all rules were enabled**	**Difference**
Attempted Admin	59980112	63527433	3547321
Attempted User	168682484	166409820	-2272664
Policy Violation	3520482	3359541	-160941
Shell code-detected	397019	399746	2727
Successful-user	122579	122579	0
Trojan-activity	31133981	29601908	-1532073
Web-application-attack	32174850	34683099	2508249

Table 5. The total impact (in terms of CPU usage) of excluding priority 3 rules on each priority 1 rules.

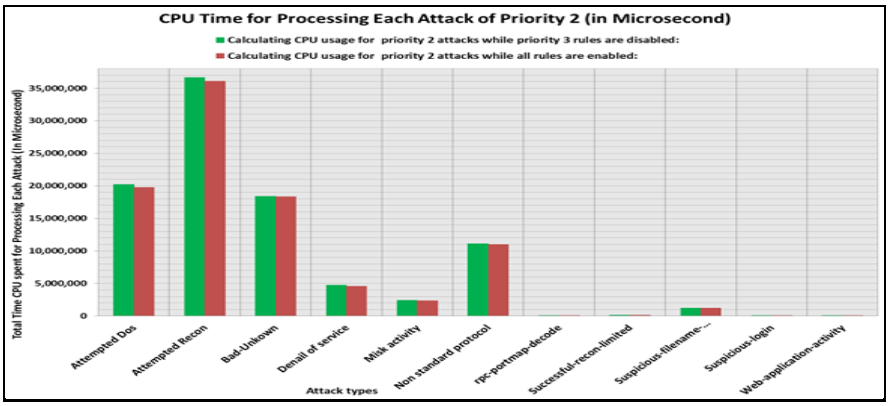

Figure 7. The effect of excluding priority 3 rules on performance (CPU usage) of each priority 2 rule.

	Processing time of each priority 2 rule after excluding priority 3 rules	**Processing time of each priority 2 rule when all rules were enabled**	**Differences**
Attempted Dos	20219682	19742745	-476937
Attempted Recon	36677853	36068530	-609323
Bad-Unknown	18420688	18311658	-109030
Denial of service	4758399	4597285	-161114
Misk activity	2410748	2372670	-38078
Non-standard protocol	11137221	11007372	-129849
rpc-portmap-decode	283	332	49
Successful-recon-limited	139156	127779	-11377
Suspicious-filename-detected	1204234	1193865	-10369
Suspicious-login	575	490	-85
Web-application-activity	48267	48724	457

Table 6. The total impact (in terms of CPU usage) of excluding priority 3 rules on each priority 2 rules.

546,963 microseconds (2092619-1545656) is the total saved CPU processing time from processing priority 1 and priority 2 rules after disabling priority 3 rules. This difference might not look significant; however the saved processing time (74,085,653) can be utilized in analyzing those attacks (critical and moderate) which are unanalyzed by Snort due to the packets drops. There is improvement in the detection of most critical (priority 1 and priority 2) traffic which was dropped when priority 3 rules were enabled which is further discussed hereafter. The variation in detection of priority 1 and priority 2 attacks after the exclusion of priority 3 rules from Snort rules database is also described. It is clear from Table 8 that by excluding priority 3 rules, there is a significant increase (2057079) in the total number of priority 1 attacks detected by Snort.

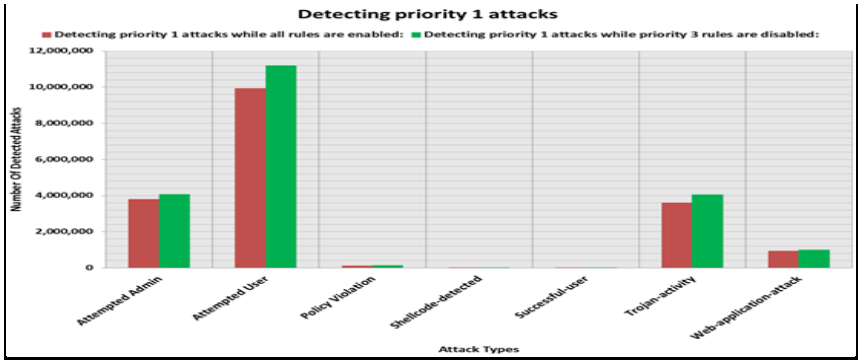

Figure 8. The effect of excluding priority 3 rules on the detection of priority 1 attacks.

	Detection of each priority 1 attacks when priority 3 rules are disabled	**Detection of each priority 1 attacks when all rules are enabled**	**Differences**
Attempted Admin	4062890	3789173	273717
Attempted User	11198115	9929952	1268163
Policy Violation	140707	125438	15269
Shell code-detected	27380	24570	2810
Successful-user	8129	7227	902
Trojan-activity	4040448	3611010	429438
Web-application-attack	1000410	933630	66780

Table 7. The impact of excluding priority 3 rules on the total detection of each priority 1 attack.

Table 8 shows that by excluding priority 3 rules, there is a significant increase (876551) in the total number of priority 2 attacks detected by Snort. Figure 10 shows that a significant increase is noticed when priority 3 rules are excluded from snort rules database. The total number of priority 1 types of attacks increase from 18421000 to 20478079 by 2057079. Similarly, the total number of priority 2 types of attacks increase from 10850327 to 11726878 by 876551. After disabling priority 3 rules, the percentage of analyzed traffic increased by 6.06% from 86.40% to 92.46%.

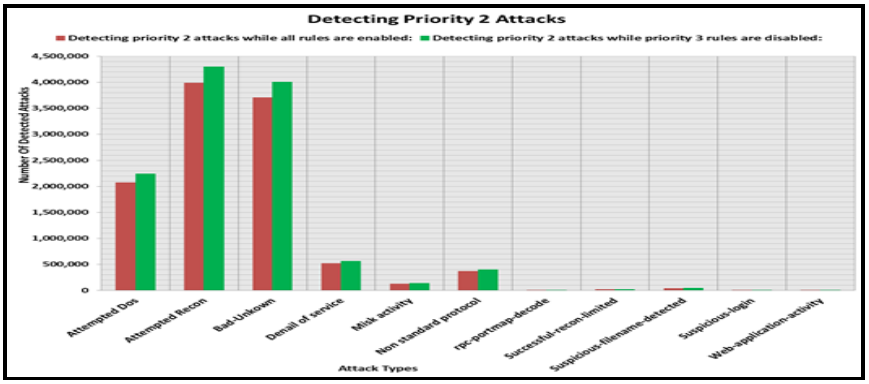

Figure 9. The effect of excluding priority 3 rules on the detection of priority 2 attacks.

	Detection of each priority 2 attacks when priority 3 rules are disable	Detection of each priority 2 attacks when all rules are enable	Differences
Attempted Dos	2238882	2074832	164050
Attempted Recon	4299495	3985337	314158
Bad-Unkown	4006822	3707002	299820
Denail of service	566726	520572	46154
Misk activity	141032	128274	12758
Non-standard protocol	402032	368055	33977
rpc-portmap-decode	107	100	7
Successful-recon-limited	22457	20455	2002
Suspicious-filename-detected	45815	42329	3486
Suspicious-login	67	59	8
Web-application-activity	3443	3312	131

Table 8. The impact of excluding priority 3 rules on the total detection of each priority 2 attack.

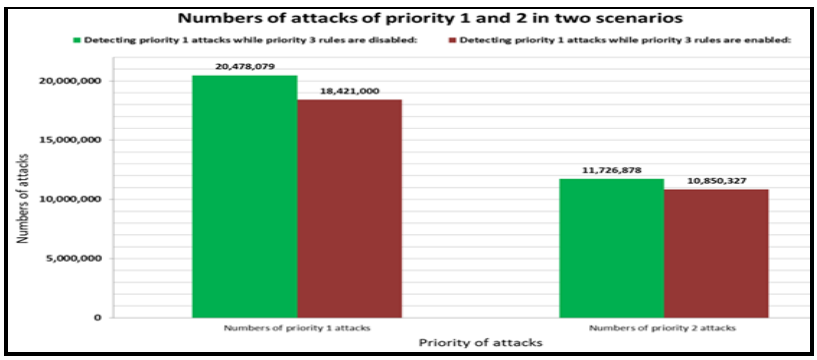

Figure 10. The progress in the total number of detected priority 1 and 2 types of attacks after disabling priority 3 rules.

5 Conclusions

The focus of this paper was to quantitatively assess the security risk level of NIDPS and to evaluate the performance security tradeoff of Snort NIDPS. Current NIDPS, such as Snort has capability to classify the alerts using CVSS based on the impact of attacks in terms of *high, medium, low* and *very low*. These security metrics are not sufficient especially to analyze the status of a network in terms of attacks. Based on the experiments performed in Scenario -1, NIDPS security metrics have been proposed which quantitatively assess the security of a network by using probability theory which can provide better results as compare to traditional measure such as *high, medium, low* and *very low*. During the performance security tradeoff of Snort NIDPS, it is observed that according to Snort team, attacks with priority 3 have low risk on the network, but still are considered as a threat on the network. These attacks are often used by intruders during information gathering phase and may not have direct impact on the network security, but can be the base of attack with higher level of risk. However, the concept of excluding priority 3 rules from Snort rules database is more appropriate in those hours when there is a heavy traffic running on the network. As soon as the traffic load decreases, it must be the duty of network administrator to re-enable priority 3 rules that will decrease the probability of occurring priority 3 attacks on the network in normal traffic load. In this regards, creating a bash file is more helpful approach for automating the above task. Another major advantage of sacrificing priority 3 rules is the increment in the performance of Snort in terms of detecting more priority 1 and priority 2 attacks which carry heavier threat levels.

6 References

[1] Alserhani, F., et al. Evaluating intrusion detection systems in high speed networks. in Information Assurance and Security, 2009. IAS'09. Fifth International Conference on. 2009. IEEE.

[2] Akhlaq, M., et al., Implementation and evaluation of network intrusion detection systems, in Network performance engineering. 2011, Springer. p. 988-1016.

[3] Alhomoud, A., et al., Performance evaluation study of intrusion detection systems. Procedia Computer Science, 2011. 5: p. 173-180.

[4] Xing, T., et al. Snortflow: A openflow-based intrusion prevention system in cloud environment. in Research and Educational Experiment Workshop (GREE), 2013 Second GENI. 2013. IEEE.

[5] Kruegel, C. and T. Toth. Using decision trees to improve signature-based intrusion detection. in Recent Advances in Intrusion Detection. 2003. Springer.

[6] Schaelicke, L., et al. characterizing the performance of network intrusion detection sensors. in Recent Advances in Intrusion Detection. 2003. Springer.

[7] Roesch, M. Snort: Lightweight Intrusion Detection for Networks. in LISA. 1999.

[8] Roesch, M. and C. Green, Snort users manual snort release: 2.0. 1. Snort Documentation, 2003.

[9] Northcutt, S., et al., Snort: IDS and IPS toolkit. 2007: Syngress Press.

[10] Roesch, M., Snort, intrusion detection system. http:/www. snort. org, 2011.

[11] Caswell, B., J. Beale, and A. Baker, Snort Intrusion Detection and Prevention Toolkit. 2007: Elsevie.