ULUSLARARASI 3B YAZICI TEKNOLOJİLERİ
VE DİJİTAL ENDÜSTRİ DERGİSİ

INTERNATIONAL JOURNAL OF 3D PRINTING
TECHNOLOGIES AND DIGITAL INDUSTRY

ISSN:2602-3350 (Online)

URL: https://dergipark.org.tr/ij3dptdi

COMPARISON OF SOFTWARE AND HARDWARE BASED INTRUSION PREVENTION SYSTEMS

Yazarlar (Authors): Cagri Yardimci, Mevlut Ersoy*

Bu makaleye şu şekilde atıfta bulunabilirsiniz (To cite to this article): *Yardimci C., Ersoy M.," Comparison Of Software And Hardware Based Intrusion Prevention Systems" Int. J. of 3D Printing Tech. Dig. Ind., 5(2): 237-250, (2021).*

DOI: 10.46519/ij3dptdi.954996

Araştırma Makale/ Research Article

Erişim Linki: (To link to this article): https://dergipark.org.tr/en/pub/ij3dptdi/archive

COMPARISON OF SOFTWARE AND HARDWARE BASED INTRUSION PREVENTION SYSTEMS

Cagri Yardimci[a,b], Mevlut Ersoy[a*]

[a]Suleyman Demirel University, Engineering Faculty, Department of Computer Engineering, TURKEY
[b]Usak University, Department of Information Technology, TURKEY

[*] *Correponding Author: mevlutersoy@sdu.edu.tr*

(**Received:** 20.06.2021; **Revised:** 05.08.2021; **Accepted:** 20.08.2021)

ABSTRACT

In this study, attacks in three different scenarios were organized by Nmap and Hping3 tools on the virtual Kali server to physical servers running two software-based, open source Intrusion Prevention Systems (IPS-A and IPS-B) and one hardware-based, closed-source Intrusion Prevention System (IPS-C). Although the software-based IPS-A has high packet capture performances, it has been observed that the detection/alarm results are below the average. Although the hardware-based IPS-C is an optimized appliance to put a minimum load on the processor, the detection/alarm figures are at very low levels. In this paper, it has been observed that the IPS-B which is the other software-based Intrusion Prevention System, has a processor usage of 100% but it has reached a far ahead result with very high analysis and detection/alarm performance. In this study, in all the scenarios, four different packet numbers and about twenty parameters were applied to all three IPSs that packet capture performance is quite high and 100%. All three IPSs achieved 100% detection results in attacks where a small number of packets were sent.

Keywords: Intrusion Prevention Systems. Security. Network. Comparison. Software. Hardware.

1. INTRODUCTION

As organizations implement distributed, critical business applications that they support with system and network activities to protect and strengthen their competitive advantages, they cannot prevent server and network security risks from increasing. Multi-layered security strategies are the most accurate solution for organizations to be preferred instead of a single product in terms of protection against known or unknown attacks or those who try to circumvent the corporate application rules [1]. The purpose of Intrusion Prevention Systems (IPS) is to identify the attacks that will come on the systems both from the internal network and from the outside, and to prevent potentially unauthorized, anonymous use and abuse in real time. This study aims to analyze globally prominent products from security systems with different architectures, both hardware-based and software-based, using different methods.

In 1980, James Anderson wrote a technical report for the US Air Force called Computer Security Threat Monitoring and Surveillance. The report showed that audit logs can be used to help identify computer abuse and threat classifications. In 1984, SRI International was funded by the US Navy for intrusion detection research. SRI has developed a prototype called IDES (Intrusion Detection Expert System) that will analyze audit trails from government systems and track user activities. In 1988, Lawrence Livermore Laboratory, a security, science, technology, and R&D organization, produced an Intrusion Detection System (IDS) that could analyze audit data according to defined models. In 1989, Todd Heberlein, a student at the University of California, founded an IPS called NSM (Network System Monitor). Unlike IDES, NSM can analyze network traffic instead of system logs. In 1998, Snort, an open source and libpcap based packet sniffer and recorder, was developed by Marty Roesch. It has led many people to learn and use intrusion detection technology [2].

Hicham et al. examined the security problems in Mobile Ad Hoc Networks (MANET). Prevention methods such as authentication and cryptography techniques cannot provide security alone in MANETs. They have classified the IPS architecture introduced for MANETs so far and compared the existing

intrusion detection techniques as well. An effective attack detection has been achieved to facilitate the identification and isolation of attacks and has made recommendations for further studies [3].

Gunasekaran has examined and compared the Snort, Tcpdump, and Network Flight Recorder systems, which are the most preferred IPSs and offer a very important security layer by monitoring the network traffic for a predefined suspicious transaction or pattern. As a result, it is revealed that Snort as a network security application is much more successful than open source tools examined in financial, technical, and managerial terms [4]. Albin et al. evaluated it under three headings: speed, memory, and accuracy of detection engines. The first is the real-time traffic on the backbone of the school, the other is on a supercomputer where packets passing through the backbone are recorded, and the third is the packet responses sent by friendly vulnerability scanning products. They concluded that Suricata can handle larger volumes of traffic than Snort with similar accuracy and that its performance scales directly with the number of processors (up to 48 processors) [5].

Kacha et al. evaluated Snort and Suricata, rule-based and open source network Intrusion Prevention Systems. They ran each product on multi-core computers and examined the accuracy of speed, memory, and detection engines in various scenarios. Suricata's multi-threaded architecture requires more memory and processor resources than Snort. Suricata's total processor usage is almost twice that of Snort, and Suricata uses more than twice the amount of memory used by Snort. Suricata has the advantage of being able to grow to accommodate increasing network traffic without requiring multiple samples. Snort, on the other hand, does not consume many resources and is fast, but it has been concluded that when it exceeds 200-300 Mbps per sample, it has problems with packet processing [6].

Park et al. analyzed and compared the processing and detection rate of Snort and Suricata to decide which is better in a single thread or multi-thread environment. The results clearly showed that Snort has better performance than Suricata with a lower CPU utilization rate. However, Suricata, which has more features than Snort, surpasses Snort with single and multi-core detection performance and single core performance. In addition, Suricata has been shown to increase its performance when the graphics processor is enabled. According to these results, it was concluded that Suricata surpassed Snort with its multi-threading and additional features [7].

Shah et al. investigate the performance of two open source IPSs, Snort, and Suricata, to accurately detect malicious traffic on the network. Suricata has been found to have faster packet processing performance with lower packet loss rates with higher system usage. Snort stands out for more advanced studies with its higher accurate detection rates [8]. Baykara et al. examined IPSs, one of the most important tools of security systems, in detail. These tools are classified according to criteria such as data source, architectural structure, and runtime. Owing to the newly developed attack patterns, it has been concluded that systems with high capture potential, learning, and low false alarm levels should be developed [9].

As explained above, in the literature, most studies focused on open source solutions. For comparison, two open source solutions are generally considered and few studies on hardware-based products have been examined. Those who included branded products in the studies, generally shared limited information within the scope of confidentiality agreements over a single brand due to plagiarism and ethical perspective. Stronger network and server security architectures can be planned and developed in computer networks and system infrastructures, owing to the analysis of attackers, attack paths, tools they use, known and unknown methods, and detections made by using predictive and/or early warning methods. This study aimed to gain a perspective on similar security architectural studies and researched which security products and methods were better according to their architecture and performance, and some application examples based on scenarios were shown. These applications were evaluated in the light of fundamentals and standard metrics. Security systems have become one of the most important layers in our networks in these harsh times when attacks on systems and networks are increasing day by day. Especially when Intrusion Prevention Systems reach the ability to be one step ahead of attackers, intrusion attempts on our servers and infrastructures will be uncovered. Analysis of the most widely used and highly preferred Intrusion Prevention Systems in the industry and the field with comparative real attack simulations will reveal the effectiveness and competence of these systems.

In this study, both software and hardware based IPSs have been compared in terms of performance in three different scenarios. Although a very powerful DoS attack occurred in scenario-1, IPS-B achieved a 100% analysis and alarm rates. Although IPS-A and IPS-C show very similar test results, the performance of IPS-B is much better than IPS-A and IPS-C. Since the attack in scenario-2 is relatively weak compared to the first scenario, the number of IPS-A analysis and the number of alarms has increased significantly. The same results partially apply to IPS-C. On the other hand, as in the first scenario, IPS-B achieved a very high value, and only 76 alarms could not be detected during the 100K attack. Although the attack in scenario-3 is much more complicated than the previous two scenarios, the analysis and detection rates of IPS-A and IPS-C slightly improved, but the average performance is still not reached. Due to the adoption of an IPS-B structure that does not compromise security, it can provide almost 100% detection and alarm rates even in the case of processing a large number of data packets.

2. METHOD

2.1. Intrusion Prevention Systems

IPS can be defined as a set of tools, methods, and resources that help identify, evaluate, and report unauthorized or unapproved network activity. Since a network is only as secure as its weakest link, using a layered approach and defense in depth with careful risk analysis is essential in an information security approach. Therefore, a network must have a multi-layered security strategy, each with its own function, to determine the overall security strategy of the organization [10].

To gain an in-depth understanding of events on the network, it is necessary to have IPS logs on both successful and unsuccessful attempts. Ideally, one of the preferred methods is to place IPSs listening for network traffic both in front of the firewall and behind the firewall, and compare the recorded data on both sides. Ideal IPS architecture is given in Figure 1 [10].

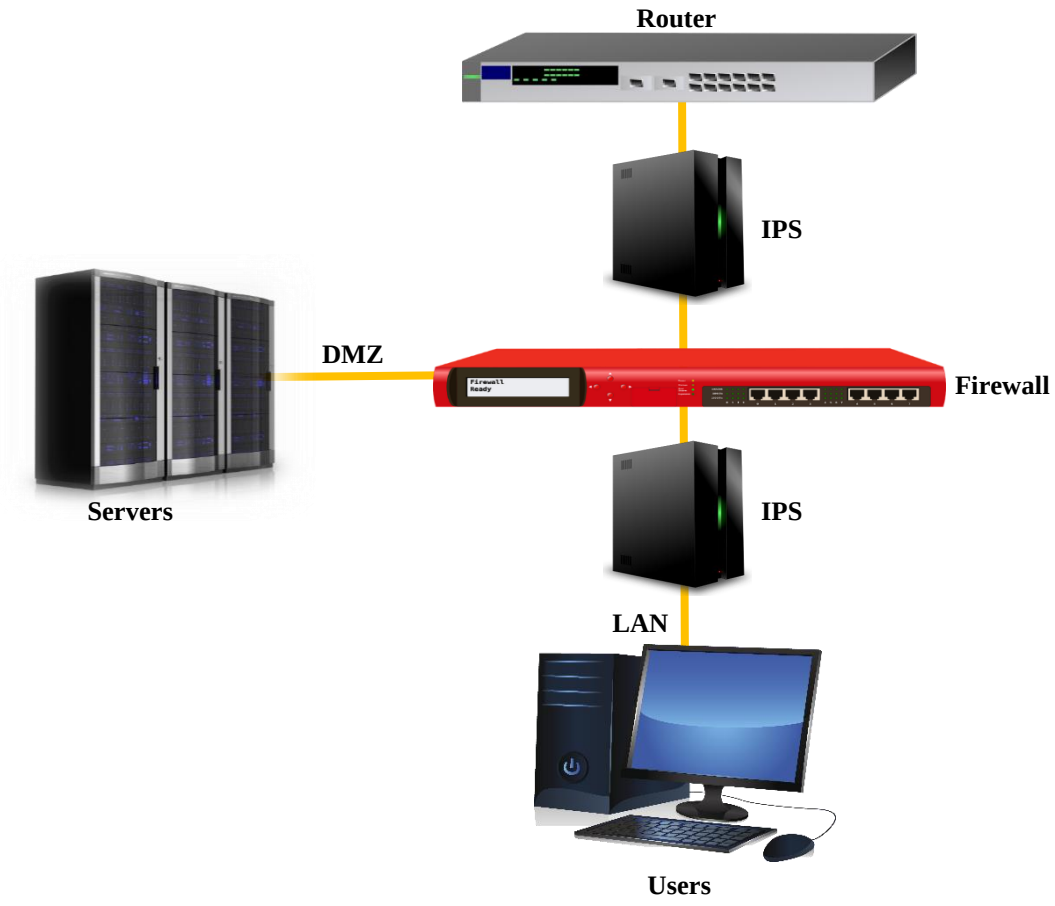


Figure 1. Ideal IPS architecture.

IPSs generally operate at the network layer of the OSI model and are usually located at bottlenecks in the network. They check and track packets to find specific attack patterns in network traffic. If they catch a pattern in traffic, the relevant alert is logged and a response or an action can be triggered based on the recorded data. IPSs are similar to antivirus software in that they use known signatures to distinguish malicious traffic patterns. IPSs fundamentally consists of 5 modules as shown in Figure 2 [11]. The packet decoder captures packets from different interfaces, protocols and services on the network and makes them ready to be sent to the preprocessor. Interfaces can be Serial Line Internet Protocol, Ethernet, Point-to-Point Protocol, and similar protocols [12]. After the packets are passed from the packet decoder to the preprocessor, the preprocessor identifies the data packets according to density and IPS rules and forwards them to the next component which is the detection engine [11]. The detection engine receives the packets from the preprocessor and checks them through a set of rules. If the rules match the data and patterns in the packets, they are sent to the alarm system and the corresponding actions are triggered. If there is no match, the packet will be dropped. By default, packets are logged into log files by the respective module. Depending on the detection engine operation, alerts and/or actions can be triggered by analyzing the packets [12]. The output module is used to save the alarm results. Depending on how the output is intended to be saved, it can perform different actions generated by the log and alert system. This module can also control the type of output produced by the log and alert system [11].

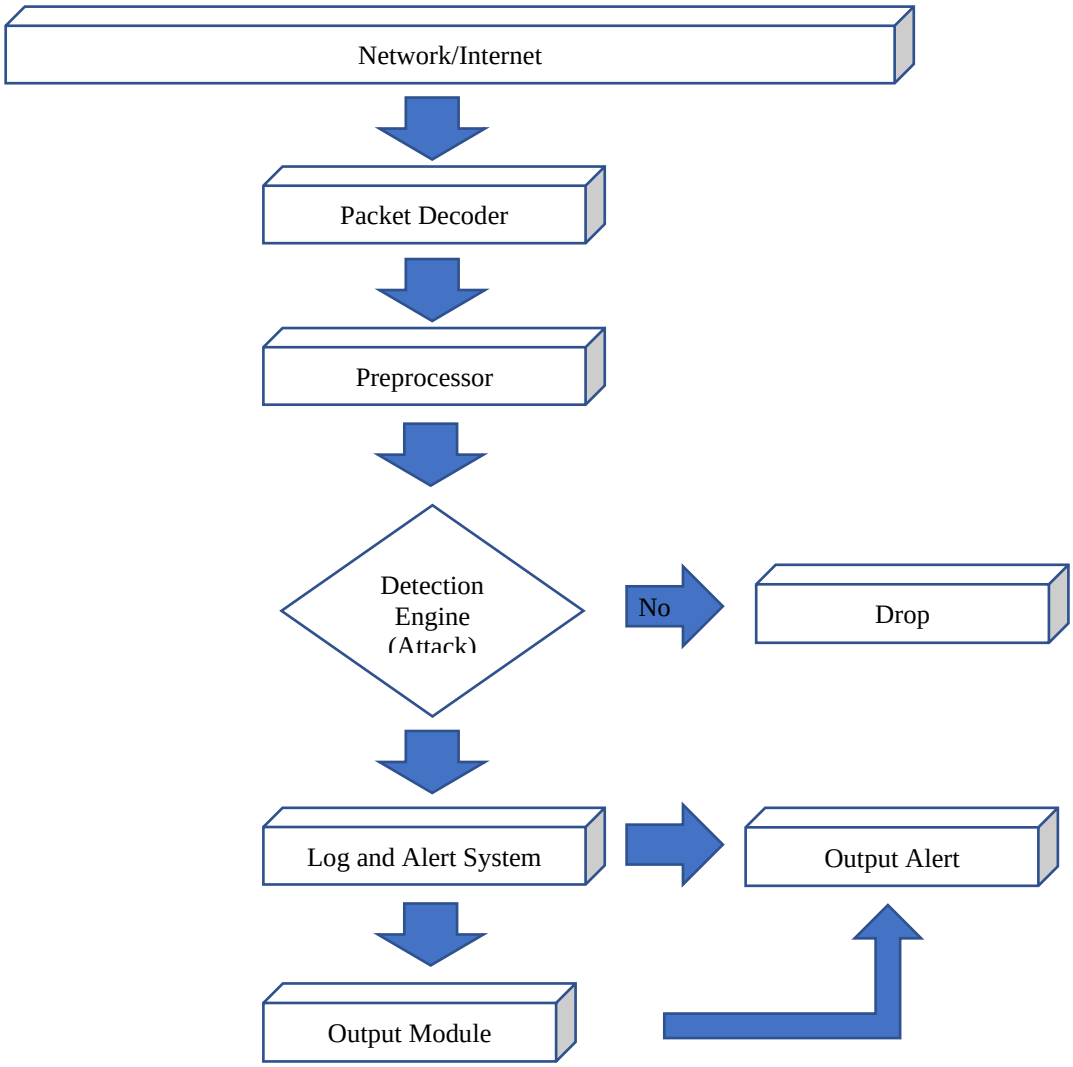


Figure 2. Structure of IPS [11].

2.2. Detection Methods

In order to detect and prevent actions correctly and effectively, it is essential to think like hackers and attackers and defend the network by experiencing the tools and applications they use. A large number of parameters and variables were encountered in the studies conducted before simulations and scenarios. This depth shows that attackers have the ability to break into the systems by using many known or unknown options and methods [13]. Software-based IPSs are installed and configured on hardware, and hardware-based ones are located directly at the bottleneck of networks in general. In this way, they can perform more accurate and consistent network analysis by passing all incoming and outgoing network traffic and data packets [14]. Depending on their structures, IPSs can detect, analyze and prevent attacks with rule, signature, and profile-based or a combination of these architectures [15].

2.2.1. Signature-based match

A signature is a kind of string that uniquely identifies the attack initiated by the attacker on the target system. Signature matching means that the input strings passed to the detection engine match a pattern in the IPS's signature database. The signature matching methodology of an IPS varies from system to system. The simplest, but most inefficient method is to use fgrep or a similar search command to compare each part of the input passed from the preprocessor to the detection engine against the signature lists. Positive identification of an attack occurs when the string search command finds a match as shown in Figure 3 [15].

Figure 3. Signature-based match.

2.2.2. Rule-based match

Rule-based IPSs act on combinations of possible attack findings because they check whether a rule condition is met. In some cases, a signature that always detects an attack may be the only indicator needed for a rule-based IPS to trigger an alarm. An anonymous FTP (File Transfer Protocol) connection attempt from an external IP address may cause the system to not trigger any alerts. However, the situation becomes more suspicious for a rule-based IPS if the FTP connection attempt comes soon after a scan from the same IP. If the FTP connection attempt is successful and the attacker starts to cd .., repeatedly in the root directory, the rule-based IPS will alarm. Rule matching of SYN-Flood attack is given in Figure 4 [1].

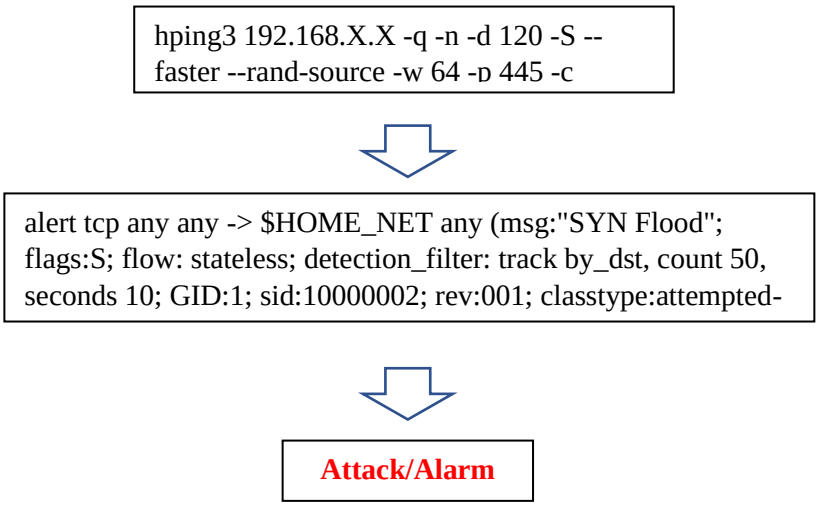


Figure 4. Rule-based match.

2.2.3. Profile-based match

Information about users' session properties is kept in system and transaction logs. Profiling routines extract information for each user and keep it in the database. These routines are measured and statistical models are created. When a user action deviates too much from the standard model, the detection engine flags this event and passes the necessary information to the output module. If a user normally logs in at 8:00 am each morning and logs out at 6:00 pm and logs in at 2:00 am one night, a profile-based IPS will alarm this event [14].

2.3. Attack Simulation

In this study, Kali, an open source Debian-based Linux distribution, which is frequently used in penetration tests, digital forensics, attacks, and defense security studies, was preferred. There are many cyber attack and defense applications in this operating system, which was first released in 2013 and has versions that support different platforms, such as Amd64 and Arm64.

Kali (Rolling 2020.4) x64 customized version for the Vmware platform was used in the simulation. For the Kali virtual machine, 2 virtual CPUs, 2 GB Ram, 100 GB hard disk, and 1 Gb ethernet are reserved as resources. Nmap and Hping3 tools on Kali were used for vulnerability detection and attack simulation. Nmap is an open source application for network discovery and security auditing. It uses raw IP packets to determine which computers and security solutions are on the network and which services and applications they are using. Generally, it is designed to scan large networks. Hping3 is an open source application written for low-level TCP/IP packet processing and analysis using an array-based engine. It is a widely used tool for penetration testing of firewalls and networks.

As IPS, a defense line has been established with 3 different systems. IPS-A and IPS-B are open source systems, while IPS-C is a hardware-based closed source system. IPS-A and IPS-B have been tested on physical and dedicated servers. Each server has an Intel 2957 processor and 4 GB of memory. On the other hand, Intel 3350 processor, which is very close to these 2 servers, is used as the CPU of IPS-C. During the comparisons, these 3 systems were considered to be conjugates with each other. The evaluation was made on standard rule sets for all 3 IPSs. The most up-to-date stable versions of each system on the relevant date were preferred. The test environments are built on the attacks on the same switch and network as shown in Figure 5.

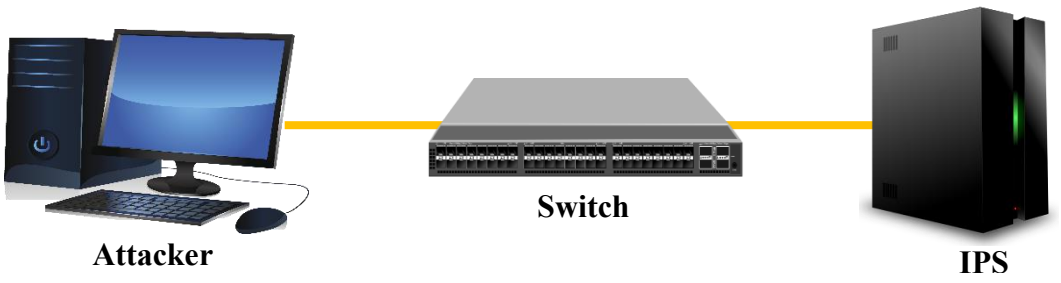


Figure 5. Network structure of the scenarios.

When the configurations are examined, it is possible to talk about an architecture that refers to the modular structure. At the first stage, it should be decided which network or networks the IPS will listen to. In Figure 6, network data is entered to cover all local IPs of the test platform. This configuration may vary depending on the use of server and service types (DNS, HTTP, SQL etc.) or network structure. Depending on the security strategies, the "any" parameter can also be used for a broader definition of the rule.

```
# Setup the network addresses you are protecting
ipvar HOME_NET 192.168.0.0/16

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET !$HOME_NET

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET

# List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET

# List of ssh servers on your network
ipvar SSH_SERVERS $HOME_NET

# List of ftp servers on your network
ipvar FTP_SERVERS $HOME_NET
```

Figure 6. IPS network configuration.

In the next step, the folder and file paths are defined where the rules, preprocessor and log etc. files are kept as shown in Figure 7.

```
# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute path
# such as:  c:\...\rules
var RULE_PATH c:\...\rules
#var SO_RULE_PATH ../so_rules
var PREPROC_RULE_PATH c:\...\preproc_rules
config logdir: c:\...\log
```

Figure 7. IPS file path configuration.

Preprocessors such as frag3, stream5, http_inspect, ftp_telnet, smtp, dns, ssl, sensitive_data are activated according to detailed usage and security needs as shown in Figure 8.

```
# Target-based IP defragmentation.  For more inforation, see README.frag3
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy windows detect_anomalies overlap_limit 10 min_fragment_length 100 timeout 180

# Target-Based stateful inspection/stream reassembly.  For more inforation, see README.stream5
preprocessor stream5_global: track_tcp yes,
   track_udp yes,
   track_icmp no,
   max_tcp 262144,
   max_udp 131072,
   max_active_responses 2,
   min_response_seconds 5
```

Figure 8. IPS preprocessor configuration.

The attack methods that have applied in the three scenarios of the simulations are Syn-Flood Attack, UDP-Flood Attack and Xmas Tree Attack.

2.3.1. SYN-Flood attack

In the SYN-Flood Attack, too many SYN packets are sent to an open TCP port. The target system will send a feedback which is an ACK packet. It then creates a record in the pending connection queue and waits for the TCP three-way handshake to complete. At this point, the connection is often described as half-open. This connection requires a certain amount of memory to be queued. If many SYN packets

are received during this process, which consume too much memory and fail to complete the three-way handshake, the amount of memory consumed will increase hence the system will crash [16].

2.3.2. UDP-Flood attack

The UDP-Flood Attack is a form of DoS attack in which the attacker saturates random ports on the target system with IP packets containing UDP packets. In this type of attack, the target system tries to find applications associated with these datagrams. When no relationship is found, the destination system sends a "Destination Host Unreachable" packet to the sender. The cumulative effect of being targeted by this type of overflow attack is that the system crashes and is therefore unable to respond to network traffic [17].

2.3.3. Xmas tree attack

In the Xmas Tree Attack, the target computer is exposed to too many TCP packets. This custom-made TCP packet has all the options to use any protocol. Based on the fact that all options in this packet can be "turned on", this type of attack is called the Christmas tree. In this attack form, attacks are carried out by assigning FIN, URG, PSH, SYN, ACK, RST, Xmas and Ymas flags. Christmas tree packets require much more processing power by routers, switches and target systems compared to other packets. The processing of these special packets consumes so many resources that after a while the target systems become inoperable [18].

3. EXPERIMENTAL RESULTS

Within the scope of planned scenarios, attack detection and analysis results were evaluated comparatively. In tables, the captured packets, analyzed packets, and attack numbers, and in graphics, CPU and RAM usages are given.

Within the scope of all attack scenarios, the open ports on the target system are primarily targeted with the NMAP tool. Open ports on the system to be attacked are detected using the command "nmap –F 192.168.X.X -Pn" as shown in Figure 9. The parameters used are as follows;

- **-F**: Fast mode
- **-Pn**: Treat all hosts as online

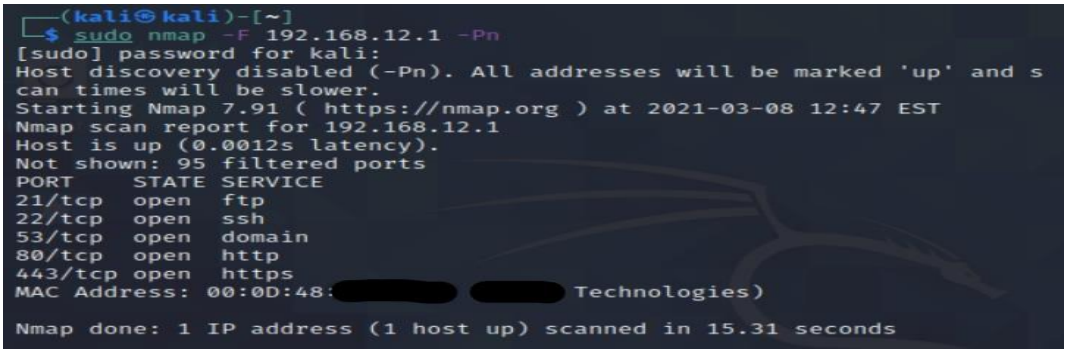


Figure 9. Open port scan with Nmap.

Syn-Flood attack was planned on the detected open ports within the scenario [19]. The Hping3 tool on Kali was used for the Syn attack. A successful Syn attack was made with the command "hping3 192.168.X.X -q -n -d 120 -S --faster --rand-source -w 64 -p 445 -c 1000000" as shown in Figure 10.

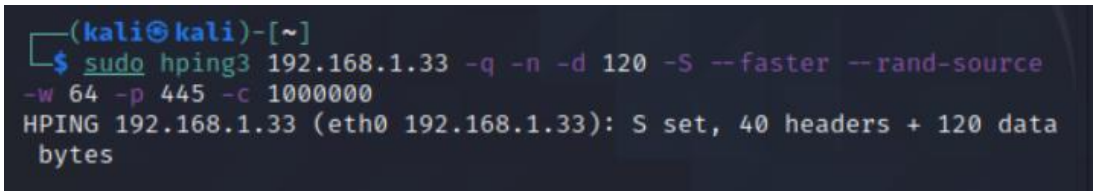


Figure 10. Syn-Flood Attack.

Simulating the real world environment, from the attacker's perspective, the attack should be quick, numerical, brief and of course anonymous. On this point of view, the parameters listed below were chosen for the scenarios. For comparison, attacks and performances were measured by sending 1.000, 10.000, 100.000, and 1.000.000 packets to all 3 systems. The parameters used are as follows;

- **-q:** Quiet output
- **-n:** Numeric output
- **-d:** Data size
- **-S:** Syn flag
- **--faster:** Faster (100p/s)
- **--rand-source:** Random source
- **-w:** Window size
- **-p:** Port number
- **-c:** Packet count

IPS successfully detected the Syn attack as shown in Figure 11. However, all 3 systems showed different performances in the analysis. IPSs have been evaluated in 2 categories as intrusion detection and resource use. The number of packets that IPS can capture in different and intense attacks varies. With this change, there are also differences in the number of packets that can be analyzed. As a natural result of analysis differences, the number of attack detections and alarms also change proportionally.

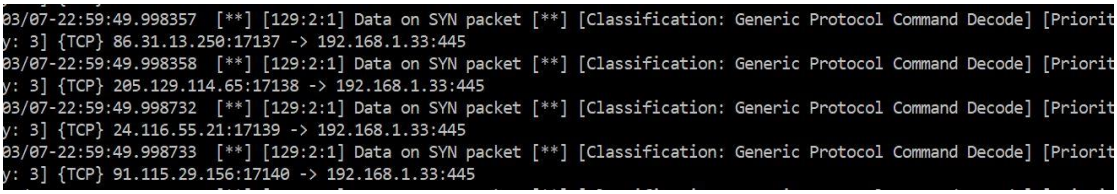
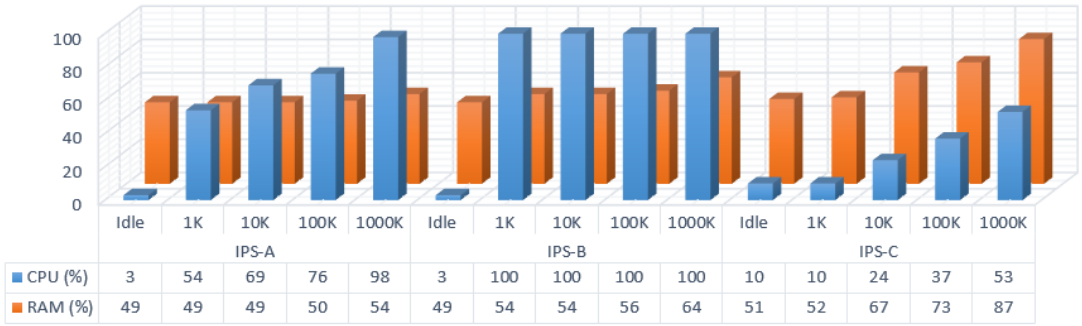


Figure 11. Detection of Syn-Flood Attack.

Intrusion detection performance comparison was made based on the captured packet, analyzed packet, and attack detection as shown in Table 1. The resource usage comparison was evaluated in terms of the processor and memory consumed during the attack as shown in Figure 12.

Table 1. Scenario-1 performance comparison.

	IPS-A				**IPS-B**				**IPS-C**			
Captured	1K	10K	100K	1000K	1K	10K	100K	1000K	1K	10K	100K	1000K
Analyzed	1000	6856	57142	257995	1000	10000	100000	1000000	1000	5487	33049	260528
Attack	1000	6737	56914	257213	1000	10000	99998	999887	1000	5412	32916	259086



	Idle	1K	10K	100K	1000K	Idle	1K	10K	100K	1000K	Idle	1K	10K	100K	1000K
	IPS-A					IPS-B					IPS-C				
CPU (%)	3	54	69	76	98	3	100	100	100	100	10	10	24	37	53
RAM (%)	49	49	49	50	54	49	54	54	56	64	51	52	67	73	87

Figure 12. Scenario-1 resource usage comparison.

IPS-A has a CPU utilization rate of 3% and a RAM utilization rate of 49% in idle. IPS-B has a CPU utilization rate of 3% and a RAM utilization rate of 49% in idle. The IPS-C has a CPU utilization rate of 10% and a RAM utilization rate of 51% in idle. In the attack made by sending 1.000 packets, IPS-A captured and analyzed 1.000 packets, and was able to detect that all of the packets it analyzed were attacks. During this process, it reached 54% CPU and 49% RAM usage rates. In the attack made by sending 1.000 packets, IPS-B captured and analyzed 1.000 packets, and was able to detect that all of the packets it analyzed were attacks. During this process, it reached 100% CPU and 54% RAM usage rates. In the attack made by sending 1.000 packets, IPS-C captured and analyzed 1.000 packets, and was able to detect that all of the packets it analyzed were attacks. During this process, it reached 10% CPU and 52% RAM usage rates.

When the data is examined in detail, it is seen that the packet capture performance of all three IPS is at a high level. No significant packet loss has been observed throughout the scenario. It has been observed that IPS-B performs much better than the other two IPSs in terms of the analyzed packet numbers and can analyze all the packets it captures. When the reasons for this finding are examined, it is striking that it actively uses the processor and processor cores up to 100%, or even fails to serve. This feature, which seems like a plus, is not very acceptable in terms of resource consumption and management. This feature, which seems like a plus, is not very acceptable in terms of resource consumption and management.

In light of this data, the system that uses its resources at the optimum level is IPS-C. The biggest reason for these stable usage results is that the software running on it is optimized according to the hardware layer in the most performance way. Although IPS-C does not compromise the appliance architecture, it is understood that it has great handicaps in terms of security. In denial of service (DoS) attacks, the system becomes inaccessible and/or inoperable as a result of resource saturation on the target system. Similarly, in these attack scenarios, IPSs have shown close to 100% success in the low number of packet attacks. However, as the number of packets increases, their analysis and alarm capabilities decrease as the resources they use to begin to run out. Although IPS-B shows poor resource optimization performance owing to its multi-core and parallel thread support, it stands out from the test results that do not compromise on security [20].

With its average resource utilization capability and above-average alarm numbers, IPS-A stands out as a preferable free solution for small and medium-sized enterprises, owing to its open source, software-based, and continuously developed, post-installation support and resource platform. In the second scenario, a UDP-Flood attack was planned on the detected open ports within the scenario [17]. A successful UDP attack was made with the command “hping3 192.168.X.X --udp -q -n -d 120 --faster --rand-source -w 64 -p 445 -c 1000000”. In this command, unlike the previous scenario, sending UDP packets was provided by using the --udp parameter instead of TCP, and IPSs detected the attack correctly as shown in Table 2.

Table 2. Scenario-2 performance comparison.

	IPS-A				IPS-B				IPS-C			
Captured	1K	10K	100K	1000K	1K	10K	100K	1000K	1K	10K	100K	1000K
Analyzed	1000	10000	79911	395799	1000	10000	100000	1000000	1000	10000	68012	218610
Attack	994	9990	79723	395517	991	9976	99924	998681	992	9890	67443	177801

When the data is examined in detail, it is understood that IPS-B's capture, analysis, and detection statistics for scenario-2 are well ahead. It is seen that IPS-A achieved much higher detection figures than scenario-1 in terms of attack type, and remained at more reasonable levels in processor and memory usage inversely proportionally. When it is observed that similar results are valid for IPS-C, it can be said that the attack in scenario-2 is somewhat weaker in terms of Denial of Service as shown in Figure 13.

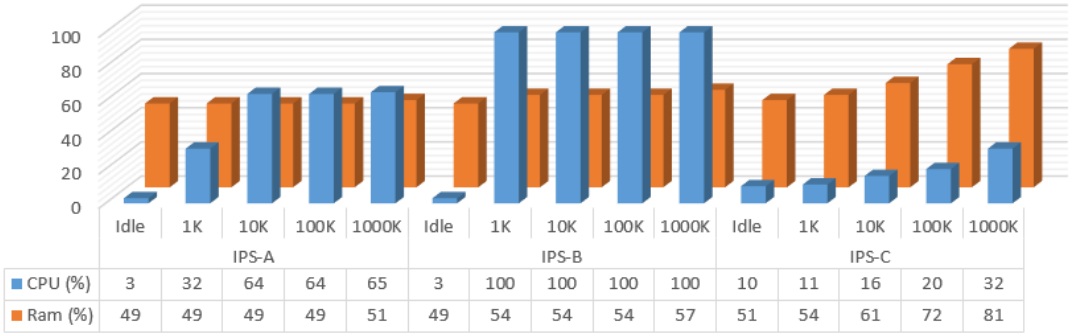


Figure 13. Scenario-2 resource usage comparison.

In the third scenario, a Christmas Tree Attack is planned for the detected open ports within the scenario [18]. A successful Christmas Tree attack was made with the command "hping3 192.168.XX -q -n -d 120 -S -A -R -P -U -F -X -Y --faster --rand-source -w 64 -p 445 -c 1000000". In this command, all flags were activated as a characteristic feature of the Christmas Tree attack, and packet sending was provided as shown in Table 3. The parameters used in the attack are as follows;

- **-q:** Quiet output
- **-n:** Numeric output
- **-d:** Data size
- **--faster:** Faster (100p/s)
- **--rand-source:** Random source
- **-w:** Window size
- **-p:** Port number
- **-c:** Packet count
- **-S:** Syn flag
- **-A:** Ack flag
- **-R:** Rst flag
- **-P:** Push flag
- **-U:** Urg flag
- **-F:** Fin flag
- **-X:** Xmas flag
- **-Y:** Ymas flag

Table 3. Scenario-3 performance comparison.

	IPS-A				IPS-B				IPS-C			
Captured	1K	10K	100K	1000K	1K	10K	100K	1000K	1K	10K	100K	1000K
Analyzed	1000	10000	60214	427272	1000	10000	100000	1000000	1000	10000	71538	368009
Attack	1000	9991	60185	426973	958	9954	99899	998647	1000	9913	70772	330842

Considering the complex attack architecture in scenario-3, it can be said that all three IPS performed well within the scenario averages. When these three scenarios are examined, it is observed that IPS-C keeps the captured packets in a queue in the memory area in order to increase the device performance and processes them in a way that puts a minimum load on the processor. Considering that the overall success of the device is very low, it is recommended to develop parameters such as the IPS rule, signature, and profile by the manufacturer, and to make improvements to the configuration. Compared to previous scenarios, IPS-A is expected to achieve above-average results against this more complex attack, and to meet security needs with more performance servers and more optimized configuration parameters as shown in Figure 14.

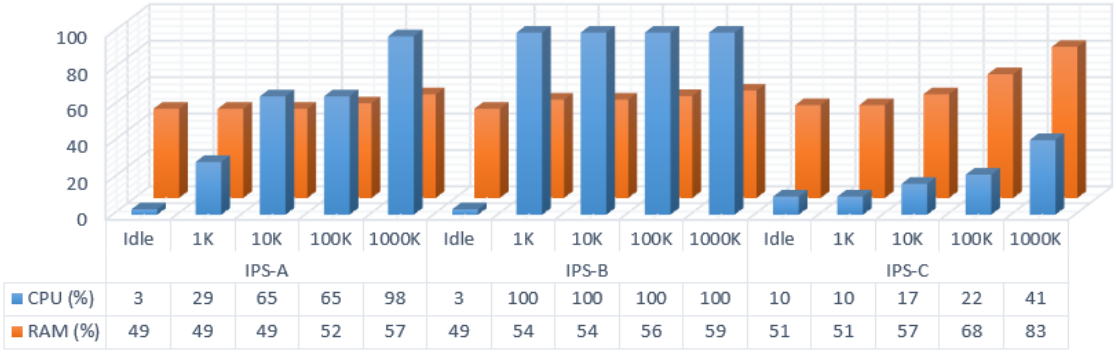


Figure 14. Scenario-3 resource usage comparison.

Based on the philosophy that there is no 100% security, although IPS-B is the worst resource usage among these three IPS, it has easily taken the first place with its stable and well-established structure, high analysis, and detection rates. It can be predicted that IPS-B, which runs on a limited and relatively small server in terms of processor and memory, will work more optimized in real life studies than in the test environment owing to its multi-threading and multi-core support. In this context, IPS-B stands out as an ideal free solution for medium and large-scale structures with its open source code and support platform opportunities that continue to be developed. In this study, although it has shown optimized performance as a hardware-based and commercial system IPS-C due to its moderate and below average results occupies the last place compared to other open source and free options.

4. CONCLUSION

In this study, both software-based and hardware-based IPSs have been compared in terms of performance. Within the scope of the study, two software-based and open source and one hardware-based and closed source IPSs were examined. Three different scenarios were determined for comparison. Four different packet numbers and about twenty parameters were used in each scenario. In all scenarios where various parameters are applied to all three IPSs, packet capture performance is quite high and 100%. Likewise, all three IPSs achieved 100% detection results in attacks where a small number (1000 packets) of packets were sent. Despite a very strong DoS attack in scenario-1, IPS-B reached analysis and alarm figures of 100%. Although IPS-A and IPS-C have shown very close detection results, IPS-B has reached much more successful figures than IPS-A and IPS-C.

Due to the relatively weak attack in scenario-2 compared to the first scenario, there has been a noticeable increase in the analysis and alarm numbers of IPS-A. A similar result partially applies to IPS-C. On the other hand, IPS-B reached very high values, as in the first scenario, and could not detect only 76 alarms in a 100K attack. Although a much more complex attack was organized in scenario-3 compared to the first two scenarios, IPS-A and IPS-C increased their analysis and detection numbers a little, but they still could not achieve average performance. As a result of IPS-B's structure that does not compromise on security, it has reached detection and alarm numbers close to 100%, even in high packet numbers.

Despite the architecture of IPS-C, in which it processes the captured packets in a queue in the memory area in order to use the device performance at the optimum level and processes them in a way that puts a minimum load on the processor, the packet analysis and detection/alarm performance have remained at very low levels. Despite the highly supported software-based platform of IPS-A and acceptable resource usage values, the analysis and detection/alarm performance that could not exceed the average put it ahead of IPS-C but left it far behind IPS-B. Despite the use of 100% processors, IPS-B with its very high analysis and detection/alarm performance, has become a free alternative for corporations in terms of security when compared to the other two systems.

REFERENCES

1. Li H. and Liu D., "Research on intelligent intrusion prevention system based on snort", International Conference on Computer, Mechatronics, Control and Electronic Engineering, Pages 251-253, 2010.

2. Innella P., "The evolution of intrusion detection systems," Tetrad Digital Integrity, Pages 1-15, 2001.

3. Hicham Z., Ahmed T., Rachid L., and Noureddin I., "Evaluating and comparison of intrusion in mobile ad hoc networks," International Journal of Distributed and Parallel Systems, Vol. 3, Page 243, 2012.

4. Gunasekaran S., "Comparison of network intrusion detection systems in cloud computing environment", International Conference on Computer Communication and Informatics, Pages 1-6, 2012.

5. Albin E. and Rowe N. C., "A realistic experimental comparison of the Suricata and Snort intrusion-detection systems", 26th International Conference on Advanced Information Networking and Applications Workshops, Pages 122-127, 2012.

6. Kacha C. and Shevade K. A., "Comparison of different intrusion detection and prevention systems," International Journal of Emerging Technology and Advanced Engineering, Vol. 2, Pages 243-245, 2012.

7. Park W. and Ahn S., "Performance comparison and detection analysis in snort and suricata environment," Wireless Personal Communications, Vol. 94,Pages 241-252, 2017.

8. Shah S. A. R. and Issac B., "Performance comparison of intrusion detection systems and application of machine learning to Snort system," Future Generation Computer Systems, Vol. 80, Pages 157-170, 2018.

9. Baykara M. and Resul D., "Saldırı tespit ve engelleme araçlarının incelenmesi," Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi, Cilt 10, Sayfa 57-75, 2019.

10. Beale J. , "Snort 2.1 Intrusion Detection", Syngress, MA, USA, 2004.

11. Elmubarak M., Karrar A., and Hassan N., "Implementation Hybrid (NIDS) System using Anomaly Holt-winter Algorithm and Signature based Scheme," 2019.

12. Roesch M., "Snort: Lightweight intrusion detection for networks", Lisa, Pages 229-238, 1999.

13. Bukac V., "IDS system evasion techniques," Master. Masarykova Univerzita, 2010.

14. Stiawan D., Abdullah A. H., and Idris M. Y., "The trends of intrusion prevention system network," in 2nd International Conference on Education Technology and Computer, Pages V4-217-V4-221, 2010.

15. Wang Y., Meng W., Li W., Li J., Liu W.-X., and Xiang Y., "A fog-based privacy-preserving approach for distributed signature-based intrusion detection," Journal of Parallel and Distributed Computing, Vol. 122, Pages 26-35, 2018.

16. Ersoy M. and Emik M. H., "Akış Tabanlı IP Ağlarda Ağ Trafiği Üzerinde Yapay Sinir Ağı Kullanılarak SYN Seli Saldırılarının Tespiti", Uluslararası Mühendislikte Yapay Zeka ve Uygulamalı Matematik Konferansı, Sayfa 77-88, Antalya, 2019.

17.Kolahi S. S., Treseangrat K., and Sarrafpour B. , "Analysis of UDP DDoS flood cyber attack and defense mechanisms on Web Server with Linux Ubuntu 13", International Conference on Communications, Signal Processing, and their Applications, Pages 1-5, 2015.

18. Bou-Harb E., Debbabi M., and Assi C., "Cyber scanning: a comprehensive survey," Ieee communications surveys & tutorials, Vol. 16, Pages 1496-1519, 2013.

19. Haris S., Ahmad R., and Ghani M., "Detecting TCP SYN flood attack based on anomaly detection", Second International Conference on Network Applications, Protocols and Services, Pages 240-244, 2010.

20.Choi S.-H., Hwang D.-H., and Choi Y.-H., "Wireless intrusion prevention system using dynamic random forest against wireless MAC spoofing attack", IEEE Conference on Dependable and Secure Computing, Pages 131-137, 2017.