Current Trends in Computer Sciences & Applications

DOI: 10.32474/CTCSA.2023.02.000148

ISSN: 2643-6744

Research Article

Smart IDS/IPS: A Novel Approach on Automating Intrusion Detection and Prevention System's Fine Tuning

Asad Raza[1]*, Asif Siddiqui[2], Haider Abbas[3] and Atif Chowhan[4]

[1]*Department of Computer Science New, Institute of Technology, USA*

[2]*Department of Computer Science, Dakota State University, USA*

[3]*Department of Information Security, National University of Sciences and Technology, Pakistan*

[4]*Department of Computer Science, Abu Dhabi Polytechnic, Institute of Technology, UAE*

***Corresponding author:** Asad Raza, Department of Computer Science, Institute of Technology, USA

Received: July 03, 2023 **Published:** July 10, 2023

Abstract

The Intrusion Detection/Prevention Systems (IDS/IPS), which are generally regarded as essential components of network based cyber security infrastructure may be deployed using open-source solutions like Snort or Commercial-Off-The-Shelf (COTS) products. Both open-source solutions as well as COTS products come with suggested recommendations and best practices put forth by the researchers' community and vendors for open source and COTS products, respectively. While these recommendations and best practices do greatly facilitate deployment and management of IDS/IPS solutions with a view to enhance efficient and accurate detection/prevention, however, they lack in offering specific solutions suited to individual needs of an organization's infrastructure, thus burdening the administrators with the added responsibility of fine-tuning the system in accordance with their specific environments. Policy management and limiting the number of false positives is key challenges for administrators in such deployments. In this paper, we introduce a new Smart IDS/IPS technology which not only relies on generic signature recommendations from the vendor but also automatically determines which signatures should be enabled/disabled according to the specific environment in which the IDS/IPS solution is deployed. The underlying idea of Smart IDS/IPS system is to apply application aware traffic profiling for efficient and accurate detection of signatures. For implementation, the application awareness engine may be deployed as an integral process of the IDS/IPS solution or as a stand-alone application inspection service within the existing network infrastructure.

Keywords: Intrusion Detection System; Intrusion Prevention System; Smart IDS/IPS; Snort

Introduction

The Intrusion Detection/Prevention Systems IDS/IPS have been in the market since 1998 when Snort was written and developed. Intrusion technology has matured a lot with time due to two main factors. One is the increase of cyber-attack sophistication and second is the interest of big players like Cisco, McAfee, and IBM to gain market share. Research and development of IDS/IPS projects significantly increased as organizations started seeing the importance of IDS/IPS as part of their cyber security strategy. Most of the organizations are still reluctant to use Intrusion's detections Systems in prevention mode and it is because intrusion technology still requires a lot of manual fine tuning of signatures. False positives and false negatives are one of the core contributors in IDS/IPS list of problems. IDS/IPS fine tuning is a long and tedious process, it could take months and immense amount of manpower to come up with meaningful results. Presently there are numerous IDS/IPS products available in the market. Host Based Intrusion Detection/Prevention systems commonly known as HIDS/HIPS are for endpoint protection that is for servers, desktops, and laptops. Network Based Intrusion Detection/Prevention devices NIDS/NIPS are deployed to inspect and protect network-based intrusions. Network behavior based or anomaly detection systems are a newer line of products and vendors usually market them as zero-day attack detection tools. Network and Host based IDS/IPS systems are built on signature detection engines, whereas behavior-based systems mostly rely on traffic flows and patterns. Smart IDS/IPS systems can address signature applicability issues associated with signature-based Intrusion Detection/Prevention Systems.

Literature Review

Several different techniques focused on enhancing the accuracy of IDS/IPS solutions have been proposed by the researchers' community. The hybrid intrusion detection prevention system [1] proposes the integration of signature and anomaly-based systems for detection and prevention of malicious attacks by correlating anomalous behavior with baseline recommendations used by the signature-based systems. The paper also suggests the implementation of a Virtual Machine Monitor based Honeypot for enhancing accuracy of the hybrid intrusion detection prevention system through utilizing Fuzzy Genetic Algorithm (GA) for analyzing the network traffic. The implementation of Fuzzy GA has also been proposed in [2], with the distinction of using signature matching algorithm for identification of internal attacks and Fuzzy GA for implementation of external attacks detection. The system relies on classification of SQL queries and incoming packets as normal or anomalous for detection of internal and external attacks, respectively. The paper also suggests the deployment of neural networks and clustering algorithms for reducing the number of false alarms and thus improving the overall accuracy of IDS/IPS.

The limitations of using a rule-based signature matching approach leading to reduced accuracy of the system has also been highlighted in [3], which proposes the isolation of true conditions for probabilistic prediction of likely conditions that could be observed. The paper proposes a probabilistic abductive reasoning approach for augmenting the existing capabilities of a rule-based signature matching IDS/IPS for detection of polymorphic attacks by predicting the rule conditions that may probably occur and generating new rules using the existing rules to relieve administrators from the burden of continually updating rules database. The technique presented in [4] proposes the implementation of Deep Packet Inspection (DPI) for generation of signatures using network packet metadata extracted from packet headers, thus offering a fast signature-based IDS/IPS solution which works equally well for encrypted network traffic. Some other techniques like the high-speed flow-based intrusion detection presented in [5] focus on performance issues and compatibility of existing solutions with modern high-speed network links for faster detection. The said paper proposes the use of flow monitoring based on the Internet Protocol Flow Information Export (IPFIX) standard in conjunction with the IDS/IPS, thus offering an improved version of the IPFIX based signature matching intrusion detection system while incorporating application layer HTTP flows. The proposed solution is capable of handling four times higher network data rates while maintaining the same event detection rate and providing application aware detection. The dependence of signature matching techniques on the available signatures and the tendency of anomaly-based techniques to produce false alarms is addressed through utilization of Clonal Selection Algorithm (CSA) as proposed in [6]. The paper also presents the evaluation results by comparing measures like recall, precision, and F-score of a Snort IDS with no added improvement versus Snort IDS improved by Negative Selection Algorithm (NSA) and the proposed approach, highlighting that the proposed approach renders better results than the other two approaches. The technique proposed in [7] presents a novel approach for achieving greater accuracy in detection of malicious network traffic mainly comprising of shell code patterns using artificial neural networks. The above techniques and several others that were studied during the course of this research [8-11] strengthened our inclination towards exploring various possibilities like utilizing complex algorithms and making use of interdisciplinary approaches for reducing the number of false alarms and incorporating intelligence in the intrusion detection/prevention system.

Intrusion Detection/Prevention Systems Signature Applicability

Intrusions Detection/Prevention systems make their decision based on signatures. Signatures are the key factor in terms of vulnerability detection and elimination [12]. These devices come up with thousands of signatures out of the box and signatures are categorized as High, Medium, and Low based on vendor recommendations and confidence level. Open-source tools like OSSEC, Snort mostly relies on security community contributions with limited research and support capabilities. Signatures are then applied to inspection rules, these rules indicate the actions i.e., Alert, Drop, Investigate etc. IDS/IPS signatures get triggered after the traffic is decoded and prefilters are applied. Signature detection engine applies rules and alarms for detection or prevention. The sequence in which these operations are executed in snort is shown below in Figure 1.

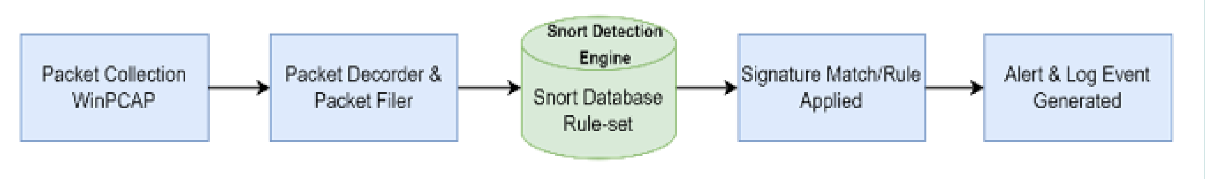

Packet Collection WinPCAP
Packet Decorder & Packet Filer
Snort Detection Engine
Snort Database Rule-set
Signature Match/Rule Applied
Alert & Log Event Generated

Figure 1: Snort Sequence of Operations.

The following are the Caveats to this flow.

a. Signature actions are static in nature and dependent on rule set. For example, if a traffic is set for "Alert" in rule set it will stay in that state until an IDS administrator manually changes it.

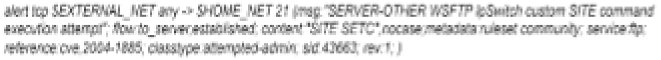
b. There is no correlation between signatures and applications currently used in the environment.

c. Dynamic activation and deactivation of signatures is not possible.

Now Let us Examine the First Sample Snort Rule Which Gets Fired When the Following Criteria are Met.

a. Source / Destination Network = EXTERNAL_NET/ HOME_NET

b. Source Port / Destination Port / Service = Any/21/FTP



```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"SERVER-OTHER WSFTP IpSwitch custom SITE command
execution attempt"; flow:to_server,established; content:"SITE SETC",nocase;metadata:ruleset community; service:ftp;
reference:cve,2004-1885; classtype:attempted-admin; sid:43663; rev:1; )
```

The second sample snort rule gets fired when the following criteria are met.

a. Source / Destination Network = EXTERNAL_NET/ HOME_NET

b. Source Port / Destination Port / Service = Any/Any

c. Pattern = /^Location\s*\x3a\s*\w+\x3a\/\/([^\n] *\x3a)?[^\n]{128}/ims

d. Content = “location”

The first rule generates an alert in case an authenticated user tries to exploit a vulnerability in WS_FTP server. The second rule generates an alter if a buffer overflow attack is launched windows 98. These signatures are very old, but they still exist in snort database [13] and there are several rules like these in the snort database. In most of the environments these signatures will not be applicable and even if they are applicable, they should be in disabled state after the vulnerability is patched. However, most of the environments do not bother to audit signature validity and relies at the mercy of vendor or snort signature updates because the number of signatures keeps going up and it will be a daunting task to remove unwanted signatures.

Application Awareness and Traffic Profiling

The latest attacks have redefined the cyber security landscape. Many of these attacks were able to bypass traditional security devices that have the capability of Layer 3 and Layer 4 inspection. The new generation of hackers are mostly interested in inserting malicious payload by considering the traditional security enforcement points. It is very common to have HTTP/HTTPS ports open in the outbound direction, and to DMZ servers in inbound direction. Attack can be initiated by modifying the payload with malicious content and sending it over open ports. This is how most of the malware, bots, injections operate in today’s world.

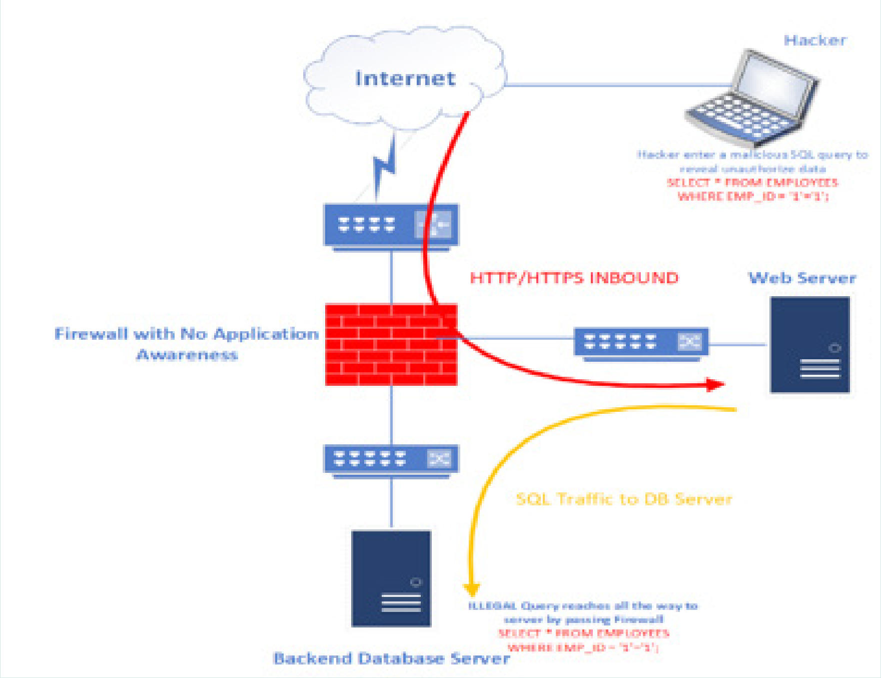

Internet
Hacker
Hacker enter a malicious SQL query to reveal unauthorize data
SELECT * FROM EMPLOYEES WHERE EMP_ID = '1'='1';
HTTP/HTTPS INBOUND
Web Server
Firewall with No Application Awareness
SQL Traffic to DB Server
ILLEGAL Query reaches all the way to server by passing Firewall
SELECT * FROM EMPLOYEES WHERE EMP_ID = '1'='1';
Backend Database Server

Figure 2: Flow of malicious SQL command reaching server bypassing application un-aware firewall.

Client server web applications are commonly deployed using three tier architecture, client, web server and backend database server. In most of the SQL injection attacks hackers pass malicious queries using web front end and lack of database security configurations can result in private data exfiltration. Figure 2 illustrates the flow of malicious SQL payload into the victim’s network using open ports on traditional firewalls. SQL injection is probably one of the most well-known remote malicious code execution attacks. According to a study conducted by Lionel [14], SQL injection attacks constitute about 33 % of all the breaches identified in 2022. It is still one of the most prevalent sources of vulnerabilities in web applications. Blaster worm is another malware that uses legitimate TCP Port 135 necessary for Windows operation to launch a distributed denial of service attack against Windows Update Service. The list of such attacks keeps growing and has become a dominating force for the next generation firewalls. Security vendors like Palo Alto, Checkpoint, Cisco, Symantec are coming up with a variety of application aware products. Application targeted attacks can be detected or blocked by Intrusion Detection/Prevention Systems if the signature exists in the database. However, the IDS/IPS inspection comes with the cost of high maintenance and regular tuning for effective results unless it is configured with some autotuning approach [15]. Some of the constant maintenance work can be automated by designing IDS/IPS devices application aware. One thing to be careful of is not to confuse signature database with application database. Signature database consists of attack definitions, traffic patterns, malicious code, and limited application detection capabilities. Application database on the other hand is comprised of application signatures, which in turn can be used for security, forensics, data analytics, quality of service and network visibility enhancements. Application databases can also be used for profiling traffic and mobile applications can have a separate profile than personal computers. Application identifications (APP-ID) can be created by using multiple techniques, signatures, behaviors, and network flows. Palo Alto application aware firewalls can even decrypt packets to determine application id.

Smart Ids/Ips Systems Architecture

Developing application aware appliances is not a new concept, for a very long time the technology was embedded in proxy solutions for inspecting web traffic and content filtering. Recently we have seen application aware firewalls by vendors like Palo Alto and Checkpoint, it is one of the key factors in Palo Alto Next Generation firewall success. Application aware Intrusion Detection/Prevention systems have also existed in one form or the other. All IDS/IPS systems have Layer-7 intelligence, but it does not mean it can detect the traffic flowing through it and label them as Skype, Office 365, Bit-torrent, Salesforce, Facebook etc. Network Behavior Analyzers and Anomaly Detection appliances like Stealth watch are based on NetFlow or network communications to identify malicious sources and destinations. It is without a doubt that the higher the visibility these security devices have, the greater will be their protection capability. Data driven from application ids can not only be used for inspection, but it can also be useful in automating tasks for intrusion analysis. The proposed Smart-IDS/IPS in this paper will take advantage of both the signatures and application database for automation and fine tuning [16].

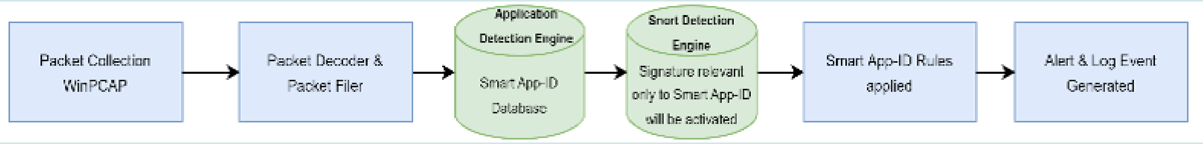

Packet Collection WinPCAP
Packet Decoder & Packet Filer
Application Detection Engine
Smart App-ID Database
Snort Detection Engine
Signature relevant only to Smart App-ID will be activated
Smart App-ID Rules applied
Alert & Log Event Generated

Figure 3: Smart IDS/IPS Order of Operations.

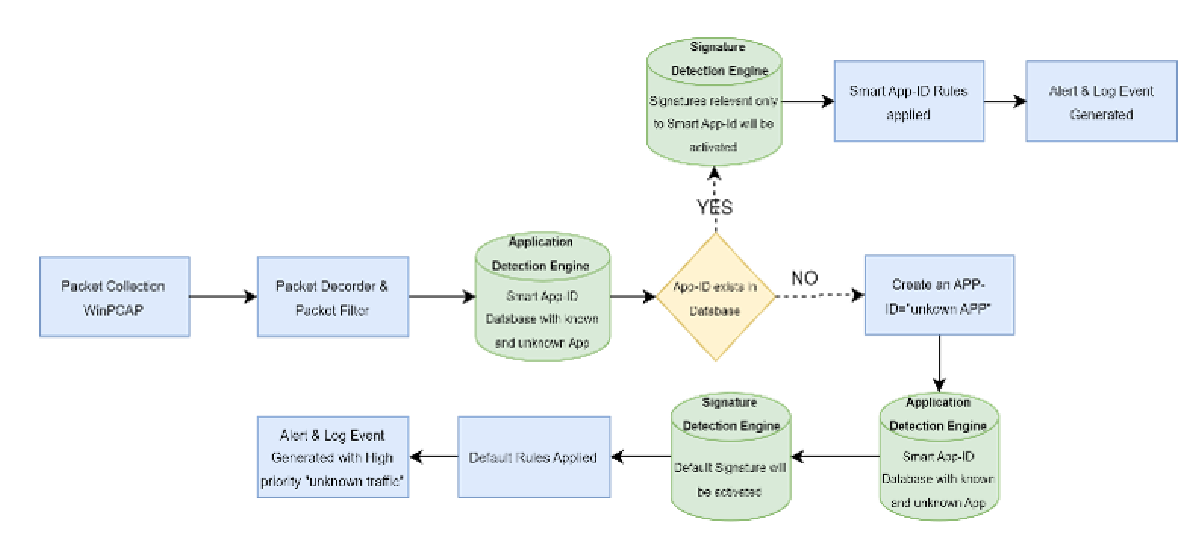

Signature Detection Engine
Signatures relevent only to Smart App-Id will be activated
Smart App-ID Rules applied
Alert & Log Event Generated
YES
Packet Collection WinPCAP
Packet Decorder & Packet Filter
Application Detection Engine
Smart App-ID Database with known and unknown App
App-ID exists in Database
NO
Create an APP-ID="unkown APP"
Application Detection Engine
Smart App-ID Database with known and unknown App
Signature Detection Engine
Default Signature will be activated
Default Rules Applied
Alert & Log Event Generated with High priority "unknown traffic"

Figure 4: Smart IDS/IPS For Known/Unknown Applications.

Proposed Smart IDS/IPS Architecture

Smart IDS/IPS systems correlates the data from Application database with traditional IDS/IPS signature database. As the traffic is processed by pre-filters, it must go through the Application database to generate "smart app-id". Once the "smart app-id" is generated it will be compared against signature database for any malicious code match for the specific application. In this way only relevant signatures will get activated at a given time, which can also reduce false positives and resource allocations. Smart App-ID is the core of the Smart IDS/IPS architecture, Figure 3 depicts the normal operation of Smart IDS/IPS if there is a match of application signature. The order of operation will be different in case there is no matching application in the database, Figure 4 shows the case where the application signature doesn't exist in application database, generic APP-ID= "Unknown-APP" signature will be created and added to the application database. Unknown-APP ID will then be processed by snort signature detection engine, default signatures and rules will be applied. Unknown Application IDs are reviewed by Smart IDS/IPS administrator for proper categorization of applications. Application profiling will help in building an up-to-date database, security alerts will be more targeted, and anomalies can easily be detected. In larger implementations Smart IDS/IPS can be integrated with third party application databases or firewall application databases for centralized management. APP-ID's do require a constant update from application/content update server just like signature updates. Once we have the application and signature database correlated, we can incorporate numerous tasks without user intervention.

Smart IDS/IPS Automation Engine

Smart IDS/IPS can perform procedures that are not possible with traditional IDS/IPS systems. Rules and signatures will be activated only for the matching App-ID as an on-demand service with the initial expiration of 30 days. After 30 days of App-ID unavailability, signatures and corresponding rules will go back to disabled state. In order to understand the dynamic behavior, consider a scenario of skype traffic flowing through Smart IDS/IPS shown in Figure 5.

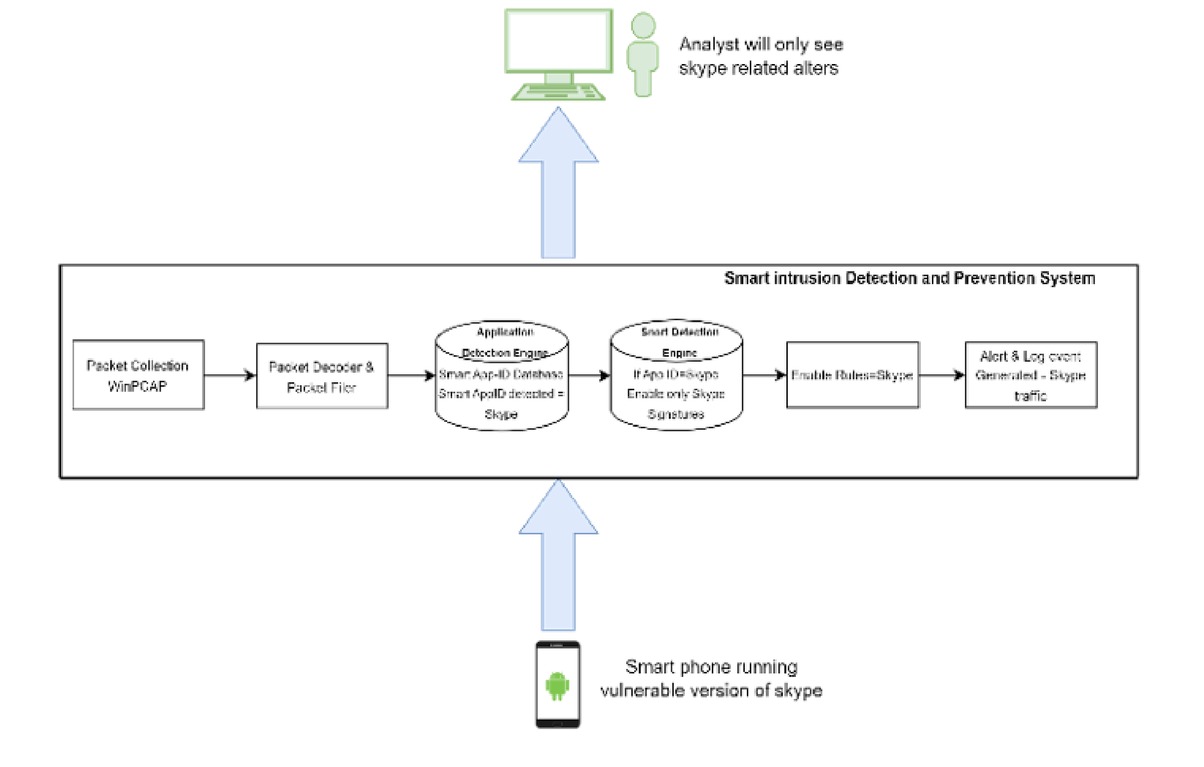


Figure 5: Smart IDS/IPS enabling Skype signature only for 30 Days.

In this scenario vulnerable skype traffic flows through a Smart IDS/IPS unit and following sequence of events occur:

a. Application detection engine matched the traffic as skype traffic based on an application signature match, App-ID detected and sent over to Snort inspection engine.

b. In the scenario above vulnerable skype traffic flows through a Smart IDS/IPS unit and following sequence of events occur:

i. Application detection engine matched the traffic as skype traffic based on an application signature match.

ii. App-ID detected and sent over to Snort inspection engine.

iii. Since the traffic is already pre-filtered by Application engine, Snort will only enable the skype related signatures.

c. There is a period of thirty days the skype signatures will be enabled for, if the IDS/IPS device doesn't see the skype traffic for 30 days (programmable feature), it will automatically disable skype signatures.

Note that the time bound signature can be very useful for IDS/IPS auditing and review from compliance point of view. For instance, Windows 98 signatures are still enabled in traditional IDS/IPS devices even though the operating system doesn't exist in most of the environments (refer-image). Smart IDS/IPS devices only activate a signature for relevant traffic with the expiration time.

Comparison Of Smart Ids/Ips to Traditional Ids

Application-aware smart Intrusion Detection Systems (IDS) have several advantages over traditional IDS like Snort: Table 1 shows the comparison between a traditional IDS (snort) vs Smart IDS/IPS.

Table 1: Comparison of Traditional Ids Vs Smart Ids.

Feature	Traditional IDS (snort)	Smart IDS
Detection Method	Signature-based	App-ID and Signature based
Application awareness	No	Yes
Rules Activation	Activated by default	On-demand Activation
False Positive Rate	Moderate to high	Low to moderate
False Negative Rate	Low to moderate	Low
Flexibility	Limited	High
Performance	Slow	Fast due to on-demand activation
CPU Utilization	High	Low to moderate
Ram Utilization	High	Low to moderate
Scalability	Limited	High
Deployment	Easy	More complex due to training requirements

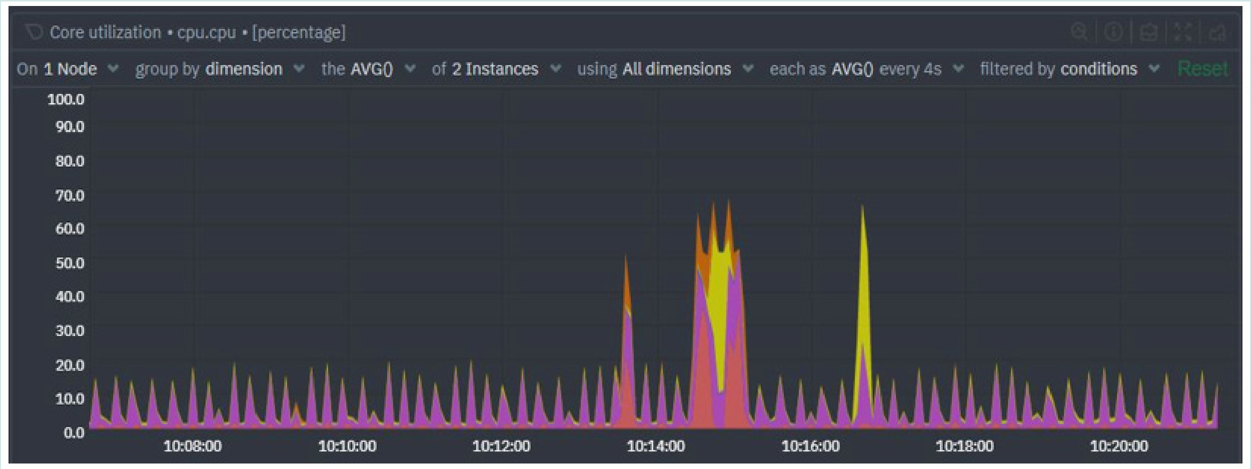
Improved accuracy: Smart IDS can detect a wider range of application-specific attacks that traditional IDS may not detect. Smart IDS can identify the different protocols and applications that are being used on the network, and analyze the data exchanged between them to detect any anomalies or threats.

Reduced false positives: Smart IDS can analyze application-level protocols and payloads, which means they can make more accurate decisions on what is normal and what is malicious. This results in fewer false positive alerts, which reduces the amount of time analysts must spend reviewing alerts.

Greater visibility: Smart IDS can provide deeper visibility into the applications and protocols that are being used on the network. This allows security teams to identify and respond to threats more quickly and efficiently.

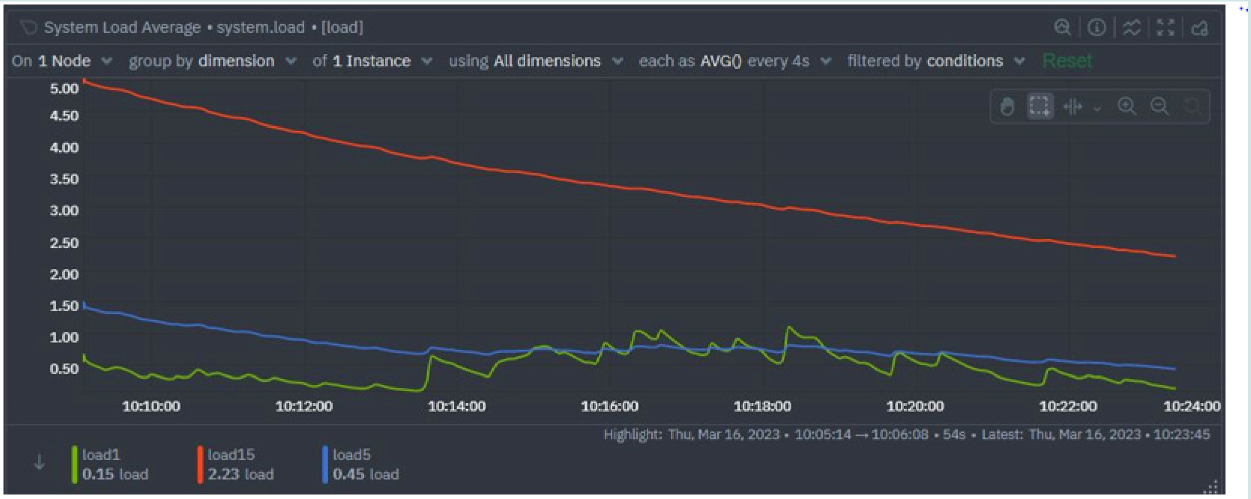
Faster incident response: Smart IDS disables the rules after 30 days and are activated only after AP-ID matches. This reduces the time to detect and respond to threats and can help prevent attacks from spreading or causing further damage.

Compliance: Smart IDS can help organizations meet regulatory compliance requirements by providing more detailed and accurate reporting on security incidents. They can also help organizations monitor and enforce policies related to specific applications and protocols (Figures 6-6c).


Core utilization • cpu.cpu • [percentage]

On 1 Node group by dimension the AVG() of 2 Instances using All dimensions each as AVG() every 4s filtered by conditions Reset

Figure 6 (a): System CPU Utilization while using Smart IDS.



System Load Average • system.load • [load]

On 1 Node group by dimension of 1 Instance using All dimensions each as AVG() every 4s filtered by conditions Reset

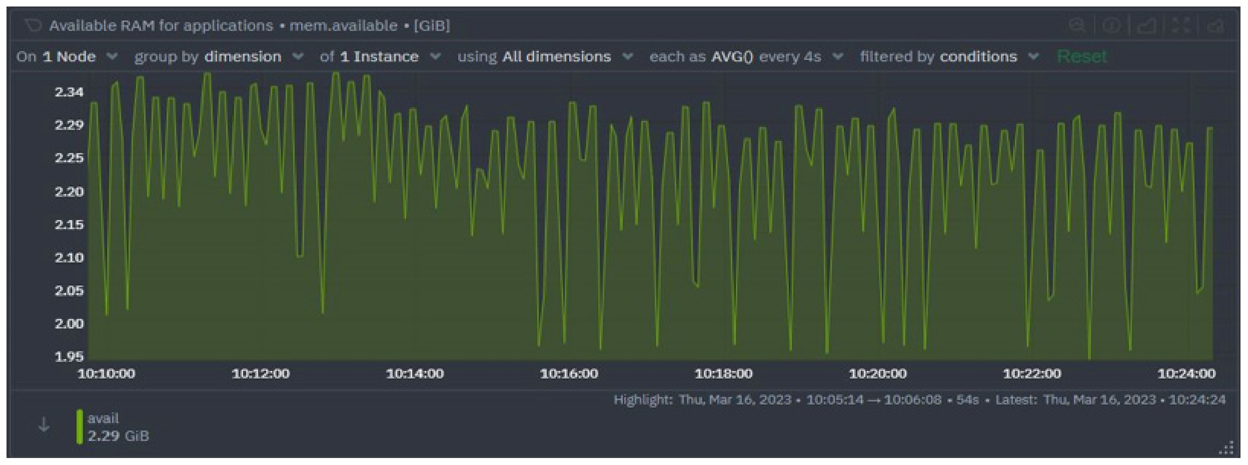
Highlight: Thu, Mar 16, 2023 • 10:05:14 → 10:06:08 • 54s • Latest: Thu, Mar 16, 2023 • 10:23:45

load1 0.15 load

load15 2.23 load

load5 0.45 load

Figure 6 (b): System Load Utilization.



Available RAM for applications • mem.available • [GiB]

On 1 Node group by dimension of 1 Instance using All dimensions each as AVG() every 4s filtered by conditions Reset

Highlight: Thu, Mar 16, 2023 • 10:05:14 → 10:06:08 • 54s • Latest: Thu, Mar 16, 2023 • 10:24:24

avail 2.29 GiB

Figure 6 (c): Available memory while using Smart IDS.

Performance Evaluation of Smart Ids/Ips

An Intrusion Detection System (IDS) can have an impact on the performance of the underlying operating system, as it involves monitoring network traffic and analyzing it for suspicious activity. The exact impact on performance can vary depending on the specific IDS implementation and the hardware and software environment it is operating in. An IDS can consume CPU resources, memory, and network bandwidth, which can impact the overall system performance. However, the impact can be minimized by optimizing the IDS settings, such as reducing the number of rules or alerts, or using hardware acceleration for processing network traffic. In this section we have evaluated the performance of Smart IDS/IPD which enables the rules based on AP-ID in comparison to the traditional IDS (snort) where all the rules are enabled by default. The system specifications for this evaluation are given in Table 2. During the evaluation process we deployed an open-source intrusion detection system (snort) on Ubuntu 0.04 LTS. The snort was configured to work in default mode which means if an attack is launched on the target system, snort will go through all the rules to generate alerts. Figure 7 (a) shows when an attack was launched. the CPU utilization jumped to 90 % and between the time intervals 10:40-10:44 the CPU utilization remained between 90%-100%. Of course, it depends on the system capacity and the CPU utilization may vary depending on the system specifications. But this deployment gives a reasonable estimate of traditional IDS resource consumption when the system is under attack. Figure 7(b) shows the three load averages in time series, and it should be noticed that the average load started increasing and jumped from 1 to 11 between the time intervals 10:36-10:48 when snort started generating alerts due to the attack. The RAM availability was around 2.5 GB when the smart IDS was working, whereas Figure 7(c) shows that the available memory declined to 1 GB and even less during the time interval 10:42-10:48 which means that snort utilized RAM more than double in comparison to Smart IDS [17]. Figure 6 (a) shows that the average CPU utilization is around 20% while smart IDS was action. We can notice some spikes between

10:14 to 10:10 but the overall average remains around 20%. Figure 6 (b) shows the 3 load averages between time interval 10:10 and 10:24. It should be noticed that while under attack smart IDS did not put significant load on the system and the average load is between 1.5 and 5 Figure 6 (c) shows that the available memory remined between 2.29 GB to 2.35 GB during the entire time interval while Smart IDS was running. The summary of results is shown in Table 3

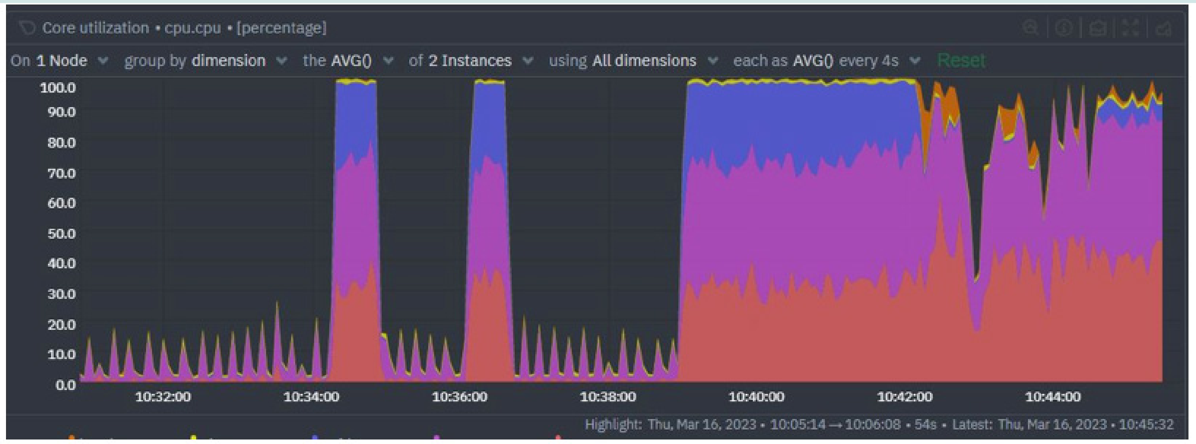


Figure 7 (a): System CPU Utilization of Snort.

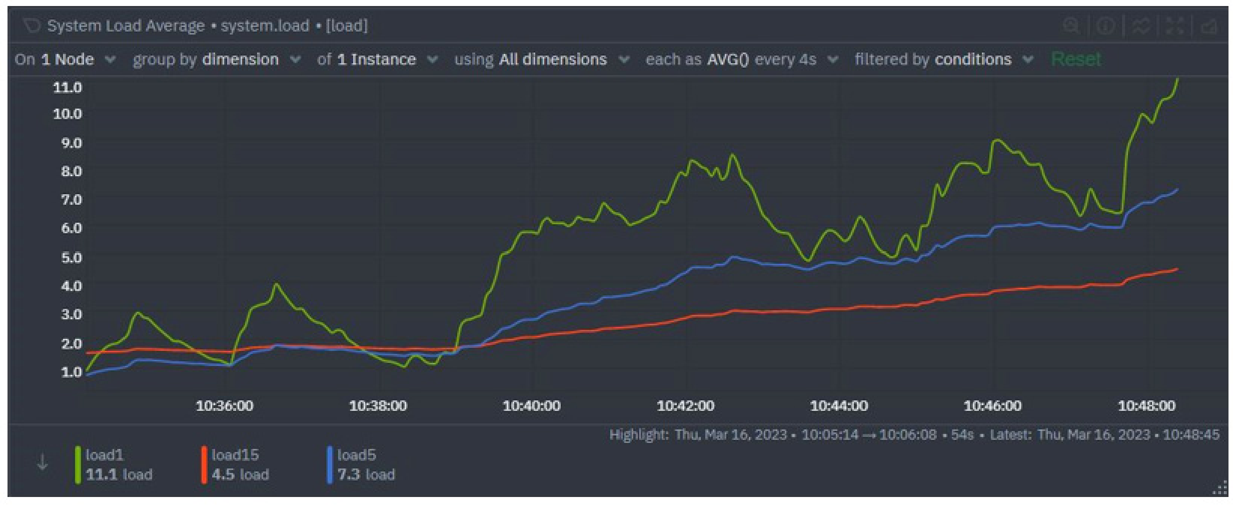


Figure 7 (b): System load utilization.

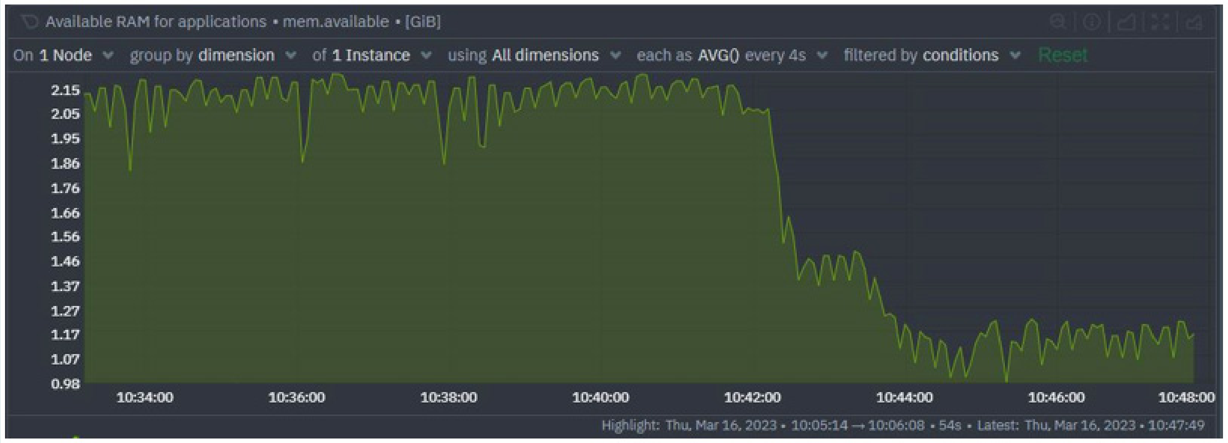


Figure 7 (c): Available memory while using Smart IDS.

Table 2: System Specifications.

Operation System	Ram	CPU	IDS	Monitoring Tool
Ubuntu 0.04 LTS	4 GB	Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz	Snort & Smart IDS	Net Data

Table 3: Performance Evaluation Summary.

Feature	System in Idle State		Traditional IDS (snort)	Smart IDS
CPU Utilization	5%		90%-100%	20%-30%
Available RAM	3 GB		0.8 GB – 1.0 GB	2.5 GB
System Load	< 1		11	Between 1.5–5
CPU interrupts	<500		4000-7000	<1000
Swap Utilization	0.01 GB		0.09 GB – 0.15	0.03 GB – 0.05GB

Conclusion

Intrusion detection systems play a significant role in the protection of organizational network infrastructure. There are two major issues with the traditional intrusion detection systems. Firstly, traditional IDS are not application aware which means that they cannot differentiate between the packets of various applications. This means that snort will process the packet even though the application is not running in the environment which will directly impact the resource utilization. Secondly, in traditional IDS all the rules are enabled by default which means that snort will match the packets will all the rules in its database until it finds a match [18]. For example, Windows 98 and XP are considered obsolete operating systems but snort has several rules related to this operating system which still exists in its database. Similarly, there are several applications which no longer exist in organizational environments but the rules for these vulnerable applications still exist and are enabled by default in short. In this research we have suggested the idea of Smart IDS which not only is application aware, but it also disables the rules which are not relevant in an environment. This automates the tuning process, consumes less resources, provides more flexibility and increases the performance of IDS in detecting malicious traffic.

Footnote

This paragraph of the first footnote will contain the date on which you submitted your paper for review, which is populated by IEEE. It is IEEE style to display support information, including sponsor and financial support acknowledgment, here and not in an acknowledgment section at the end of the article. For example, "This work was supported in part by the U.S. Department of Commerce under Grant 123456." The name of the corresponding author appears after the financial information, e.g. (Corresponding author: Second B. Author). Here you may also indicate if authors contributed equally or if there are co-first authors.

References

1. Rizvi, Syed, Gabriel Labrador, Matt Guyan, Jeremy Savan (2016) Advocating for hybrid intrusion detection prevention system and framework improvement. Procedia Computer Science 95: 369-374.
2. Desai Anuja S, D P Gaikwad (2016) Real time hybrid intrusion detection system using signature matching algorithm and fuzzy-GA. In 2016 IEEE international conference on advances in electronics, communication, and computer technology (ICAECCT) pp. 291-294.
3. Ganesan Ashwinkumar, Pooja Parameshwarappa, Akshay Peshave, Zhiyuan Chen, Tim Oates (2019) Extending Signature-based Intrusion Detection Systems with Bayesian Abductive Reasoning pp. 1-10.
4. Papadogiannaki, Eva, Dimitris Deyannis, Sotiris Ioannidis (2020) Head (er) Hunter: Fast Intrusion Detection using Packet Metadata Signatures. In 2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD) pp. 1-6.
5. Erlacher Felix, Falko Dressler (2020) On High-Speed Flow-based Intrusion Detection using Snort-compatible Signatures 99(1): 1-1.
6. Elshafie Hussein M, Tarek M Mahmoud, Abdelmgeid A Ali (2019) Improving the performance of the snort intrusion detection using clonal selection. In 2019 International Conference on Innovative Trends in Computer Engineering (ITCE) pp. 104-110.
7. Shenfield, Alex, David Day, Aladdin Ayesh (2018) Intelligent intrusion detection systems using artificial neural networks. ICT Express (2): 95-99.
8. Aldweesh Arwa, Abdelouahid Derhab, Ahmed Z Ema (2020) Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. Knowledge-Based Systems 189(1): 105124-105124.
9. Qureshi Aqsa Saeed, Asifullah Khan, Nauman Shamim, Muhammad Hanif Durad (2019) Intrusion detection using deep sparse auto-encoder and self-taught learning. Neural Computing and Applications 32(1): 3135-3147.
10. Naseer Sheraz, Yasir Saleem (2018) Enhanced Network Intrusion Detection using Deep Convolutional Neural Networks. TIIS 12(10): 5159-5178.
11. Vieira Kleber, Fernando L Koch, João Bosco M Sobral, Carlos Becker Westphall, Jorge Lopes de Souza Leão (2019) Autonomic Intrusion Detection and Response Using Big Data. IEEE Systems Journal14(2): 1984-1991.
12. Rafath Samrin, D Vasumathi (2017) Review on anomaly-based network intrusion detection system , International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT) pp.141-147.
13. Snort Rules, Available at, Accessed on Oct 13,2022.
14. Lionel Sujay Vailshery (2022) Global web application critical vulnerability taxonomy.

15. Zhenwei Yu, Jeffrey J P Tsai, Thomas Weigert (2007) An Automatically Tuning Intrusion Detection System , IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 37(2): 373-384.

16. Hussein Alnabulsi, Md Rafiqul Islam, Quazi Mamun (2014) Detecting SQL injection attacks using SNORT IDS, Asia-Pacific World Congress on Computer Science and Engineering.

17. Yi-Ying Zhang, Jian Luo, Yeshen He, Shengguo Ma, Kun Liang, et al. (2021) Smart Meter Intrusion Detection Based on APSO-DBN Model , 2021 Smart City Challenges & Outcomes for Urban Transformation (SCOUT).

18. Arshid Ali, Shahtaj Shaukat, Muhammad Tayyab, Muazzam A Khan, Jan Sher Khan et al. Network Intrusion Detection Leveraging Machine Learning and Feature Selection , IEEE 17th International Conference on Smart Communities: Improving Quality of Life Using ICT, IoT and AI (HONET) pp. 14-16.

DOI: 10.32474/CTCSA.2023.02.000148