ISSN 2229-3531
(Print)

JUCIT Vol. 13(1), 1-10 (2022). Periodicity-2-Monthly

ISSN 2455-9997
(Online

JOURNAL OF ULTRA COMPUTER & INFORMATION TECHNOLOGY
An International Open Free Access Peer Reviewed Research Journal of Computer Science Engineering & Information Technology
website:- www.compitjournal.org

Estd. 2010

Web Intrusion Detection Systems Comparison: Techniques and usage

ABDULLAH HAMAD ALQAHTANI

Department of Computer Science The University Of Sheffield United Kingdom
Corresponding Author Email:- Alqahtani.a@live.com

http://dx.doi.org/10.22147/jucit/130101

Acceptance Date 21st March, 2022, Online Publication Date 4th April 2022

Abstract

Web attacks are one of the most concern these days. Vulnerable applications require protection, which can be provided through web application firewalls (WAF) and web intrusion detection systems (WIDS). Some of them are signature based and some detect / protect through anomaly detection. Various commercial solutions have been offered by vendors like CISCO ACE application firewall, Barracuda application firewall *etc*. Open source community has also contributed some formidable solutions like ModSecurity, PHPIDS, Ironbee, WebKnight and Snort etc. No solution has yet proven to be the silver bullet and this area is still a subject of active research. Inability to detect any novel attack has been the common weakness and has lead to various techniques being proposed for identifying zero-day attacks. In this paper, we analyze various commercial and open source web application protection solutions and make comparative analyses of their strengths and weaknesses, identifying any areas that still need attention of the research community.

Key word : Web IDS, Intrusion Detection System, Web Application Firewall, Web attacks prevention systems.

1. Introduction

The rush to bring everything online resulted in myriad applications and web application platforms to be deployed online. The major focus of all these applications was usability with very less or no focus on security. As all these applications were a gateway to different business functions, they became vulnerable to be exploited by cyber criminals and malicious activity. A need was felt to secure these web applications without re-engineering them. This requirement of retrofitting security for web applications was fulfilled by Intrusion detection systems and web application firewalls. Numerous initiatives have been taken in this regards both in open source community and by commercial vendors of security products. But still no perfect solution has been found for this problem. Each product excels in one area but lags in others. This paper is an attempt to introduce notable initiatives and list their strengths and weaknesses.

1.1. *Web Attacks :*

Nowadays, many of applications and services are working on the web. Vulnerabilities on websites can be exploited by attackers to do their goals. Attackers can get data or damage it by using one of many types of attacks on the web. These attacks are differing in their mechanisms and impacts. Some of the most popular attacks are Injection flows (such as SQL injection, Local File Inclusion, Remote File Inclusion, LDAP injection, and Header injection), Cross Site Script (XSS), Cross-Site Request Forgery (CSRF), Directory Traversal (DT), Denial of Service (DoS) and Buffer Overflow[1-8].

2. *Intrusion Detection Systems :*

Intrusion Detection System IDS is security system or device used to monitor system or network to detect any attack or malicious behavior and analyze it. Attacks stored in log file or database and system administrator may alerted by that attack. The next generation of IDS is Intrusion Prevention System IPS which furthermore what traditional IDS do can prevent attack[9].

2.1. *Intrusion Detection System types :*

Intrusion detection systems can be categorized in four main categories based on IDS location. These types differ in their capabilities and their advantages and disadvantages.

2.1.1. *Network based IDS (NIDS) :*

This type of IDS based on capturing packets on network traffic and analyzing it to detect any malicious behavior. The NIDS can monitor all traffic enter the network it installed or placed before[13].

2.1.2. *Host based IDS (HIDS) :*

They monitor malicious activities on one single host. The HIDS is installed on the host and can analyze file system modifications, system calls, application logs and network traffic used by the host. Every HIDS is responsible only onthe host that HIDS installed on[11] as in figure 2.

2.1.3. *Combining NIDS and HIDS :*

Is the hybrid of combining network based IDS and host based IDS which produces a securely system that used commonly by organizations. The idea is to put sensor of traffic on network segment and on host segment. This will monitor malicious behavior or attack attempt on entire the network[9].

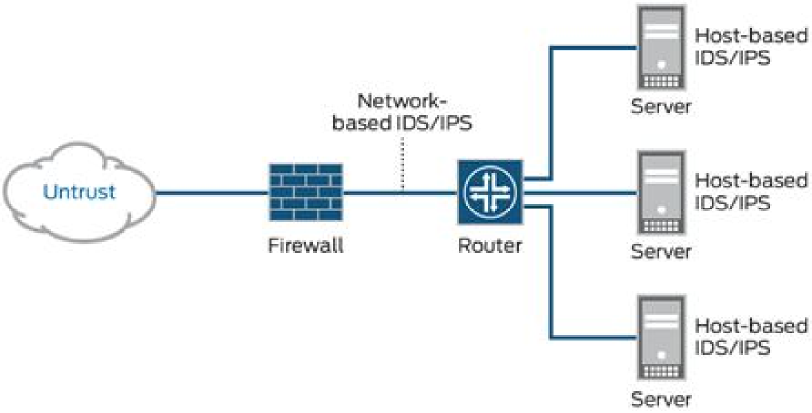

Figure 1 NIDS is Placed before network to detect any malicious data in traffic[33].

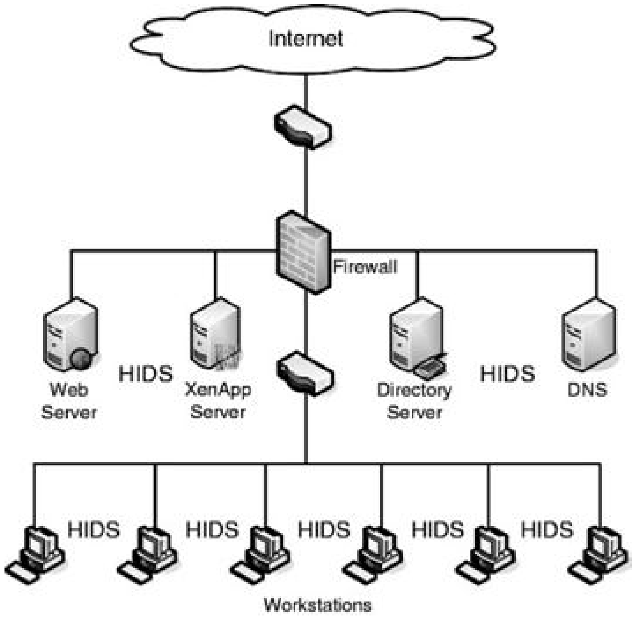

Figure 2: HIDS installed on every host in the network[34].

Table-1. The advantage of using Web IDS over NIDS to detect web attacks

Network IDS	Web IDS
Process huge data traffic	Process part of data traffic (only HTTP/HTTPS traffic)
Can not handle encrypted data	Can handle encrypted data
Can not handle compressed data	Can handle compressed data
Some of NIDS has problem with evasion problem when attacker use characters like Unicode and ASCII code.	Detecting done after data format conversion
Dedicated to works with TCP/IP layer.	Dedicated to works with application layer
Detect limited kinds of web attacks.	Most of web attacks can be detected

Actually, a few of applications use web based IDS like ModSecurity[10] and PHPIDS[11].

2.1.4. *Web Intrusion Detection System :*

Web IDS is one type of intrusion detection systems that can detect attacks on websites. This kind of IDS differs from NIDS in that Web IDS works on application layer level, http and https protocols, instead of network layer level as NIDS. Also, it differs from HIDS because it is not concerned to hosts. The advantages of using WIDS over NIDS for detecting web attacks are shown in table1.

2.2. *IDS techniques :*

To define and detect attacks and malicious behavior, there are two main techniques used to be applied on incoming data.

2.2.1. ***Misuse detection :*** It is also called signature or knowledge based detection. The idea is to compare the incoming data with predefined patterns (signature). The predefined patterns are a collection of patterns of different attacks. If any incoming data match pattern defined in signature, the incoming data is considered as attack.

2.2.2. ***Anomaly detection:*** This technique is based on monitoring the historical activities of the system to determine its behavior if it is normal or malicious. These activities can be anomaly general size of data traffic, general protocols used, ports connected between devices and other activities done by users or network. Any varies of these general activities can be considered as malicious[12].

2.3. *WEB IDS :*

This type of IDS is analysing request and reply traffic going to the web server to protect a specific website. The following are examples of some web-IDS tools.

2.3.1. *PHPIDS :*

PHPIDS is PHP web based intrusion detection and prevention system written in PHP and open source under LGPL license in 2007. It is developed as a security layer to detect and prevent intrusions that can affect PHP web pages. Its idea is to detect incoming data to web server and analysis it to decide if it has malicious code to prevent it. Every attack can be presented with its details and then stored it in PHPIDS log file.

Different kinds of attacks that can be detected by PHPIDS are SQL injection, LDAP injection, XSS, CSRF, directory traversal, header injection, RFE, LFI, and DoS attacks[11].

2.3.1.1. *PHPIDS Detection technique :*

PHPIDS uses signature technique by using regular expressions to detect malicious pattern of attacks. These regular expressions are used as rules in XML file named default_filter.xml updated frequently. Default filter file uses about 77 rules with different kind of malicious patterns and different impacts. Before data coming from client to web server, PHPIDS analyzes these data and converting encoding and formats if it necessary before comparing and matching malicious patterns in default_filter.xml file. If there is a match, the attacker will be prevented and a report for that attack stored in log file or database. Furthermore, PHPIDS Centrifuge component used to detect unknown malicious patterns. By analyzing incoming strings to find special characters that indicate that any attack. If the incoming strings are more than 25 characters, the ratio between the number of spaces, word characters, punctuation and the non-word characters are calculated. If that ratio less than or equal 3.5 then the incoming string considered as attack[11].

2.3.1.2. *PHPIDS pros and cons :*

PHPIDS is helpful software that can benefit network administrator or web owner to detect and prevent attacks on his website. It is easy to use and has a good community and support. On the other hand there are some drawbacks belong to PHPIDS use such as it just work on PHP based website in addition to the high positive alarms.

2.3.2. *Mod Security :*

It is an open source web application firewall/ IDS which is installed as a module in Apache web server[10]. Now IIS7 and Nginx and Java Servlet version are also available. ModSecurity installs as an embedded component of the web server and also can be placed as a reverse proxy server to filter and monitor every request. It is not language dependent like PHPIDS and provides an external protective layer to stop the attacks from reaching the applications. It provides extensive logging facilities and is able to log full HTTP requests and responses,, which can then be used to detect and prevent attacks. It can function both as a WAF or as a WIDS where real time monitoring of the HTTP traffic is carried out to detect attacks and then alerts can be generated to react to them.

2.3.2.1. ***Detection Techniques*** - ModSecurity ModSecurity uses a rule set to analyze requests and make decisions. ModSecurity rule engine implements ModSecurity Rule Language, which can be used to make rule set. This rule set supports both positive security model and negative security model. In negative security model the anomaly score of each request is calculated and monitoring is done for each request, user, source IP Address, session, maintaining their category wise anomaly scores. Any request crossing the threshold for anomalies is rejected or logged based on configured preference. ModSecurity Core Rule Set (CRS)[15] is developed and maintained by OWASP which provides rules to detect malicious activities.

2.3.2.2. *ModSecurity - Strengths :*

ModeSecurity has many advantages including the easy of use and the ability to detect high range of attacks. However, it is difficult to detect zero-day attacks.

2.3.3. *The Automatic Identification of Web Attacks System (AIWAS) :*

This is an anomaly detection intrusion detection system[16] specifically designed for web applications. It uses machine learning to build usage profiles of each web application being protected. After the learning phase is over the requests are matched against this usage profile and request is categorized either as attack or otherwise. It does not rely on signatures and thus has the potential to stop hitherto unknown attacks (zero day attacks). It uses Instance Models[9], which is a model for input values passed as parameters with the request

The AIWAS consists of two components: Sentinel which intercepts the message flow between the client and web server, maps the request to Instance Model. The another component is Oracle which classifies the input Instance Model as attack or valid request.

AIWAS can be used both as IDS and IPS. For learning phase AIWAS can be deployed in a live system for some time and under administrator supervision IM classifications can be monitored. This may take long time but the training data set is most accurate and representative. The other option could be the use of stored web server logs, provided they contain all the necessary data like POST request data *etc.*

2.3.3.1. *AIWAS – Strengths :*

One of the strength points in AIWAS is that it does not need signatures and thus able to detect novel attacks. And trains for particular application, thus no generic solution.

2.3.3.2. *AIWAS – Weaknesses :*

Focused on input validation only, introduces a single point of failure and brings in latency issues because of network communication and interception are most weaknesses in AIWAS.

2.3.4. *Snort :*

It is an open source and free NIDS[18], which performs network traffic analysis in real time by packet capturing. Snort has been subject of active research and various extensions have been made to it by the research community, making it a complete IDS / IPS solution packed with features. Snort performs as WIDS as it decodes application layer of a packet and various rules can be applied to the content of the decoded data. In addition to this it performs protocol analysis and is able to detect a vast variety of attacks.

2.3.4.1. *Snort – Detection Technique :*

Initially Snort was limited to signature based detection but then extensions were made to make it a hybrid system by incorporating anomaly detection[19]. Snort is essentially a sniffer which captures packets and then hands them over to decoder module which decodes the application layer of the packet and then passes it on to the preprocessor module which preprocesses the packet to validate packet headers, carries out packet defragmentation and reassembly of TCP streams and formats the packet information and data for use by the detection engine. Detection engine uses Boyer-Moore[20] string matching algorithm for application of rules stored in snort.conf file. Three types of rules in Snort that are Alert rules, Pass rules and Log rules.

2.3.4.2. *Snort – Strengths :*

Cross platform and open source and easy to use. The ability to support both signature and anomaly detection. In addition to Application level analysis.

2.3.4.3. *Snort – Weaknesses :*

Growing number of rules brings in latency issues as all these rules need be matched. Another drawback is the anomaly detection not effective with low probability of detection. In addition to one wrongly crafted rule will prevent Snort from starting and it will only be known on seeing log files[21].

2.3.5. *Bro :*

Bro is a UNIX based and open-source Network Intrusion Detection System (NIDS)[22]. It monitors network traffic for any malicious activity in passive mode. It uses the contents and attributes of the network traffic to make decisions. Its emphasis is on speed of analysis and preventing any dropped packets. It is achieved through isolating the mechanism from the policy aspects. It has the capability to decode and analyze the application layer for specific protocols like FTP, HTTP, telnet *etc.* It supports signature based as well anomaly based detection, though the primary detection mechanism relies on policy scripts written in Bro language.

2.3.5.1. *Bro – Detection Technique :*

Bro is a policy based IDS which dynamically detects the protocol though analyzing the payload. It parses the network traffic and takes out the semantic information about application layer[23]. This information is then used by event engine to detect intrusions through execution of event oriented analyzers which match it with patterns of malicious activity. The detection of attacks takes place through defined events and patterns of activity and also by signatures.

2.3.5.2. *Bro – Strengths :*

Does not rely on signatures for detection. Low probability of packet drops. Supports detection of attacks against web applications[15]. and the ability to detect novel attacks through policy scripts.

2.3.5.3. *Bro – Weaknesses :*

Limited to UNIX only in addition to steep learning curve is the most concern of using Bro.

2.3.6. *Suricata :*

Suricata is a highly scalable, cross platform and open source IDS / IPS developed by the Open Information Security Foundation[24]. It is able to detect the underlying protocol stream so that rules are not port specific rather protocol specific (e.g. HTTP running on port 8080 and not on port 80). It includes support for file identification using file types and MD5 checksums; it can be used for preventing data loss or exfiltration detection. It supports multithreading and is able take advantage of multiple cores in present day systems[25]. It supports both rule based and anomaly detection approaches. There is an embedded HTTP library to support decoding and analysis of application layer [16]. It also supports analysis of SSL/TLS streams. Like ModSecurity, it provides IP reputation functionality. Snort rule set can be easily used with Suricata.

2.3.6.1. *Suricata – Strengths :*

Highly scalable and superior performance under heavy volume environment because of multithreading[27], low packet drop rate[26] and automatic protocol detection and validation.

2.3.6.2. *Suricata – Weaknesses :*

Inability to detect novel attacks is the most weakness in Suricata.

2.3.7. *Web Application Intrusion Detection System (WAIDS) :*

WAIDS[28] is a web intrusion detection system which functions on the principle of anomaly detection. It is specially designed to detect input validation attacks against the applications. It uses a learning model and generates a profile of a web applications with the help of arguments passed with the web requests to the application during normal working of the application. This profile is then used to evaluate future requests for anomaly presence.

2.3.7.1. *WAIDS – Detection Technique :*

The technique used by WAIDS consists of four steps: In first step data about parameters contained in HTTP requests is collected. These parameters are used to pass values as arguments to the web application along with POST or GET requests. In the second step keywords are extracted from the data collected in the first step. These keywords may include SQL query operators and important parameter names. In the third step similarity is measured by using Optimal Sequence Detection (OSD). Any request received is matched with similar requests in the normal profile of the application built during learning phase. In the last step the runtime web requests are compared against the normal profile of the web application. Any malicious and anomalous activity is detected if it is no in conformity with the normal profile of the application.

2.3.7.2. *WAIDS – Strengths :*

The strengths in WAIDS are it is effective against Input validation attack, and able to detect novel attacks.

2.3.7.3. *WAIDS – Weaknesses :*

The drawback of WAIDS are only effective against input validation attacks, does not provide protocol validation, and introduces a single point of failure as per architecture proposed[28].

2.4. *Web Application Firewalls (WAF) :*

While WIDS detect the intrusions, the WAF[29] uses a set of rules to stop attacks directed at web applications like SQL Injection, Cross Site Scripting (XSS). The rules need to be created for every attack and there is no ability to identify novel attacks. Few notable WAFs are:

2.4.1. *WebKnight :*

It is an open source WAF for IIS and other web servers[30]. It scans all HTTP requests and applies filter rules to them. These rules are framed by application administrators. The rules correspond to general category of attacks like buffer overflow, SQL injections etc and not on specific threat. This enables protection against all known and novel attacks. It is essentially an ISAPI filter and thus integrated into the web server. This allows analysis of even encrypted traffic.

2.4.1.1. *WebKnight –Strengths :*

Some advantages of WebKnight are open source and free, perform protocol validation as per RFC, SSL protection, can be updated without restarts, and supports Authentication scanning for brute force attempts.

2.4.1.2. *WebKnight –Weaknesses :*

Only runs with IIS and other ISAPI filter supported web servers and only supports negative security model.

2.4.2. *NAXSI :*

Naxsi stands for (Nginx Anti XSS SQL Injection)[31]. It is an open source free WAF designed to work with NGINX only. It is characterized by white listing approach and aims at providing high performance with low maintenance and updation requirements. It does not depend upon signatures of attacks rather it builds a profile of the web application during learning phase and then detects variations fro that profile by detecting unexpected characters in HTTP requests and parameters. Not much documentation and support is available for it.

2.4.3. *Barracuda Web Application firewall :*

It is a hardware appliance, commercially available[32]. It protects the web applications against all known vulnerabilities. It supports positive security model where adaptive profile (white list) for the application is built and applied for detection of malicious activity. It supports rate limiting and IP reputation services. Also provide Data Loss Prevention service.

Table 2 : Comparison of IDS

	PHPIDS	Mod Security	Snort	AIWAS	Suricata	Bro	WAIDS
Platform	PHP based only	Cross platform	Cross platform	Cross platform	Cross platform	Unix only	Cross platform
Detection Technique	Signature-based	Rule, Anomaly and whitelisting	Rules and anomaly based	ML	Rule and anomaly based	Policy, and signature based	Anomaly-based
License	LGPL	GPL 2	GPL	-	GPL 2	BSD	-
IPS feature	No	Yes	Yes	Yes	Yes	No	No
Rules	PHP IDS rules	Mod Security, Snort, and PHPIDS rules	Snort, SO and emerging threats rules	-	Snort and emerging threats rules	Policy scripts	-
Offline log analysis	Yes	Yes	Yes (pcap files)	Yes	Yes (pcap files)	Yes (pcap files)	-
Documentation	Yes	Yes	Yes	No	Yes	Yes	No
User Interface	Yes	Third part only	Third part only	-	Third party only	Yes	-
Multithreading	No	No	No	-	Yes	No	-
Security Model	Negative	Negative Positive	Negative	Positive	Negative	Negative	Positive
Ability to detect novel attacks	No	Partly	Partly	Partly	No	Yes	Yes

3. *Analysis :*

It can be seen that every IDS discussed has something that distinguishes it from the others and no IDS can be said to be the soloution to all aspects of the problem of web application security. Table 2 below summarizes the various properties of these IDS.

4. Conclusion and future works

Attackers are adept at finding new ways to attack the web applications. These web applications present an attractive and vulnerable target to them because they have been engineered to usability and not security and have a potential to give them access to precious assets both monetary and intellectual property. Web Intrusion systems (WIDS) have also been evolving according to this threat. The WIDS analyzed in this paper are the most widely used and supported across the open source community. Though most of them work on signature based detection principle, the signatures / rules for any new threat are available within hours of its detection, thanks to the large open source community. But this clearly shows that they are most suitable for use by small and medium enterprises which are not threatened directly by zero day attacks. Large enterprises, which have to brace themselves for situations where novel zero day attacks are engineered keeping the vulnerabilities of their web applications in mind, have to stay one step ahead of the attackers to prevent themselves. For this reason they cannot rely on a reactive model of signature based detection rather have to be proactive to prevent intrusions. The anomaly detection capability may offer a solution in this regards, but its effectiveness is questionable. Bro may be a better choice in this situation because of its policy based approach. Also white listing or profiling approach used by ModSecurity through the use of ModProfiler is also a good option for such large enterprises. However, where theses open source WIDS stand when compared with the commercial IDS can only be made clear when a thorough evaluation of their detection capabilities is made in a test environment. In the future we are going to include different types of IDS and more comprehensive analysis and comparison. In addition to do evaluation of these tools on real network.

References

1. W. G. Halfond, A. Orso, "Detection and Prevention of SQL Injection Attacks," Malware Detection Advances in Information Security, Vol. *27,* pp.85-109 (2007).
2. sG. Johnson, (2008, Jan) "Remote and Local File Inclusion Explained", [OnLine]. Available: http://hakin9.org/remote-and-local-file-inclusion-explained/
3. J. M. Alonso, R. Bordon, M. Beltran and A. Guzman, "LDAP Injection Techniques," in 11th IEEE Singapore International Conference on Communication Systems, Singapore, 2008, pp. 980-986.
4. Jatinder. "Sending emails in PHP & email injection attacks." Internet: http://phpsense.com/2006/php-email-injection-attacks/ Oct. 15, 2006 [Oct. *19,* 2012].
5. G. A. Dilucca, A. R. Fasolino, M. Mastroianni, P. Tramontana, "Identifying Cross Site Scripting Vulnerabilities in Web Applications," Proceedings of the Sixth IEEE International Workshop on Web Site Evolution, Washington, USA, (Sept. 2007).
6. Z. Mao, N. Li, I. Molloy, "Defeating Cross-Site Request Forgery Attacks with Browser-Enforced Authenticity Protection," Financial Cryptography and Data Security, Berline: Springer, (2009).
7. A. R. Zade, and S. H. Ratil, "A Survey On Various Defense Mechanisms Against Application Layer Distributed Denial Of Service Attack," International Journal on Computer Science and Engineering, Vol. *3,* no. 11, (Nov. 2011).
8. C. Cowan, *et al*, "StackGuard Automatic Adaptive Detection and prevention of Buffer-Overflow Attacks," in SSYM'98 Proceedings of the 7th conference on USENIX Security Symposium, Vol. *7,* pp. 5-5, (1998).
9. R. Weaver, Guide to Network Defense and Countermeasures. Course Technology, (2006).
10. ModSecurity www.modsecurity.org, (Last accessed 10 March 2022).
11. "PHPIDS Web application security 2.0," www.phpids.org, (Last accessed 10 March 2022).
12. M. E. Whitman, and H. J. Mattord, Principles of Information Security, Boston: Course Technology, (2009).
13. H. Hochheiser, and B. Shneiderman, "Using Interactive visualizations of WWW log data to characterize access patterns and inform site design," Journal of the American Society for Information Science and Technology. Vol. *52*, no. 4, pp. 331-343, (Feb. 2001).
14. OWASP Core Rule Set – Rule Template for ModSecurity https://www.owasp.org/index.php/ModSecurity_CRS_Rule_Description Template (Last accessed 11 March 2022).
15. OWASP Core Rule Set – ModSecurity. https://www.owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project (Last accessed 11 May 2022).
16. Huynh, Toan and James Miller. "AIWAS: The Automatic Identification of Web Attacks System," International Journal of Systems and Service-Oriented Engineering (IJSSOE) 3 (2012): 1, accessed (March 10, 2022), doi:10.4018/jssoe.2012010105
17. Huynh, Toan Nguyen Duc. 2010. Empirically driven investigation of dependability and security issues in Internet-centric systems. Edmonton, Alta: University of Alberta. http://hdl.handle.net/10048/1112.
18. Roesch, Martin. "Snort-lightweight intrusion detection for networks." In Proceedings of the 13th USENIX conference on System administration, pp. 229-238. (1999).
19. Gómez, J., C. Gil, N. Padilla, R. Baños, and C. Jiménez. "Design of a snort-based hybrid intrusion detection system." In Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living, pp. 515-522. Springer Berlin Heidelberg, (2009).
20. Algorithms in C: Fundamentals, Data Structures, Sorting, Searching, Robert Sedgewick, Addison-

Wesely Publishing Company, (1997).

21. Tenhunen, Thomas. Implementing an Intrusion Detection System in the MYSEA architecture. NAVAL POSTGRADUATE SCHOOL MONTEREY CA, (2008).
22. Paxson, Vern. "Bro: a system for detecting network intruders in real-time." Computer networks 31, no. 23 (1999): 2435-2463.
23. Dreger, Holger, Anja Feldmann, Michael Mai, Vern Paxson and Robin Sommer. "Dynamic application-layer protocol analysis for network intrusion detection." In USENIX Security Symposium, pp. 257-272. (2006).
24. Suricata http://suricata-ids.org/ (Last accessed 14 Feb 2022).
25. Rødfoss, Jonas Taftø. "Comparison of open source network intrusion detection systems." (2011).
26. Albin, Eugene. "A comparative analysis of the snort and suricata intrusion-detection systems." PhD diss., Monterey, California. Naval Postgraduate School, (2011).
27. Day, David, and Benjamin Burns. "A performance analysis of snort and suricata network intrusion detection and prevention engines." In ICDS 2011, The Fifth International Conference on Digital Society, pp. 187-192. (2011).
28. Park, Yong Joon, and Jae Chul Park. "Web Application Intrusion Detection System for Input Validation Attack." In Convergence and Hybrid Information Technology, 2008. ICCIT'08. Third International Conference on, vol. 2, pp. 498-504. IEEE, (2008).
29. OWASP - Web Application Firewalls https://www.owasp.org/index.php/Web_Application_Firewall (Last accessed 10 March 2022).
30. WebKnight http://www.aqtronix.com/?Page ID=99 (Last accessed 14 March 2022).
31. NXSI https://www.owasp.org/index.php/OWASP_NAXSI_Project (Last accessed 14 March 2022).
32. Barracuda Web Application Firewall https://www.barracuda.com/products/webapplicationfirewall/features (Last accessed 9 March 2022).
33. https://www.juniper.net/us/en/research-topics/what-is-ids-ips.html (Last access 10 March 2022)
34. Tariq Bin Azad, in Securing Citrix Presentation Server in the Enterprise, (2008).