



Real-time Hypertext Transfer Protocol Intrusion Detection System on Web Server using Firebase Cloud Messaging

Agus Tedyyana¹^a and Osman Ghazali²^b

¹Department of Informatic Engineering, Politeknik Negeri Bengkalis, Jl. Bathin Alam, Bengkalis, Indonesia

²School of Computing, University Utara Malaysia, Kedah, Malaysia

Keywords: HTTP IDS, Cyber Attack, Telegram Bot, OWASP ZAP, Firebase Cloud Messaging.


Abstract: Behind the rapid development of the internet in the current era, various types of crime also target vital players in the internet industry. The types of attacks on the internet, for example, are website attacks and data theft on servers. With the many types of crimes rampant in the internet world, an antidote is also needed to suppress internet crime, and users can use it safely. Therefore, the researcher proposes a solution in the form of Keris, namely HTTP Intrusion Detection System, that runs on the server where our website is running. Keris works by detecting rampant intrusions attacking servers or websites utilizing resources from outside parties. When the intrusion is detected, the Keris will notify the system admin using the Keris Telegram chatbot. From the results of tests carried out using the OWASP ZAP test tool, Keris can detect intrusions in the form of directory brute force then send notifications to the system admin with a delay time of 0.72 seconds.


1 INTRODUCTION

With the rapid growth and success of the Internet at this time, the Internet has now become an important aspect in almost every field and has significantly brought changes to the basic aspects. As time goes on, it is not just a tool for researchers or communicators. Governments use the Internet in e-governance and in some ways to provide information to the citizens of a country or the world in general and this trend is running smoothly. Governments continue to use the Internet to provide better and more transparent governance. Many Companies also use the Internet for various needs such as exchanging information with business units, suppliers, customers, and partners in a very efficient and fast manner.. Meanwhile, research and educational institutions also use the internet as a tool in problem-solving and as a platform to collaborate and disseminate their findings and research around the world. Now with such adoption, it is becoming clear that many elements are increasingly dependent on the Internet and network-based information systems(Mir et al., 2018).

There are various types of attacks that are used as tools to commit cybercrime. In the web world, it is at least divided into several categories, for example:

1. Common Web Attack, is a criminal attack that is generally done against web-based applications. The most common are Cross-Site Scripting (XSS)(Rathore et al., 2017), SQL Injection (SQLi) (Pham & Subburaj, 2020), Path/Directory Traversal (Potinteu & Varga, 2020), Local File Inclusion (LFI) (Abdur Rahman et al., 2020), and Distributed Denial of Service (DDoS) (Haider et al., 2020).
2. Common Vulnerabilities and Exposures (CVE), is a system that provides reference methods for commonly known information security vulnerabilities and exposures in the form of data sets. This system can be used as a reference to protecting existing systems, but can also be a reference for attackers to attack systems (Fan et al., 2020).
3. Bad Crawler, is a bot used for criminal acts, such as illegally collecting (memo/crawler) website content or taking email addresses listed on

^a <https://orcid.org/0000-0001-6872-7490>

^b <https://orcid.org/0000-0003-4637-1117>

websites with the intention of sending spam to victims. In addition, there are also types of bots that are used to perform DDOS actions, which use up bandwidth and hurt website owners.

4. Directory Brute Force, is a frequently used attack on websites and web servers. Brute Force directories are used to find hidden and often overlooked directories on websites to find security holes on websites.

Identifying unauthorized access from a computer network system is known as intrusion detection. The reason behind these activities is security threats and dangerous consequences from illegal access, which can be triggered by intruders either from internal or external parties who attempt to steal valuable information from the computer system without the owner's permission (Zarpelão et al., 2017). Intrusion detection systems (IDS) are new security mechanisms that use sophisticated techniques to secure computer networks from ongoing intrusions or illegal access that have already occurred. In addition, cybersecurity work requires a robust IDS design, which can withstand large scale of hacking and ethical testing in a real-time (Sohal et al., 2018).

IDS consists of various categories, namely Network-based IDS (NIDS), Wireless IDS, and Host-based IDS (HIDS). There is also a Hybrid IDS which combines various IDS categories into one. Host-based IDS performs single host activity and if any malicious activity occurs on the server. HIDS policy in the process and policy policies system files, system logs, and registry keys from outside network intrusion. A host-based intrusion detection system that runs on the system which includes techniques to analyze and analyze information on the system (Jose et al., 2018).

In intrusion detection, web-based applications typically use weblogs from a web server to detect intrusion. A weblog file is usually contain as a raw text file that save information each time a user of system clicks a link URL from a website, which provides important and meaningful information to the web administrator, usually contains data about the IP address, timestamp (time and date), bytes that transferred, HTTP status codes, access requests, and user agents, etc. In the research of Tanaka et al., they created a model to detect bots that enter the website by utilizing the User-Agent and User Behavior recorded in the webserver log (Tanaka et al., 2020). Log is very useful that the system administrator can make a recover or at least find out about the cause of the web server failure to running normally. Also,

weblogs may be used as an additional measure against common hacker attacks such as cross-site scripting (XSS), denial of service (DOS), session hijacking, SQL injection and etc. By analyzing of these log files, we were able to get information about some potential web attack patterns that may be happen (Ma et al., 2017). Web servers usually produce number of excellent logs, between 1KB to 100 MB or even larger, that cannot be analyzed by our self manually. Weblogs are stored using human-oriented text format, sometimes containing noisy and unnecessary data that can affect the quality of the intrusion detection process and thus need to process raw log files.

There are some big challenges in building an IDS that runs on a web server. Intrusion detection methodologies are still immature in the domain of web application security. Tracking systems are mainly used as network security tools. However, IDS web design requires a different approach from traditional network IDS to address the complexities associated with web-based applications (Agarwal & Hussain, 2018).

In this research, the researcher propose an IDS in the form of HTTP IDS, namely Keris. Keris is an IDS that detecting real-time intrusion detection tool and web log-based alerts that run on the terminal with resources collected. By analyzing the ongoing weblog, it will alert the admin in real-time if any suspicious activity is recorded in the weblog. The hope of the researchers in this study is to be able to get the results of intrusion notifications that can be sent in real-time with the shortest available delay time. This IDS was built using the Go programming language to build something that works natively without using a lot of RAM (Dymora & Paszkiewicz, 2020). In this research, Keris still uses external resources to detect intrusions into the system. For example, for types of attacks such as Common Web Attack using PHP-IDS (Chora & Kozik, 2017) resources, CVE uses Nuclei Templates (ProjectDiscovery, n.d.). Bad IP Address & Bad Referrer is obtained from the Nginx Ultimate Bad Bot Blocker (Krog, 2017) collection, and Bad Crawler takes from project resources from Crawler Detect (Bizzle, n.d.). The last one is Directory Brute force leveraging resources from Dirsearch (Soria, 2014).

2 RESEARCH METHOD

As explained in the previous section, HTTP-IDS processes the logs to detect intrusions on the system. This section will explain how logs on the webserver

are utilized in the intrusion process, as well as how the Keris system works and how the Keris system utilizes external resources used to help detect whether an intrusion occurs or not.

2.1 Proposed Mechanism Design

The output obtained from this stage is a design of how the IDS Keris will work where the data is obtained from the results of analysis of literature studies. As shown in Figure 1, the following flowchart will explain how the intrusion process is detected and reported to the system admin.

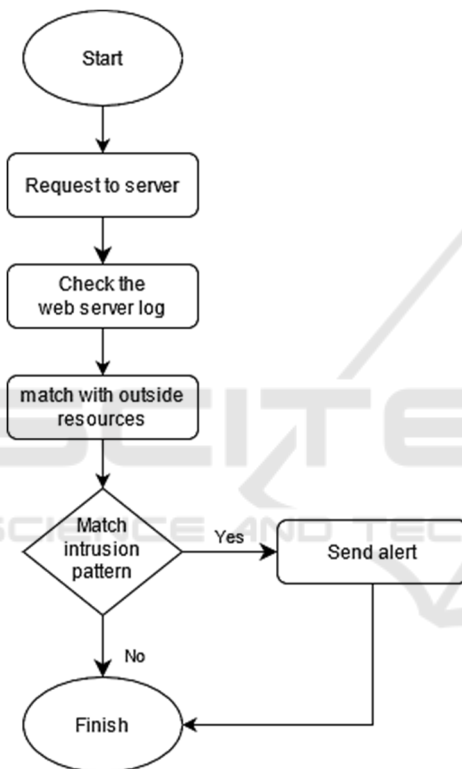


Figure 1: Flowchart of the Keris HTP IDS.

The flow chart illustrates how the Keris HTP IDS workflow developed in this study. The request from the user will go to the webserver that is used as the hosting of the web application. When the HTPP request comes in, information on the activity carried out by the client will be recorded in the webserver log. The logs will then be matched with external resources that are used to check whether the incoming request was classified as an intrusion or not. If an incoming HTPP request is declared suspicious by IDS based on these resources, the system will send a warning notification to the system admin about an intrusion attempt to the webserver.

2.2 Use of Web Server Logs on the Intrusion Process

Web servers include several parts that control how web users access hosted files. The minimum is an HTTP server. An HTTP server understands URLs (web addresses) and HTTP (protocols used by your browser to view web pages) (Suchacka et al., 2021). The HTTP server can be accessed through the domain name of the hosted website and passes the content of this hosted website to the end user's device.

A web server log file written as an activity generated by a web server. Log files collect various data about information requests to a web server. Some examples of data collected and stored are Date, Time, Customer IP Address, Referrer, User Agent, Service Name, Server Name, Server IP, etc. The log on the web server acts as a visitor login sheet. Logs can answer questions such as: Who is browsing the website? What browser is used? Where do visitors browse this site? What pages do visitors see?

2.3 Rule Method Used in the HTTP-IDS Keris

As explained in the previous section, the system script to be created runs by analyzing the webserver log files. The main reason for this selection is as an opportunity for a detailed analysis of the actions of the user on the system. By examining past data, information security policies for web applications can be created and implemented properly (Baş Seyyar et al., 2018). In addition, further exploitation can be prevented in advance. However, working with logs has some drawbacks. Because the log file does not contain all HTTP request and response data, some critical data cannot be analyzed. For example, POST parameters that are vulnerable to injection attacks cannot be recorded by the webserver. Other negative aspects are log size and parsing difficulty.

The proposed method can be described as a rule-based system. Unlike detection based on anomalies, the rules are static, i.e. using a blacklist and whitelist approach. A rule-based system is detection based on a database of known intrusion or attack alerts crafted by humans based on their expert knowledge (Sadikin et al., 2020). The database used here is an outside resource that has been explained in the previous section. If IDS detects an intrusion, then according to the data from the database, immediate detection is categorized as an intrusion.

2.4 Intrusion Warning Mechanism

There are various ways to warn users if an intrusion is detected in the IDS. Starting from SMS Gateway, e-Mail, or chatbot. Chatbots can be developed in any programming language. The backend receives the message, figures out what to answer, and returns a response to the user using a web API from the chatbot service provider. An API (application programming interface) is a set of classes, procedures, functions, structures, and constants provided by an application, library, service, or operating system for use in external software. In this case, the API is provided to developers in the public domain (Kozhevnikov et al., 2017).

The use of chatbots as an intrusion warning medium is now widely used as its implementation is easy as it is spoiled with documentation from existing developers and communities. For example, in a Telegram boat, there are two ways to run a Telegram Boat, Long Polling and Webhook(Candra et al., 2020).

The advantage of using the Webhook method is that data exchange is done in real-time and the admin only needs to configure the callback URL, then the sending server will send data to the configured URL until there is an update. And the weakness of the Webhook method is the need for a server that has an HTTPS connection or uses hosting if it doesn't already exist. The results show that the Long Polling method has a higher response time or in this case slower than the Webhook method.

In addition to the Telegram chatbot, there is another alternative as a delivery mechanism for intrusion detection, namely using Firebase Cloud Messaging (FCM) technology. FCM is a cross-platform solution for messages and notifications for Android, iOS, and web applications. The FCM architecture is presented in Figure 2. FCM supports notification and data messages (Albertengo et al., 2019). Notification messages are automatically handled by the FCM SDK to show a notification on behalf of the client application. They contain a predefined set of user-visible keys (and an optional data payload of custom key-value pairs). The fundamental components in the FCM architecture are FCM connection server, Trusted environment with HTTP and Xtensible Messaging and Presence Protocol (XMPP), and Client application.

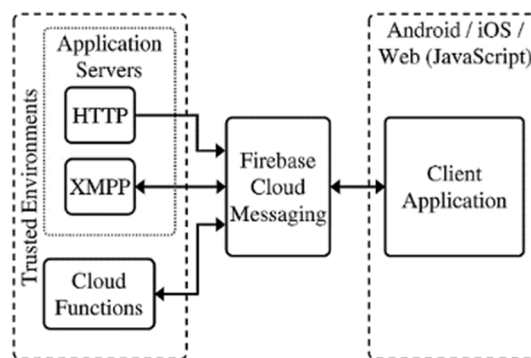


Figure 2: Flowchart of the Keris HTP IDS.

3 RESULT AND ANALYSIS

This section describes the results of the research carried out along with the results of the application test analysis which is divided into 3 sub-sections.

3.1 Keris

The result of the Keris application is an HTTP-IDS built using the Go programming language. Keris is installed on a server which is the base of the website. Keris is an application built without a GUI, so to run it, you have to use commands in the terminal. Keris can handle websites that are deployed on Apache and Nginx web servers. Figure 3 below is a display of how Keris runs and reads the logs on the webserver to determine whether there is an intrusion or not.

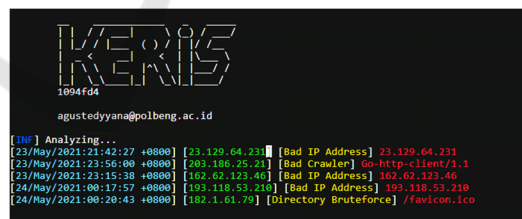


Figure 3: Keris running on Nginx Webserver.

Figure 3 above explains how the Keris looks when it is run through the terminal. When you run it for the first time, Keris will present you with a logo of the application and the contact you can address. Then below shows the analysis process done by Keris by reading the logs from the webserver. The results of the analysis data look like the format in the figure, namely the date and time of the log are written in this case the time the intrusion occurred along with the time zone. Then there is the IP address of the client who intruded. Next is the type of intrusion carried out by the client. And lastly is a description of the

information about the intrusion. For example, the Bad IP Address will certainly display the Bad IP Address, and the Bad Crawler will display the type of crawler used for the intrusion.

3.2 Keris Telegram Bot

Not always web admins or system admins work in front of their computers to monitor intrusions from outside. Therefore, we also developed a Telegram bot to meet the needs of monitoring for 24 hours with its real-time notification feature. First, we provide a template to present information that is easy to understand for the web admin. Then enter the token from the telegram bot that Telegram bot created and the chat id from our account into the existing configuration file. Figure 4 below shows how the information displayed by Keris Telegram Bot.

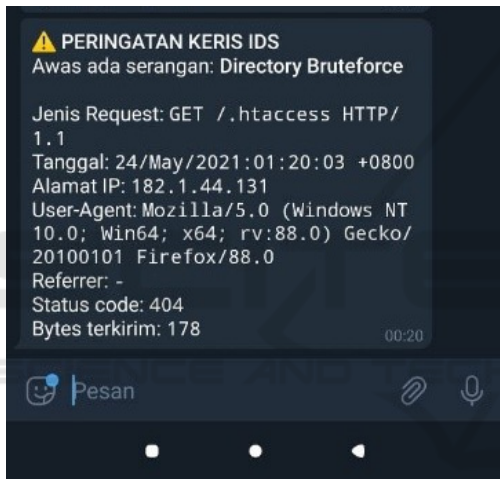


Figure 4: Keris Telegram Bot.

This image is an illustration of how the bot will notify the web admin through informative notifications. The information sent includes the type of incoming intrusion, the type of HTTP request, the attacker's IP address, the User-Agent used, the HTTP status response code from server to client, and bytes sent from server to client.

3.3 Keris Client App

In addition to using the Telegram chatbot, researchers also used FCM to examine which delay level was lower between FCM and Telegram webhook. As a notification recipient from FCM, the researcher built an Android application that is useful as a client to display the contents of intrusion detection messages as shown in Figure 5.

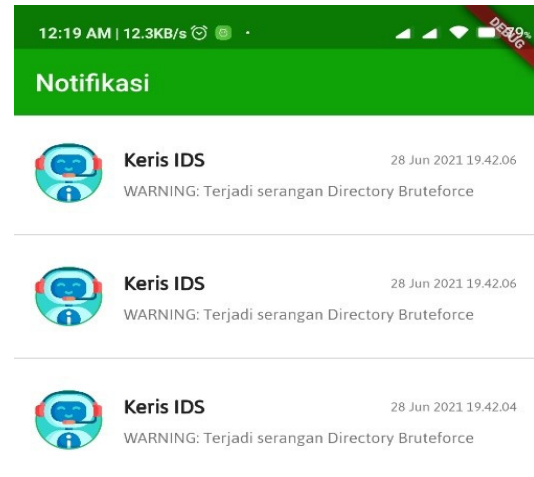


Figure 5: Keris Client App.

The picture above is a display of the application that displays basic information such as the type of intrusion and the time of the intrusion down to the seconds format.

3.4 Performance Metric

In order to analyze the real time performance of the HTTP IDS used, the researcher uses the One-Way Delay (OWD) metric for the two types of intrusion sending mechanisms used, namely Telegram Webhook and FCM. OWD based on the time taken for a packet to be transmitted across a network from source which mean the HTTP IDS to destination in this case is client. In testing the application in this research, the researcher used the OWASP ZAP tool to launch attacks on the website. the researcher chose OWASP ZAP because it has proven reliable in intrusion testing (Kritikos et al., 2019). From the results of the timestamp on log earlier, the researcher then compared the timestamp between the time the attack was detected and the time System Admin received the alert on telegram to get the results on whether Keris was able to generate real-time alerts when attacking using the OWASP ZAP tool as in table 1 below.

Table 1: Results of OWD Using Telegram Webhook.

No	Attack Type	Time of Detect	Time of Alert	Time Difference(s)
1	Directory Brute force	00:24:55	00:24:55	0
2	Directory Brute force	00:24:55	00:24:55	0

Table 1: Results of OWD Using Telegram Webhook (cont.).

No	Attack Type	Time of Detect	Time of Alert	Time Difference(s)
3	Directory Brute force	00:25:03	00:25:04	1
4	Directory Brute force	00:25:11	00:25:11	0
5	Directory Brute force	00:25:11	00:25:12	1
6	Directory Brute force	00:25:12	00:25:12	0
7	Directory Brute force	00:25:12	00:25:12	0
8	Directory Brute force	00:25:12	00:25:13	1
9	Directory Brute force	00:25:14	00:25:15	1
10	Directory Brute force	00:25:14	00:25:15	1
11	Directory Brute force	00:40:35	00:40:36	1
12	Directory Brute force	00:40:42	00:40:43	1
13	Directory Brute force	00:40:42	00:40:43	1
14	Directory Brute force	00:40:43	00:40:44	1
15	Directory Brute force	00:40:43	00:40:44	1
16	Directory Brute force	00:40:43	00:40:44	1
17	Directory Brute force	00:40:45	00:40:46	1
18	Directory Brute force	00:40:45	00:40:46	1
The difference in time average				0.72

Based on the results of the table above, it is found that of the last 18 of attacks detected using OWASP ZAP, there is an average time difference of 0.72 seconds. This means a delay of about 0.72 until 1

seconds to send a warning to the telegram since Keris detected the attack. After testing on the Telegram Webhook, the next researcher investigated using FCM. The results of OWD data on FCM are shown in table 2 below. Which shows that the delay rate on FCM is lower than on Telegram Webhook. Sama amount of data with the Telegram Hook, the FCM have an average time difference of 0.44 seconds.

Table 2: Results of OWD Using FCM.

No	Attack Type	Time of Detect	Time of Alert	Time Difference(s)
1	Directory Brute force	02:16:24	02:16:24	0
2	Directory Brute force	02:17:31	02:17:32	1
3	Directory Brute force	02:17:31	02:17:32	1
4	Directory Brute force	02:17:32	02:17:32	0
5	Directory Brute force	02:17:32	02:17:32	0
6	Directory Brute force	02:18:53	02:18:54	1
7	Bad IP Address	02:20:55	02:20:56	1
8	Directory Brute force	02:24:43	02:24:43	0
9	Directory Brute force	02:24:43	02:24:43	0
10	Directory Brute force	02:24:43	02:24:43	0
11	Directory Brute force	02:24:43	02:24:43	0
12	Directory Brute force	02:26:40	02:26:41	1
13	Directory Brute force	02:26:41	02:26:41	0
14	Directory Brute force	02:26:41	02:26:41	0
15	Directory Brute force	02:26:41	02:26:41	0
16	Directory Brute force	02:28:15	02:28:16	1
17	Directory Brute force	02:28:15	02:28:16	1
18	Directory Brute force	02:28:15	02:28:16	1
The difference in time average				0.44

4 CONCLUSIONS

From the results and analysis of the application testing that has been carried out, it is found that the proposed HTTP-IDS Keris has successfully detected

intrusions carried out by OWASP ZAP by utilizing external resources to detect the types of incoming attacks. From the results of the OWASP ZAP test, the type of intrusion was Directory Brute force. Keris can also alert the system admin about the dangers of intrusion in real-time with a delay of 0.72 seconds using the Telegram Webhook and 0.44 seconds when using the FCM. This shows that FCM is suitable more than Telegram Webhook when we want to use real-time notification.

For future research, it is expected to find out about the other technology as the alerting mechanism to resolve causes of the delay and then reduce the delay time. Besides that, research can also be carried out by utilizing the latest knowledge, such as machine learning, to maximize Keris abilities.

REFERENCES

- Abdur Rahman, M., Amjad, M., Ahmed, B., & Saeed Siddik, M. (2020). Analyzing web application vulnerabilities: An empirical study on e-commerce sector in Bangladesh. *ACM International Conference Proceeding Series*, 5–10. <https://doi.org/10.1145/3377049.3377107>
- Agarwal, N., & Hussain, S. Z. (2018). A Closer Look at Intrusion Detection System for Web Applications. *Hindawi Security and Communication Networks*, 2018, 1–27.
- Albertengo, G., Debele, F. G., Hassan, W., & Stramandino, D. (2019). On the performance of web services, google cloud messaging and firebase cloud messaging. *Digital Communications and Networks*, 6(1), 31–37. <https://doi.org/10.1016/j.dcan.2019.02.002>
- Baş Seyyar, M., Çatak, F. Ö., & Gül, E. (2018). Detection of attack-targeted scans from the Apache HTTP Server access logs. In *Applied Computing and Informatics* (Vol. 14, Issue 1, pp. 28–36). Elsevier B.V. <https://doi.org/10.1016/j.aci.2017.04.002>
- Bizzle, J. (n.d.). *Crawler Detect*. Retrieved May 6, 2021, from <https://github.com/JayBizzle/Crawler-Detect>
- Candra, H., Rino, & Riki. (2020). Designing a Chatbot Application for Student Information Centers on Telegram Messenger Using Fulltext Search Boolean Mode. *Jurnal Bit-Tech*, 2(2), 10–18.
- Chora, M., & Kozik, R. (2017). Machine Learning Techniques for Threat Modeling and Detection. *Security and Resilience in Intelligent Data-Centric Systems and Communication Networks*, 179–192. <https://doi.org/10.1016/B978-0-12-811373-8.00008-2>
- Dymora, P., & Paszkiewicz, A. (2020). Performance analysis of selected programming languages in the context of supporting decision-making processes for industry 4.0. *Applied Sciences (Switzerland)*, 10(23), 1–17. <https://doi.org/10.3390/app10238521>
- Fan, J., Li, Y., Wang, S., & Nguyen, T. N. (2020). A C/C++ Code Vulnerability Dataset with Code Changes and CVE Summaries. *Proceedings - 2020 IEEE/ACM 17th International Conference on Mining Software Repositories, MSR 2020*, 508–512. <https://doi.org/10.1145/3379597.3387501>
- Haider, S., Akhunzada, A., Mustafa, I., Patel, T. B., Fernandez, A., Choo, K. K. R., & Iqbal, J. (2020). A Deep CNN Ensemble Framework for Efficient DDoS Attack Detection in Software Defined Networks. *IEEE Access*, 8, 53972–53983. <https://doi.org/10.1109/ACCESS.2020.2976908>
- Jose, S., Malathi, D., Reddy, B., & Jayaseeli, D. (2018). A Survey on Anomaly Based Host Intrusion Detection System. *Journal of Physics: Conference Series*, 1000(1). <https://doi.org/10.1088/1742-6596/1000/1/012049>
- Kozhevnikov, V. A., Sabinin, O. Y., & Shats, J. E. (2017). Library Development For Creating Bots On Slack, Telegram And Facebook Messengers. *International Scientific Journal Theoretical & Applied Science P-ISSN*: 50(06), 59–62. <https://doi.org/10.15863/TAS>
- Kritikos, K., Magoutis, K., Papoutsakis, M., & Ioannidis, S. (2019). A survey on vulnerability assessment tools and databases for cloud-based web applications. *Array*, 3–4(October), 100011. <https://doi.org/10.1016/j.array.2019.100011>
- Krog, M. (2017). *Nginx Ultimate Bad Bot and User-Agent Blocker*. <https://github.com/mitchellkroga/nginx-ultimate-bad-bot-blocker>
- Ma, K., Jiang, R., Mianxiong, D., Jia, Y., & Li, A. (2017). Neural Network Based Web Log Analysis for Web Intrusion Detection. *Springer International Publishing*, 194–204. <https://doi.org/10.1007/978-3-319-72395-2>
- Mir, S. Q., Mir, I. A., & Beigh, B. M. (2018). Investigating the denial of service attack : A major threat to internet and the security of information. *JK Research Journal in Mathematics and Computer Sciences Investigating*, 1(1), 121–131.
- Pham, B. A., & Subburaj, V. H. (2020). *Experimental setup for Detecting SQLI Attacks using Machine Learning Algorithms*. 8(1), 79016.
- Potinteu, I. C., & Varga, R. (2020). Detecting Injection Attacks using Long Short Term Memory. *Proceedings - 2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing, ICCP 2020*, 163–169. <https://doi.org/10.1109/ICCP51029.2020.9266177>
- ProjectDiscovery. (n.d.). *Nuclei - Community Powered Vulnerability Scanner*. Retrieved May 5, 2021, from <https://nuclei.projectdiscovery.io/templating-guide/>
- Rathore, S., Sharma, P. K., & Park, J. H. (2017). XSSClassifier: An efficient XSS attack detection approach based on machine learning classifier on SNSs. *Journal of Information Processing Systems*, 13(4), 1014–1028. <https://doi.org/10.3745/JIPS.03.0079>
- Sadikin, F., Deursen, T. van, & Kumar, S. (2020). A ZigBee Intrusion Detection System for IoT using Secure and Efficient Data Collection. *Internet of Things*, 12, 100306. <https://doi.org/10.1016/j.iot.2020.100306>
- Sohal, A. S., Sandhu, R., Sood, S. K., & Chang, V. (2018). A cybersecurity framework to identify malicious edge

- device in fog computing and cloud-of-things environments. *Computers and Security*, 74, 340–354. <https://doi.org/10.1016/j.cose.2017.08.016>
- Soria, M. (2014). *Dirsearch*. <https://github.com/maurosoria/dirsearch>
- Suchacka, G., Cabri, A., Rovetta, S., & Masulli, F. (2021). Knowledge-Based Systems Efficient on-the-fly Web bot detection. *Knowledge-Based Systems*, 223, 107074. <https://doi.org/10.1016/j.knosys.2021.107074>
- Tanaka, T., Niibori, H., Li, S., Nomura, S., Kawashima, H., & Tsuda, K. (2020). Bot detection model using user agent and user behavior for web log analysis. *Procedia Computer Science*, 176, 1621–1625. <https://doi.org/10.1016/j.procs.2020.09.185>
- Zarpelão, B. B., Miani, R. S., Kawakani, C. T., & de Alvarenga, S. C. (2017). A survey of intrusion detection in Internet of Things. *Journal of Network and Computer Applications*, 84(February), 25–37. <https://doi.org/10.1016/j.jnca.2017.02.009>

