

Apache web server security with security hardening

Dega Surono Wibowo¹, Hepatika Zidny Ilmadina², Ardi Susanto^{3*}, Fariq Fadillah Gusti Insani⁴
^{1,2,3,4}Departement of Informatics, Politeknik Harapan Bersama, Indonesia

Article Info

Article history:

Received October 24, 2023
Revised November 10, 2023
Accepted November 13, 2023

Keywords:

Apache web server
Intrusion detection system
Security hardening

ABSTRACT

With the internet network, we can get information very quickly. The information we obtain is not changed by people not authorized to access the system or platform. Apache is a web server often used to connect users to the website where the information is located. The more users, the more crimes will occur when irresponsible people attack the web server. Due to limited time for web administrators, to improve security on the Apache web server, an intrusion detection system is needed to help monitor network traffic, detect the type of attack that occurs, and then forward the notification to the mobile application in real-time. Time, because an attack can occur at any time. This research aims to create an intrusion detection system on a server that uses Apache as its web server and can notify the server manager if the server is indicated to have been hacked. This application is an application of the security hardening method. The results of this research are that the system will detect intrusion attempts based on the rules created, and users will receive notifications on the Telegram application and can see details of incoming reports such as I.P. address, intrusion description, security hole name, time of intrusion, and payload used.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Ardi Susanto,
Informatics, Politeknik Harapan Bersama,
Tegal, Central Java, Indonesia.
Email: ardisusanto@poltektegal.ac.id
<https://doi.org/10.52465/joscecx.v4i4.230>

1. INTRODUCTION

With the internet network, we can get information very quickly. It is hoped that the information obtained will not be modified by people who do not have the right to access the system or platform—other forms of threats such as eavesdropping, theft of confidential data, and malware attacks. Therefore, web server security is essential, considering the rapid development of information technology. However, the limited ability of the Web Administrator to monitor the server is one of the obstacles; additional systems are needed to secure the web server. One way to secure a web server is to install an intrusion detection system to detect intrusion attempts on the web server in real-time. This system is known as the Intrusion Detection System (I.D.S.).

Computer Network Security is the most critical process because it can maintain the integrity and validity of data and guarantee service availability for its users [1]. Information provided and accessible via the network is very vulnerable and easy to infiltrate; intruders can easily change or take the information provided by the system [2]. Because of this, information security on a website is critical, especially for a website creator. Website creators must also know the initial stages of a hacker who wants to take over a website [3]. The initial

stage of a hacker is information gathering. Information Gathering is a comprehensive stage in identifying a target. The target can include information on the operating system, network topology, I.P. address used, open ports, and can find out the DNS used. This stage can also determine the ownership of a website. In the assessment process, vital security gaps can be found at the initial stage, namely Information Gathering. This is because there was an error in the system development process called Misconfiguration [4].

The N.I.S.T. Cybersecurity Framework is a best practice for a cyber resilience framework. The framework is divided into five main parts: the first is identification, the second is protection, the third is detection, the fourth is response, and the fifth is recovery [5]. These five main sections offer a holistic perspective on the timeline for mitigating cybersecurity risks. Implementing N.I.S.T. cyber security gives website managers the authority to identify and manage cyber security risks by assigning a value to each threat [6]. N.I.S.T. Cybersecurity also contains security hardening.

Security Hardening is a method for minimizing security gaps in the system. This method can be used in a variety of available systems. In carrying out security hardening, some stages must be passed; the first stage is Access; access is the stage for identifying security gaps in a system. The next stage is analysis, which analyzes the impact caused by existing security gaps. Next is the remediate stage; this stage is where and how the solution is to close existing security gaps. The final stage of security hardening is management; management is the stage to close the security gaps that have been determined [7].

Intrusion detection is the process of monitoring and detecting intrusions on a network or computer system by matching network traffic patterns with known attack patterns (misuse) or by looking for abnormal network traffic patterns (anomalies) [8]. Whenever an intrusion attempt is detected on a network, a notification will be sent to the administrator or collected as a database.

A web server is software that responds to requests from browsers to connect the server to the user. In a survey conducted by a cybercrime protection service provider, Netcraft, one of the web servers widely used is Apache [9]. Apache is a free and open-source web server. The more users, the more crimes will appear to attack web servers by irresponsible people. A vulnerability recently discovered in the Apache Web Server is being widely exploited, called Path Traversal and Remote Code Execution, which allows attackers to view root files only via the URL [10].

Based on the background that has been explained, an intrusion detection application is needed that can help monitor network traffic on the Apache web server, detect the type of attack that occurs, and then forward the notification to a mobile application in the form of a Telegram application in real time, because the use of a mobile application can be used anytime and anywhere which allows users to respond more quickly to notifications.

The results of the rules formulated by Suricata have worked well and can detect interference such as Port Scanning and Brute Force [11]. This helps the network admin take preventive action against Nmap attacks [12]. Then, a survey and research focused on DDoS attacks entitled Detection of HTTP Flooding DDoS Attack using Hadoop with MapReduce, and it was decided to prevent it using the latest Hadoop techniques with MapReduce [13]. Demonstrate and analyze the direct and indirect impact of the slow HTTP DOS and DDOS on the virtual environment. We will launch the Slowloris DOS and DDOS on V.M.s and analyze the direct impact of the attack on the target V.M. and the indirect impact of the attack on the neighbor V.M. [14].

Intrusion Detection System, or what is usually called I.D.S., is a system that monitors network traffic for suspicious activity and provides warnings when such activity is discovered. To solve this problem, in this Final Project, an I.D.S. process was analyzed using KDD99 as a dataset using a genetic algorithm as feature selection, and the K.N.N. algorithm as classification and evaluation. Based on the tests carried out, feature selection using a genetic algorithm obtained 18 of the best features to be used from 41 features, with an average accuracy of 84.17%, and classification using the K.N.N. algorithm with an accuracy of 99.98% on training data, testing data with an average of average 97.52% and average manual calculation with parameters $k=1$ 78.57%, $k=3$ 76.40%, $k=5$ 76.86%, $k=7$ 76.71%, $k=9$ 77.57 % [15].

In this research, a network security system was tested using the I.D.S. Snort application with two Snort placement positions on the local network of Laboratory VI Network Campus 3 IST AKPRIND Yogyakarta; results were obtained by testing Port Scanning, S.S.H., and DoS attacks. It was concluded that inner Snort placement was better than outer Snort placement [16].

Research on Network Intrusion Detection System Notification Using Telegram Application Media (Case Study: Tasikmalaya Immigration Office), which was built using the Network Development Life Cycle (N.D.L.C.) method, Snort as an I.D.S. sensor with a MySQL database, Acid base as a web front-end to manage alerting data which Snort detects, then uses the Telegram Bot API account as a notification medium to the

administrator. By implementing Telegram as a notification medium for this Intrusion Detection System, it is hoped that administrators will be able to find out whether or not there is suspicious activity that threatens network security so that administrators can quickly restore the network system [17].

Several classification methods were carried out on I.D.S. (Intrusion Detection System) data; the methods used were Support Vector Machine (SVM), Genetic Algorithm (G.A.) [18], K-Nearest Neighbor (K.N.N.) [19], Artificial Neural Network (A.N.N), Bayesian Method, Decision Tree, and Fuzzy Logic [20]. Slow testing time and accuracy are still challenges for research on classification methods in I.D.S. This I.D.S. has been researched extensively, especially the integration with the Fuzzy-genetic Algorithm; J. Gómez and E. León proposed a Fuzzy-genetic Algorithm to categorize intrusion activities on the network [21]. I.D.S. uses the KDD data processing process, where the algorithm is applied to the feature selection and model build process [22].

Hackers are currently not only attacking government agencies like in 2019 but have also carried out attacks on educational agencies. This is due to the monitoring and identification of the National Cyber and Crypto Agency where 38% of educational institutions were attacked in 2020. As a form of preventive action related to cyber attacks on educational institutions, it is necessary to carry out an information security analysis of the installed systems. In this article, we propose technical stages for conducting information security analysis using software with a Free Open Source Software license, namely Sudomy and OWASP ZAP. The results of the analysis of potential security gaps in the information system installed at Duta Bangsa University were obtained using these two software [23].

The vulnerabilities identified were vulnerability to DoS attacks, unencrypted communications, and outdated use of SSL/TLS. Penetration testing simulates SQL injection, DoS, Session hijacking, and Interception attacks. The penetration testing results found that the system was safe from SQL injection attacks because there was already a firewall to withstand these attacks. On the other hand, the system was not secure for other types of attacks. It needed reconfiguring on the webserver to minimize the security holes in the tap2go—Cloudflare website-based application [24].

Because there were weaknesses in previous research, especially in the less effective notification feature, this research was made, namely intrusion detection using the security hardening method, so that this software could send notifications to the Telegram application. This is necessary so web administrators can take preventive actions more quickly and identify each type of incoming attack. The reference referred to is O.W.A.S.P., an organization in the field of website application security.

2. METHOD

Conceptual Approach

Namely the set of relationships between certain factors that influence a particular situation. The conceptual approach is a description carried out in understanding, implementing, and evaluating research in information systems [25]. The method for implementing this activity can be shown in the flow diagram in Figure 1.

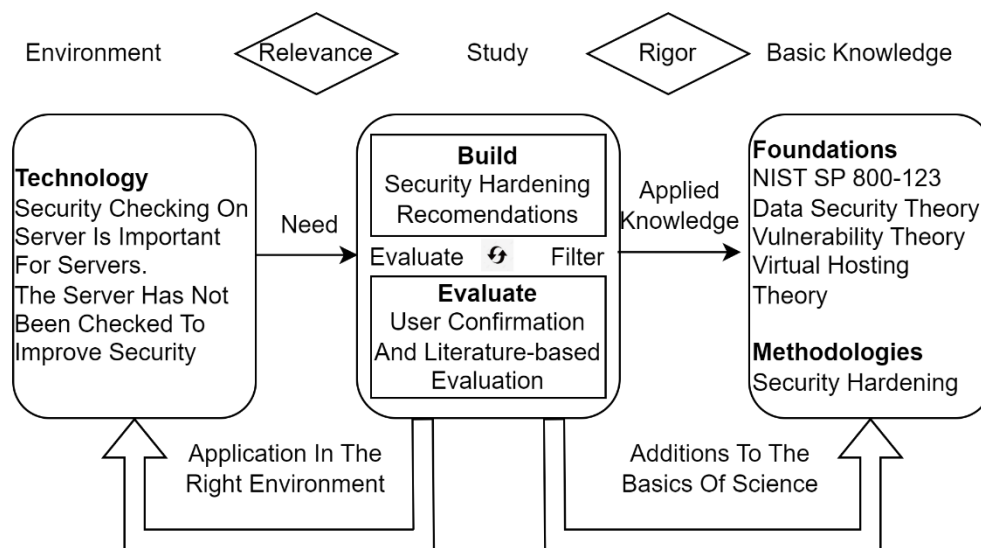


Figure 1. Conceptual approach

Systematic Approach

A systematic approach is a way to solve a problem; in other words, it is a planned process to achieve research objectives. The systematic solution carried out in this research is shown in the flow diagram in Figure 2.

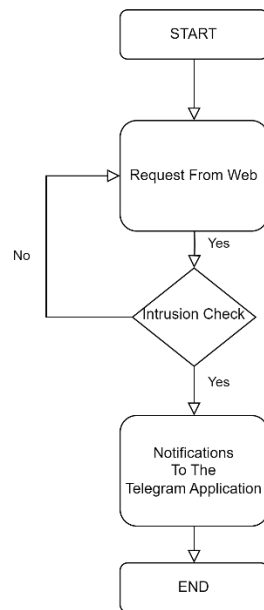


Figure 2. Application flowchart

Research Data

The data used in this research was taken from O.W.A.S.P. (Open Web Application Security Project) [26]. Data in the form of general vulnerabilities that often occur on a website and their identification, such as Cross-Site Request Forgery [27], Directory Transversal, Local File Inclusion [28], Remote File Execution, SQL Injection [29], Cross-Site Scripting [30]. The data used is vulnerability data that often occurs in creating websites, namely HTML codes that can cause bugs. For example data, the general rule is to HTML Attribute encode untrusted data (data from the database, HTTP request, user, back-end system, etc.) placed in an HTML Attribute. This is the appropriate step when outputting data in a rendering context; however, using HTML Attribute encoding in an execution context will break the application display of data.

SAFE but BROKEN example:

```

var ESAPI = require('node-esapi');
var x = document.createElement("input");
x.setAttribute("name", "company_name");
// In the following line of code, companyName represents untrusted user input
// The ESAPI.encoder().encodeForHTMLAttribute() is unnecessary and causes double-encoding
x.setAttribute("value",
'<%=ESAPI.encoder().encodeForJavascript(ESAPI.encoder().encodeForHTMLAttribute(companyName))%>');
var form1 = document.forms[0];
form1.appendChild(x);
  
```

The problem is if companyName had the value "Johnson & Johnson." What would be displayed in the input text field would be "Johnson & Johnson." The appropriate encoding to use in the above case would be only JavaScript encoding to disallow an attacker from closing out the single quotes and in-lining code or escaping to HTML and opening a new script tag.

SAFE and FUNCTIONALLY CORRECT example:

```

var ESAPI = require('node-esapi');
var x = document.createElement("input");
x.setAttribute("name", "company_name");
x.setAttribute("value", '<%=ESAPI.encoder().encodeForJavascript(companyName)%>');
  
```

```
var form1 = document.forms[0];
form1.appendChild(x);
```

It is important to note that when setting an HTML attribute that does not execute code, the value is set directly within the object attribute of the HTML element, so there are no concerns with injecting up.

3. RESULTS AND DISCUSSIONS

Intrusion Detection Rules

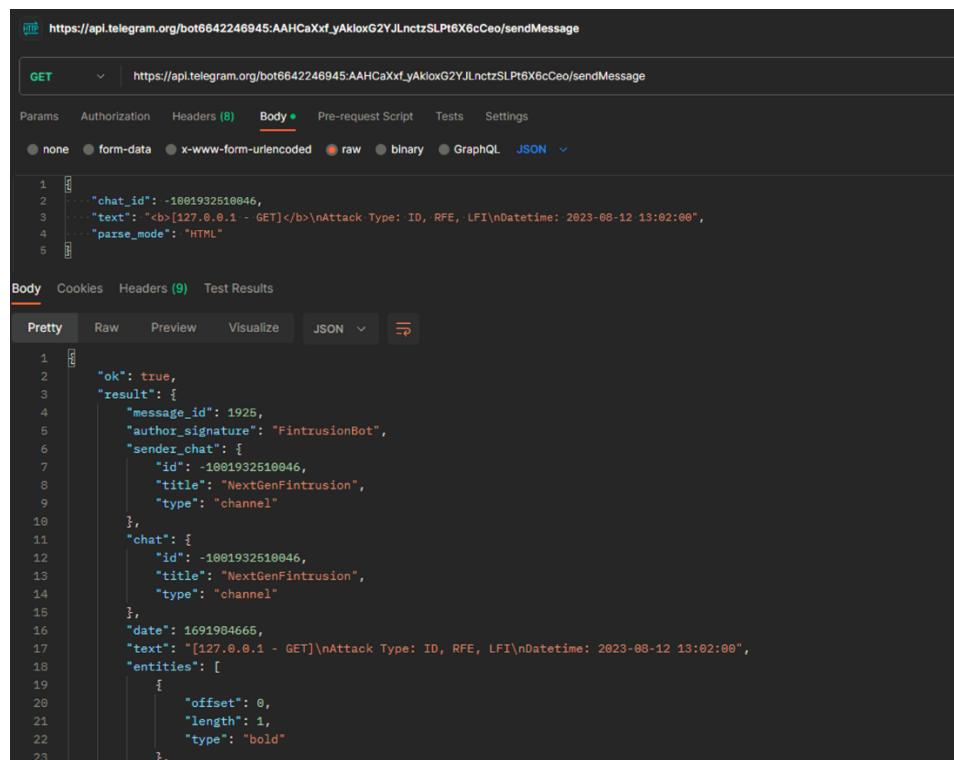
These intrusion detection rules are created as a JSON file and placed on the server, see Figure 3. For the rules structure, it contains I.D., rules, descriptions, and vulnerability tags. These rules use a string-matching method with the help of Regular Expressions; in other words, the system will detect by matching intrusions based on the rules that have been created.

```
{
  "id": "1",
  "rule": "(?:\\^[^\\"]*\\^>)(?:[\\^\\w\\s\\s*\\s/>])(?:>\\")",
  "description": "Finds html breaking injections including whitespace attacks",
  "tags": {
    "tag": [
      "xss",
      "csrf"
    ]
  }
}
```

Figure 3. Intrusion detection rules

Send Request API Telegram

Created to make Telegram API requests using the post method to send results if the intrusion detection rules detect an intrusion carried out on the server; it can be seen in Figure 4.



The screenshot shows a REST client interface with a GET request to the Telegram API endpoint. The request body is a JSON object with the following fields:

```
{
  "chat_id": -1001932510046,
  "text": "[127.0.0.1 - GET]<b>\nAttack Type: ID, RFE, LFI\nDatetime: 2023-08-12 13:02:00",
  "parse_mode": "HTML"
}
```

The response body is a JSON object with the following fields:

```
{
  "ok": true,
  "result": {
    "message_id": 1925,
    "author_signature": "FintrusionBot",
    "sender_chat": {
      "id": -1001932510046,
      "title": "NextGenFintrusion",
      "type": "channel"
    },
    "chat": {
      "id": -1001932510046,
      "title": "NextGenFintrusion",
      "type": "channel"
    },
    "date": 1691984665,
    "text": "[127.0.0.1 - GET]<b>\nAttack Type: ID, RFE, LFI\nDatetime: 2023-08-12 13:02:00",
    "entities": [
      {
        "offset": 0,
        "length": 1,
        "type": "bold"
      }
    ]
  }
}
```

Figure 4. Send request API telegram

Discussion

The resulting rule model has been tested by observing the software input and output results without knowing the system source code. From the results of the tests, the rule model can quickly detect attack attempts in the form of Cross-Site Scripting, SQL Injection, Local File Inclusion, and Directory Transversal. Sending notifications to Telegram is also fast.

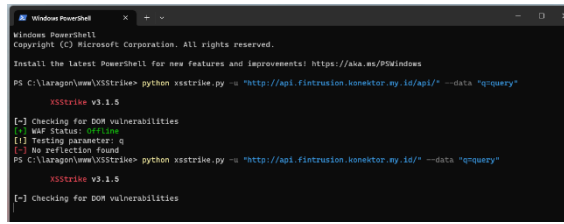


Figure 5. An attempted attack with XSSStrike

Testing is carried out to find out whether the system is functioning correctly or not, using an XSSStrike attack on a server that has intrusion detection installed; an example of an XSSStrike attack can be seen in Figure 5 can be seen in Table 1.

Table 1. Intrusion detection system testing

No	Tested Object	Testing Scenarios	Results
1.	Cross-Site Scripting Attack Detection.	We are carrying out attacks on websites deliberately using the Cross-Site Scripting method.	Detects attempted Cross-Site Scripting attacks.
2.	SQL Injection attack detection.	We are carrying out attacks on websites deliberately using the SQL Injection method.	I am detecting attempted SQL Injection attacks.
3.	Detect Local File Inclusion attacks.	We are intentionally attacking websites using the Local File Inclusion method.	Detects attempted Local File Inclusion attacks.
4.	Deteksi serangan <i>Directory Traversal</i> .	Directory Traversal attack detection.	Detects attempted Directory Traversal attacks.

The results of this intrusion detection application are notifications sent via Telegram using the Telegram API and then the POST (postman) method. The display of notifications sent via Telegram can be seen in Figure 6.

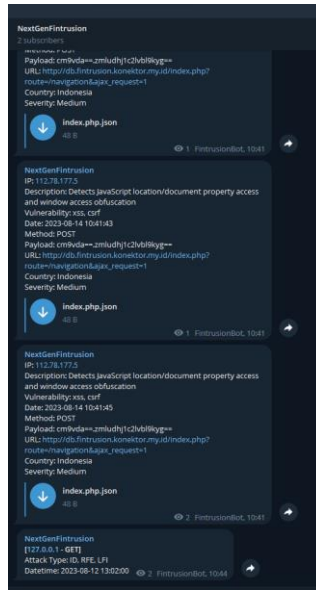


Figure 6. Telegram result report notification

Based on testing attacks that have been carried out using XSSStrike, this system can detect Cross-Site-Scripting attacks with payloads, as shown in Table 2.

Table 2. Payload cross-site-scripting

Data Type	Context	Code Sample	Defense
String	HTML Body	UNTRUSTED DATA	HTML Entity Encoding (rule #1)
String	Safe HTML Attributes	<input type="text" name="fname" value="UNTRUSTED DATA ">	Aggressive HTML Entity Encoding (rule #2): Only place untrusted data into a list of safe attributes (listed below); strictly validate unsafe attributes such as background, I.D., and name
String	G.E.T. Parameter	clickme	URL Encoding (rule #5).
String	Untrusted URL in a S.R.C. or H.R.E.F. attribute	clickme<iframe src="UNTRUSTED URL " />	Canonicalize input, URL Validation, Safe URL verification, Allow-list HTTP and HTTPS URLs only (Avoid the JavaScript Protocol to Open a new Window), Attribute encoder
String	CSS Value	HTML <div style="width: UNTRUSTED DATA;">Selection</div>	Strict structural validation (rule #4), CSS Hex encoding, and Good design of CSS Features.
String	JavaScript Variable	<script>var currentValue='UNTRUSTED DATA';</script><script>someFunction('UNTRUSTED DATA');</script>	Ensure JavaScript variables are quoted, JavaScript Hex Encoding, JavaScript Unicode Encoding, Avoid backslash encoding (\ " or \ ' or \ \).

HTML	HTML Body	<div>UNTRUSTED HTML</div>	HTML Validation (JSoup, AntiSamy, HTML Sanitizer...).
String	DOM XSS	<script>document.write("UNTRUSTED INPUT: " + document.location.hash);</script>	D.O.M. based XSS Prevention Cheat Sheet

For SQL Injection testing, this system can detect attempted SQL Injection attacks, both SQL Injection Bay passing W.A.F., Blind SQL, Injection, Code Injection, Double Encoding, and ORM injection.

Local File Inclusion (L.F.I.) testing of this system can also detect L.F.I. attacks by hackers. L.F.I. attacks usually aim to find view access and include directories or root folders.

Meanwhile, testing Directory Traversal path traversal, this system can detect attacks that aim to enter the principal document containing a web application's subdirectories.

4. CONCLUSION

The following are some conclusions obtained from the discussion and research that has been carried out: the application created can detect attempted attacks on servers that use the Apache web server; attacks that can be detected are Cross-Site-Scripting, SQL Injection, Local File Inclusion, and Directory Traversal. This application needs to update its detection rules because the rules hackers use are increasingly developing.

REFERENCES

- [1] Zen Munawar, and Novianti Indah Putri, "Keamanan Jaringan Komputer Pada Era Big Data," *J-SIKA/Jurnal Sist. Inf. Karya Anak Bangsa*, vol. 2, no. 01 SE-Articles, pp. 14–20, Jul. 2020, [Online]. Available: <https://ejournal.unibba.ac.id/index.php/j-sika/article/view/275>
- [2] B. Fachri and F. H. Harahap, "Simulasi Penggunaan Intrusion Detection System (IDS) Sebagai Keamanan Jaringan dan Komputer," *J. MEDIA Inform. BUDIDARMA*, vol. 4, no. 2, p. 413, Apr. 2020, doi: 10.30865/mib.v4i2.2037.
- [3] M. Zeeshan, S. U. Nisa, T. Majeed, N. Nasir, and S. Anayat, "Vulnerability Assessment and Penetration Testing: A proactive approach towards Network and Information Security," *Int. J. Digit. Inf. Wirel. Commun.*, pp. 124–142, 2017.
- [4] R. Sahtyawan, "PENERAPAN ZERO ENTRY HACKING DIDALAM SECURITY MISCONFIGURATION PADA VAPT (VULNERABILITY ASSESSMENT AND PENETRATION TESTING)," *J. Inf. Syst. Manag.*, vol. 1, no. 1, pp. 18–22, Jul. 2019, doi: 10.24076/JOISM.2019v1i1.18.
- [5] NIST, "Cybersecurity Framework (CSF) 2.0," NIST, 2023.
- [6] M. Antunes, M. Maximiano, R. Gomes, and D. Pinto, "Information Security and Cybersecurity Management: A Case Study with SMEs in Portugal," *J. Cybersecurity Priv.*, vol. 1, no. 2, pp. 219–238, Apr. 2021, doi: 10.3390/jcp1020012.
- [7] G. F. Laurensius, "Security Hardening dengan Cloud Web Service untuk Pengamanan Website Berbasis Wordpress," Universitas Dian Nuswantoro, 2016.
- [8] A. Jacobus, "Sistem Deteksi Intrusi Jaringan Dengan Metode Support Vector Machine," Universitas Gadjah Mada, 2013.
- [9] Netcraft, "June 2021 Web Server Survey," *Netcraft*, 2021. <https://www.netcraft.com/blog/june-2021-web-server-survey/> (accessed Jul. 02, 2021).
- [10] Apache, "Apache HTTP Server 2.4 vulnerabilities," *Apache*, 2021. https://httpd.apache.org/security/vulnerabilities_24.html (accessed Oct. 28, 2021).
- [11] Adam Dwi Ralianto and S. Cahyono, "Perbandingan Nilai Akurasi Snort dan Suricata dalam Mendeteksi Intrusi Lalu Lintas di Jaringan," *Info Kripto*, vol. 15, no. 2, pp. 69–75, Aug. 2021, doi: 10.56706/ik.v15i2.10.
- [12] Nazwita and S. Ramadhani, "Analisis Sistem Keamanan Web Server Dan Database Server Menggunakan Suricata," in *Seminar Nasional Teknologi Informasi, Komunikasi dan Industri*, 2017, pp. 308–317.
- [13] Z. R. Alashhab, M. Anbar, S. D. A. Rihan, B. A. Alabsi, and K. Ateeq, "Enhancing Cloud Computing Analysis: A CCE-Based HTTP-GET Log Dataset," *Appl. Sci.*, vol. 13, no. 16, p. 9086, Aug. 2023, doi: 10.3390/app13169086.
- [14] O. Yevsieieva and S. M. Helalat, "Analysis of the impact of the slow HTTP DOS and DDOS attacks on the cloud environment," in *2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*, 2017, pp. 519–523. doi: 10.1109/INFOCOMMST.2017.8246453.
- [15] M. A. Fauzi, A. T. Hanuranto, and C. Setianingsih, "Sistem Deteksi Intrusi Menggunakan Algoritma Genetik Pada Serangan Dos Di Protokol Tcp Dan Udp," *eProceedings Eng.*, vol. 6, no. 2, p. 4800, 2019.
- [16] Y. Abdullah, J. Triyono, and U. Lestari, "PENGARUH PENEMPATAN SNORT TERHADAP KEAMANAN JARINGAN (STUDI KASUS LABORATORIUM VI JARINGAN KAMPUS 3 IST AKPRIND YOGYAKARTA)," *J. Jarkom*, vol. 8, no. 1, 2020.
- [17] F. Nuraeni and I. Nurfajri, "Notifikasi Network Intrusion Detection System Menggunakan Media Aplikasi Telegram (Studi Kasus: Kantor Imigrasi Tasikmalaya)," *J. Sist. Inf. dan Teknol. Inf.*, vol. 6, no. 1, pp. 87–98, 2017.
- [18] V. M. Hashemi, Z. Muda, and W. Yassin, "Improving Intrusion Detection Using Genetic Algorithm," *Inf. Technol. J.*, vol. 12,

- no. 11, pp. 2167–2173, May 2013, doi: 10.3923/itj.2013.2167.2173.
- [19] S. E. Benaicha, L. Saoudi, S. E. B. Guermeche, and O. Lounis, “Intrusion detection system using genetic algorithm,” in *2014 Science and Information Conference*, IEEE, Aug. 2014, pp. 564–568. doi: 10.1109/SAL.2014.6918242.
- [20] J.-Z. Zhao and H.-K. Huang, “An intrusion detection system based on data mining and immune principles,” in *Proceedings. International Conference on Machine Learning and Cybernetics*, 2022, pp. 524–528 vol.1. doi: 10.1109/ICMLC.2002.1176811.
- [21] P. Jongsuebsuk, N. Wattanapongsakorn, and C. Charnsripinyo, “Real-time intrusion detection with fuzzy genetic algorithm,” in *2013 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, IEEE, May 2013, pp. 1–6. doi: 10.1109/ECTICon.2013.6559603.
- [22] Y. Danane and T. Parvat, “Intrusion detection system using fuzzy genetic algorithm,” in *2015 International Conference on Pervasive Computing (ICPC)*, IEEE, Jan. 2015, pp. 1–5. doi: 10.1109/PERVASIVE.2015.7086963.
- [23] D. Hariyadi and F. E. Nastiti, “Analisis Keamanan Sistem Informasi Menggunakan Sudomy dan OWASP ZAP di Universitas Duta Bangsa Surakarta,” *J. Komtika (Komputasi dan Inform.*, vol. 5, no. 1, pp. 35–42, Jul. 2021, doi: 10.31603/komtika.v5i1.5134.
- [24] R. Z. Nufal, U. Y. K. S. Herdianto, and M. Fathinuddin, “Hardening Cloudfri Dengan Metode Security Hardening Pada Aplikasi Berbasis Website Tap2go.cloudfri.id,” *eProceedings Eng.*, vol. 8, no. 5, 2021.
- [25] T. R. Peltier, *Information Security Risk Analysis*. Auerbach Publications, 2010. doi: 10.1201/EBK1439839560.
- [26] Owasp, “Top 10 Web Application Security Risks,” *Owasp*. <https://owasp.org/www-project-top-ten/> (accessed Mar. 23, 2023).
- [27] A. Sudhodanan, R. Carbone, L. Compagna, N. Dolgin, A. Armando, and U. Morelli, “Large-Scale Analysis & Detection of Authentication Cross-Site Request Forgeries,” in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, Apr. 2017, pp. 350–365. doi: 10.1109/EuroSP.2017.45.
- [28] M. Chawda, D. P. Sharma, and M. J. Patel, “Deep Dive into Directory Traversal and File Inclusion Attacks leads to Privilege Escalation,” *Int. J. Sci. Res. Sci. Eng. Technol.*, pp. 115–120, May 2021, doi: 10.32628/IJSRSET218384.
- [29] A. Begum, M. M. Hassan, T. Bhuiyan, and M. H. Sharif, “RFI and SQLi based local file inclusion vulnerabilities in web applications of Bangladesh,” in *2016 International Workshop on Computational Intelligence (IWCI)*, IEEE, Dec. 2016, pp. 21–25. doi: 10.1109/IWCI.2016.7860332.
- [30] P. Nagarjun and S. Shakeel, “Cross-site Scripting Research: A Review,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 4, 2020, doi: 10.14569/IJACSA.2020.0110481.