

Analysis of SIEM and NIDS Integration for Network Monitoring

Suryayusra¹⁾, Delfin Christofa^{*2)}, Ilman Zuhri Yadi³⁾, Rahmat Novrianda Dasmen⁴⁾

^{1,2)}Informatics Engineering, Faculty of Science and Technology, Universitas Bina Darma, Indonesia

³⁾Information Systems, Faculty of Science and Technology, Universitas Bina Darma, Indonesia

⁴⁾Electrical Engineering, Faculty of Science and Technology, Universitas Bina Darma, Indonesia

¹⁾suryayusra@binadarma.ac.id, ²⁾delfinchristofa1@gmail.com, ³⁾ilmanzuhriyadi@binadarma.ac.id,

⁴⁾rahmat_novrianda@binadarma.ac.id

Submitted : Jul 31, 2026 | **Accepted :** Sep 15, 2026 | **Published :** Oct 1, 2026

Abstract: Network security monitoring is generally still carried out manually (dashboard-gazing), creating a time gap between the moment an incident is detected and the moment it is recognised and responded to by the administrator. This condition is aggravated by a Network Intrusion Detection System (NIDS) that works on its own, because its detection logs remain raw and unstructured, making them difficult to manage and analyse. This study proposes the integration of a Suricata-based NIDS with a Wazuh-based Security Information and Event Management (SIEM), on the grounds that Wazuh is able to manage and organise the raw Suricata logs into structured data while correlating them centrally. The integration was carried out by connecting the Suricata sensor to the Wazuh Server, so that the detection logs (eve.json), which were originally in raw JSON format, could be parsed into alerts with rule IDs and levels that are easy to analyse. Testing was conducted through the simulation of five attack scenarios (brute force, LFI, directory brute force, SQL injection, XSS) over 12 days, producing a correlation of 25,414 security logs covering all scenarios. The security readiness measurement using the KAMI Index v5.0 in the Technology Area shows that 5 of 35 criteria (14.3%) are directly fulfilled and 7 criteria (20%) are partially fulfilled. As an added value, all alerts resulting from the integration were forwarded through webhook-based Discord notifications without any delivery failure, shortening the time between a threat being detected and that threat becoming known to the administrator.

Keywords: *Discord Notification; KAMI Index; NIDS; SIEM; Suricata; Wazuh*

INTRODUCTION

Network security monitoring is currently still generally performed manually, where the administrator has to actively inspect a dashboard in order to find out whether any suspicious activity is taking place (Suryantoro et al., 2022). This kind of approach is less effective because it creates a delay between the moment a threat is detected by the security sensor and the moment that threat is actually known and responded to by the administrator, which may potentially enlarge the impact of an incident if it is not immediately followed up.

To answer that problem, this research focuses on the analysis of the integration of SIEM (Security Information and Event Management) and NIDS (Network Intrusion Detection System) as a solution designed to create a network monitoring system that is not only able to detect attacks, but is also able to manage activity logs centrally. As an added value, the system is also equipped with notification through Discord, so that threat information can reach the administrator more quickly. This research uses Suricata as the NIDS whose task is to perform deep inspection of network traffic in order to find attack patterns (Nandaputra et al., 2024), while its detection data is forwarded to Wazuh as the SIEM platform to be analysed and correlated with other system logs.

Although it is called notification, the process of sending alerts through instant messaging platforms such as Discord fundamentally still cannot take place instantaneously, because it involves several stages ranging from detection by the security sensor, data processing and correlation in the SIEM system, up to message transmission through the internet network, so that there is always a delivery latency (Ishaq & Firmansyah, 2023).

The problems underlying this research can be identified from two sides. First, from the operational side, the conventional network security monitoring mechanism requires the administrator to actively inspect the dashboard (*dashboard-gazing*), so that there is a time gap between a threat being detected and the threat being known and responded to (Amami et al., 2024). Second, from the academic side, there is a *research gap*: previous studies have integrated SIEM and NIDS with Telegram or Discord as the notification media (Damanik & Anggraeni, 2024);

*name of corresponding author



This is anCreative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Suryayusra & Christofa, 2026), but have not yet included an evaluation of information security readiness using a standard instrument such as the KAMI Index. Conversely, other studies have applied the KAMI Index to evaluate information security readiness, but these are administrative self-assessments in the government sector, without being accompanied by technical evidence from the actual implementation of a detection and log management system (Ramadhan et al., 2025).

Based on those two problems, this research offers a *novelty* compared with similar previous studies, namely by extending the integration of SIEM and NIDS, which in previous research generally stopped at the stage of detection and log correlation, into a single workflow that covers: (1) the delivery of alerts through the Discord notification platform, and (2) an analysis of the resulting information security readiness using the KAMI Index v5.0 instrument based on log data evidence, so that the technical integration result does not merely stop as functional evidence, but can also be mapped into standardised information security governance evidence.

Based on the description above, the objectives of this research are: (1) to integrate Suricata (NIDS) and Wazuh (SIEM) in order to centralise and correlate security logs against various network threats; (2) to test the system using 5 attack scenarios in order to ensure that the automatic notification based on the Discord webhook successfully forwards threat alerts to the administrator; (3) to measure the detection success level of the system through rule-triggering coverage as well as the fulfilment level of information security criteria in the Technology Area using the KAMI Index v5.0 instrument; and (4) to analyse the contribution and effectiveness of the Wazuh-Suricata integration together with the Discord notification in supporting the network monitoring process.

To achieve those objectives, the diagnosed problems are resolved through a series of real actions in a simulation environment, starting from the design of the system architecture and topology, the technical implementation of the Wazuh-Suricata integration together with the Discord notification mechanism, the execution of testing using the predetermined attack scenarios, up to a comprehensive evaluation of the results of the actions in order to obtain the learning that will become the input for improvement in the next cycle. This series of cyclical and continuous stages is described in more detail in the research method section.

LITERATURE REVIEW

Several previous studies have become the basis of this research. Research on the integration of SIEM and NIDS with Telegram and Discord notification media has shown that alerts can be forwarded automatically to the administrator, but those studies have not yet included an evaluation of information security readiness using a standard instrument such as the KAMI Index (Adrian et al., 2026; Dewangga et al., 2025). On the other hand, studies that apply the KAMI Index have succeeded in evaluating information security readiness, but they are administrative self-assessments in the government sector and are not accompanied by technical evidence from the actual implementation of a detection and log management system (Wulansari & Novandi, 2022). Other research also shows that Wazuh as an open-source SIEM can be deployed to strengthen cyber defence and to detect threats, including in cloud environments (Moiz et al., 2024), and that attacks on a web server can be analysed using Wazuh-based SIEM (Chamkar et al., 2025). From this review, the gap that is addressed by the present research can be seen, namely combining the technical evidence of SIEM-NIDS integration and notification with the mapping of that evidence to the KAMI Index v5.0 criteria in the Technology Area.

METHOD

This research uses the Action Research method, which is cyclical and continuous, to solve practical problems related to detection accuracy and the speed of cyber incident response in a real setting (Novrianda Dasmen & Kurniawan, 2021). Action Research was chosen because this research requires the researcher to be directly involved in taking action in the form of designing, implementing, and testing the system in a trial environment, which is then evaluated comprehensively in order to obtain learning as the basis for improvement in the next cycle. This method consists of five stages, namely *diagnosing*, *action planning*, *action taking*, *evaluating*, and *learning*.



Fig. 1 Action Research Diagram

*name of corresponding author



This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Diagnosing

The initial stage of this method is identifying the main problems, namely: Wazuh as a SIEM has limitations in detecting activity at the network traffic level if it is not combined with a network sensor; Suricata as a NIDS produces alerts that are local in nature and not yet correlated with logs from other sources; the process of monitoring alerts is still carried out manually through the dashboard, which creates a response time gap; and there is not yet an automatic and notification mechanism for the administrator.

Action Planning

The research environment was built using the *Google Cloud Platform* (GCP) cloud computing service (Krishnan et al., 2023; Lusita et al., 2022), with four virtual machine (VM) instances, namely Wazuh-Server as the SIEM Server, Suricata as the NIDS, Wazuh Agent as the target/endpoint (ubuntu-desktop-1), and an Attacker Node (Kali Linux) to simulate the attacks. Because Suricata is placed on a separate VM, the network traffic going to the target VM is mirrored to the Suricata VM using the Packet Mirroring feature of the GCP VPC so that it can be inspected out-of-band (Subhan et al., 2023). Furthermore, the Suricata VM is also equipped with an additional Wazuh Agent as a log shipper to forward the Suricata detection log file (eve.json) to the Wazuh Manager. The topology of the research system architecture is shown in Fig. 2.

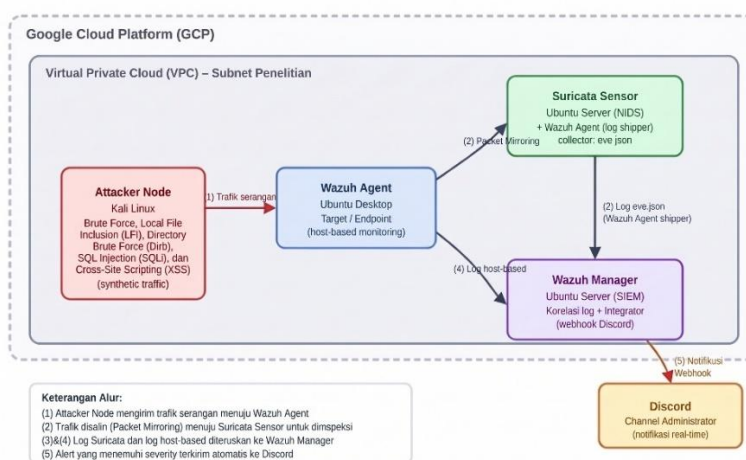


Fig. 2 Research System Topology

The action plan covers: the determination of the VPC network configuration, subnet, and firewall rules; the design of the Packet Mirroring configuration; the design of the log integration architecture (Suricata as log shipper and Wazuh Agent for host-based monitoring); the design of the automatic notification mechanism from the Wazuh Manager to Discord through the Integrator/Webhook feature; the design of the penetration simulation scenarios in the form of *Brute Force*, *Local File Inclusion (LFI)*, *Directory Brute Force (Dirb)*, *SQL Injection (SQLi)*, and *Cross-Site Scripting (XSS)*; as well as the preparation of the documentation study instrument in the form of the KAMI Index v5.0 questionnaire from the National Cyber and Crypto Agency (BSSN) for the Technology Area.

Action Taking

This stage is the technical implementation of the entire plan, starting from the installation of Wazuh 4.12 on the Wazuh-Server and Suricata 8.0.5 on the sensor VM, the configuration of *packet mirroring* on the GCP VPC, the configuration of the eve.json log integration to the Wazuh Manager, the configuration of the Discord webhook notification, up to the execution of three data collection techniques, namely penetration simulation (*pentesting*), technical observation (*automated logging and monitoring*), and the KAMI Index documentation study. The testing was carried out over 12 days, with each of the five types of attack executed once per day, so that in total there were 60 attack simulations (5 types of attack × 12 days).

Evaluating

The evaluation stage was carried out to assess the extent to which the integration actions that had been implemented succeeded in answering the problem formulation, by establishing a success indicator in the form of rule-triggering coverage, namely whether the detection rules on Suricata and/or Wazuh were successfully triggered at least once in every daily testing session for each type of attack. The evaluation was also carried out by mapping the evidence from the technical observation against the criteria of the KAMI Index v5.0 in the Technology Area, in order to assess the contribution of the integration result to information security readiness more broadly, not only in the functional aspect of detection.

*name of corresponding author



This is anCreative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Learning

The learning stage is used to reflect on the findings, the benefits, and the technical obstacles that emerged during the implementation of the actions, particularly those related to the limitations of traffic monitoring in a cloud environment. The results of this learning become the basis for formulating technical improvement recommendations as well as the direction of development in the next action research cycle, in line with the cyclical and continuous nature of action research.

RESULT

This section presents the results of the technical observation and documentation study obtained from the implementation of the actions (action taking) in the simulation environment, covering the log integration result, the Discord notification mechanism, the validation of the attack scenarios, the contribution to traffic monitoring, and the evaluation using the KAMI Index v5.0.

Analysis of the Wazuh and Suricata Integration

Based on the raw log data extracted from the Wazuh database, the Wazuh Manager system successfully collected, normalised, and centralised a total of 25,414 *events* (security events) during the 12-day trial period, namely from 1 to 12 July 2026. This threat visibility was obtained from two main monitoring points (agents). Suricata functions to monitor traffic at the network layer as a whole and contributed 8,946 logs (35.20%), while the host-based agent ubuntu-desktop-1 monitors activity at the operating system level and contributed 16,468 logs (64.80%), as summarised in Table 1 and visualised in Fig. 3.

Table 1. Distribution of Threat Logs Based on VM

No	VM	VM Role	Number of Logs (Events)	Percentage
1	ubuntu-desktop-1	Host-based monitoring (Endpoint)	16,468	64.80%
2	suricata	Network-based monitoring (NIDS)	8,946	35.20%
	Total		25,414	100%

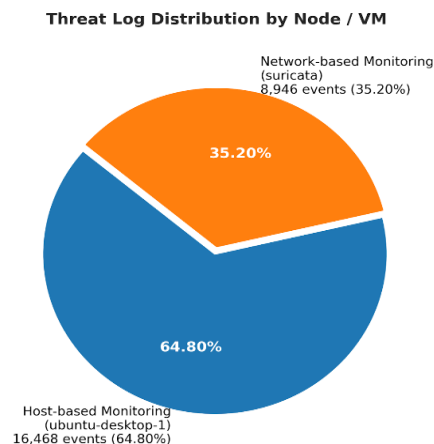


Fig. 3 Distribution of Threat Logs Based on VM

Based on Table 1 and Fig. 3, it can be concluded that the system integration runs very well. The system is able to provide complete (360-degree) visibility of the security posture, because threats are not only detected while they are on the network (through Suricata), but are also monitored while they target the endpoint infrastructure (through the Wazuh Agent). This confirms that the use of Suricata as a complement to Wazuh successfully covers the network-level detection weakness of a single host-based agent, which, if it ran alone, would lose visibility of attack patterns at the network payload layer.

Of these 25,414 correlated events, all were confirmed to have been forwarded through the Discord webhook without a single delivery failure, confirming that the notification pipeline is fully aligned with the log correlation result.

The need for this integration is also based on the condition of the Suricata logs which, by default, are still in raw format and not neatly arranged, so that they are difficult to read, filter, or analyse manually once the volume

*name of corresponding author



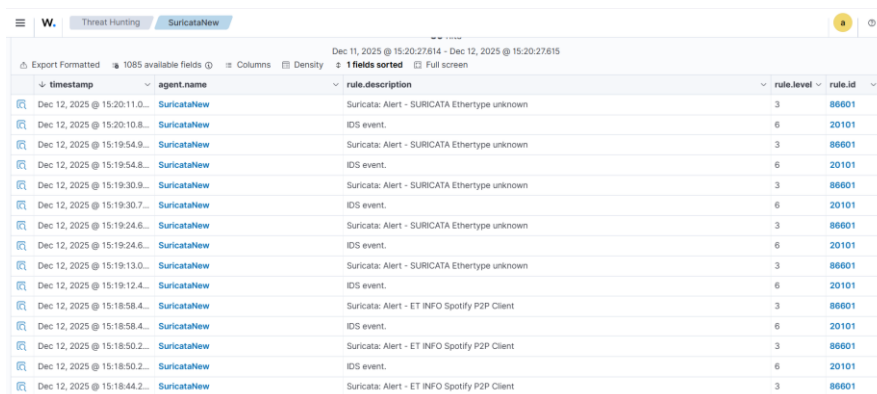
This is anCreative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

of the generated logs is already very large. As a comparison, Fig. 4 shows the condition of the Suricata log before integration (fast.log), in which all log information such as the time of the event, the signature ID, the classification, the priority, as well as the source and destination IP addresses are still merged into one long line of text without a clear data structure.

```
suricata@suricata:~$ sudo tail -f /var/log/suricata/fast.log
12/12/2025-08:21:88.189725 [**] [1:2280121:1] SURICATA Etype unknown [**] [Classification: Generic Protocol Command Decode] [Priority: 3] [**] [Raw pkt
: FF FF FF FF FF FF 38 17 C3 C3 80 FA 8F FD BE EF 07 00 22 01 62 53 14 0A 54 4A 81 00 00 14 C0 A8 ]
12/12/2025-08:21:52.429222 [**] [1:2280121:1] SURICATA Etype unknown [**] [Classification: Generic Protocol Command Decode] [Priority: 3] [**] [Raw pkt
: FF FF FF FF FF FF 00 4E 8C C1 00 14 0F FD 8C EF 07 00 22 01 62 53 14 0A 54 4A 81 00 00 14 C0 A8 ]
12/12/2025-08:23:36.379257 [**] [1:2027397:1] ET INFO Spotify P2P Client [**] [Classification: Not Suspicious Traffic] [Priority: 3] [UDP] 172.16.0.177:576
21 -> 172.16.255.255:57621
12/12/2025-08:23:37.464639 [**] [1:2027397:1] ET INFO Spotify P2P Client [**] [Classification: Not Suspicious Traffic] [Priority: 3] [UDP] 172.16.0.66:5762
1 -> 172.16.255.255:57621
12/12/2025-08:24:05.458836 [**] [1:2027397:1] ET INFO Spotify P2P Client [**] [Classification: Not Suspicious Traffic] [Priority: 3] [UDP] 172.16.0.5:57621
-> 172.16.255.255:57621
12/12/2025-08:24:22.968471 [**] [1:2027397:1] ET INFO Spotify P2P Client [**] [Classification: Not Suspicious Traffic] [Priority: 3] [UDP] 172.16.1.132:576
21 -> 172.16.255.255:57621
12/12/2025-08:24:35.248689 [**] [1:2027397:1] ET INFO Spotify P2P Client [**] [Classification: Not Suspicious Traffic] [Priority: 3] [UDP] 172.16.1.0:57621
-> 172.16.255.255:57621
12/12/2025-08:24:35.787360 [**] [1:2027397:1] ET INFO Spotify P2P Client [**] [Classification: Not Suspicious Traffic] [Priority: 3] [UDP] 172.16.0.146:576
21 -> 172.16.255.255:57621
12/12/2025-08:24:39.138083 [**] [1:2027397:1] ET INFO Spotify P2P Client [**] [Classification: Not Suspicious Traffic] [Priority: 3] [UDP] 172.16.0.246:576
21 -> 172.16.255.255:57621
12/12/2025-08:26:10.308131 [**] [1:2027397:1] ET INFO Spotify P2P Client [**] [Classification: Not Suspicious Traffic] [Priority: 3] [UDP] 172.16.0.150:576
21 -> 172.16.255.255:57621
12/12/2025-08:26:10.308131 [**] [1:2100498:7] GPL ATTACK_RESPONSE id check returned root [**] [Classification: Potentially Bad Traffic] [Priority: 2] [TCP]
217.160.0.187:80 -> 172.16.2.39:77918
12/12/2025-08:26:18.068821 [**] [1:2100498:7] GPL ATTACK_RESPONSE id check returned root [**] [Classification: Potentially Bad Traffic] [Priority: 2] [TCP]
217.160.0.187:80 -> 172.16.2.39:57634
12/12/2025-08:26:19.235981 [**] [1:2100498:7] GPL ATTACK_RESPONSE id check returned root [**] [Classification: Potentially Bad Traffic] [Priority: 2] [TCP]
217.160.0.187:80 -> 172.16.2.39:57638
12/12/2025-08:26:28.188822 [**] [1:2100498:7] GPL ATTACK_RESPONSE id check returned root [**] [Classification: Potentially Bad Traffic] [Priority: 2] [TCP]
217.160.0.187:80 -> 172.16.2.39:57648
12/12/2025-08:26:21.067869 [**] [1:2100498:7] GPL ATTACK_RESPONSE id check returned root [**] [Classification: Potentially Bad Traffic] [Priority: 2] [TCP]
217.160.0.187:80 -> 172.16.2.39:57654
```

Fig. 4 Suricata Log Before Integration (fast.log, raw text)

Conversely, after the same data is forwarded and processed by Wazuh, the log is parsed into separate columns on the Wazuh Threat Hunting dashboard, namely timestamp, agent.name, rule.description, rule.level, and rule.id, as shown in Fig. 5. This neat JSON data structure makes it easier for the analyst to search, sort, and filter the logs based on particular columns, thus proving that the role of Wazuh in managing and organising the Suricata logs successfully eliminates the previously disorganised log condition.



timestamp	agent.name	rule.description	rule.level	rule.id
Dec 12, 2025 @ 15:20:11.0...	SuricataNew	Suricata: Alert - SURICATA Etype unknown	3	86601
Dec 12, 2025 @ 15:20:10.8...	SuricataNew	IDS event.	6	20101
Dec 12, 2025 @ 15:19:54.9...	SuricataNew	Suricata: Alert - SURICATA Etype unknown	3	86601
Dec 12, 2025 @ 15:19:54.8...	SuricataNew	IDS event.	6	20101
Dec 12, 2025 @ 15:19:30.9...	SuricataNew	Suricata: Alert - SURICATA Etype unknown	3	86601
Dec 12, 2025 @ 15:19:30.7...	SuricataNew	IDS event.	6	20101
Dec 12, 2025 @ 15:19:24.6...	SuricataNew	Suricata: Alert - SURICATA Etype unknown	3	86601
Dec 12, 2025 @ 15:19:24.6...	SuricataNew	IDS event.	6	20101
Dec 12, 2025 @ 15:19:13.0...	SuricataNew	Suricata: Alert - SURICATA Etype unknown	3	86601
Dec 12, 2025 @ 15:19:12.4...	SuricataNew	IDS event.	6	20101
Dec 12, 2025 @ 15:18:58.4...	SuricataNew	Suricata: Alert - ET INFO Spotify P2P Client	3	86601
Dec 12, 2025 @ 15:18:58.4...	SuricataNew	IDS event.	6	20101
Dec 12, 2025 @ 15:18:50.2...	SuricataNew	Suricata: Alert - ET INFO Spotify P2P Client	3	86601
Dec 12, 2025 @ 15:18:50.2...	SuricataNew	IDS event.	6	20101
Dec 12, 2025 @ 15:18:44.2...	SuricataNew	Suricata: Alert - ET INFO Spotify P2P Client	3	86601

Fig. 5 Log After Integration into the Wazuh Dashboard (structured columns)

Validation and Testing of the Attack Scenarios

The system validation was carried out by simulating five cyber attack scenarios in order to test the detection capability of Suricata and the forwarding response of Wazuh. The testing was carried out over 12 days, with each of the five types of attack (*Brute Force*, *Dirb*, *LFI*, *SQL Injection*, and *XSS*) executed once on every testing day, so that in total there were 60 attack simulations (5 types of attack × 12 days). The results of the data processing show that all detection rules configured on Suricata successfully produced alerts for every attack category tested, as summarised in Table 2.

Table 2. Recapitulation of Detection Success Indicators per Attack Scenario

No	Type of Attack	Number of Cases	Detected by
1	SQL Injection (SQLMap)	12 of 12 days	Suricata & Wazuh (Ubuntu)
2	Directory Brute Force (Dirb)	12 of 12 days	Suricata & Wazuh (Ubuntu)
3	Local File Inclusion (LFI)	12 of 12 days	Suricata & Wazuh (Ubuntu)
4	Brute Force SSH	12 of 12 days	Suricata & Wazuh (Ubuntu)
5	Cross-Site Scripting (XSS)	12 of 12 days	Suricata & Wazuh (Ubuntu)

It needs to be emphasised that the table above proves the *scenario coverage* of the detection, namely that every attack category successfully triggered at least one related rule; it is not a formal measurement of detection accuracy. This daily repetition gives an overview of the consistency of detection over time, but a quantitative measurement of detection accuracy (including the statistical rate of correct and incorrect detections, as well as a

comparison with normal traffic as a negative control) has still not been formally carried out in this research, so this is stated as a limitation of the study and as a planned action for the next cycle.

Given that limitation, the research success indicator was determined using the rule-triggering coverage approach, namely whether the detection rules on Suricata and/or Wazuh were successfully triggered at least once in every daily testing session for each type of simulated attack. The recapitulation of the detection success indicators per attack scenario is shown in Table 3.

Table 3. Recapitulation of Detection Success Indicators per Attack Scenario

No	Type of Attack	Number of Test Sessions	Days Detected	Detection Rate
1	SQL Injection (SQLMap)	12	12	100%
2	Local File Inclusion (LFI)	12	12	100%
3	Cross-Site Scripting (XSS)	12	12	100%
4	Directory Brute Force (Dirb)	12	12	100%
5	Brute Force SSH (Hydra)	12	12	100%
Total		60	60	100%

Table 3 shows that all five types of attack scenario successfully triggered at least one detection *rule* in every daily testing session, so that the research success indicator in the form of rule-triggering coverage was achieved at 100% (60 of 60 testing sessions) during the 12-day testing period. The achievement of this indicator becomes the basis for the conclusion that the SIEM-NIDS-Discord integration architecture that was built has functioned as planned in the action planning stage.

Analysis of the Integration with Respect to Traffic Monitoring

The final objective of the Wazuh and Suricata integration is to analyse its contribution to monitoring network traffic. In a conventional monitoring system, the administrator has to read raw traffic data, which requires a long analysis time. By using the SIEM approach, the system automatically performs risk scoring by mapping every incident into a severity level (rule level). Out of the total of 25,414 *events* (security events), the system has grouped the incidents hierarchically as shown in Table 4.

Table 4. Distribution of Incidents Based on Severity (Rule Level)

Rule Level	Severity	Number of Incidents	Percentage
Level 3	Low	8,610	33.88%
Level 4	Low	120	0.47%
Level 5	Medium	8,293	32.63%
Level 6	Medium	7,621	29.99%
Level 10	High	756	2.97%
Level 12	Critical	14	0.06%
Total		25,414	100%

The conclusion of the analysis of Table shows that the SIEM integration contributes to network traffic monitoring through the automation of risk grouping for the 25,414 *events* (security events), without the administrator needing to read the raw logs one by one manually. The system successfully grouped the raw logs into rule levels, with the majority classified as low-to-medium level threats ($\pm 96.97\%$), while anomalies with high to critical risk (Level 10 and 12) were identified separately at $\pm 3.03\%$, so that the administrator can immediately prioritise handling based on the severity that has already been categorised automatically by the system.

Evaluation of Information Security Readiness Using the KAMI Index v5.0

The evaluation of the readiness and maturity of the security system in the Technology Area was measured using the KAMI (Information Security) Index v5.0 instrument from BSSN. The system integration built in this research directly fulfils and answers a number of critical question indicators in that questionnaire. For example, control 6.1 concerning whether ICT services are protected by more than one layer of security is declared proven, because the NIDS (Suricata), the SIEM (Wazuh), and the Discord notification form three layers of defence in depth, whereas other controls such as network segmentation are declared indirect because they are procedural in the design stage and are not the result of observation from the testing log data.

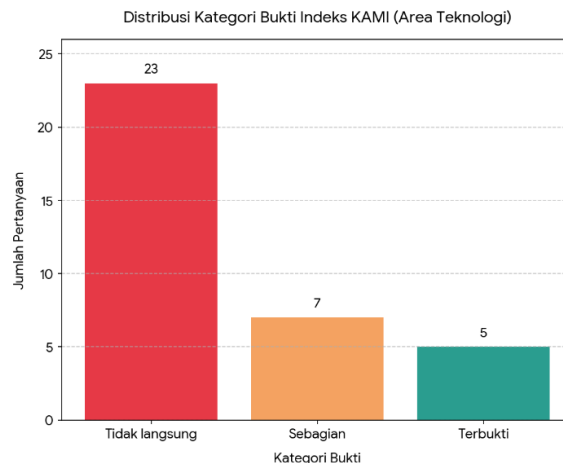


Fig. 6 Measurement Result of the KAMI Index v5.0 in the Technology Area

Fig. 6 shows that out of a total of 35 questions/controls in the Technology Area of the KAMI Index v5.0, the implemented Wazuh and Suricata integration system is proven to fulfil 5 criteria (14.3%) directly, namely the controls related to layered protection, automatic log recording, log analysis for forensic trails, and time synchronisation. In addition, there are 7 criteria (20.0%) that are partially fulfilled, where the architecture that was built only detects and notifies without an automatic prevention mechanism, or is only partially covered by the scope of the testing. Meanwhile, the remaining 23 criteria (65.7%) have the status of indirect or outside the scope of the simulation, especially the controls in the KAMI Index v5.0 that demand evidence in the domain of secure application development (*secure coding*, *source code review*), encryption, host antivirus/antimalware, and Data Leakage Prevention (DLP), which are indeed not the focus of the SIEM-NIDS-Discord architecture in this research.

The result of this mapping is a self-assessment that has not been verified by BSSN, and it confirms that the implementation of SIEM and NIDS provides partial technical evidence of the fulfilment posture of the Technology Area in the simulation environment that was built, without claiming a calculation of the overall *maturity level*.

DISCUSSIONS

Based on the trial results, the researchers found that if only one security tool is relied upon, the security officer has to read and inspect thousands of log lines manually, which is very troublesome. The combination of the two security systems (Suricata and Wazuh) successfully overcomes this problem, because this combined system can be likened to an assistant that automatically sorts, matches, and analyses all suspicious activity from various angles, so that the officer can see the security condition as a whole more clearly.

The process of forwarding the alerts also ran smoothly, where the threat alert messages were successfully received. This unification of notifications simplifies the process of delivering information so that the administrator is not required to monitor the Wazuh dashboard continuously, in line with the findings of previous research that also made use of webhook integration for instant security notifications.

In addition, the presence of the automatic notification feature through the Discord messaging application changes the way the administrator works from previously reactive (continuously staring at the monitor screen waiting for alerts) into proactive, because threat alerts come directly to the administrator's phone or computer just like an ordinary chat message, so that handling can be carried out much more quickly. Using a government assessment standard (the KAMI Index v5.0) also provides the important lesson that a good security system is not only a matter of the ability to detect attack simulations, but must also comply with the official rules and standards established by the National Cyber and Crypto Agency (BSSN), such as the ability to record events automatically, to store strong forensic evidence, and to have accurate time settings.

During the implementation process, the researchers also realised that installing a security system in a *cloud* computing environment is far more complicated than in a physical network, because the cloud infrastructure applies very strict segmentation rules so that direct traffic tapping on the cable path cannot be carried out. To overcome this, the researchers made use of the *packet mirroring* technique to reflect a copy of the data traffic to the Suricata VM so that it could be inspected out-of-band without disturbing the main data path. This experience becomes an important lesson that can serve as a reference for other parties who intend to move their network security system to *cloud* infrastructure safely.

The limitation of this study is that the measurement of detection accuracy has not been carried out formally in a quantitative manner, including the statistical rate of correct and incorrect detections and the comparison with normal traffic as a negative control, so that the success indicator used is limited to rule-triggering coverage. In

addition, the mapping to the KAMI Index v5.0 is a self-assessment that has not been verified by BSSN, and the testing was carried out in a simulation environment on a cloud platform, so that the results cannot yet be generalised directly to a production network environment.

CONCLUSION

Based on the results of the testing, evaluation, and learning from the action research on the integration of the network security system in the simulated cloud environment, it can be concluded that the integration between Suricata and Wazuh is proven to be effective in providing complete security visibility and in overcoming data fragmentation, because Wazuh successfully acted as a correlation engine that cross-references network sensor data with host-level data automatically in order to produce alerts for attack categories such as *SQL Injection*, *XSS*, *LFI*, and *Brute Force* out of a total of 25,414 *events* (security events) recorded during the simulation, with rule-triggering coverage reaching 100% in all 60 testing sessions.

Regarding the contribution to the fulfilment of information security criteria, this system architecture successfully supports national BSSN compliance by fulfilling 5 of 35 criteria (14.3%) directly and 7 criteria (20%) partially in the Technology Area of the KAMI Index v5.0, while the remaining criteria are not fulfilled because they are administrative/policy in nature or are outside the development focus of this architecture. Finally, regarding the effectiveness of the notification, the use of the Discord webhook is proven effective in transforming the administrator's network monitoring culture from previously reactive into more proactive through the delivery of accurate, fast, and incident threat alerts. Future research is recommended to expand the attack simulation scenarios, to perform continuous ruleset calibration, and to implement a deduplication and notification aggregation mechanism in order to reduce the potential for *notification flooding*.

REFERENCES

- Adrian, A. Z. A., Megantara, R. A., & Al Zami, F. (2026). Hybrid Multilayer Architecture Integrating Suricata, Wazuh, and Cyber Threat Intelligence for Drive-by-Download Malvertising Detection. *Sinkron: Jurnal Dan Penelitian Teknik Informatika*, 10(1), 161–168.
- Amami, R., Charfeddine, M., & Masmoudi, S. (2024). Exploration of Open Source SIEM Tools and Deployment of an Appropriate Wazuh-Based Solution for Strengthening Cyberdefense. *2024 10th International Conference on Control, Decision and Information Technologies (CoDIT)*, 1–7.
- Chamkar, S. A., Zaydi, M., Maleh, Y., Gherabi, N., Chamkar, S. A., Zaydi, M., Maleh, Y., & Gherabi, N. (2025). Improving Threat Detection in Wazuh Using Machine Learning Techniques. *Journal of Cybersecurity and Privacy* 2025, Vol. 5, 5(2). <https://doi.org/10.3390/JCP5020034>
- Damanik, H. A., & Anggraeni, M. (2024). Sistem Deteksi Intrusi Hybrid dan Mitigasi Kerentanan Infrastruktur Jaringan Menggunakan Teknik Active Response (XDR) Wazuh dan Suricata. *Jurnal Pekommas*, 9(2), 309–322. <https://doi.org/10.56873/jpkm.v9i2.5829>
- Dewangga, O., Pramudya, D., Hatta, P., & Budiyo, C. W. (2025). MODELING INTRUSION DETECTION AND PREVENTION SYSTEM TO DETECT AND PREVENT NETWORK ATTACKS USING WAZUH. *Jurnal Teknik Informatika (Jutif)*, 6(1), 173–186. <https://doi.org/10.52436/1.JUTIF.2025.6.1.1830>
- Ishaq, M. Y., & Firmansyah, F. (2023). Implementasi Sistem Monitoring Menggunakan Zabbix Dan Notifikasi Realtime Telegram. *Jurnal INSAN Journal of Information System Management Innovation*, 3(2), 72–77. <https://doi.org/10.31294/JINSAN.V3I2.2432>
- Krishnan, P., Jain, K., Aldweesh, A., Prabu, P., & Buyya, R. (2023). OpenStackDP: a scalable network security framework for SDN-based OpenStack cloud infrastructure. *Journal of Cloud Computing*, 12(1), 26.
- Lusita, D., Anissa, F., & Andryani, R. (2022). Penerapan Cloud Computing Dalam Aplikasi Panggil Teknisi Berbasis Android Menggunakan Google Cloud Platform. *Jurnal Sains Komputer & Informatika (J-SAKTI)*, 6(2), 1292–1300.
- Moiz, S., Majid, A., Basit, A., Ebrahim, M., Abro, A. A., & Naeem, M. (2024). Security and threat detection through cloud-based Wazuh deployment. *2024 IEEE 1st Karachi Section Humanitarian Technology Conference (KHI-HTC)*, 1–5.
- Nandaputra, J. R., Sukarno, P., & Wardana, A. A. (2024). Detection and Prevention System on Computer Network to Handle Distributed Denial-Of-Service (Ddos) Attack in Realtime and Multi-Agent. *ACM International Conference Proceeding Series*, 237–241. <https://doi.org/10.1145/3674558.3674592>
- Novrianda Dasmen, R., & Kurniawan, F. (2021). *Digital Forensik Deleted Cyber Crime Evidence pada Pesan Instan Media Sosial Digital Forensics Deleted Cyber Crime Evidence on Social Media Instant Messaging*. 20(4), 527–539.
- Ramadhan, R., Widiyasono, N., & Rahmatullah, A. (2025). Evaluasi Keamanan Informasi Sistem Informasi Manajemen RSUD KHZ Musthafa Menggunakan Indeks KAMI 5.0 Berbasis ISO/IEC 27001: 2022. *JoMMIT: Jurnal Multi Media Dan IT*, 9(1), 80–86.

- Subhan, A., Kunang, Y. N., & Yadi, I. Z. (2023). Analyzing the attack pattern of brute force attack on SSH port. *2023 International Conference on Information Technology and Computing (ICITCOM)*, 67–72.
- Suryantoro, T., Purnomosidi, B. D. P., & Andriyani, W. (2022). The analysis of attacks against port 80 webserver with SIEM Wazuh using detection and OSCAR methods. *2022 5th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 1–6.
- Suryayusra, & Christofa, D. (2026). Integrasi Wazuh File Integrity Monitoring Dan Suricata Dengan Notifikasi Telegram Untuk Keamanan Jaringan. *Jurnal Ilmiah Matrik*, 28(1), 49–57. <https://doi.org/10.33557/99M23A40>
- Wulansari, T. T., & Novandi, D. (2022). Evaluation of information security management using the KAMI index framework. *2022 International Conference of Science and Information Technology in Smart Administration (ICSINTESA)*, 173–177.