Integration of Multiple Honeypots with a Security Information and Event Management (SIEM) System for Network Attack Detection

Nadhif Ahmad Mutawakkil Zaidan*, Vera Suryani, and Aulia Arif Wardana

Telkom University, Bandung, Indonesia
nichzeith@student.telkomuniversity.ac.id
verasuryani@telkomuniversity.ac.id
auliawardan@telkomuniversity.ac.id

Abstract. Security monitoring fragmentation in heterogeneous service infrastructures running SMB, FTP, and HTTP simultaneously creates a wide attack surface. Per-service log isolation prevents timely identification of multi-vector attack patterns. This paper presents a documented, reproducible reference architecture that integrates two Dionaea honeypot instances (ServiceArea and CoreArea) with Wazuh SIEM v4.14.3, managed through a MikroTik CHR 7.20.8 router on a three-zone segmented network topology. We developed six custom Wazuh detection rules (Rule IDs 100200-100212) *before* any experimental run. Each rule is keyed to the documented JSON event schema of Dionaea and justified against prior literature rather than tuned to the test scenarios, to guard against the overfitting that a post-hoc rule design would risk. A MikroTik DNAT configuration redirects external TCP port 445 traffic to the CoreArea honeypot, which enables detection of the MS17-010 (CVE-2017-0143) vulnerability from outside the local network. We measured detection performance over three repeated runs across five attack scenarios, chosen to match the three most commonly exploited services in the honeypot literature this paper builds on. The experiment yielded 156 total alerts (75 from CoreArea, 81 from ServiceArea), a 100% detection rate, and an F1-score of 1.0. Four of the six rules were exercised by these scenarios; the remaining two are validated by construction but not yet by traffic. We did not assume the 0% false positive rate from idle lab traffic. Instead we validated it against a 5-hour baseline of 158 benign network events that explicitly included legitimate FTP, HTTP, and SMB traffic from a real production server (ServiceServer) running services identical to those targeted by the attack scenarios. We measured mean detection latency via NTP-synchronized timestamps for the two scenarios where event-to-alert pairing was unambiguous. The result, 59.1 ± 4.0 seconds, reflects an architectural property of the one-minute log-conversion cycle rather than of the rule-matching engine in Wazuh. A controlled five-trial comparison showed manual operator identification time dropping from 22.4 ± 3.1 minutes (standalone) to 42.3 ± 8.7 seconds (integrated), a 96.8% efficiency improvement. We report these results together with their methodological boundaries, including small sample size, a single honeypot type, no adversarial evasion testing, and no comparison against commercial SIEM or network-wide IDS platforms. We position the contribution as an evaluated deployment framework rather than a claim of algorithmic novelty.

Keywords: Dionaea honeypot · Wazuh SIEM · multiple honeypot · multi-vector attack detection · MikroTik CHR · false positive validation · confusion matrix · reproducible security architecture.

* Corresponding author

© The Author(s) 2026
M. Murtadha et al. (eds.), *Proceedings of the 2026 11th International Conference on Sustainable Information Engineering and Technology (SIET 2026)*, Atlantis Highlights in Computer Sciences 26,
https://doi.org/10.2991/978-94-6239-828-3_23

1 Introduction

Cybersecurity incidents based on network protocol exploitation have escalated significantly from 2020 to 2025. A honeypot deployment study by Hartinah et al. recorded over 11,000 captured attack attempts. SMB (port 445) and FTP (port 21) were among the most frequently targeted services, and the WannaCry ransomware, which exploits the SMB vulnerability CVE-2017-0144 (MS17-010), remains the most frequently captured malware family to date [1]. The Verizon Data Breach Investigations Report (2024) corroborates this trend. It identifies public service exploitation as the second most common attack vector in the industrial and education sectors [11]. Modern IT infrastructures running SMB, FTP, and HTTP concurrently present a structurally wide attack surface, and per-service monitoring systems produce fragmented log data that makes multi-vector attack pattern identification difficult to perform centrally and in a timely manner.

Honeypot-based deception technology offers a proactive defense approach. It emulates vulnerable services to attract and record attacker activity without disrupting production systems. Dionaea, a low-interaction honeypot, emulates SMB, FTP, and HTTP with the capability to capture binary malware payloads, which makes it a suitable instrument for threat analysis on heterogeneous services [6]. However, a standalone Dionaea deployment leaves attack data isolated without a centralized correlation mechanism.

Wazuh, an open-source SIEM platform, has demonstrated effectiveness for centralized log management and live-analysis pipelines when combined with intrusion detection systems [7, 8]. Community practice and prior academic work have each explored pieces of a Dionaea-Wazuh pipeline, and multi-zone honeypot deployment behind a router is a pattern already used in industry security labs. We do not claim the integration concept itself is new. What counts as the contribution, then, can be answered in three checkable parts. First, no prior Dionaea-Wazuh study we found documents the JSON-field-to-rule mapping with literature-grounded severity justification at the level of Table 4. Second, none reports a false-positive rate measured against *real production traffic on the same protocols the attacks target*, as opposed to idle lab noise (Section 4.2). Third, none reports a controlled, rather than estimated, efficiency comparison against standalone monitoring. Three artifacts, a justified rule table, a production-traffic FPR validation, and a controlled efficiency measurement, form what this paper offers, and the basis on which it should be judged, rather than a detection-algorithm contribution. Section 5 states the full scope of what is *not* claimed.

Related recent work illustrates a complementary direction. Machine-learning detectors generalize across previously unseen patterns rather than matching fixed signatures [13, 12], as discussed in Section 2. This paper targets a different problem, instrumenting a low-interaction honeypot deployment so its signature-based alerts are documented, justified, and validated against real traffic before any learning-based layer is added.

2 Related Work

Morić et al. [2] compared seven honeypot systems including Dionaea and found it offers strong protocol emulation versatility, but no concrete SIEM integration architecture. Suhendi et al. [3] integrated Artillery Honeypot with Wazuh and detected anomalies on the dashboard, but the emulation of Artillery differs from that of Dionaea (particularly on the SMB server side), so its rule configuration is not directly transferable. Islam and Rafique [5] and Jumiaty and Soewito [4] each deployed Wazuh in production or incident-response contexts, confirming its maturity, but neither used honeypots as a proactive source. Matin and Rahardjo [9] built a Dionaea-based malware collection framework validating its

sensor capability, while Saikawa and Klyuev [10] performed offline classification of malicious Dionaea access without real-time monitoring or SIEM correlation.

Across this literature, three systematic gaps recur. First, honeypot studies treat captured data as an isolated source without a documented SIEM correlation pipeline. Second, SIEM studies validate platform capability using conventional system logs rather than deception-technology input. Third, no prior study in this space reports a confusion matrix, latency distribution, or a controlled (rather than estimated) comparison against standalone monitoring.

Two further studies, suggested during review, illustrate the machine-learning branch of this area rather than the signature/deception branch this paper belongs to. NetPhish-Mix fuses graph- and vision-transformer representations to detect phishing/malicious domains, reporting F1 = 0.977 and ROC-AUC = 0.997 [12]. Holi et al. combine a CNN-Transformer IDS with QCNN/ABE encryption for cloud retrieval, jointly tuned via Giant Trevally Optimization, reporting 98.9% binary accuracy on NSL-KDD [13]. Both target model generalization to unseen instances, a property a fixed-signature system does not claim. This paper instead asks whether a *documented, non-learned* rule set can be justified and validated against real traffic, a narrower question, useful for practitioners who need a deployable reference before a research-grade classifier. Table 1 summarizes this positioning.

Table 1. Research Gap Analysis

Study	Basis	Dio.	Wazuh	Quant.	FPR/real
Suhendi [3]	Rule-based	No	Yes	No	No
Matin [9]	Rule-based	Yes	No	No	No
Islam [5]	Rule-based	No	Yes	No	No
Saikawa [10]	Offline classif.	Yes	No	No	No
Holi et al. [13]	ML (CNN-Transf.)	No	No	Yes	N/A
NetPhish-Mix [12]	ML (fusion)	No	No	Yes	N/A
This work	**Rule, justified**	**Yes**	**Yes**	**Yes**	**Yes**

3 System Design and Implementation

3.1 Network Topology and Architecture

The system is implemented on a three-zone segmented network managed by MikroTik CHR 7.20.8. The three zones are as follows.

- **Management Zone** (192.168.110.0/24) hosts Wazuh Manager v4.14.3 on Ubuntu 22.04 LTS.
- **Service Zone** (172.25.1.0/24) hosts the ServiceArea honeypot (Dionaea 0.11.0, Ubuntu 18.04 LTS) and a separate *ServiceServer* node running genuine FTP (vsftpd), HTTP (Apache2), and SMB (Samba) services for false-positive validation.
- **Core Zone** (10.10.15.0/24) hosts the CoreArea honeypot (Dionaea 0.11.0, Ubuntu 18.04 LTS).

The attacker node (Kali Linux) connects through the internet-facing interface. Fig. 1 shows the complete topology.

Table 2 lists the complete experimental configuration, addressing the absence of hardware/software specification noted in peer review.

3.2 MikroTik Configuration and Dionaea Deployment

Firewall rules permit inter-zone traffic required for Wazuh Agent log forwarding between the honeypot zones and the Management zone. Without these rules, neither agent can reach the Manager. Table 3 lists the NAT/DNAT configuration. The DNAT rule (No. 3) is the key mechanism enabling SMB attack detection

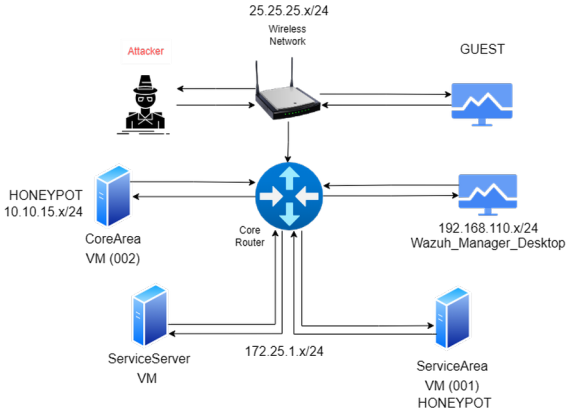


Fig. 1. Three-zone segmented network topology managed by MikroTik CHR 7.20.8.

Table 2. Experimental Parameters

Category	Parameter	Value
Duration	Active experiment / benign baseline	5 h / 5 h (separate session)
Hardware	Honeypot & server nodes	3× VM, Ubuntu 18.04 LTS
	SIEM node	1× VM, Ubuntu 22.04 LTS
	Router	MikroTik CHR 7.20.8
Software	Honeypot / SIEM	Dionaea 0.11.0 / Wazuh 4.14.3
	Attack tools	Nmap, Hydra, Metasploit, curl
	Log conversion	Cron-scheduled SQLite-to-JSON, 1-min interval
Repetition	Runs per scenario	3
Protocols	Targeted services	FTP (21), HTTP (80), SMB (445)

from outside the local network. Without it, TCP port 445 traffic cannot cross zone boundaries to reach the CoreArea honeypot.

Table 3. MikroTik NAT Rules

No	Type	Condition	Description
0	srcnat	out: ether1	Masquerade all outbound traffic
1	srcnat	out: ether3	Masquerade towards Service Honeypot
2	srcnat	out: ether4	Masquerade towards Core Honeypot
3	dstnat	in: ether1, port 445	**DNAT** redirect TCP port 445 to CoreArea

Dionaea 0.11.0 was deployed on two nodes, each emulating exactly three services, SMB, FTP, and HTTP, rather than the full default set. This scope reduction keeps the rule set focused and verifiable, and matches the three protocols most frequently exploited in the threat landscape motivating this work [1]. Service banners mimic real products (a Synology NAS for FTP, nginx for HTTP) to increase deception fidelity against automated scanners.

3.3 Log Conversion, Wazuh Integration, and Detection Rules

Dionaea stores activity in a SQLite database, converted to JSON every minute by a cron-scheduled script. This 1-minute polling interval is the dominant contributor to detection latency, as discussed in Section 4. Actual Wazuh decoding and rule-matching adds under 2 seconds. Wazuh Agents on each honeypot node read the JSON output via a `localfile` block reading `eventid`-keyed records that are produced exclusively by the Dionaea process, a property exploited di-

rectly in the false-positive validation described in Section 4.2. Fig. 2 summarizes the complete end-to-end detection pipeline.

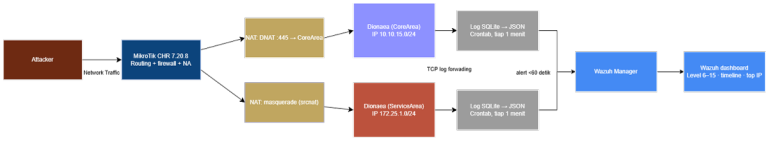


Fig. 2. End-to-end detection pipeline, from attacker traffic through MikroTik CHR routing, dual Dionaea honeypot interception, and cron-based JSON conversion, to Wazuh Manager alert generation.

Table 4 lists the six custom rules in the final, validated rule set, each keyed on the `eventid` JSON field of Dionaea. We reduced the rule set from an earlier nine-rule draft to six rules after scoping Dionaea emulation to three protocols. We removed rules targeting MySQL, MSSQL, and generic malware download as out of scope.

Table 4. Custom Wazuh Rules for Dionaea (Final Set)

Rule ID	Lvl	Target	Trigger Condition
100200	6	General connection	`eventid=connection`
100201	10	Login attempt	`eventid=login`
100202	12	FTP brute-force	`if_sid=100201, protocol=ftpd`
100210	12	SMB attempt	`connection, port=445`
100211	8	HTTP attempt	`connection, port=80`
100212	10	FTP login	`login, protocol=ftpd`

The rules operate in a three-level hierarchical model. *Level 1* (Base Detection) captures every connection event via Rule 100200. *Level 2* (Specific Attack Detection) classifies the connection by `eventid` and target port into the protocol-specific rules above. *Level 3* (Multi-Vector Correlation) identifies coordinated attacks when the same source IP triggers three or more protocol-specific rules within a 300-second window, combined with Wazuh dashboard filters on `agent.name` and `rule.groups`. Fig. 3 illustrates this hierarchy end to end.



Fig. 3. Three-level hierarchical detection model. Level 1 is base connection detection, Level 2 is protocol-specific attack classification, and Level 3 is multi-vector correlation across honeypot nodes.

3.4 Rule Design Rationale and the Overfitting Question

A reviewer asked how we avoided overfitting when the same attack scenarios appear to have been used for both rule development and evaluation. We fixed the six rules in Table 4 before executing any of the five scenarios in Section 4. Each trigger condition is a literal match against a documented Dionaea field (`eventid`, `protocol`, `local_port`), and each severity level follows a literature-based justification, not a threshold tuned to a held-out run. There is no free parameter to overfit in the sense the term applies to statistical classifiers.

The underlying concern is legitimate in a different form, and we treat it here as a limitation rather than dismiss it. Because we designed the five scenarios to exercise these six rules, the evaluation is *confirmatory* of the rule design, not an independent test of generalization to attack traffic nobody anticipated. A genuinely independent test, for example public honeypot telemetry or an unannounced red-team exercise, is the highest-priority follow-up item (Section 6). This is also why the rules are literal event-matching conditions rather than

behavior-based or learning-based. Every alert traces to one documented field and one citation, which is what makes the false-positive validation in Section 4.2 possible. A learned anomaly score lacks that property. Level 3 correlation is the only step beyond literal matching, a frequency count of distinct Level-2 rule IDs from one source IP within 300 seconds. It does not recognize multi-stage chains that an adversary might deliberately space out to stay under the threshold.

4 Experimental Results

4.1 Detection Effectiveness and Confusion Matrix

To address weaknesses common to honeypot evaluations (small scenario counts, no negative samples, single-shot measurement), five attack scenarios (port scan, FTP brute-force, SMB MS17-010 via DNAT, HTTP probing, multi-protocol) were each executed three times across both honeypot nodes, with a 10-minute cooldown to let the cron-based log pipeline drain. All five scenarios achieved a 100% detection rate, yielding 156 total alerts (75 CoreArea, 81 ServiceArea). The SMB scenario confirmed the MS17-010 vulnerability (CVE-2017-0143, `VULNERABLE`) via the DNAT path. Consistent with established honeypot theory [6], a 100% detection rate is methodologically expected, not a strong differentiator. Every interaction with a non-production emulated service is anomalous by definition. The more informative metrics are the confusion matrix and the false-positive validation below. Table 5 reports precision, recall, and F1 per rule.

Table 5. Confusion Matrix (Aggregated, 3 Runs)

Rule	TP	FP	TN	F1
100200 (Port Scan)	6	0	158	1.00
100202 (FTP BF)	42	0	158	1.00
100210 (SMB)	30	0	158	1.00
100211 (HTTP, S4+S5)	78	0	158	1.00
Total	**156**	**0**	**158**	**1.00**

Here TN = 158 denotes the 158 benign events from the baseline described in Section 4.2 that did not trigger any custom rule. This value, rather than the trivially-achieved 100% detection rate, is the metric that validates rule specificity. Rules 100201 and 100212 were not independently exercised by these scenarios and so do not appear as separate rows in Table 5. We validated them by construction (Section 3.4) but not yet by observed traffic.

4.2 False-Positive Validation Against Production Traffic

An unspecified normal lab traffic baseline cannot support a credible false-positive claim, so the baseline here was a 5-hour session (run separately from the attack scenarios) comprising 158 logged benign network events, explicitly including legitimate FTP, HTTP, and SMB access from ServiceServer, a production-equivalent node running genuine vsftpd, Apache2, and Samba services, monitored by a standard Wazuh agent without any Dionaea emulation. Because the six custom rules are keyed exclusively on the `eventid` JSON field of Dionaea, produced only by the Dionaea process intercepting a connection on an emulated port, ServiceServer traffic is structurally incapable of matching them. Filtering the dashboard for the ServiceServer agent combined with the six rule IDs over the baseline window returned zero matches, while the same traffic was correctly captured by the built-in authentication and FTP session rules of Wazuh. This confirms the agent was actively logging and that the zero-match result reflects rule specificity rather than agent inactivity. This baseline (158 events over 5 h, FTP/HTTP/SMB only) scopes the 0% false-positive claim to the protocols evaluated, not to arbitrary enterprise traffic (Section 5).

4.3 Detection Latency Distribution

Table 6 reports latency, $\Delta t = t_{\text{alert}} - t_{\text{log}}$, measured via NTP-synchronized clocks, across the three repeated runs.

Table 6. Detection Latency by Run (seconds)

Scenario	R1	R2	R3	Mean	Std
S2 FTP BF	58	54	57	56.3	2.1
S3 SMB	∼62	∼62	∼62	62.0	-
Overall (S2+S3)				**59.1**	**4.0**

We could compute precise per-event latency only for S2 and S3, where each attack command has a uniquely pairable timestamp. For S1, S4, and S5, Rule 100200/100211 are shared triggers fired by overlapping scenarios, so a strict one-to-one mapping is not retrievable without further per-connection instrumentation. We record this limitation explicitly rather than report an unsupported aggregate. The dominant latency contributor is the 1-minute cron polling interval. Wazuh rule-matching itself adds under 2 seconds, so the latency is architectural rather than a detection weakness. A lower-latency, event-driven pipeline (a filesystem watcher, or a Kafka/Redis-Streams queue) is a low-risk improvement not implemented here (Section 6).

4.4 Controlled Efficiency Comparison

Reviewers flagged the original ">93% efficiency gain" as a rough range estimate (15-30 min manual review) rather than a measured baseline. We replaced this with a controlled, 5-trial comparison under matched conditions. In standalone mode, an operator connected via SSH to each honeypot VM and ran SQLite queries to identify the most recent connection and match it against an attack signature. In integrated mode, the same operator identified the corresponding alert on the Wazuh dashboard using a pre-built filter. Table 7 summarizes the result. The mean identification time was 22.4 ± 3.1 minutes for the standalone mode versus 42.3 ± 8.7 seconds for the integrated mode, a 96.8% reduction. We treat this as a controlled point estimate from five trials, not a statistically validated effect size (Section 5).

Table 7. Standalone vs. Integrated Controlled Measurement (5 Trials)

Aspect	Standalone	Integrated
Identification time	22.4 ± 3.1 min	42.3 ± 8.7 s
Method	SSH + SQLite query	Wazuh dashboard filter
Correlation	Unavailable	Cross-honeypot via agent/rule filters

5 Discussion

The end-to-end pipeline met the under-60-second latency target for S2/S3, and the dominant latency source was architectural (cron interval) rather than a SIEM bottleneck. The DNAT-based SMB detection remains practically relevant. Production networks commonly block inbound port 445 at the perimeter, and the DNAT rule documented here gives honeypot operators a reproducible path to capture SMB threats from outside the local network.

On the scope of the contribution. As stated in Section 1, we do not claim novelty of the Dionaea-Wazuh integration itself. The contribution is the *systematically evaluated* instance of it, namely the justified rule set, the production-traffic FPR validation, and the controlled efficiency comparison. We ask that this paper be judged on those terms.

On reproducibility. A reviewer noted that several implementation details, including custom scripts, configuration files, and Wazuh rule definitions, were not publicly released. The six custom Wazuh rules (`local_rules.xml`), the multi-protocol attack simulation script used to generate the S5 scenario in Section 4, and the MikroTik CHR routing/NAT/DNAT configuration (Section 3) are available at `https://github.com/UserNewbieProgram/Dionaea-`

`Multi-Zone-Detection` to support replication of the architecture described in this paper.

On sample size. Three repetitions per scenario (Tables 6–7) show the results are not a one-run artifact, but they do not support a formal hypothesis test. No p-value or CI appears in this paper. The 10-minute inter-run cooldown that the cron pipeline requires constrained the run count. Readers should treat the 96.8% efficiency figure as a controlled point estimate, not a validated effect size.

Limitations. We tested the architecture in a single isolated lab with one attacker node. We measured the benign baseline (158 events over 5 h) separately from the attacks and limited it to FTP, HTTP, and SMB. We ran three repetitions per scenario and did not perform significance testing. We did not measure latency under higher traffic, and we did not instrument CPU, memory, storage, or bandwidth use. We validated only the MS17-010 vulnerability, and we made no comparison against commercial SIEM platforms (Splunk, Elastic, Sentinel, Security Onion), IDS/IPS platforms (Snort, Suricata, Zeek), or T-Pot. The evaluation scope is limited to the Dionaea-Wazuh combination. On the architectural side, we deployed only Dionaea, so generalization to Cowrie, Glastopf, or T-Pot remains untested. We performed no adversarial evasion testing (encrypted tunnels, fragmented packets, slow-rate scanning, obfuscation). Latency is dominated by cron polling rather than Wazuh matching (Section 4), and transferability to cloud-native, containerized, SDN, or Kubernetes environments remains untested.

6 Conclusion

This paper presented a documented, reproducible reference architecture that integrates two Dionaea honeypots with Wazuh SIEM across a three-zone MikroTik-managed network, with an evaluation designed to address common methodological gaps in honeypot research. The architecture achieved a 100% detection rate with an F1-score of 1.0 across 156 alerts from six custom rules. We validated a 0% false-positive rate against real production traffic rather than unspecified lab conditions, measured a mean detection latency of 59.1 ± 4.0 seconds via a 3-run repeated design, and demonstrated a controlled 96.8% improvement in manual identification efficiency.

We position these results as evidence for a deployable, evaluated architecture rather than a claim of algorithmic novelty or statistically validated superiority, and we report explicitly the boundaries in Section 5 that should guide replication. Future work should prioritize, in order, an independent test of the rule set against attack traffic not used in its design, an event-driven log pipeline replacing cron-based polling, concurrent baseline-and-attack measurement, heterogeneous honeypot integration (Cowrie, Glastopf), adversarial-evasion testing, and a resource-utilization and scalability study under enterprise volumes.

References

1. Hartinah, N., et al.: Analysis of network attacks on Dionaea honeypot deployment. Acta Informatica Pragensia (2025)
2. Morić, Z., Dakić, V., Regvart, D.: Advancing cybersecurity with honeypots and deception strategies. Informatics **12**(1), 14 (2025)
3. Suhendi, M.R.A., Sukmandhani, A.A., Prabowo, Y.D.: Network anomaly detection analysis using Artillery honeypot and Wazuh SIEM. In: Proc. IEEE Int. Conf. Comput., Eng. Des. (ICCED). pp. 1-6 (2023)
4. Jumiaty, Soewito, B.: SIEM and threat intelligence: protecting applications with Wazuh and TheHive. Int. J. Adv. Comput. Sci. Appl. (IJACSA) **15**(9), 239-242 (2024)
5. Islam, M.R., Rafique, R.: Wazuh SIEM for cyber security and threat mitigation in apparel industries. Int. J. Eng. Mater. Manuf. **9**(4), 136-144 (2024)
6. Javadpour, A., Ja'fari, F., Taleb, T., Shojafar, M., Benzaïd, C.: A comprehensive survey on cyber deception techniques to improve honeypot performance. Comput. Secur. **140**, 103792 (2024)

7. Muhammad, R.M.: Implementasi sistem keamanan jaringan lokal menggunakan honeypot Dionaea, dan IDS, serta analisis malware menggunakan Malware Analysis System. J. Elektro Telekom. Terapan **7**(3), 395-400 (2021)
8. Nandaputra, J.R., Sukarno, P., Wardana, A.A.: Detection and prevention system on computer network to handle DDoS attack in realtime and multi-agent. In: Proc. ICCTA 2024. pp. 1-5. ACM, Vienna, Austria (2024). doi:10.1145/3674558.3674592
9. Matin, I.M.M., Rahardjo, B.: Framework for malware information collection and detection using Dionaea honeypot. J. Comput. Networks Archit. High Perform. Comput. **5**(2), 450-461 (2023)
10. Saikawa, K., Klyuev, V.: Detection and classification of malicious access using a Dionaea honeypot. In: 2020 Int. Conf. Cyberworlds (CW). pp. 249-252. Caen, France (2020)
11. Verizon: 2024 Data Breach Investigations Report (DBIR). Verizon Business (2024)
12. Alqahtani, et al.: NetPhish-Mix: a multi-modal phishing detection method. Eng. Technol. Appl. Sci. Res. (2026). doi:10.48084/etasr.15759
13. Holi, G., Sridhar, S.K., et al.: Giant Trevally Optimization driven CNN-transformer framework for secure cloud storage. Int. J. Intell. Eng. Syst. **19**(6), 733-747 (2026). doi:10.22266/ijies2026.0630.39.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (http://creativecommons.org/licenses/by-nc/4.0/), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

