Jurnal Informatika: Jurnal pengembangan IT

Vol. 11, No. 2, 2026
ISSN : 2477-5126, DOI:10.30591/jpit.v11i2.10390

Centralized Orchestration for Agent-Based Host Intrusion Detection System with Threat Intelligence

Gede Ananda [1*], Sawali Wahyu [2], Muhamad Hadi Arfian [3], Nugroho Budhi Santoso [4]
[1, 2, 3, 4] Department of Informatics Engineering, Faculty of Computer Science, Esa Unggul University, Indonesia
[1] gedeananda03@student.esaunggul.ac.id, [2] sawaliwahyu@esaunggul.ac.id, muhamad.arfian@esaunggul.ac.id [3], nugroho.budhisantosa@esaunggul.ac.id [4]

Info Artikel

Riwayat Artikel:

Received 2026-02-14
Revised 2026-04-25
Accepted 2026-05-19

Abstract— *Cyber threats such as malware injection, web shell exploitation, phishing, and lateral movement increasingly challenge host-level security mechanisms. Conventional Host-Based Intrusion Detection Systems (HIDS) are commonly deployed in stand-alone configurations, resulting in limited cross-host visibility, delayed incident response, and lack of integration with external threat intelligence. This study proposes an agent-based HIDS architecture with centralized orchestration and real-time threat intelligence integration to improve detection accuracy and response efficiency. The system is developed using an experimental approach based on the NIST SP 800-61 Rev.2 incident handling framework, covering preparation, detection and analysis, containment and recovery, and post-incident evaluation. Each host deploys a lightweight agent that monitors file system activities, generates cryptographic hash values, and sends artifact metadata to a centralized orchestration server. The server performs parallel validation using external threat intelligence services and executes automated containment actions. Experimental results in a multi-host virtual environment show a False Positive Rate (FPR) of 3.2%, Mean Time to Detect (MTTD) of 4.8 seconds, and Mean Time to Respond (MTTR) of 6.5 seconds, with a 38% improvement compared to manual monitoring. These findings indicate that centralized orchestration combined with threat intelligence integration enhances detection precision, scalability, and incident response effectiveness in HIDS.*

Keywords: *Agent-Based Security; Host-Based Intrusion Detection; Incident Response; Malware Detection; Threat Intelligence.*

Corresponding Author:

Gede Ananda
Email:
gedeananda03@student.esaunggul.ac.id

Abstrak— *Perkembangan ancaman siber seperti injeksi malware, eksploitasi web shell, phishing, dan lateral movement semakin menantang efektivitas keamanan pada tingkat host. Implementasi Host-Based Intrusion Detection System (HIDS) konvensional umumnya masih bersifat stand-alone, sehingga menyebabkan keterbatasan visibilitas lintas host, keterlambatan respons insiden, serta minimnya integrasi dengan threat intelligence eksternal. Penelitian ini mengusulkan arsitektur HIDS berbasis agen dengan orkestrasi terpusat dan integrasi threat intelligence secara real-time untuk meningkatkan akurasi deteksi dan efisiensi respons. Metode penelitian menggunakan pendekatan eksperimental berdasarkan kerangka kerja NIST SP 800-61 Rev.2 yang mencakup tahap persiapan, deteksi dan analisis, penanganan dan pemulihan, serta evaluasi pasca insiden. Setiap host dipasangi agen ringan yang memonitor aktivitas sistem berkas, menghasilkan nilai hash kriptografis, dan mengirimkan metadata artefak ke server orkestrasi pusat. Server melakukan validasi paralel menggunakan layanan threat intelligence eksternal serta mengeksekusi tindakan karantina secara otomatis. Hasil pengujian pada lingkungan virtual multi-host menunjukkan nilai False Positive Rate (FPR) sebesar 3,2%, Mean Time to Detect (MTTD) 4,8 detik, dan Mean Time to Respond (MTTR) 6,5 detik, dengan peningkatan efisiensi sebesar 38% dibandingkan pemantauan manual. Hasil ini menunjukkan bahwa orkestrasi terpusat yang terintegrasi dengan threat intelligence mampu meningkatkan presisi deteksi, skalabilitas, dan efektivitas respons insiden pada HIDS.*

Kata Kunci *: Deteksi Intrusi Berbasis Host; Deteksi Malware; Intelijen Ancaman; Keamanan Berbasis Agen; Respons Insiden.*

I. INTRODUCTION

The rapid digital transformation of organizational infrastructures has fundamentally reshaped the cybersecurity landscape. Modern enterprises increasingly rely on distributed endpoints, virtualized servers, cloud-based services, and interconnected applications to support operational continuity. While these developments improve scalability and efficiency, they also expand the attack surface available to adversaries. Recent reports indicate that a large proportion of successful intrusions originate from compromised endpoints rather than perimeter-level breaches [1], [2]. This shift underscores the strategic importance of host-level monitoring in contemporary security architectures.

Cyberattacks have also become more sophisticated. Threat actors increasingly employ multi-stage techniques such as malware injection, web shell exploitation, phishing-based credential theft, privilege escalation, and lateral movement [3], [4]. These attacks often exploit legitimate system processes and encrypted channels to evade traditional network-based detection mechanisms. As a result, Network-Based Intrusion Detection Systems (NIDS), although still important for traffic monitoring, are insufficient to provide comprehensive protection against endpoint-centric threats

[5]. Host-Based Intrusion Detection Systems (HIDS) therefore remain a relevant defensive layer because they observe file-system activity, process execution, system logs, registry changes, and user behavior at the operating-system level [6], [7].

Despite this advantage, conventional HIDS deployments still face major limitations. Many implementations operate as stand-alone agents on individual hosts without centralized coordination or cross-host correlation [8]. This fragmented architecture reduces visibility across distributed environments and complicates incident analysis when multiple hosts are affected simultaneously. Prior studies show that the absence of centralized orchestration contributes to longer Mean Time to Detect (MTTD) and Mean Time to Respond (MTTR), which in turn increases the operational impact of security incidents [9], [10]. In addition, many HIDS solutions depend on static validation mechanisms such as signature-based scanning or local rule-based checks. Although effective against known threats, these approaches are less reliable for obfuscated malware, polymorphic payloads, and previously unseen attack variants [11]. They also tend to generate false positives, which can overload analysts and reduce response quality [12].

To improve detection performance, several studies have proposed machine learning-based HIDS models, including Support Vector Machines, deep neural networks, Siamese networks, and hybrid ensemble methods [13], [14], [15], [16], [17]. These approaches are effective in improving classification accuracy on curated datasets. However, most of them focus on algorithmic performance in controlled environments and do not sufficiently address architectural scalability, orchestration, or standardized incident response workflows. In other words, better classification does not automatically translate into better operational security outcomes.

Another limitation in current HIDS research is the insufficient integration of external threat intelligence services. Threat intelligence platforms aggregate Indicators of Compromise (IOC), such as malicious hashes, URLs, and other reputation data, which can improve contextual awareness during artifact validation [18]. However, many existing HIDS solutions remain isolated from real-time intelligence feeds and therefore cannot adapt quickly to emerging threats. In parallel, Security Orchestration, Automation, and Response (SOAR) frameworks have been recognized as an effective way to accelerate incident handling, standardize response actions, and reduce manual intervention [19]. Nevertheless, SOAR principles are still rarely implemented directly within host-based intrusion detection architectures.

Evaluation practices in HIDS studies also remain incomplete. Many studies emphasize accuracy-oriented metrics such as precision, recall, and F1-score, but these indicators do not fully represent operational efficiency. Metrics such as False Positive Rate (FPR), Mean Time to Detect (MTTD), and Mean Time to Respond (MTTR) provide a more practical assessment of real-world incident handling performance [20], [21]. However, only a limited number of studies evaluate detection accuracy and response efficiency within the same experimental framework.

Based on these issues, three research gaps can be identified. First, there is a lack of HIDS architectures that combine agent-based host monitoring with centralized orchestration for cross-host coordination. Second, the operational use of external threat intelligence in host-level decision-making remains limited. Third, comprehensive evaluation using incident-response-oriented metrics is still uncommon in architectural HIDS studies.

To address these gaps, this research proposes an agent-based Host-Based Intrusion Detection System with centralized orchestration and integrated threat intelligence validation. Each host deploys a lightweight agent that monitors file-system events and generates cryptographic fingerprints of detected artifacts. These artifacts are transmitted to a centralized orchestration server, where parallel validation is performed using external threat intelligence services. When malicious artifacts are confirmed, automated containment actions are executed in accordance with the NIST SP 800-61 Rev.2 incident handling lifecycle [22].

To clearly position the proposed approach within existing literature, Table 1 presents a comparison with related studies. The comparison highlights that previous works mainly rely on machine learning-based detection without incorporating centralized orchestration, real-time threat intelligence integration, or operational performance evaluation using incident response metrics. In contrast, this study integrates these components into a unified architecture, addressing both detection and response dimensions.

TABLE 1.
COMPARISON WITH RELATED STUDIES

Study	ML-Based	Centralized	Threat Intel	MTTD/MTTR Evaluated	Real Deployment
Memon et al. [13] (2022)	✓	✗	✗	✗	Lab
Park et al. [14] (2021)	✓	✗	✗	✗	Lab
Baz et al. [2] (2022)	✓	✗	✗	✗	IoT
Singh et al. [17] (2024)	✓	✗	✗	✗	Cloud
This Study	✗	✓	✓	✓	Multi-Host

The novelty of this study lies in the integration of four key components: (1) agent-based host monitoring, (2) centralized multi-host orchestration, (3) real-time threat intelligence enrichment, and (4) operational performance evaluation using False Positive Rate (FPR), Mean Time to Detect (MTTD), and Mean Time to Respond (MTTR). Unlike prior studies that predominantly focus on improving detection accuracy through algorithmic approaches, this research emphasizes architectural integration to enhance both detection precision and incident response efficiency.

From a practical perspective, this study provides a deployable architecture for enhancing endpoint security monitoring without relying solely on computationally intensive machine learning models. From an academic perspective, this research contributes to the advancement of host-based intrusion detection by emphasizing system-level orchestration, integration of threat intelligence, and the use of operational performance metrics, rather than focusing exclusively on classification accuracy.

II. METHOD

A. Research Stages

The methodological framework of this study is structured in accordance with the NIST SP 800-61 Rev.2 incident handling lifecycle, which consists of four primary stages: Preparation, Detection and Analysis, Containment and Recovery, and Post-Incident Activity [22].

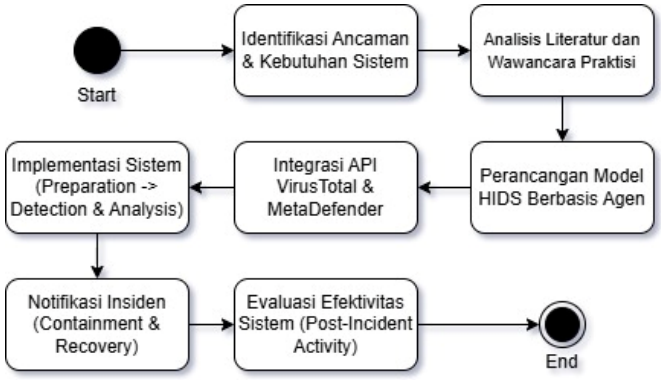

Figure 1. Research Stages Based on NIST SP 800-61 Rev.2 Framework

Figure 1 presents the sequential workflow adopted in this study, starting from threat identification and system requirement analysis, followed by system design, implementation, integration with threat intelligence services, incident handling, and post-incident evaluation. This structured approach ensures that the system design and evaluation process align with standardized incident response practices.

B. Theoretical Framework of the Methodology

1. Preparation Phase

The Preparation phase focuses on defining system requirements, identifying threat models, and designing the system architecture. According to NIST SP 800-61 Rev.2 [22], effective incident handling begins with proper preparation, including monitoring infrastructure, communication mechanisms, and response procedures.

The primary threats considered in this study include malicious file uploads such as web shell injection, disguised executable malware artifacts, phishing-related URL artifacts stored locally, and unauthorized modification of critical system directories. These attack scenarios are selected based on common endpoint exploitation patterns reported in recent cybersecurity studies [3], [15].

An agent-based monitoring architecture is adopted to enable lightweight telemetry collection at the host level. Unlike conventional stand-alone HIDS deployments [8], the proposed system integrates centralized orchestration to

provide cross-host visibility and coordinated response capabilities. To evaluate system performance, operational metrics—False Positive Rate (FPR), Mean Time to Detect (MTTD), and Mean Time to Respond (MTTR)—are defined in this phase. These metrics ensure that evaluation is not limited to detection accuracy but also reflects real-world incident response efficiency [22], [23].

2. Detection and Analysis Phase

The Detection and Analysis phase is responsible for identifying suspicious activities and validating detected artifacts. The monitoring mechanism operates using an event-driven file system observation model. When a file creation or modification event occurs, the host-level agent computes cryptographic hash values using MD5 and SHA-256 to uniquely identify the artifact [18].

The generated metadata, including file path, timestamp, file size, host identifier, and hash values, are transmitted securely to the centralized orchestration server via RESTful communication. Rather than performing classification locally, artifact validation is delegated to the centralized server to optimize endpoint resource utilization. The server performs parallel validation by querying external threat intelligence platforms, which provide reputation data for submitted hashes and URLs [21], [24]. This approach enhances detection confidence and reduces uncertainty compared to static rule-based methods.

3. Containment, Eradication, and Recovery Phase

Once an artifact is classified as malicious, the system automatically initiates containment procedures. The centralized orchestration server sends remote execution commands to the corresponding host agent. These commands instruct the agent to move the malicious artifact into a secure quarantine directory, revoke execution permissions, record incident metadata, and notify administrators.

This automated containment mechanism follows the principles of Security Orchestration, Automation, and Response (SOAR), which aim to minimize manual intervention and accelerate incident handling processes [19]. By eliminating manual delays, the system reduces Mean Time to Respond (MTTR) and limits potential lateral movement risks [3]. Recovery procedures ensure that affected directories are restored to a secure state and that no residual malicious artifacts remain. All containment and recovery activities are recorded in centralized logs to support subsequent forensic analysis.

4. Post-Incident Activity Phase

The Post-Incident Activity phase evaluates detection accuracy and operational performance. In accordance with incident response best practices [22], this phase emphasizes quantitative performance measurement. The False Positive Rate (FPR) is calculated as:

$$FPR \frac{FP}{(FP + TN)}$$

where FP represents false positives and TN represents true negatives.

Mean Time to Detect (MTTD) is defined as the average time between file event occurrence and malicious classification confirmation. Mean Time to Respond (MTTR) represents the average time between detection confirmation and successful execution of containment actions. These metrics provide a comprehensive evaluation of both detection reliability and response efficiency, addressing limitations in prior studies that focus solely on classification accuracy [14], [17].

C. System Architecture Design

The proposed agent-based Host Intrusion Detection System architecture with centralized orchestration is illustrated in Figure 2.

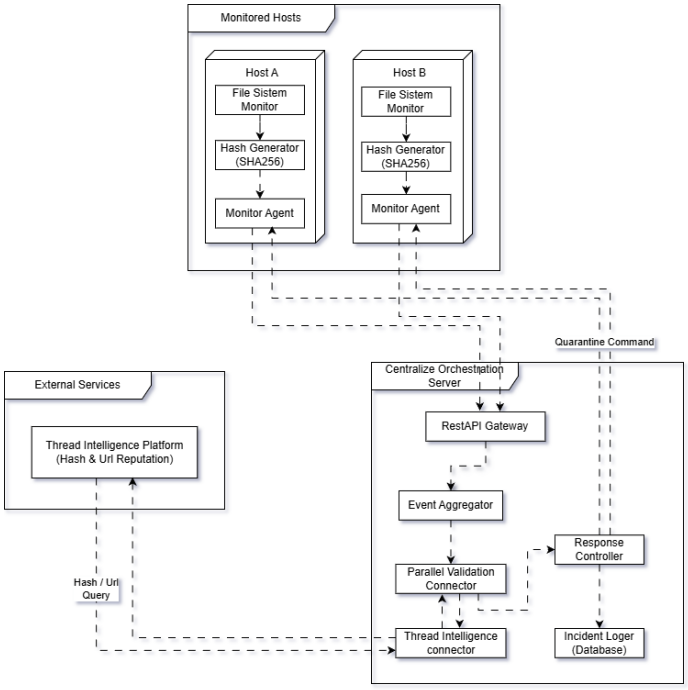

Figure 2. Proposed Agent-Based HIDS Architecture with Centralized Orchestration

Figure 2 shows that each monitored host is equipped with a file system monitoring module and a hash generation component. The monitoring agent detects file system events and generates cryptographic fingerprints, which are transmitted to the centralized orchestration server. The server processes incoming data using multiple modules, including REST API Gateway, Event Aggregator, Parallel Validation Connector, Threat Intelligence Connector, Response Controller, and Incident Logger. External threat intelligence services are used to validate artifact reputation based on global intelligence data[21], [24]. This architecture enables cross-host coordination, asynchronous validation, and automated response execution, thereby improving detection speed and operational efficiency compared to conventional stand-alone HIDS implementations [8], [19].

D. Experimental Environment

The evaluation environment was designed to simulate a distributed enterprise infrastructure. Multiple virtual machines were deployed to represent independent hosts, each running the monitoring agent. Each host was configured with a dual-core virtual CPU, 4 GB RAM, and 40 GB storage capacity. The centralized orchestration server was configured with a quad-core virtual CPU and 8 GB RAM to support concurrent validation processing. A total of 50 test artifacts were used in the experiment, consisting of 25 malware samples and 25 benign files. This dataset was used to evaluate detection accuracy and operational performance under controlled multi-host conditions.

III. RESULTS

The results of this study are presented according to the four stages defined in the NIST SP 800-61 Rev.2 incident handling lifecycle to ensure alignment between system design, implementation, and evaluation.

A. Preparation Stage Results

The preparation stage aimed to validate system deployment stability and ensure reliable communication between host-level monitoring agents and the centralized orchestration server. Each monitoring agent was successfully deployed across multiple virtual hosts and registered through secure HTTPS-based communication. Continuous heartbeat synchronization confirmed stable connectivity between hosts and the orchestration server.

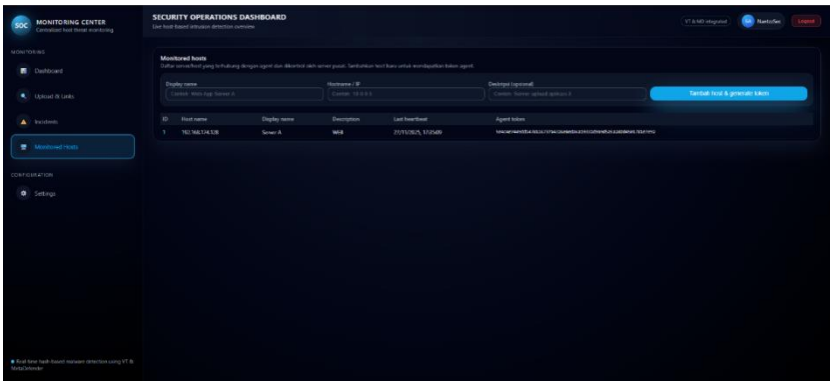

Figure 3. Centralized Orchestration Dashboard of Monitored Hosts

Figure 3 presents the centralized dashboard displaying registered hosts along with their operational status, including host identity, IP address, and activity timestamps. This centralized visibility overcomes the limitations of stand-alone HIDS deployments, which typically lack cross-host monitoring capability [8].

A 24-hour stability test showed no communication failure between agents and the server. In addition, the average CPU utilization remained below 5%, indicating that the monitoring agent introduces minimal performance overhead. These results confirm that the system is stable and suitable for continuous deployment in a distributed environment.

B. Detection and Analysis Result

To evaluate detection capability under realistic attack conditions, a controlled experiment was conducted using 50 artifacts consisting of 25 malware samples and 25 benign files. The attack scenarios included web shell uploads, disguised executable files, and script-based exploitation attempts.

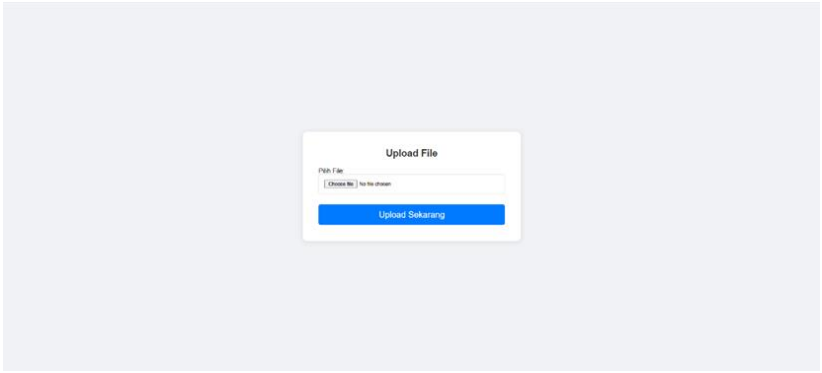

Figure 4. Web Shell Upload Simulation on Target Host

Figure 4 illustrates a web shell upload scenario used to simulate a common endpoint exploitation technique. The uploaded malicious file triggered the monitoring agent, which immediately detected the file creation event through an event-driven mechanism.

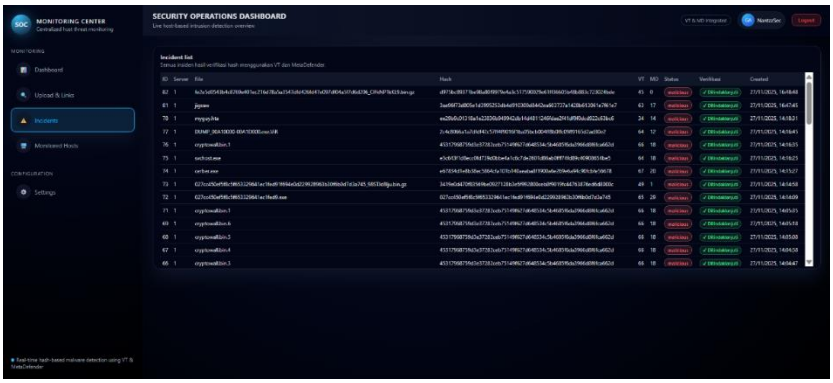

Figure 5. Real-Time Detection Log on Monitoring Agent

Figure 5 shows the detection log generated after the file upload. The log records file events, cryptographic hash values, and metadata transmission status. This confirms that the system captures events in real time without relying on periodic scanning, which significantly reduces detection latency [6].

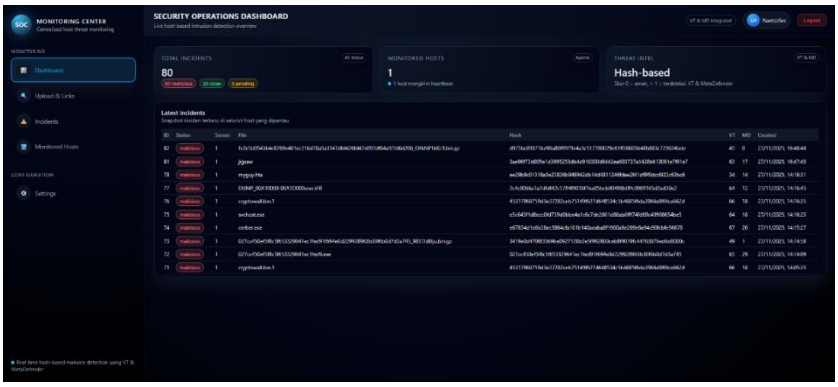

Figure 6. Threat Intelligence Validation Response

Figure 6 presents the validation results obtained from external threat intelligence services. Each artifact is classified based on global reputation data. This integration ensures that detection decisions are not solely based on local analysis but are enriched with external intelligence, improving classification reliability [24].

1. *Detection Accuracy Evaluation*

A total of 50 artifacts were tested. The detection results are summarized in Table 2.

TABEL 2
DETECTION ACCURACY METRICS

Parameter	Value
True Positive	25
False Positive	1
True Negative	24
FPR	3.2%

Table 2 explain the system successfully detected all malware samples. One benign file was incorrectly classified due to mislabeling in external threat intelligence data. The False Positive Rate (FPR) of 3.2% indicates a relatively low misclassification rate, which helps reduce alert fatigue for security analysts [12].

However, it should be noted that the system relies on external threat intelligence. Therefore, classification accuracy may be affected by incorrect or outdated reputation data. This dependency introduces a potential limitation in detection reliability. The False Positive Rate (FPR) is calculated using:

$$FPR \frac{FP}{(FP + TN)}$$

where FP represents the number of benign files incorrectly classified as malicious, and TN represents the number of correctly identified benign files. Based on the experimental results:

$$FPR = \frac{1}{1 + 24} = 3.2\%$$

This result indicates that only a small proportion of legitimate files were misclassified. A low FPR confirms that the integration of threat intelligence enrichment improves classification reliability while minimizing unnecessary alerts [12].

2. *Mean Time to Detect (MTTD)*

The average Mean Time to Detect (MTTD) recorded was 4.8 seconds. This time includes event detection, hash generation, metadata transmission, and threat intelligence validation.

The low MTTD value is attributed to the event-driven monitoring mechanism and centralized validation architecture. Compared to conventional stand-alone HIDS, which rely on periodic scanning, the proposed system significantly reduces detection delay [9].

C. Containment, Eradication, and Recovery Stage

Once a malicious artifact is confirmed, the system automatically executes containment procedures.

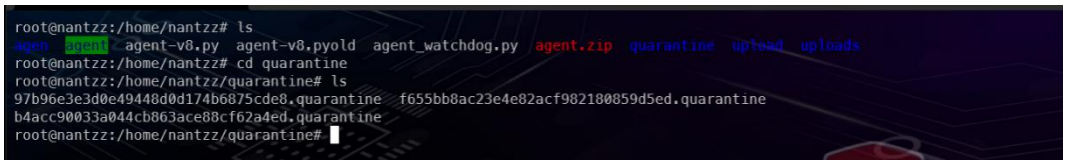

Figure 7. Automated Quarantine Execution on Host

Figure 7 shows that the malicious artifact is immediately moved to a quarantine directory and execution permissions are revoked. This automated response reduces the time required to mitigate threats.

1. Response Efficiency Evaluation

TABLE 3
RESPONSE EFFICIENCY METRICS

Metric	Result
Mean Time to Detect (MTTD)	4.8 sec
Mean Time to Respond (MTTR)	6.5 sec
Quarantine Time	2.1 sec
Improvement vs Manual	38%

Table 3 explain about manual baseline monitoring required approximately 10.5 seconds to complete containment under identical conditions. The proposed system reduced response time to 6.5 seconds, representing a 38% improvement in overall incident handling efficiency.This improvement confirms that centralized orchestration and automated containment significantly enhance operational performance [25].

D. Post-Incident Evaluation

Post-incident analysis assessed system behavior under concurrent event scenarios. Simultaneous file uploads were generated across multiple hosts to evaluate scalability. The centralized asynchronous validation engine maintained stable throughput without significant latency increase. No detection failures or data loss were observed during concurrent processing.

These findings demonstrate that the proposed architecture supports distributed deployment and can scale horizontally by allocating additional orchestration server resources. Overall, the system achieves reliable detection accuracy, rapid automated containment, and stable performance in a multi-host environment.

E. Discussion

This section analyzes the significance of the experimental findings and positions the contribution of this study within the broader context of host-based intrusion detection research. Recent studies on HIDS predominantly focus on improving detection accuracy through machine learning-based classification models[13], [17]. Although these approaches often achieve high accuracy in controlled environments, they rarely evaluate operational metrics such as Mean Time to Detect (MTTD) and Mean Time to Respond (MTTR). Consequently, improvements in classification performance do not necessarily translate into faster incident containment in real-world environments.

The results of this study demonstrate that architectural orchestration plays a critical role in improving operational security performance. By integrating lightweight host-level monitoring, centralized multi-host coordination, real-time threat intelligence enrichment, and automated containment aligned with the NIST SP 800-61 framework [22], the proposed system achieved a 38% reduction in incident handling time while maintaining a low False Positive Rate (FPR) of 3.2%. These findings indicate that system-level coordination contributes directly to measurable improvements in both detection responsiveness and response efficiency. Rate of 3.2%. These results indicate that system-level coordination contributes directly to measurable improvements in detection and response efficiency.

Compared to conventional stand-alone HIDS, the proposed architecture provides enhanced cross-host visibility and coordinated response capabilities. Unlike machine learning-based HIDS approaches that prioritize classification accuracy, this study emphasizes operational performance through reduced detection latency and automated response execution. Therefore, rather than replacing machine learning approaches, the proposed system complements them by addressing architectural and operational limitations.

From an operational perspective, centralized orchestration improves situational awareness, enables coordinated automated containment, and reduces analyst workload. These characteristics align with Security Operations Center (SOC) practices and the principles of Security Orchestration, Automation, and Response (SOAR) [19]. By minimizing

manual intervention and standardizing response workflows, the system significantly reduces response latency and improves consistency in incident handling.

Despite these advantages, several limitations must be acknowledged. First, the system depends on external threat intelligence services for artifact validation. As observed in the experimental results, incorrect or outdated reputation data may lead to misclassification, including potential false positives or false negatives. Second, the centralized architecture introduces a potential scalability constraint, as system performance may degrade under high-volume concurrent events if orchestration resources are limited. Third, the relatively small dataset (50 artifacts) limits the generalizability of the findings and may not fully represent large-scale enterprise environments.

These limitations suggest that future improvements should focus on enhancing system robustness and scalability. Potential directions include integrating local threat intelligence caching mechanisms to reduce dependency on external services, incorporating behavioral and process-level monitoring to detect advanced or fileless attacks, and evaluating the system in large-scale production environments. Additionally, combining architectural orchestration with adaptive machine learning models may further improve detection accuracy while maintaining operational efficiency. Overall, this study demonstrates that architectural integration rather than algorithmic complexity alone is a critical factor in improving the effectiveness of host-based intrusion detection systems.

IV. CONCLUSION

This study proposes and evaluates an agent-based Host-Based Intrusion Detection System (HIDS) with centralized orchestration and integrated threat intelligence to address the limitations of conventional stand-alone HIDS architectures. Experimental results demonstrate that the proposed system achieves a False Positive Rate (FPR) of 3.2%, a Mean Time to Detect (MTTD) of 4.8 seconds, and a Mean Time to Respond (MTTR) of 6.5 seconds. Compared to manual monitoring, the system reduces overall incident handling time by 38%, indicating improved detection responsiveness and response efficiency. The main contribution of this research lies in the integration of agent-based monitoring, centralized orchestration, real-time threat intelligence, and operational performance evaluation. Unlike prior studies that primarily emphasize machine learning-based detection accuracy, this work highlights the importance of architectural coordination and automation in improving real-world cybersecurity operations. However, this study has several limitations, including dependency on external threat intelligence services, limited dataset size, and evaluation within a controlled environment. These constraints may affect system scalability and generalizability. Future research should focus on large-scale deployment, integration with behavioral detection mechanisms, and hybrid approaches that combine architectural orchestration with adaptive machine learning models to further enhance system robustness and detection capability.

ACKNOWLEDGMENT

The authors would like to acknowledge Esa Unggul University for providing research facilities and academic support for this study. The authors also thank the supervisor and contributors who provided valuable technical insights during the research process.

REFERENCES

[1] European Union Agency for Cybersecurity (ENISA), "ENISA Threat Landscape 2024," 2024.
[2] M. Baz, "SEHIDS: Self-Evolving Host-Based Intrusion Detection System for IoT Networks," *Journal of Network and Computer Applications*, vol. 198, 2022.
[3] M. Smiliotopoulos, G. Kambourakis, and S. Gritzalis, "Lateral Movement Detection: A Survey," *Comput. Secur.*, vol. 127, 2023.
[4] Y. Dong and Y. Li, "PHP Web Shell Detection Based on Abstract Syntax Tree and Deep Forest," *Journal of Information Security and Applications*, vol. 75, 2024.
[5] G. Khraisat and A. Alazab, "A Critical Review of Intrusion Detection Systems in IoT Environment," *IEEE Internet Things J.*, vol. 8, no. 6, 2021.
[6] S. Thakkar and R. Lohiya, "A Review on Intrusion Detection System for Network Security," *Int. J. Comput. Appl.*, vol. 174, no. 7, 2022.
[7] A. Satılmış, S. Akleylek, and G. Yüce Tok, "Lightweight Agent-Based Intrusion Detection Architecture," *Security and Communication Networks*, 2023.
[8] A. Nuswantoro, R. Prasetyo, and M. Hidayat, "Privilege Escalation Detection Using Integrated HIDS and SIEM," *Indonesian Journal of Computing and Cybernetics Systems*, vol. 18, no. 2, 2024.
[9] M. Mollahosseini, S. Homayoun, and M. Dehghantanha, "Reducing Incident Response Time Using Automated Security Orchestration," *Future Generation Computer Systems*, vol. 129, 2022.
[10] A. Al-Sabaawi, "Performance Analysis of Cryptographic Hash Algorithms," *International Journal of Information Security Science*, vol. 9, no. 2, 2020.
[11] M. Mat, N. Rahman, and Z. Ismail, "Advanced Persistent Threat Detection: A Systematic Review," *ACM Comput. Surv.*, vol. 56, no. 3, 2024.
[12] A. Sharma, S. Gupta, and R. Kumar, "A Critical Review of Intrusion Detection Systems in IoT," *Journal of Information Security and Applications*, vol. 71, 2023.
[13] S. Memon, M. U. Khan, and A. Abbas, "Host-Based Intrusion Detection System Using Support Vector Machine," *Appl. Soft Comput.*, vol. 112, 2022.
[14] D. Park, J. Kim, and S. Lee, "Host-Based Intrusion Detection Model Using Siamese Network," *IEEE Access*, vol. 9, 2021.

[15] Y. Pang, H. Wang, and J. Liu, "Web Shell Detection Using GAN-Based Data Augmentation," *Comput. Secur.*, vol. 114, 2022.
[16] A. Razaq, M. Rehman, and S. Khan, "Hybrid Ensemble Learning Model for IoT Intrusion Detection," *Sensors*, vol. 22, no. 4, 2022.
[17] S. G. et al. Singh, "Machine Learning for Cloud-Based Privilege Escalation Attack Detection," *IEEE Transactions on Cloud Computing*, vol. 12, no. 1, 2024.
[18] L. Lima, R. Souza, and P. Fernandes, "Distributed Host-Based Intrusion Detection in Containerized Environments," *Future Internet*, vol. 16, no. 1, 2024.
[19] P. Krishnapriya and R. Singh, "APT Detection Challenges in Cloud and Fog Environments," *Journal of Cloud Computing*, vol. 13, no. 2, 2024.
[20] A. Pinto, R. Pires, and J. Barbosa, "Survey on Intrusion Detection Systems Based on Machine Learning Techniques for the Protection of Critical Infrastructure," *Comput. Secur.*, vol. 123, 2023.
[21] M. Rehman and S. Manickam, "SHA-256-Based Malware Detection Approaches," *Procedia Comput. Sci.*, vol. 132, 2018.
[22] National Institute of Standards and Technology, "Computer Security Incident Handling Guide (SP 800-61 Rev.2)," 2020.
[23] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 8th ed. Pearson, 2018.
[24] A. et al. Ahmad, "Security Orchestration and Automated Incident Response: State-of-the-Art Review," *IEEE Access*, 2023.
[25] S. Zhao, H. Wu, and Q. Zhang, "Automated Threat Intelligence Integration for Endpoint Protection," *IEEE Trans. Dependable Secure Comput.*, 2025.