ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

ADAPTIVE INTRUSION DETECTION AND AUTOMATED CRYPTOGRAPHIC RESPONSE FRAMEWORK FOR REAL-TIME CYBER THREAT MITIGATION

[1]Jamaluddin✉, [1]Darwis Robinson Manalu, [2]Syahril Efendi, [3]Jonson Manurung, [1]Roni Jhonson Simamora

[1]Informatics, Universitas Methodist Indonesia, Medan, Indonesia
[2]Computer Science, Universitas Sumatera Utara, Medan, Indonesia
[3]Informatics, Universitas Pertahanan Republik Indonesia, Bogor, Indonesia
Email: jamaluddin@methodist.ac.id

DOI: https://doi.org/10.46880/jmika.Vol10No2.pp863-876

ABSTRACT

The increasing dynamics of cyberattacks require security systems that can not only detect threats but also select proportionate responses with minimal computational overhead. This study proposes the Adaptive Intrusion Detection and Automated Cryptographic Response (AID-ACR) Framework, a closed-loop prototype integrating intrusion detection, stateful threat-risk assessment, response selection, cryptographic mechanisms, and feedback. In this prototype, the term adaptive refers to updates in risk and response state based on threat frequency and session context; the IDS model itself is fixed during each run and is not retrained online. The prototype was evaluated through a reproducible simulation study using 32,000 synthetic CICIoT-like samples with 39 numerical features and eight classes. Across three repeated stratified runs (seeds 42, 43, and 44), LightGBM achieved a mean accuracy of 98.50% ± 0.25% and a mean Macro-F1 of 97.25% ± 0.62%, marginally higher than the DNN mean Macro-F1 of 97.23% ± 0.51%. Validation selected a conservative threshold configuration of 0.22/0.42/0.62/0.82, which achieved 95.40% risk-level agreement, 95.88% response-selection policy agreement, and a 0.418% Unnecessary Response Rate (URR). Relative to static response (0.9749% URR), the revised policy reduced unnecessary active responses by 57.14%. The measured 0.659 ms per event is reported as end-to-end computational prototype latency because response execution was locally emulated and excluded firewall/SDN, KMS/HSM, identity-provider, network round-trip, and physical-device delays. Actual attack-mitigation effectiveness was not measured; the reported response metric quantifies agreement with the reference simulation policy. These findings support the feasibility of integrating detection evidence, stateful risk assessment, and contextual network/cryptographic response in a measurable prototype pipeline.

Keyword: Adaptive Intrusion Detection, Automated Response, Cryptographic Response, Cybersecurity, Threat Risk Assessment.

INTRODUCTION

Digital transformation through the Internet of Things (IoT), cloud services, cyber-physical systems, and distributed networks has expanded organizational capacity to collect and process data while simultaneously increasing the attack surface. Heterogeneous devices, diverse communication protocols, resource constraints at edge devices, and high traffic volumes create new challenges for network protection. Under these conditions, static preventive mechanisms such as firewalls, access control, and encryption remain necessary but are insufficient to respond to rapidly changing threats. Security systems must also be able to recognize abnormal behavior, assess threat context, and determine mitigation actions within a short time interval (Kikissagbe & Adda, 2024; Komal & Li, 2026).

An Intrusion Detection System (IDS) is an important component for identifying activities that deviate from normal traffic patterns. Advances in machine learning and deep learning have improved the ability of IDSs to recognize complex attack patterns, including DDoS, DoS, spoofing, reconnaissance, brute-force, botnet, and web-based attacks. Tree-ensemble and gradient-boosting models are effective for tabular network-traffic data, while deep-learning models can learn nonlinear relationships at large scale. Numerous studies using modern IoT datasets report very high classification performance; however, classification performance alone does not guarantee operational readiness because latency, class imbalance, false positives, adaptability, and mitigation mechanisms must also be considered (Yaras & Dener, 2024; Hizal et al., 2024; Adewole et al., 2025).

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

A major limitation of many conventional IDSs is that their workflow ends with Detect-Classify-Alert. After an alert is generated, an administrator must still assess the threat level and select an action, such as blocking a source address, restricting a session, isolating a device, or replacing keying material. This manual stage increases the response gap, namely the interval between threat detection and actual mitigation. Sallam et al. (2026), in a review of 32 IoT-IDS proposals, reported that most studies still relied on offline evaluation, mitigation capability was not yet a common component, and operational adaptability remained limited. These findings reinforce the need to move from detection-centric IDSs toward security systems that integrate detection and mitigation.

Security Orchestration, Automation, and Response (SOAR) provide an approach for reducing dependence on manual intervention by automating actions based on rules, policies, or threat information. Integrating artificial intelligence into SOAR can accelerate analysis and response, although response policies in many implementations still focus on administrative and network actions such as blocking, quarantine, firewall-rule modification, or account locking (Gustina DM & Ananda, 2024; Waelchli & Walter, 2025). In contrast, when an attack indicates possible compromise of identity, a session, or keying material, retaining the same cryptographic context may prolong exposure. Session re-keying, key rotation, key revocation, and secure session re-establishment are therefore relevant as contextual response actions.

NIST SP 800-61 Rev. 3 positions incident response as an integrated component of cybersecurity risk management rather than as a separate process performed only after detection (Nelson et al., 2025). At the cryptographic layer, NIST SP 800-57 Part 1 Rev. 5 emphasizes management of keying material throughout its life cycle, including circumstances in which compromise is suspected (Barker, 2020). These principles allow IDS outputs to be used not only to trigger alerts but also to modify the status and security context of a session as risk increases.

Previous work by Jamaluddin et al. (2025) integrated cryptographic mechanisms and IDS as two complementary protection layers. However, that integration left room for further development in threat-risk scoring, threat-to-action mapping, automated cryptographic response, and temporal feedback. Based on this gap, this study proposes the Adaptive Intrusion Detection and Automated Cryptographic Response (AID-ACR) Framework. The main contributions are: (1) using the IDS as a decision sensor that provides attack probabilities and class labels; (2) dynamic threat-risk scoring based on probability, severity, confidence, and frequency; (3) context-aware automated response that combines network controls and cryptographic actions; and (4) end-to-end evaluation incorporating detection latency, response latency, and Unnecessary Response Rate (URR).

LITERATURE REVIEW

Machine Learning- and Deep Learning-Based Intrusion Detection

Anomaly-based IDSs increasingly employ machine learning because it can learn the separation between normal and malicious traffic without relying entirely on signatures. Kikissagbe and Adda (2024) showed that supervised learning, unsupervised learning, deep learning, and hybrid models have been widely applied in IoT environments, although generalization, computational requirements, feature selection, and real-time evaluation remain challenging. Newer datasets, including CICIoT2023, aim to represent large-scale IoT attack scenarios and are therefore relevant for evaluating modern models (Neto et al., 2023).

Yaras and Dener (2024) combined 1D-CNN and LSTM for IoT traffic classification and demonstrated that deep learning can achieve high accuracy on CICIoT2023 and TON_IoT. Hizal et al. (2024) employed a two-level deep-learning architecture to detect and classify DDoS attacks. Gheni and Al-Yaseen (2024) reduced the complexity of CICIoT2023 through two-step clustering and dimensionality-reduction optimization. These approaches improved the detection stage, but mitigation was not a primary component of their models.

Adewole et al. (2025) extended IDS functionality through rule induction to improve explainability. Meanwhile, Chandnani et al. (2025) developed federated multi-layered deep learning to keep data distributed and improve generalization across several IoT datasets. These developments indicate that IDS research is moving toward explainability and privacy-preserving learning, although model outputs still predominantly consist of predictions or explanations of predictions.

Adaptive and Real-Time Intrusion Detection

The term adaptive in IDS research may refer to model updating, incremental learning, threshold adjustment, class-weight modification, or online learning. Cerasuolo et al. (2025a) showed that class-incremental learning can help a network IDS adapt to new attack classes without completely losing knowledge of previously learned classes. Another

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

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

study from the same group showed that performance can decline substantially when the network domain changes and can be improved through incremental adaptation (Cerasuolo et al., 2025b). These findings demonstrate that adaptability concerns not only accuracy but also the ability to maintain performance as the traffic distribution changes.

In addition to model adaptation, latency is important in real-time settings. Almahaqeri et al. (2026) evaluated gradient-boosting models in an eight-class CICIoT2023 scenario and reported that XGBoost achieved an accuracy of 0.9959 and a Macro-F1 of 0.8903, while feature reduction substantially lowered inference time with only minor changes in accuracy. Riandari et al. (2024) demonstrated that dynamic optimization combined with adaptive control mechanisms can improve network resilience against distributed cyberattacks by adjusting defensive strategies in real time. Although their study focused on blockchain networks, the findings reinforce the importance of adaptive security mechanisms capable of modifying defensive actions according to evolving threat conditions. These results confirm that highly accurate models must also be assessed in terms of computational cost. Recent reviews likewise emphasize the need for IDSs that are lightweight, scalable, adaptive, and validated in realistic environments (Sallam et al., 2026; Komal & Li, 2026).

Automated Response and Cryptographic Adaptation

Automated response converts detection outputs into security actions without requiring the entire decision process to be performed manually. SOAR combines orchestration, automation, and response so that routine tasks can be executed according to playbooks or policies. In this study, automated response does not mean that every attack is treated identically. Availability-oriented threats such as DDoS require rate limiting, blocking, or isolation, whereas threats involving sessions or authentication may require re-keying, key rotation, re-authentication, or revocation.

AES-GCM was selected for the simulated data channel because it provides authenticated encryption, combining confidentiality with message authentication and integrity (Dworkin, 2007). Management of keying material follows the principles of NIST SP 800-57, particularly the requirement that a key suspected of compromise should not continue to protect new data (Barker, 2020). Accordingly, cryptographic response in AID-ACR functions as one context-dependent mitigation mechanism rather than as a replacement for network-defense controls.

Table 1. Summary of Previous Studies and the Research Gap

Study	Data	Focus	Gap/Position
Yaras & Dener (2024)	CICIoT2023, TON_IoT	Hybrid 1D-CNN-LSTM	Detection-focused; no adaptive response
Hizal et al. (2024)	CICIoT2023	Two-level DL IDS	Classification does not trigger mitigation
Gheni & Al-Yaseen (2024)	CICIoT2023	Clustering + dimensionality reduction	Focuses on preprocessing efficiency
Adewole et al. (2025)	CICIoT2023	Ensemble + rule induction	Explainable, but no automated response
Cerasuolo et al. (2025a)	4 IoT datasets	Class-incremental learning	Model adaptation, not cryptographic response
Jamaluddin et al. (2025)	Framework	Cryptography + IDS	No dynamic risk or closed-loop response
Almahaqeri et al. (2026)	CICIoT2023	Optimized gradient boosting	Optimization ends at the detection stage
AID-ACR	Simulation study	Adaptive IDS + risk + response	Addresses the detection-to-mitigation gap

Table 1 identifies three primary gaps. First, the detection-response gap arises because most IDSs stop at classification or alert generation. Second, the adaptation gap arises because security decisions often do not consider threat intensity and history. Third, the cryptographic-response gap exists because key-management mechanisms generally operate as policies separate from IDS outputs. AID-ACR is positioned to connect these three areas through a closed loop of detection, assessment, response, feedback, and adaptation.

Conceptual Positioning of AID-ACR

Conceptually, AID-ACR differs from three common groups of approaches. The first is the pure detection system, which focuses on improving accuracy, F1-score, or zero-day detection. The second is the adaptive learning system, which updates a model when the traffic distribution changes. The third is the automated response platform, which executes playbooks based on alerts. AID-ACR lies between these groups by treating IDS output as evidence rather than as the final decision and then calculating a risk score that determines response intensity. This positioning allows the detection model to be replaced without redesigning the overall response policy, provided that class probabilities and labels remain available.

The architecture also avoids the assumption that higher confidence should always trigger a more aggressive response. Confidence is only one component of risk. An event with high confidence but

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

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

low severity may remain at the Medium level, whereas a sequence of events with moderate confidence may escalate to High because of the frequency factor. Thus, the risk score acts as a mediation layer between machine-learning output and operational security decisions. This mediation is important for reducing automation bias, namely the tendency to treat every model prediction as a direct command for action.

From a key-management perspective, AID-ACR introduces a threat-triggered key-lifecycle principle. Key rotation is not performed only according to a schedule or cryptoperiod; it may also be triggered when security evidence indicates increasing session risk. Nevertheless, the mechanism remains subject to sound key-management principles: if keying material is suspected of compromise, a new key should not be derived solely from an old key that may have been exposed. Secure session re-establishment therefore uses fresh key agreement and a new derivation context.

The closed-loop design of AID-ACR does not imply unrestricted self-modification. In the present prototype, adaptivity is located in the threat-risk and response layers: frequency state, session-trust state, and escalation context are updated as events arrive, which can change the selected response for subsequent events. The IDS model parameters, feature representation, and learned decision function remain fixed during each experimental run; no online or incremental retraining is performed. Thus, the term adaptive in this study should be interpreted as stateful risk and response adaptation rather than online adaptive learning. Future work can combine this layer with class-incremental or concept-drift-aware IDS updating.

RESEARCH METHOD

AID-ACR Framework Design

This study uses an experimental simulation-study approach to evaluate the feasibility of the AID-ACR concept. Because the official CICIoT2023 files were not executed in the computational environment used for this study, the experimental data are not claimed to represent a full CICIoT2023 evaluation. Instead, 32,000 synthetic CICIoT-like samples were generated with 39 numerical features, eight classes, an imbalanced distribution, and class-separation characteristics designed to simulate IoT traffic. CICIoT2023 serves as the reference for attack categories and benchmark context, while the synthetic data are used to test the complete end-to-end pipeline reproducibly.

The AID-ACR architecture comprises the Traffic and Preprocessing Layer, Intrusion Detection Layer, Adaptive Threat Assessment Layer, Response Decision Layer, Network and Cryptographic Response Layer, and Feedback and Adaptation Layer. IDS output does not directly become the final action; instead, it is forwarded as evidence to the risk-assessment mechanism. The architecture is shown in Figure 1.

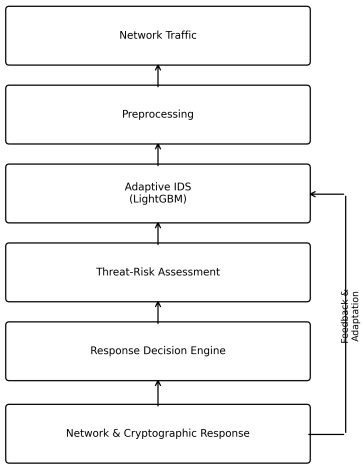


Figure 1. Architecture of the Adaptive Intrusion Detection and Automated Cryptographic Response (AID-ACR) Framework

Data Generation and Preprocessing

The simulation uses eight labels: Benign, DDoS, DoS, Recon, Web, Brute Force, Spoofing, and Mirai. Synthetic samples were generated with scikit-learns make classification using 32,000 observations and 39 numerical features: 27 informative features, 8 redundant features, and 4 non-informative features. The generator used eight classes, one cluster per class, class_sep = 2.10, flip_y = 0.008, and class weights of 0.15 (Benign), 0.40 (DDoS), 0.15 (DoS), 0.07 (Recon), 0.04 (Web), 0.02 (Brute Force), 0.07 (Spoofing), and 0.10 (Mirai). The intentionally imbalanced distribution tests whether performance remains stable for minority classes. Data are stratified into 70% training, 15% validation, and 15% testing sets. The reference detailed analysis uses seed 42, while repeated stratified runs use seeds 42, 43, and 44 to quantify model variability.

Transformation parameters are estimated only from the training data to prevent data leakage. For algorithms requiring scaling, standardization is performed using training-set parameters. Class imbalance is handled through class weighting. The validation set is used to select the threat-risk scoring configuration and thresholds, while the test set is reserved exclusively for final evaluation.

Intrusion Detection Engine

Four models are compared: Random Forest, XGBoost, LightGBM, and a DNN based on a

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

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

multilayer perceptron. Random Forest uses 70 estimators with balanced subsampling; XGBoost uses 70 estimators, maximum depth 6, learning rate 0.10, subsample 0.85, and colsample_bytree 0.85; LightGBM uses 90 estimators, 45 leaves, learning rate 0.08, and class weighting; and the DNN uses two hidden layers containing 64 and 32 neurons with early stopping. These configurations are intended to provide representative baselines without excessive hyperparameter optimization.

The best model is selected primarily on the basis of Macro-F1 and secondarily on latency. For each event x_t, the model produces a class-probability distribution p(c|x_t). Attack probability is defined as one minus the Benign probability, while confidence is computed as the margin between the highest and second-highest class probabilities. This approach makes confidence reflect the decision margin rather than only the magnitude of the predicted class probability.

Adaptive Threat-Risk Assessment

The threat-risk score combines four components: attack proba bility P_t, severity S_t, model confidence C_t, and frequency F_t within a sliding window. Severity is defined as a simulation-policy parameter, with lower values for Recon and higher values for DDoS, DoS, Spoofing, and Mirai. Frequency is normalized to capture escalation caused by repeated threats. The general formulation is shown in Equation (1).

$$R_t = w_p P_t + w_s S_t + w_c C_t + w_f F_t \quad (1)$$

The weights satisfy $w_p + w_s + w_c + w_f = 1$ Several weight combinations are evaluated on the validation set and selected according to their agreement with the reference simulation policy. The resulting R_t value is mapped to five risk levels: Normal, Low, Medium, High, and Critical. Threshold sensitivity is evaluated on the validation data before test-set analysis. Based on the reviewer-motivated re-analysis, the primary configuration is the more conservative threshold set 0.22/0.42/0.62/0.82 because it produced the highest validation risk-level agreement and response-selection policy agreement while also producing the lowest URR among the tested configurations. The thresholds are then fixed before test-set evaluation to prevent test-set tuning.

Automated Network and Cryptographic Response

The Response Decision Engine uses the threat category, risk level, and session context to select an action. The policy is deliberately context-aware: availability attacks are not forced to execute cryptographic operations, whereas threats indicating possible session compromise can trigger changes to keying material. The response mapping is shown in Table 2.

Table 2. AID-ACR Adaptive Response Policy

Level	Cryptographic Response	Network Response
Normal	Maintain active key/session	Normal monitoring
Low	No key rotation	Enhanced monitoring
Medium	Re-keying for session-related threats	Rate limiting/restriction
High	Key rotation + re-authentication	Blocking/isolation
Critical	Key revocation + new secure session	Isolation/quarantine

The prototype uses AES-256-GCM for authenticated encryption of a 1 KB payload. Session re-keying uses fresh input keying material generated independently of the current session key and HKDF-SHA256 to derive K_{t+1}; the previous transmit key is retired after an atomic key-version handover. Key rotation similarly derives a fresh key and increments the active key version, with re-authentication required for high-risk events. Key revocation changes the status of the affected key identifier to revoked, terminates the associated session, and prevents new data from being accepted under the revoked key version. Secure session re-establishment is simulated using fresh X25519 ephemeral key pairs, X25519 key agreement, HKDF-SHA256, and construction of a new AES-GCM context. The prototype therefore avoids deriving replacement keys solely from a potentially compromised old key.

Algorithm: Operational Cryptographic Response Procedure

1) Input: predicted class, risk level, and current session/key state.
2) Availability-only attacks: apply rate limiting, blocking, or isolation; do not rotate keys unless session compromise is also indicated.
3) Medium session risk: generate fresh input keying material, derive K_{t+1} using HKDF-SHA256, activate the new key version, then retire the previous transmit key after handover.
4) High risk: rotate the key using fresh material, increment the key version, require re-authentication, and reject new messages under the superseded version.
5) Critical/key-compromise risk: revoke the key identifier, terminate the session, isolate the endpoint, perform fresh X25519 key agreement, derive a new key with HKDF-SHA256, and construct a new AES-GCM context after re-authentication.

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

6) Audit: record old/new key IDs, reason, timestamp, action result, and session state; update frequency and escalation context.
7) Failure handling: if a critical cryptographic action fails, maintain isolation and generate an administrative alert.

Operational Flow and Decision Logic

The operational flow begins when a traffic event is received and transformed into a feature vector. The Intrusion Detection Engine produces a predicted class, probability distribution, and confidence margin. If the predicted class is Benign with low attack probability, the event proceeds to normal monitoring. When attack evidence is present, Threat-Risk Assessment calculates R_t and assigns a level. The Response Decision Engine then determines whether the attack category concerns availability, session integrity, authentication, or suspected compromise. This assessment determines whether the system executes a network control, a cryptographic control, or a combination of both.

Procedurally, AID-ACR can be summarized as follows: (1) receive event x_t; (2) calculate $p(c \mid x_t)$, the predicted class, and confidence; (3) update the frequency within the sliding window; (4) calculate R_t; (5) map R_t to a risk level; (6) select a policy based on attack family and session context; (7) execute the response; (8) record the outcome; and (9) update the feedback state. This sequence ensures that no cryptographic operation is executed before risk assessment is completed.

Algorithm: AID-ACR Decision Logic

Step	Process
1	Receive traffic event x_t and perform preprocessing.
2	Predict class and probability using the IDS.
3	Calculate the confidence margin and attack probability.
4	Update the frequency within the sliding window.
5	Calculate threat-risk score R_t and determine the level.
6	Evaluate the attack family and session/key context.
7	Select a network response and/or cryptographic response.
8	Execute the action and record response time.
9	Update the feedback state for the next event.

The policy engine is deterministic in the simulation study so that response-selection policy agreement can be evaluated objectively. The adaptive behavior therefore does not arise from a second learning model or from online IDS retraining. Instead, it arises because frequency state, session context, and risk level can change over time, causing the deterministic policy to select a different action for later events. In a production implementation, the policy can be represented as policy-as-code, a risk matrix, or a digitally signed rule engine so that policy changes remain auditable.

Threat Model and Simulation Assumptions

The threat model assumes that the IDS sensor obtains a flow or event representation before a security decision is enforced. An attacker may generate volumetric attacks, reconnaissance, authentication attempts, spoofing, web attacks, or botnet behavior. The framework does not assume that every alert indicates key compromise. Instead, attack category and risk level determine whether response is sufficient at the network layer or should be extended to the cryptographic context. Under these assumptions, DDoS and DoS are mapped primarily to rate limiting, blocking, and isolation, whereas Spoofing, Brute Force, Web attacks, and Mirai can trigger re-keying or key-state changes when the risk level increases.

The simulation also assumes that AID-ACR has access to an enforcement point, a session manager, and a key-management function. In a production environment, these components may reside at a gateway, reverse proxy, software-defined networking controller, security service, or host endpoint. Communication time to external components is not included in the prototype benchmark; therefore, the reported latency represents local computational latency. This distinction is important to avoid implying that isolation of a device in a physical network can always be completed within sub-millisecond time.

The reference simulation policy is used as the response-policy ground truth, not as a universal security ground truth. It is constructed from the true class, severity, and temporal recurrence to evaluate whether the risk formula can reproduce consistent decisions. In real deployment, severity and thresholds should be derived from an organizational threat model, asset criticality, business impact, or a validated risk framework. Thus, the experiment evaluates the consistency of AID-ACR against a defined policy rather than asserting that thresholds from 0.20 to 0.80 are universally applicable.

Experimental Environment and Reproducibility

Experiments were executed using Python 3.13.5 on Linux 6.18.44 (x86_64) with an Intel Xeon Platinum 8573C processor, five available logical CPU cores, and approximately 5.8 GB of memory. The implementation used NumPy 2.3.5, pandas 2.2.3, scikit-learn 1.8.0, XGBoost 3.1.3, LightGBM 4.6.0, and cryptography 46.0.4. The reference detailed experiment uses random seed 42. To assess model stability, the complete synthetic-data generation, stratified split, training, and

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

testing workflow was independently repeated with seeds 42, 43, and 44, and mean ± standard deviation is reported for the four IDS models. Latency measurements used a high-resolution performance counter and repeated cryptographic benchmarks to reduce one-off execution variability.

For cryptographic operations, benchmarks were performed on a 1 KB payload using 500 repetitions for AES-GCM, re-keying, rotation, and revocation, and 200 repetitions for the more expensive secure session re-establishment operation. Mean, Median, and P95 values are reported to characterize latency variation. For IDS evaluation, batch inference is used to compare model efficiency, while streaming single-event inference is used as a parameter closer to the AID-ACR pipeline. This strategy supports reproducibility while distinguishing batch throughput from individual-event latency.

Simulated Deployment Scenario

The reference deployment scenario is an IoT/edge gateway positioned between local devices and an external network. The IDS is located at the gateway or network sensor, while the Response Decision Engine communicates with a firewall/SDN policy point and a session/key service. In this scenario, a DDoS flow classified as Critical is directed to isolation and blocking, whereas a High-risk Spoofing event may trigger key rotation and re-authentication. If Mirai or a similar compromise reaches the Critical level, key revocation may be performed before the device is isolated.

The gateway is selected as the conceptual enforcement point because it combines traffic observability with enforcement capability. In a distributed deployment, multiple sensors may send evidence to a central decision engine while the key-management service remains separate. The framework does not require all components to reside on the same host; it only requires interfaces for communicating class, risk, action requests, and outcomes. The architecture can therefore be adapted to enterprise, edge-computing, or IoT environments provided that trust mechanisms between components are available.

This scenario also explains why response latency in the experiment does not include network round-trip time. The purpose of the simulation study is to measure the minimum computational cost of the decision pipeline. In a future testbed, T_response should be decomposed into policy-dispatch time, enforcement-propagation time, cryptographic-service time, and confirmation time. Such decomposition would help determine whether the bottleneck lies in the IDS, the network controller, or the key-management infrastructure.

Evaluation Metrics

IDS performance is evaluated using Accuracy, Macro-Precision, Macro-Recall, and Macro-F1. Macro-F1 is prioritized because the class distribution is imbalanced. Model stability is assessed through three repeated stratified runs and summarized using mean ± standard deviation. Latency is measured for both batch inference and streaming single-event inference. For the complete prototype, end-to-end computational prototype latency is defined as the sum of local detection, risk assessment, response decision, and locally emulated response execution. It explicitly excludes external enforcement and communication delays.

$$T_{E2E} = T_{det} + T_{risk} + T_{decision} + T_{response} \quad (2)$$

Unnecessary Response Rate (URR) measures the percentage of Benign traffic that incorrectly triggers an active response. This metric is important because an overly aggressive automated system may cause unnecessary key rotation, blocking, or isolation. All experiments use random seed 42 to support reproducibility.

RESULTS AND DISCUSSION

Simulation Study Data Distribution

After data generation, 32,000 samples were obtained and divided into 22,400 training, 4,800 validation, and 4,800 testing samples. DDoS was the dominant class with 12,745 samples, whereas Brute Force was the smallest class with 669 samples. The complete distribution is presented in Table 3.

Table 3. Data Distribution in the Simulation Study

Category	Total	Train	Valid.	Test
Benign	4,788	3,352	718	718
DDoS	12,745	8,921	1,912	1,912
DoS	4,798	3,359	719	720
Recon	2,262	1,583	340	339
Web	1,294	906	194	194
Brute Force	669	468	100	101
Spoofing	2,247	1,573	337	337
Mirai	3,197	2,238	480	479
Total	32,000	22,400	4,800	4,800

The distribution intentionally preserves class imbalance to test whether the models can still recognize

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

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

classes with fewer samples. This also explains why Macro-F1 is more informative than accuracy alone.

Intrusion Detection Performance

The comparison of the four models is presented in Table 4 and Figure 2. Across three repeated stratified runs, LightGBM and the DNN both achieved mean Accuracy of 98.50%, but LightGBM produced the highest mean Macro-F1 at 97.25% ± 0.62%, marginally above the DNN at 97.23% ± 0.51%. XGBoost achieved 96.60% ± 0.69% Macro-F1, while Random Forest showed greater variability at 94.40% ± 1.23%. These repeated results support retaining LightGBM as the Intrusion Detection Engine because its class-balanced performance remained marginally strongest across runs. Figure 2 presents the reference seed-42 run for visual comparison, while Table 4 reports mean ± standard deviation.

Table 4. Comparison of Intrusion Detection Model Performance

Model	Acc. % (mean ± SD)	Prec. % (mean ± SD)	Rec. % (mean ± SD)	F1 % (mean ± SD)	µs/sample*
Random Forest	97.19 ± 0.40	97.75 ± 0.31	92.13 ± 1.76	94.40 ± 1.23	8.673
XGBoost	98.03 ± 0.31	97.11 ± 0.77	96.18 ± 0.60	96.60 ± 0.69	9.010
LightGBM	98.50 ± 0.25	98.16 ± 0.44	96.46 ± 0.72	97.25 ± 0.62	22.340
DNN (MLP)	98.50 ± 0.24	98.19 ± 0.46	96.36 ± 0.60	97.23 ± 0.51	1.416

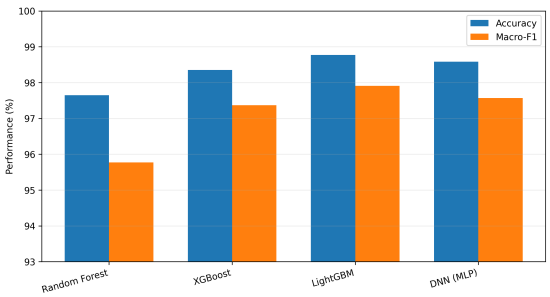


Figure 2. Comparison of Accuracy and Macro-F1 Across IDS Models

Although LightGBM batch inference was slower than the DNN in the simulation implementation, single-event evaluation yielded a median detection latency of 0.552 ms per event. This value remains very small relative to human response times and is sufficient for near-real-time experimentation. LightGBM was therefore selected on the basis of balanced cross-class performance and latency rather than solely on the highest Accuracy value.

As an external benchmark, Almahaqeri et al. (2026) reported an XGBoost accuracy of 0.9959 and Macro-F1 of 0.8903 in an eight-class CICIoT2023 scenario. This comparison is not intended to imply that the synthetic data are equivalent to CICIoT2023. On the contrary, the difference in Macro-F1 underscores that simulation results are more controlled and must be validated on actual data before making generalization claims.

Per-Class Analysis

Per-class analysis was performed to ensure that the aggregate Macro-F1 did not obscure weaknesses in minority classes. Table 5 presents LightGBM precision, recall, F1-score, and support on the test set. DDoS produced the highest F1-score at 99.53%, followed by DoS at 99.24% and Mirai at 98.74%. Benign traffic achieved 99.03% recall, indicating that most normal traffic remained correctly recognized as normal and only a small portion entered the threat-response path.

Table 5. LightGBM Performance for Each Class

Class	Prec. %	Recall %	F1 %	Support
Benign	97.53	99.03	98.27	718
DDoS	99.27	99.79	99.53	1,912
DoS	99.03	99.44	99.24	720
Recon	98.21	97.05	97.63	339
Web	99.45	93.81	96.55	194
Brute Force	98.94	92.08	95.38	101
Spoofing	97.92	97.92	97.92	337
Mirai	98.95	98.54	98.74	479

Web and Brute Force had the lowest recall values, at 93.81% and 92.08%, respectively. This result is consistent with their smaller sample sizes and the similarity of some events to other classes. In an automated response system, a false negative in a class such as Brute Force has consequences beyond a reduced F1-score because missed detection may prevent re-authentication or key rotation that should otherwise be activated. Therefore, AID-ACR evaluation must treat confusion patterns as part of operational risk rather than only as classification statistics.

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

The confusion matrix showed that Benign errors were distributed mainly toward Recon and Spoofing, whereas some Web errors were assigned to Benign, DDoS, DoS, Recon, or Mirai. Brute Force produced eight false negatives among 101 test samples. These findings explain why the risk engine does not use the predicted label as the sole decision factor. Probability and confidence preserve information about model uncertainty, while frequency can increase the risk score when similar events recur.

Adaptive Threat-Risk Assessment

Calibration on the validation set produced weights of $w_s = 0.40$, $w_c = 0.10$, and $w_f = 0.20$. Severity received the largest weight because the AID-ACR policy must consider the impact of the attack type rather than model confidence alone. The calibrated formula is shown in Equation (3).

$$R_t = 0.30P_t + 0.40S_t + 0.10C_t + 0.20F_t \quad (3)$$

Threshold sensitivity analysis selected 0.22, 0.42, 0.62, and 0.82 as the primary thresholds. This configuration achieved 95.40% risk-level agreement with the reference simulation policy on the validation set and 95.88% overall response-selection policy agreement on the test set, while reducing URR to 0.418%. The corresponding test-set risk distribution is shown in Table 6.

Table 6. Threat-Risk Level Distribution

Risk Level	Events	Percentage (%)
Normal	729	15.19
Low	1	0.02
Medium	312	6.50
High	1,059	22.06
Critical	2,699	56.23
Total	4,800	100.00

The Critical level remained dominant because the simulation contains large proportions of DDoS, DoS, and Mirai combined with high severity values. Under the revised conservative thresholds, the Critical share decreased from 60.40% to 56.23%, Medium and High shares increased, and one event entered the Low level. These values are properties of the simulation and should not be interpreted as assumptions about production risk distributions. In operational deployment, thresholds must be calibrated using representative traffic, asset criticality, and response costs.

Automated Response and Response-Selection Policy Agreement

The risk-assessment output was subsequently translated into security actions. The reported response-selection policy agreement measures whether the action selected by AID-ACR matches the reference simulation policy derived from the true class and temporal context. It is not a measure of whether the action actually stops an attack in a real network. The results are presented in Table 7.

Table 7. Response-Selection Policy Agreement by Threat Category

Threat	Level	Dominant Response	Policy Agree. %
DDoS	Critical	Isolation + blocking	99.63
DoS	Critical	Isolation + blocking	90.28
Recon	Medium	Enhanced monitoring	92.63
Web	High	Key rotation + re-authentication	90.72
Brute Force	High	Key rotation + re-authentication	87.13
Spoofing	High	Key rotation + re-authentication	98.22
Mirai	Critical	Key revocation + isolation	89.14

DDoS achieved the highest response-selection policy agreement at 99.63%, followed by Spoofing at 98.22% and Recon at 92.63%. Recon improved substantially under the revised conservative thresholds because fewer borderline events crossed into overly aggressive response levels. The remaining differences indicate that policy agreement is sensitive to the interaction among severity, model confidence, and temporal frequency and should be calibrated against operational response costs.

The context-aware policy also prevents a design error in which key replacement is performed for every attack. For DDoS and DoS, the primary actions remain availability controls, whereas for Spoofing, Brute Force, Web attacks, and Mirai, cryptographic actions are used when the context suggests possible session compromise. Cryptographic response therefore strengthens rather than replaces network response.

Actual mitigation effectiveness was not measured in this prototype because the experiment did not replay attacks through a live enforcement point or observe post-response traffic. Therefore, policy-agreement percentages must not be interpreted as attack-blocking success rates. A subsequent testbed evaluation should measure time-to-containment, malicious-packet reduction after enforcement, session-

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

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

termination success, key-revocation propagation success, and residual attack traffic after the selected action.

Cryptographic Response Performance

Cryptographic operations were benchmarked using a 1 KB payload. The median AES-256-GCM encryption and decryption times were 0.00189 ms and 0.00174 ms, respectively. Session re-keying and key rotation had median times of 0.00376 ms and 0.00484 ms, while secure session re-establishment was the most expensive operation, with a median of 0.11042 ms. Complete results are shown in Table 8.

Table 8. Cryptographic Response Performance

Operation	Mean (ms)	Median (ms)	P95 (ms)
AES-256-GCM Encryption	0.00273	0.00189	0.00229
AES-256-GCM Decryption	0.00258	0.00174	0.00201
Session Re-keying	0.00688	0.00376	0.00411
Key Rotation	0.00508	0.00484	0.00530
Key Revocation	0.00033	0.00029	0.00055
Secure Session Re-establishment	0.18471	0.11042	0.19039

The difference in computational cost supports a tiered response mechanism. Re-keying and rotation can be used for medium- or high-risk threats because their overhead is very small, whereas establishing an entirely new session should be reserved for situations that truly require termination of the previous security context. This principle is consistent with risk-based response: action intensity increases as risk increases.

End-to-End Computational Prototype Latency

Local prototype execution produced an end-to-end computational latency of 0.65933 ms per event. Intrusion detection accounted for 83.78% of the total local computation time, response execution for 15.58%, and risk assessment plus response decision together for less than 1%. Table 9 and Figure 3 present the latency composition.

Table 9. End-to-End Computational Prototype Latency

Component	Latency (ms)	Contribution (%)
Intrusion Detection	0.55238	83.78
Risk Assessment	0.00253	0.38
Response Decision	0.00169	0.26
Response Execution	0.10273	15.58
Total	0.65933	100.00

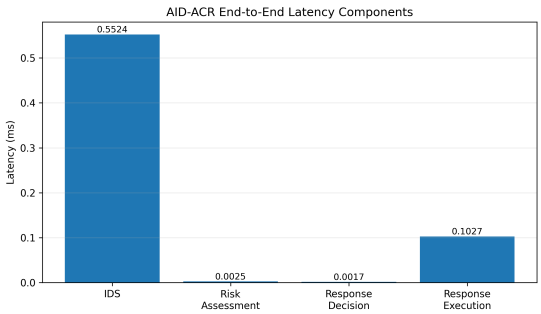


Figure 3. Decomposition of AID-ACR Local Computational Prototype Latency

These findings show that adding dynamic risk scoring does not create the primary local-computation bottleneck. However, response execution in this study is local-operation emulation only. The 0.659 ms value excludes communication with distributed firewalls or SDN controllers, remote KMS/HSM infrastructure, identity providers, network round-trip time, device actuation, and confirmation that the selected control has taken effect. Therefore, the value is reported as end-to-end computational prototype latency and must not be interpreted as end-to-end operational threat-mitigation latency.

Unnecessary Response Rate and Framework Implications

Static response produced a URR of 0.9749%, whereas the revised AID-ACR threshold policy produced 0.4178%. The relative reduction was 57.14%. This result indicates that risk-based action selection can reduce unnecessary active responses to Benign traffic in the simulation. Lower URR is important because false responses can reduce service availability by terminating legitimate sessions or performing unnecessary key rotations.

$$\text{URR Reduction} = \frac{0.9749 - 0.4178}{0.9749} \times 100\% = 57.14\% \quad (4)$$

Ablation Study of Risk Score Components

An ablation study was conducted using the revised 0.22/0.42/0.62/0.82 threshold policy to evaluate the contributions of severity and frequency. The complete P+S+C+F configuration was compared with variants without frequency, without severity, and using probability only. Table 10 shows that the complete configuration achieved 95.40% validation risk-level agreement, 95.88% overall response-selection policy agreement, and 0.418% URR. Removing frequency reduced policy agreement to 74.50%; removing severity produced 84.71%; and the probability-only configuration produced 73.10%.

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

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

Table 10. Ablation Study of Threat-Risk Score Components under the Revised Threshold Policy

Configuration	Risk Agree. %	Policy Agree. %	URR %
P+S+C+F	95.40	95.88	0.418
Without Frequency	75.38	74.50	0.696
Without Severity	84.00	84.71	0.696
Probability Only	73.08	73.10	0.696

The revised ablation results confirm that both temporal recurrence and severity contribute materially to response selection. Removing frequency produced the largest loss in policy agreement, indicating that event history is important for escalation. Removing severity also reduced agreement because attacks with similar probabilities do not necessarily have equivalent security impact. The complete formulation therefore remains preferable to a simple probability threshold for proportionate response selection.

Threshold Sensitivity Analysis

Sensitivity analysis compared three threshold policies: more sensitive (0.18/0.38/0.58/0.78), the original baseline (0.20/0.40/0.60/0.80), and the more conservative configuration (0.22/0.42/0.62/0.82). The conservative configuration is adopted as the revised primary policy because it achieved the highest validation agreement (95.40%), highest response-selection policy agreement (95.88%), and lowest URR (0.418%) among the tested configurations, while also reducing the Critical rate to 56.23%.

Table 11. Sensitivity Analysis and Revised Threshold Selection

Policy	Threshold	Agree. %	Policy %	URR %	Critical %
More sensitive	0.18/0.38/0.58/0.78	88.02	87.54	0.696	64.67
Original baseline	0.20/0.40/0.60/0.80	94.44	94.27	0.557	60.40
Selected conservative	0.22/0.42/0.62/0.82	95.40	95.88	0.418	56.23

These results demonstrate the expected trade-off among sensitivity, false-response risk, and escalation intensity. Because the conservative configuration performed better on all three validation-oriented criteria used in this study, it is selected as the primary threshold policy in the revised manuscript. This choice is empirical rather than universal: a real deployment should recalibrate thresholds using an organizational threat model, asset criticality, false-response costs, and tolerance for service disruption.

Comparison of Conventional IDS, Static Response, and AID-ACR

To clarify the architectural contribution, Table 12 compares a conventional IDS, an IDS with static response, and AID-ACR. A conventional IDS ends at alert generation. Static response automates actions but applies fixed rules to each category. AID-ACR adds dynamic risk, temporal context, response escalation, and cryptographic adaptation. This distinction is important because automation alone is not equivalent to adaptability.

Table 12. Comparison of Security Response Architectures

Capability	Conventional IDS	Static	AID-ACR
Attack detection	Yes	Yes	Yes
Dynamic risk	No	No	Yes
Temporal context	No	No	Yes
Automated mitigation	No	Yes	Yes
Context-aware crypto	No	Limited	Yes
Response escalation	No	No	Yes
Feedback state	No	No	Yes
URR control	No	Limited	Yes

The comparison shows that adding a static response engine can indeed shorten reaction time, but it may execute overly aggressive actions when an alert is a false positive. This is reflected in the higher URR of the static-response configuration. AID-ACR introduces a risk gate so that not every predicted attack immediately becomes a disruptive action. This mechanism is particularly important when available actions include isolation or key revocation, because the cost of an incorrect response is substantially greater than the cost of adding one additional log entry.

Threats to Validity and Limitations

Internal validity is affected by the synthetic-data design and the reference simulation policy. Class separation, class imbalance, severity, and thresholds were constructed so that the pipeline could be evaluated under controlled conditions. Although the random seed and configurations are recorded, model-performance results should not be interpreted as estimates of performance on the actual CICIoT2023 dataset. Accuracy and Macro-F1 in this simulation are more appropriately used to compare the four models within

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

the same environment than to compare them directly with external benchmarks.

Construct validity concerns the distinction between response-selection policy agreement and actual mitigation effectiveness. The former uses a reference policy constructed from the true class and temporal context and therefore measures decision consistency with that policy. It does not measure whether an action actually stops an attack in a live network. Similarly, key revocation in the prototype is measured as a local registry-state change rather than propagation to distributed enforcement components. Actual mitigation effectiveness must therefore be validated using traffic replay or a live testbed with post-response observations.

External validity is the most significant limitation. Synthetic data cannot capture all noise, concept drift, inter-device variation, protocol diversity, and adversarial behavior found in real IoT networks. In addition, single-host computational latency does not represent deployments involving an SDN controller, hardware firewall, HSM, remote KMS, or identity provider. The sub-millisecond result only indicates that the core framework logic does not add substantial computational overhead in the test environment.

Reliability is addressed through recorded software and hardware specifications, explicit synthetic-data generation parameters, repeated cryptographic benchmarks, separate training/ validation/testing sets, and three independent repeated stratified runs using seeds 42, 43, and 44. The repeated results show that LightGBM achieved 97.25% ± 0.62% Macro-F1 and DNN achieved 97.23% ± 0.51%, indicating that the ranking is stable but very close. Hardware variation can still change absolute latency values. Future work should additionally provide source code, the exact data-generation script, and a real-network replay configuration.

These limitations do not diminish the role of the simulation study as a proof of concept, but they restrict the claims that can be made. The primary contribution of this study is to demonstrate that detection output, risk assessment, network response, and cryptographic response can be integrated into a measurable pipeline. Claims of superiority in real networks can only be made after validation using the official CICIoT2023 dataset, cross-dataset evaluation, and a real-time testbed.

Comparison with Current Research Directions

The AID-ACR results complement adaptive-IDS research rather than claiming online learning. Cerasuolo et al. (2025a, 2025b) place adaptation inside the learning model through incremental updating, whereas this prototype keeps the IDS fixed and places adaptation in stateful risk assessment and response selection. These approaches are complementary: incremental learning can update the detector when concepts or classes change, while the AID-ACR risk/response layer determines context-dependent actions after detection.

Compared with SOAR, AID-ACR does not attempt to replace an orchestration platform. The framework is narrower and focuses on mapping IDS evidence to network and cryptographic controls. It can therefore operate as a decision component within a SOAR playbook. Under this design, the research contribution lies in linking detection confidence, threat severity, recurrence, and key-state actions. It also extends Jamaluddin et al. (2025) from integrating cryptography with IDS to an explicit decision loop that can be evaluated through response accuracy, URR, and end-to-end latency.

Overall, the simulation-study results support three propositions. First, a high-performing IDS can serve as a decision sensor but need not be the final decision maker. Second, dynamic threat-risk scoring enables the system to assign different response levels to the same attack label on the basis of severity, confidence, and recurrence patterns. Third, cryptographic actions can be integrated as contextual responses without dominating computational overhead when they are used selectively.

AID-ACR therefore forms a Detect-Assess-Respond-Observe-Adapt pattern. This pattern is closer to modern incident-response principles than an IDS that terminates at alert generation. However, generalization cannot yet be claimed because the evaluation data remain synthetic. The next validation stage should use the official CICIoT2023 files, cross-dataset testing, traffic replay on a testbed, and integration with actual enforcement points. Severity weights and thresholds should also be derived from an organizational threat model or multi-criteria decision-making technique rather than functioning only as experimental parameters.

CONCLUSION

This study proposes the Adaptive Intrusion Detection and Automated Cryptographic Response (AID-ACR) Framework as a closed-loop cybersecurity prototype that integrates a fixed intrusion-detection model with stateful risk assessment, response selection, cryptographic actions, and feedback. Across three repeated stratified runs, LightGBM achieved mean Accuracy of 98.50% ± 0.25% and mean Macro-F1 of 97.25% ± 0.62%. The revised conservative threshold

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

policy (0.22/0.42/0.62/0.82) achieved 95.40% validation risk-level agreement, 95.88% response-selection policy agreement, and reduced URR from 0.9749% under static response to 0.4178%, corresponding to a 57.14% relative reduction. The term adaptive in this work refers specifically to updates in threat-frequency, session state, and response escalation; the IDS is not retrained online.

In terms of efficiency, AID-ACR produced 0.659 ms per event of end-to-end computational prototype latency, with intrusion detection as the largest local-computation component. This value excludes firewall/SDN, KMS/HSM, identity-provider, network round-trip, and physical-enforcement delays. Session re-keying and key rotation introduced low local overhead, whereas secure session re-establishment was more expensive and is therefore more appropriate for High or Critical threats. The reported response metric represents agreement with a reference simulation policy and not actual attack-mitigation effectiveness. Validation using the official CICIoT2023 dataset, attack replay, real enforcement points, and post-response traffic measurements is required before operational effectiveness can be claimed.

REFERENCES

Adewole, K. S., Jacobsson, A., & Davidsson, P. (2025). Intrusion detection framework for Internet of Things with rule induction for model explanation. *Sensors, 25*(6), 1845. https://doi.org/10.3390/s25061845

Almahaqeri, S. A., Almourish, M. H., Nasser, A. A., Alghawli, A. S. A., Elsayed, A. A. K., & Alhejoj, A. N. (2026). An optimized gradient boosting framework for IoT intrusion detection: A comprehensive evaluation on the CICIoT2023 dataset. *Scientific Reports, 16*, 16909. https://doi.org/10.1038/s41598-026-47399-5

Barker, E. (2020). *Recommendation for key management: Part 1 - General (NIST Special Publication 800-57 Part 1 Revision 5)*. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-57pt1r5

Cerasuolo, F., Bovenzi, G., Ciuonzo, D., & Pescapè, A. (2025a). Attack-adaptive network intrusion detection systems for IoT networks through class incremental learning. *Computer Networks, 263*, 111228. https://doi.org/10.1016/j.comnet.2025.111228

Cerasuolo, F., Bovenzi, G., Ciuonzo, D., & Pescapè, A. (2025b). Adaptable, incremental, and explainable network intrusion detection systems for Internet of Things. *Engineering Applications of Artificial Intelligence, 144*, 110143. https://doi.org/10.1016/j.engappai.2025.110143

Chandnani, P., et al. (2025). A physics-based hyper parameter optimized federated multi-layered deep learning model for intrusion detection in IoT networks. *IEEE Access*, 13, 21992-22010. https://doi.org/10.1109/ACCESS.2025.3535952

Dworkin, M. (2007). *Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC (NIST Special Publication 800-38D)*. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-38D

Gheni, H. Q., & Al-Yaseen, W. L. (2024). Two-step data clustering for improved intrusion detection system using CICIoT2023 dataset. *e-Prime - Advances in Electrical Engineering, Electronics and Energy, 9*, 100673. https://doi.org/10.1016/j.prime.2024.100673

Gustina DM, V., & Ananda, A. (2024). Kecerdasan buatan untuk Security Orchestration, Automation and Response: Tinjauan cakupan. *Jurnal Komputer Terapan, 10*(1), 36-47. https://doi.org/10.35143/jkt.v10i1.6247

Hizal, S., Çavusoglu, Ü., & Akgün, D. (2024). A novel deep learning-based intrusion detection system for IoT DDoS security. *Internet of Things, 28*, 101336. https://doi.org/10.1016/j.iot.2024.101336

Jamaluddin, Manalu, D. R., & Rumapea, H. (2025). An integrated framework of cryptographic mechanisms and intrusion detection systems for enhanced cybersecurity. In *2025 Tenth International Conference on Informatics and Computing (ICIC) (pp. 1-4)*. IEEE. https://doi.org/10.1109/ICIC6805 4.2025.11309624

Kikissagbe, B. R., & Adda, M. (2024). Machine learning-based intrusion detection methods in IoT systems: A comprehensive review. *Electronics, 13*(18), 3601. https://doi.org/10.3390/electronics13183601

Komal, A., & Li, S. (2026). Intrusion detection in the Internet of Things: A comprehensive review of techniques, architectures, datasets, and emerging trends. *Sensors, 26*(11), 3405. https://doi.org/10.3390/s26113405

Nelson, A., Rekhi, S., Scarfone, K., & Souppaya, M. (2025). *Incident response recommendations and considerations for cybersecurity risk management: A CSF 2.0 community profile (NIST Special Publication 800-61 Revision 3)*. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-61r3

Neto, E. C. P., Dadkhah, S., Ferreira, R., Zohourian, A., Lu, R., & Ghorbani, A. A. (2023). CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment. *Sensors, 23*(13), 5941. https://doi.org/10.3390/s23135941

Riandari, F., Afrisawati, A., Afifa, R. M., Syahputra, R., & Ginting, R. (2024). Dynamic optimization algorithms for enhancing blockchain network resilience against distributed attacks. *International Journal of Basic and Applied*

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

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

Science, 13(2), 96–111. https://doi.org/10.35335/ijobas.v13i2.499

Sallam, S., El Barachi, M., & Li, N. (2026). Intrusion detection on the Internet of Things: A comprehensive review and gap analysis toward real-time, lightweight, adaptive, and autonomous security. *IoT, 7*(1), 16. https://doi.org/10.3390/iot7010016

Waelchli, S., & Walter, Y. (2025). Reducing the risk of social engineering attacks using SOAR measures in a real-world environment: A case study. *Computers & Security, 148*, 104137. https://doi.org/10.1016/j.cose.2024.104137

Yaras, S., & Dener, M. (2024). IoT-based intrusion detection system using new hybrid deep learning algorithm. *Electronics, 13*(6), 1053. https://doi.org/10.3390/electronics13061053

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