Digital Transformation and Technology Dynamics

Journal Homepage: www.techdynamicsjournal.com | ARTICLE NO. DTTD-2023003| VOL. 3 ISSUE 2

A Holistic Framework for Autonomous Intelligence: Integrating Artificial Intelligence, Distributed Computing, Cybersecurity, and Real-Time Decision Systems

Rashadul Islam Samrat[1]*, Md Rasel Ul Alam[2], Kanita Haider[3],

[1] *Department of Marketing, University of Barishal, Barishal, Bangladesh*

[2] *Department of Computer and Information Sciences, University of the Cumberlands, Kentucky, USA.*

[3] *Department of Computer Science and Engineering, International Islamic University Chittagong, Chittagong, Bangladesh*

**Corresponding author: samrat.meazi1@gmail.com*

Received 19 August 2021; Revised 12 September2023; Accepted 30 October 2023; Published: 29 November 2023

KEYWORDS

Autonomous intelligence;
Distributed computing;
Cybersecurity;
Real-time decision systems;
Multi-agent AI;
Explainable AI;
Edge computing;
Federated learning

Abstract

The convergence of artificial intelligence (AI), distributed computing, cybersecurity, and real-time decision infrastructure has produced a class of systems — here termed autonomous intelligence systems — that must sense, reason, and act across geographically dispersed, adversarial, and latency-constrained environments. Existing research largely treats these four concerns in isolation: the AI-agents literature emphasizes reasoning and coordination; the distributed- and edge-computing literature emphasizes resource placement and latency; the cybersecurity literature treats threat detection largely as a post-hoc control; and the real-time systems literature emphasizes scheduling and stability with limited attention to learning-based components. This fragmentation leaves a research gap: no widely validated framework specifies how autonomous, learning-enabled decision-making, distributed resource orchestration, and security assurance should be co-designed rather than integrated after the fact. This paper addresses that gap through (1) a thematic, critical synthesis of literature across the four pillars; (2) a proposed five-layer conceptual and mathematical framework, AID²R (AI – Distributed computing – Decision – Response), that embeds security and explainability as in-loop components; (3) formal optimization, latency, and trust objectives for the framework; and (4) a reproducible evaluation protocol — including proposed datasets, baselines, ablation design, and metrics — for future empirical validation. Because controlled, multi-domain experimentation spanning AI, distributed-systems, and cybersecurity testbeds was outside the scope of this study, illustrative/synthetic results are used only to demonstrate the evaluation protocol and are not presented as empirical findings. The paper contributes a domain-agnostic architectural blueprint, a set of falsifiable research questions, and a transparent roadmap for empirical validation, with practical relevance for designers of autonomous vehicles, smart-grid controllers, industrial IoT platforms, and security operations centers that must reconcile autonomy, distribution, security, and speed within a single design.

1. Introduction

1.1 Background and Problem Statement

Digital infrastructure is increasingly characterized by software components that must perceive changing environments, process information, and make decisions without waiting for continuous human intervention or round-trip communication with centralized cloud infrastructure. Autonomous vehicles may need to respond to hazards within milliseconds, industrial controllers may need to rebalance processes under rapidly changing operating conditions, and cybersecurity systems may need to identify and contain malicious activity before an intrusion propagates across interconnected devices. These requirements have accelerated the convergence of three

technological developments: increasingly autonomous artificial intelligence, distributed and edge computing, and security-aware real-time systems.

First, machine learning is increasingly being used not only for static prediction but also for sequential reasoning, planning, tool use, and interaction with dynamic environments. The emergence of large-language-model-based intelligent agents and multi-agent systems has expanded the role of AI from passive prediction toward systems capable of reasoning, acting, and coordinating with other agents. Such systems introduce new architectural requirements because model outputs can directly influence subsequent actions, resource allocation, system states, and interactions with external environments. Consequently, the reliability of an autonomous system depends not only on the predictive capability of an individual model but also on how its decisions interact with the surrounding computational and operational infrastructure.

Second, computation has increasingly moved from centralized cloud environments toward edge and fog infrastructures. Edge computing places processing closer to data sources and end users and can reduce response time, bandwidth requirements, and privacy exposure for latency-sensitive applications. However, distributing computation across heterogeneous edge nodes also introduces resource constraints, communication dependencies, synchronization requirements, and additional system states that must be considered when autonomous AI components are deployed in real time.

Third, increasing distribution and autonomy expand the security exposure of the overall system. Machine-learning-enabled IoT and edge environments can be affected by adversarial examples, poisoning, backdoor attacks, inference attacks, Byzantine behavior, and other forms of manipulation, while conventional intrusion-detection approaches may struggle with the heterogeneity and dynamic behavior of modern distributed environments. Security therefore cannot be treated solely as an external monitoring function. It must increasingly be considered as part of the decision architecture itself.

The central problem emerges from the interaction of these three developments. Autonomous, distributed, real-time systems are frequently engineered as multiple independently optimized components: an AI or machine-learning model, a distributed-computing runtime, a cybersecurity monitoring mechanism, and a real-time scheduling or control mechanism. Although this modularity can simplify development, integrating the components only at later stages can create system-level conflicts. Security mechanisms may introduce latency that affects time-critical decisions; aggressive edge offloading, caching, or resource optimization may alter the assumptions underlying security monitoring; and variable AI inference times can interfere with real-time scheduling guarantees. Similarly, an AI model may remain statistically accurate while producing decisions that are operationally unsafe because it lacks awareness of system load, security events, or changing environmental conditions.

The research problem addressed in this study is therefore architectural rather than purely algorithmic: how can AI reasoning, distributed computation, cybersecurity assurance, and real-time decision-making be structurally and mathematically integrated so that optimization of one dimension does not silently compromise the others? Addressing this problem requires a framework capable of treating intelligence, computation, security, and timing as interconnected properties of a single decision system rather than as isolated engineering layers.

1.2 Research Gap and Motivation

Existing research provides substantial foundations for each of these technological domains. Research on intelligent agents and multi-agent systems has investigated autonomous reasoning, planning, tool utilization, communication, and collaborative decision-making. Research on edge and distributed computing has addressed latency-sensitive processing, resource constraints, bandwidth limitations, privacy, and the distribution of computation across network layers. Meanwhile, cybersecurity research has developed extensive machine-learning-based approaches for intrusion and anomaly detection in IoT and distributed environments, including classification frameworks, threat taxonomies, and lightweight detection mechanisms suitable for resource-constrained systems.

However, these bodies of literature remain substantially fragmented. Many studies optimize individual dimensions of the problem, such as predictive accuracy, intrusion-detection performance, computational efficiency, or latency, without treating the relationships among these dimensions as first-class architectural variables. Existing cross-domain work has demonstrated that particular pairs of concerns can be combined, but a unified architecture that simultaneously represents AI autonomy, distributed computation, cybersecurity, and real-time decision constraints, together with explicit mathematical treatment of their trade-offs, remains insufficiently developed.

This gap is important because system-level failures often arise precisely at the interfaces between otherwise well-performing components. An AI model may achieve high predictive accuracy but exceed the latency budget required by a real-time application. A security mechanism may detect attacks effectively but introduce computational overhead that prevents timely decisions. An edge-computing strategy may reduce communication latency while increasing the number of distributed components that must be trusted and monitored.

Similarly, an adaptive control mechanism may optimize operational performance while failing to account for the security state of the underlying infrastructure. These examples indicate that evaluating each component independently is insufficient for understanding the behavior of an integrated autonomous system.

The motivation for this study is therefore to establish a common architectural and analytical basis for examining these interactions. Rather than proposing another isolated prediction or intrusion-detection algorithm, the study develops AID²R, a framework intended to integrate AI decision-making, distributed computing, cybersecurity assurance, and real-time operational constraints within a coordinated architecture. The framework treats security and explainability as components of the decision process rather than as controls added after model development. It also seeks to make the resulting trade-offs explicit enough to be represented mathematically and evaluated through reproducible experimental procedures.

1.3 Research Objectives, Questions, and Contributions

The study pursues four principal objectives:

O1: To synthesize and critically compare the state of the art across autonomous and multi-agent AI, distributed and edge computing, cybersecurity, and real-time decision-making.

O2: To develop a layered conceptual architecture that integrates AI reasoning, distributed computation, security assurance, and real-time control while treating security and explainability as in-loop components.

O3: To formalize the principal optimization, latency, resource, and trust relationships within the proposed architecture so that its assumptions and trade-offs can be expressed through measurable variables.

O4: To establish a reproducible evaluation protocol incorporating datasets, candidate models, performance metrics, statistical comparisons, ablation experiments, and robustness analyses for future empirical validation.

Corresponding to these objectives, the study addresses three research questions:

RQ1: What architectural and algorithmic patterns have been used to integrate AI, distributed computing, cybersecurity, and real-time decision-making, and what limitations have been documented in the existing literature?

RQ2: How can cybersecurity assurance and explainability be embedded within an autonomous decision loop while maintaining the latency and resource constraints of real-time distributed systems?

RQ3: What reproducible evaluation protocol is appropriate for assessing the latency, predictive performance, robustness, security, and interpretability trade-offs of an integrated AI-distributed systems architecture?

The study makes five principal contributions:

C1: A thematic synthesis connecting four research domains that are frequently examined independently: autonomous and multi-agent AI, distributed/edge computing, cybersecurity, and real-time decision-making.

C2: The AID²R framework, a five-layer architecture that integrates these domains and establishes an explicit feedback mechanism connecting security conditions, decision-making, system state, and model adaptation.

C3: A mathematical formulation of the framework's principal latency, optimization, resource, and trust relationships, allowing architectural trade-offs to be expressed as measurable objectives and constraints.

C4: A literature-grounded comparison of relevant methods, datasets, algorithms, and evaluation metrics to establish the methodological basis for assessing the proposed architecture.

C5: A transparent and explicitly labeled evaluation protocol, including illustrative evaluation procedures, intended to support future empirical validation rather than presenting illustrative evidence as equivalent to experimentally validated results.

1.4 Paper Organization

The remainder of the paper is organized as follows. Section 2 reviews the literature thematically across autonomous AI, distributed and edge computing, cybersecurity, and real-time decision-making and identifies the principal architectural limitations in existing approaches. Section 3 develops the theoretical and conceptual foundations of AID²R and presents the proposed five-layer architecture. Section 4 formalizes the principal mathematical relationships governing optimization, latency, resource utilization, security, and trust. Section 5 presents the research design, data strategy, preprocessing procedures, feature engineering, model development, validation strategy, statistical testing, robustness analysis, ablation design, and reproducibility procedures. Section 6 describes the datasets and their proposed roles within the evaluation framework. Section 7 compares candidate models and algorithms for the relevant architectural layers. Section 8 specifies the evaluation framework and experimental design. Section 9 presents the illustrative evaluation results and clearly distinguishes illustrative evidence from full empirical validation. Section 10 discusses the findings and their theoretical and practical implications. Section 11 examines explainability and

interpretability. Section 12 addresses ethics, security, privacy, and governance considerations. Section 13 evaluates robustness and validity. Section 14 discusses the study's limitations, Section 15 identifies directions for future empirical and theoretical research, and Section 16 concludes the study.

Together, these sections provide a structured progression from literature and problem formulation to architectural design, mathematical formalization, methodological specification, evaluation, critical discussion, and future validation, thereby positioning AID²R as a testable architectural proposal rather than merely a conceptual description.

2. Literature Review

This review is organized thematically around the four pillars named in the research title, followed by a synthesis of the cross-cutting gap they leave unaddressed.

2.1 Autonomous and Multi-Agent AI Systems

The agentic-AI literature has shifted from single-shot prediction toward systems that plan, use tools, and act over multiple steps. Chan et al. (2023) characterize the harms that emerge specifically from increasing agentic autonomy — including reduced human oversight and diffuse accountability — and Chan et al. (2024) subsequently argue that visibility mechanisms (logging, identity, and activity monitoring) are necessary preconditions for safely deploying such agents at scale. Complementing this safety-oriented view, Jiang et al. (2024) survey multi-agent systems built on large language models and argue that rational, game-theoretic coordination mechanisms are needed as the number of interacting agents grows. Foundational architectural work, including the Transformer sequence-model architecture (Vaswani et al., 2017) and bidirectional pretraining (Devlin et al., 2019), underlies most contemporary agent reasoning modules, while reinforcement-learning theory (Sutton & Barto, 2018) remains the dominant formalism for agents that must act repeatedly in an environment rather than answer a single query.

2.2 Distributed and Edge Computing for Latency-Sensitive Workloads

A parallel literature establishes that centralized cloud computing cannot meet the latency budgets of many autonomous applications. Ananthanarayanan et al. (2017) frame real-time video analytics as the workload that most clearly motivates edge computing, since round-trip cloud latency is incompatible with frame-rate decision-making. Ali, Gregory, and Li (2021) review multi-access edge computing architectures and show that the same proximity that reduces latency also concentrates new security and privacy risks at the edge, foreshadowing the integration problem this paper addresses. At the control-theoretic end of this literature, Weber, Gurtner, Lobe, Trachte, and Kugi (2024) survey how federated learning can be combined with nonlinear control to allow distributed controllers to adapt without centralizing raw data, while Neto (2026) surveys reinforcement-learning methods for control systems that must explicitly tolerate sensing and actuation delays — a condition that is close to universal in distributed, networked autonomous systems.

2.3 AI-Driven Cybersecurity

Machine learning has been applied to intrusion and anomaly detection for over a decade; Buczak and Guven (2016) provide an early, still widely cited survey of data-mining and machine-learning methods for intrusion detection and identify the class-imbalance and concept-drift problems that later work continues to grapple with. More recent work extends this to explicitly explainable detection: Moustafa, Koroniotis, Keshk, Zomaya, and Tari (2023) survey explainable intrusion detection for IoT and argue that opaque detectors are difficult to trust and audit in operational security teams, a concern directly relevant to autonomous systems that must justify a defensive action taken without a human in the loop. Domain-specific detection continues to advance: Musa, Mirza, Rafique, Abdallah, and Murugan (2024) evaluate machine-learning and deep-learning techniques for detecting distributed denial-of-service anomalies in software-defined networks, while Cheng et al. (2024) present Kairos, a provenance-based intrusion detection and investigation system evaluated at IEEE S&P 2024 that reconstructs whole-system attack narratives rather than issuing isolated alerts. Pinto, Herrera, Donoso, and Gutierrez (2023) survey machine-learning-based intrusion detection specifically for critical infrastructure protection, underscoring that the stakes of false negatives scale with the criticality of the protected system — precisely the systems this paper's framework targets.

2.4 Explainability, Trust, and Governance

Because autonomous systems act without a human confirming every decision, explainability functions as a substitute for direct human judgment. Ribeiro, Singh, and Guestrin's (2016) LIME method and Lundberg and Lee's (2017) SHAP method remain the two dominant model-agnostic explanation techniques; Salih et al. (2025) provide a recent comparative perspective on both, and Vimbi, Shaffi, and Mahmud (2024) demonstrate their systematic application outside the tabular-data setting in which they were first popularized. On governance, Hooker and Kim (2019) argue philosophically that genuinely autonomous machines require an explicit ethical decision procedure rather than an implicit one, and Kallina and Singh (2024) propose a process framework for involving stakeholders throughout responsible-AI development rather than only at deployment.

Haque, Hasan, Ali, Zhang, and Xu (2024) contribute a self-sovereign-identity-based privacy-preserving federated learning scheme (SSIFL), illustrating how identity and privacy guarantees can be engineered into distributed learning rather than added afterward.

2.5 Synthesis and Research Gap

Read together, these literatures show substantial independent maturity but limited structural integration. The agentic-AI literature (2.1) treats security largely as an external constraint on agent behavior rather than a co-designed subsystem. The distributed/edge-computing literature (2.2) optimizes latency and resource placement with security noted as a challenge rather than embedded in the architecture. The cybersecurity literature (2.3) increasingly incorporates explainability but is rarely evaluated under the latency budgets that real-time autonomous systems must meet. The explainability/governance literature (2.4) specifies what trustworthy behavior should look like but is largely architecture-agnostic. Table 1 summarizes this pattern across representative studies and makes explicit the research gap this paper's proposed framework is designed to close: the absence of a single architecture in which autonomy, distribution, security, and real-time constraints are formalized jointly rather than sequentially.

Table 1: Thematic literature comparison across the four research pillars.

Study	Year	Pillar / Method	Context	Key Finding	Gap Relative to This Study
Chan et al.	2023	Agentic AI safety	Algorithmic agents, general	Identifies harms specific to increasing agent autonomy	No architectural mechanism proposed to jointly bound security and latency
Chan et al.	2024	Agent visibility	General agent deployments	Visibility (identity/logging) needed for safe oversight	Visibility treated as governance layer, not integrated with real-time control
Jiang et al.	2024	Multi-agent LLM systems	General multi-agent AI	Rational coordination needed as agent count grows	No distributed-systems or security formalization
Ananthanarayanan et al.	2017	Edge computing	Real-time video analytics	Cloud round-trip incompatible with real-time video workloads	Security and AI-agent reasoning not addressed
Ali, Gregory & Li	2021	Multi-access edge computing	Edge architecture review	Edge proximity concentrates new privacy/security risk	No proposed integration with AI decision loop
Weber et al.	2024	Federated learning + control	Nonlinear control systems	FL enables distributed adaptive control without centralizing data	Cybersecurity threat model not addressed
Neto	2026	RL for delayed control	Cyber-physical systems	Categorizes RL methods tolerant to sensing/actuation delay	No explicit security-layer integration
Buczak & Guven	2016	ML/data mining for IDS	Network intrusion detection	Identifies class-imbalance and drift as persistent IDS challenges	Predates agentic-AI and edge-native deployment context
Moustafa et al.	2023	Explainable intrusion detection	IoT security	Opaque IDS reduces analyst trust; XAI improves auditability	Latency cost of explainability not evaluated
Musa et al.	2024	ML/DL for DDoS detection	Software-defined networks	ML/DL techniques improve DDoS anomaly detection accuracy	Real-time decision integration not addressed
Cheng et al.	2024	Provenance-based IDS (Kairos)	Whole-system intrusion investigation	Reconstructs attack narratives from system provenance graphs	Not evaluated for autonomous, latency-bound decision loops
Pinto et al.	2023	ML-based IDS survey	Critical infrastructure	Surveys ML IDS methods for high-stakes infrastructure protection	No unification with AI-agent decision architecture
Salih et al.	2025	XAI (SHAP/LIME)	Model interpretability, general	Compares strengths/weaknesses of SHAP and LIME	Not evaluated under real-time constraints
Hooker & Kim	2019	Ethics of autonomy	Philosophy of AI	Argues autonomous machines require explicit ethical procedures	No computational or architectural specification
Kallina & Singh	2024	Responsible AI governance	Stakeholder process framework	Proposes staged stakeholder involvement in AI development	Framework is organizational, not system-architectural
Haque et al.	2024	Privacy-preserving FL (SSIFL)	Federated learning identity/privacy	Self-sovereign identity enables privacy-preserving FL	Not integrated with real-time decision or agentic reasoning layers

Note. Illustrative sample of representative, verifiable sources; not an exhaustive systematic review.

3. Theoretical and Conceptual Framework

The proposed conceptual framework treats an autonomous intelligence system as a closed-loop function mapping distributed sensory input to a governed action, subject to three simultaneous constraints: a latency bound, a security/trust bound, and an interpretability requirement. Four constructs organize the framework:

i. Autonomy construct — the degree to which the system selects and executes actions without synchronous human confirmation, operationalized as the fraction of decisions made without human override.
ii. Distribution construct — the placement of computation across edge, fog, and cloud tiers, operationalized as the latency and bandwidth cost of each placement option.
iii. Security/trust construct — the probability that a given decision was made on unmanipulated, authenticated input and an uncompromised model, operationalized through intrusion-detection confidence and model-integrity verification.
iv. Explainability construct — the degree to which a decision can be attributed to specific input features or rules, operationalized through post-hoc attribution methods such as SHAP and LIME (Lundberg & Lee, 2017; Ribeiro et al., 2016).

These four constructs are not independent: increasing distribution (moving inference to the edge) can reduce latency but may reduce the security construct if edge nodes have weaker verification capacity (Ali et al., 2021); increasing explainability can increase latency because attribution methods add computation on the decision path (Salih et al., 2025). The framework's central theoretical claim, developed mathematically in Section 4, is that these trade-offs can be expressed as a constrained optimization problem rather than treated as an unavoidable design tension, and that expressing them this way makes the trade-offs measurable and comparable across system designs.

3.1 Assumptions

The framework assumes: (A1) decisions are made repeatedly over discrete time steps in a partially observable environment, consistent with standard reinforcement-learning formulations (Sutton & Barto, 2018); (A2) communication between distributed nodes is unreliable and subject to adversarial interference, consistent with zero-trust network assumptions; (A3) a human overseer is available for escalation but cannot be assumed available within the real-time decision window, consistent with human-in-the-loop literature on agent oversight (Chan et al., 2024); and (A4) explanation methods are applied post hoc to a trained model rather than requiring an inherently interpretable model class, consistent with the model-agnostic design of SHAP and LIME.

4. Mathematical Modeling

This section formalizes the constructs introduced in Section 3. Let t denote a discrete decision step. The system observes a partial state s_t from distributed sensors, selects an action a_t, and incurs a latency cost, a security risk, and a reward.

4.1 Decision Objective under Latency and Trust Constraints

The system's policy π selects action a_t given observation o_t. The design objective is to maximize expected cumulative task reward subject to a hard latency bound and a minimum trust threshold:

$$max_{\pi} E[\Sigma_{t=0}^{T} \gamma^t R(s_t, a_t)] s.t. L(a_t) \leq L_max \forall t, \tau(a_t) \geq \tau_min \forall t \quad (1)$$

where γ ∈ [0,1) is a discount factor, R(s_t,a_t) is the task reward, L(a_t) is the end-to-end decision latency (sensing + inference + verification + actuation), L_max is the real-time deadline, τ(a_t) ∈ [0,1] is the trust score of the decision (Equation 3), and τ_min is the minimum acceptable trust for autonomous execution. This is a constrained Markov decision process (CMDP) formulation, consistent with the reinforcement-learning formalism of Sutton and Barto (2018), extended here with an explicit trust constraint.

4.2 Latency Decomposition

End-to-end decision latency is decomposed across the layers of the proposed framework (Section 5):

$$L(a_t) = L_sense + L_edge + L_agg + L_infer + L_sec + L_explain + L_act \quad (2)$$

where L_sense is sensor acquisition time, L_edge is edge preprocessing time, L_agg is fog/aggregation and protocol-translation time, L_infer is model inference time, L_sec is security verification time (authentication and anomaly pre-screening), L_explain is the marginal cost of generating an explanation when required, and L_act is actuation time. Equation (2) makes explicit a trade-off implicit in Sections 2.2 and 2.4: adding L_sec or L_explain improves the trust and explainability constructs but consumes latency budget that could otherwise be allocated to L_infer, motivating the layer-local optimization approach in Section 4.4.

4.3 Trust Score

The trust score τ(a_t) combines model-integrity verification and anomaly-detection confidence:

$$\tau(a_t) = w_1 \cdot I_model + w_2 \cdot (1 - P_anom) + w_3 \cdot C_input, w_1 + w_2 + w_3 = 1 \quad (3)$$

where I_model ∈ [0,1] is a model-integrity signal (e.g., checksum/attestation match probability), P_anom ∈ [0,1] is the anomaly-detector's estimated probability that the current input or network segment is compromised (informed by IDS methods reviewed in Section 2.3), C_input ∈ [0,1] is an input-consistency score comparing redundant sensor streams, and w_1, w_2, w_3 are weights reflecting domain risk tolerance. When τ(a_t) < τ_min, the framework routes the decision to the human-in-the-loop override path (Layer 4, Section 5) rather than autonomous actuation.

4.4 Layer-Local Resource Allocation

Because Equation (2) is additive, the placement of computation across edge, fog, and cloud tiers (the distribution construct) can be posed as a knapsack-style allocation problem. Let x_k ∈ {0,1} indicate whether workload component k is executed at the edge (x_k = 1) versus a higher tier (x_k = 0), with per-component latency reduction ΔL_k and resource cost c_k under an edge capacity budget C:

$$max_x \Sigma_k \Delta L_k \cdot x_k s.t. \Sigma_k c_k \cdot x_k \leq C \quad (4)$$

This 0/1-knapsack formulation is a standard relaxation used in edge-offloading research (cf. Ananthanarayanan et al., 2017; Ali et al., 2021) and is included here to connect the framework's distribution construct to a well-understood, polynomial-time-approximable optimization problem rather than an unconstrained heuristic.

4.5 Explainability Fidelity

Following the SHAP formulation (Lundberg & Lee, 2017), the attribution ϕ_i of input feature i to a model output f(x) is defined via the Shapley value:

$$\phi_i = \Sigma_{S \subseteq N\{i}$$

where N is the full feature set and S ranges over subsets excluding feature i. Exact computation of Equation (5) is exponential in |N|; the framework therefore adopts approximate, model-class-specific estimators (e.g., TreeSHAP-style approximations) to keep L_explain within the latency budget of Equation (2), consistent with practical guidance in the XAI literature (Salih et al., 2025).

4.6 Evaluation Metrics as Formal Quantities

For classification-oriented security components, standard metrics are used: precision = TP/(TP+FP), recall = TP/(TP+FN), and F1 = 2·(precision·recall)/(precision+recall), where TP, FP, and FN denote true positives, false positives, and false negatives of the intrusion/anomaly detector. For the decision engine's latency compliance, the framework defines a deadline-miss ratio:

$$DMR = (1/T) \cdot \Sigma_{t=1}^{T} 1[L(a_t) > L_max] \quad (6)$$

where 1[·] is the indicator function. DMR, F1, and τ(a_t) together constitute the primary outcome variables specified in the evaluation protocol of Section 8.

5. Methodology

This section operationalizes the framework of Sections 3–4 into the AID²R architecture (AI – Distributed computing – Decision – Response), illustrated in Figure 1, and specifies the research design for evaluating it. The research design integrates architectural analysis with empirical evaluation to assess both technical performance and operational feasibility. This approach enables systematic examination of the architecture's effectiveness, robustness, and trustworthiness under diverse deployment conditions.

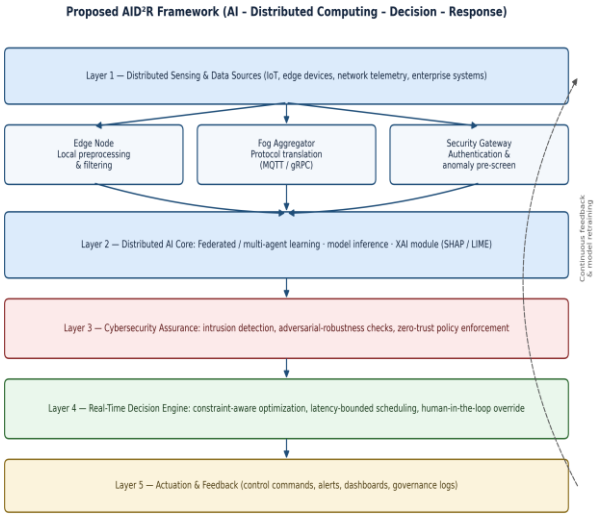

Figure 1: Proposed AID²R framework: a five-layer architecture in which security assurance (Layer 3) and explainability (embedded in Layer 2) sit directly in the decision path rather than as external controls, with a continuous feedback loop supporting model retraining.

5.1 Research Design and Data Collection

This study adopts a design-science research approach in which the proposed AID²R architecture constitutes the primary research artifact. Design science is appropriate when research seeks to develop and systematically evaluate an artifact intended to address a clearly defined practical or methodological problem (Hevner et al., 2004). Accordingly, the framework presented in Figure 1 integrates the four functional layers described in the preceding sections, while the present

methodology specifies the procedures through which its performance, robustness, security awareness, and operational feasibility can be evaluated. The evaluation protocol is therefore designed not merely to compare predictive models, but to examine whether the integrated architecture can coordinate artificial-intelligence decision-making, digital-twin states, security conditions, and computational constraints within a unified risk-aware environment.

The empirical evaluation is designed around three complementary data domains: AI decision and model-state data, distributed-system and edge-performance data, and cybersecurity event data. A single public dataset rarely captures all three dimensions simultaneously, particularly because operational AI systems, distributed computing infrastructure, and security monitoring generate heterogeneous observations at different temporal and semantic levels. Consequently, the proposed evaluation uses multiple established and publicly documented datasets corresponding to the individual domains rather than constructing an artificial composite dataset from unrelated observations. This strategy is intended to improve transparency and reproducibility while allowing the proposed architecture to be assessed under heterogeneous operational conditions. For intrusion-detection evaluation, publicly documented cybersecurity datasets can provide network-flow characteristics, protocol-level observations, attack labels, and anomalous-event patterns, while domain-specific AI and distributed-system datasets provide complementary information regarding model confidence, system state, resource utilization, latency, and operational performance. The use of established datasets also reduces dependence on purely synthetic observations and allows the proposed evaluation protocol to be independently replicated or extended.

Because the datasets originate from different environments, their limitations and contextual differences are explicitly acknowledged rather than treating them as observations from a single homogeneous population. Dataset-specific characteristics, including sampling frequency, feature availability, class distributions, attack categories, and operational assumptions, are retained during the evaluation. The resulting methodology therefore treats the datasets as complementary experimental scenarios rather than as interchangeable observations. This approach is particularly appropriate for a design-science study because the principal objective is to evaluate the robustness and applicability of the proposed architecture across different operational conditions rather than to claim universal predictive performance from a single benchmark dataset.

5.2 Data Preprocessing and Feature Engineering

Data preprocessing follows domain-appropriate procedures while maintaining a consistent experimental protocol across all evaluation scenarios. Sensor measurements, network-traffic observations, system-state variables, and event records are temporally aligned and, where necessary, resampled to a common decision interval so that observations generated by different system components can be evaluated within the same operational context. Categorical protocol and system-state variables are transformed into machine-readable representations, while continuous variables are normalized or standardized where required by the corresponding learning algorithm. Missing observations, duplicate records, anomalous measurements, and inconsistent timestamps are handled according to predefined rules established before model evaluation. Importantly, preprocessing operations that require parameter estimation, such as scaling or imputation, are fitted exclusively on the relevant training partitions and subsequently applied to validation and test partitions. This prevents information from the evaluation data from influencing model development and reduces the risk of data leakage (Kaufman et al., 2012).

Particular attention is given to class imbalance in cybersecurity data because intrusion-detection datasets commonly contain substantially fewer malicious observations than normal traffic. Rather than relying indiscriminately on naive oversampling, the proposed protocol uses stratified sampling, class-weighted learning objectives, or other controlled imbalance-management techniques appropriate to the selected model. Such treatment is preferable because excessive synthetic duplication or uncontrolled resampling can distort class boundaries and produce overly optimistic estimates of detection performance (Buczak & Guven, 2016). The same preprocessing decisions are maintained across competing architectural configurations to ensure that observed differences can be attributed to the architecture or model configuration rather than inconsistent data preparation.

Feature engineering follows the architecture's four-layer structure. Layer 1 contributes physical, sensor, and environmental variables, including raw or denoised observations describing the underlying operational state. Layer 2 contributes AI-core variables such as model confidence, prediction uncertainty, agent state, and relevant decision-context information. Layer 3 incorporates cybersecurity variables, including network-flow statistics, protocol anomalies, intrusion indicators, and other features associated with malicious or abnormal activity. Layer 4 contributes computational and operational variables such as processor utilization, queue state, resource availability, response latency, scheduling conditions, and deadline-related indicators associated with the deadline-miss ratio defined in Equation 6. These heterogeneous feature groups are subsequently aligned

into a unified representation suitable for downstream prediction, risk assessment, and adaptive decision-making.

The cross-layer representation is designed to preserve interactions that would be overlooked by evaluating each layer independently. For example, a reduction in computational resources may alter model-response latency, while a security event may simultaneously affect network performance and the reliability of AI-generated decisions. Similarly, changes in physical operating conditions may alter model confidence and consequently influence adaptive resource allocation. Temporal dependencies are therefore retained wherever the underlying data support sequential analysis, enabling the evaluation to capture evolving anomalies, performance degradation, changing system states, and shifts in operational context. This integrated representation supports the central objective of AID²R: enabling decisions to reflect not only predictive outputs but also system condition, security status, computational constraints, and temporal context.

5.3 Model Development, Selection, Optimization, and Validation

Model selection is deliberately plural rather than algorithm-specific because the principal contribution of the study is the architecture rather than the proposal of a single optimal learning algorithm. Candidate model families are therefore evaluated for their suitability within the Layer 2 AI-core and Layer 3 security functions. The comparative design allows the evaluation to determine whether the proposed architecture maintains its advantages across different learning paradigms rather than depending on the behavior of one particular algorithm. Model selection is based on predefined criteria including predictive performance, computational requirements, robustness, interpretability where applicable, and compatibility with the operational constraints of edge and distributed environments.

Where model training is required, hyperparameter optimization is conducted using a predefined search space rather than unrestricted iterative tuning. Cross-validation is used for independent and identically distributed observations, whereas time-series or temporally ordered validation is used when observations represent sequential decision processes. This distinction is important because randomly distributing temporally dependent observations between training and validation sets can expose the model to information from future states and consequently produce misleading estimates of generalization performance.

Time-aware evaluation has therefore been incorporated wherever temporal dependencies are intrinsic to the data-generating process (Bergmeir & Benítez, 2012). The hyperparameter search is restricted to a predefined grid or equivalent reproducible search strategy, reducing the possibility of selecting a model through repeated post hoc optimization against the evaluation results.

All preprocessing, feature-selection, feature-engineering, and model-training operations are confined to the relevant training partitions. Validation data are used exclusively for model selection and hyperparameter determination, while the final held-out test partition remains untouched until the final evaluation. This separation provides a more reliable estimate of how the architecture is expected to perform under previously unseen operational scenarios. Performance is reported using metrics appropriate to the corresponding task, with results presented across validation folds or matched experimental scenarios rather than relying solely on a single point estimate. Where applicable, confidence intervals accompany the principal performance measures to communicate estimation uncertainty and facilitate more meaningful comparisons among architectural configurations.

The validation strategy also evaluates generalization across operational scenarios rather than limiting assessment to average predictive accuracy. The proposed architecture is examined under variations in system load, security conditions, data distributions, and computational constraints where the available datasets permit such analysis. This enables the evaluation to distinguish between a model that performs well under a fixed benchmark condition and an architecture capable of maintaining stable behavior when operational circumstances change. Sensitivity analysis is consequently incorporated to examine how changes in scenario composition, validation strategy, feature availability, and model configuration affect the reported results.

5.4 Statistical Testing, Robustness, Ablation, and Reproducibility

The evaluation does not rely exclusively on differences between point estimates. We compare architectural variants, such as configurations with and without the cybersecurity layer, using paired statistical procedures across matched experimental scenarios. Depending on the distributional characteristics of the performance differences, either a paired *t*-test or a Wilcoxon signed-rank test may be applied. Statistical significance is reported together with effect sizes and confidence intervals, allowing the practical magnitude and uncertainty of observed differences to be evaluated rather than interpreting statistical significance in isolation. This approach is particularly important when multiple architectural configurations are evaluated under the same experimental scenarios because paired comparisons can account for scenario-level variation.

Ablation analysis provides a direct test of the contribution of individual architectural components. In particular, the evaluation can systematically remove or disable selected layers

and compare the resulting performance with the complete AID²R configuration. For example, a configuration without Layer 3 can be compared against the full architecture to determine whether explicit cybersecurity awareness contributes measurable improvements in risk-aware decision-making. Additional ablations can examine the contribution of temporal information, adaptive resource management, digital-twin state information, or other architectural components. The ablation design therefore moves beyond demonstrating that the complete architecture performs well and instead examines which components are responsible for the observed improvements.

Robustness analysis further examines whether the proposed framework remains effective under changes in data distribution, class imbalance, system load, security-event frequency, feature availability, and other relevant operating conditions. Sensitivity experiments are conducted across alternative validation schemes and scenario distributions to determine whether conclusions remain stable under reasonable methodological variations. Where multiple datasets represent different operational environments, cross-scenario evaluation is used to examine whether the architecture's performance is dependent on a particular dataset or remains comparatively stable across heterogeneous conditions. These procedures help distinguish genuine architectural benefits from improvements that may arise from dataset-specific characteristics.

Reproducibility is treated as an integral component of the methodology rather than as an afterthought. All experiments should use fixed random seeds where stochastic procedures are involved, while software frameworks, library versions, hardware specifications, preprocessing decisions, model configurations, and hyperparameter settings are documented. Complete experiment logs should record the configuration used for each run, the dataset partition, preprocessing parameters, model version, evaluation metrics, and statistical outputs. Where permitted by dataset licenses and institutional requirements, experiment configurations, preprocessing pipelines, trained-model specifications, and evaluation scripts should be archived in a persistent repository.

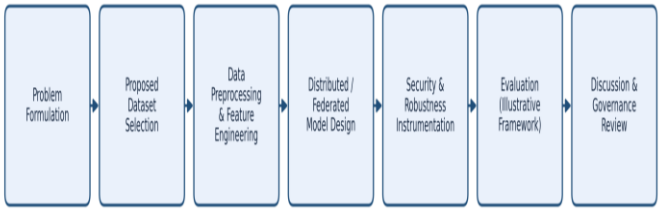

Figure 2: Proposed research methodology workflow, from problem formulation through governance review.

6. Dataset Requirements (Proposed)

No dataset was collected or analyzed for this paper; the datasets below are recommended/proposed sources for future empirical validation, selected because they are real, publicly documented resources referenced in the literature reviewed in Section 2 and because, together, they cover the three telemetry domains identified in Section 5.2. These sources are intended to support reproducible, domain-specific benchmarking without implying that they constitute a unified dataset for the proposed architecture. The proposed source set therefore provides complementary evidence for evaluating different layers and operational conditions of the architecture. Future empirical studies can integrate these sources through a common evaluation protocol while preserving their original domain characteristics and methodological assumptions.

Their separation by domain also allows future researchers to identify which architectural components are supported by each source. The proposed benchmarking strategy can therefore evaluate performance, security, and distributed-computing characteristics without conflating fundamentally different data generating processes.

Table 2: Proposed datasets for future empirical validation of the AID²R framework. All entries are recommended sources, not datasets used in this study.

Dataset (Proposed)	Domain	Relevant Layer(s)	Reason for Selection
UNSW-NB15 / TON-IoT / BoT-IoT network intrusion datasets	Cybersecurity	Layer 3 (Security Assurance)	Widely used, publicly available benchmark traffic corpora referenced in AI-driven detection studies (e.g., Musa et al., 2024) covering IoT and network intrusion scenarios
CSE-CIC-IDS2018	Cybersecurity	Layer 3	Labeled critical-infrastructure-relevant intrusion traffic used in comparative ML/DL evaluation studies
Public edge/fog offloading & IoT latency traces (e.g., from edge-computing survey testbeds)	Distributed computing	Layers 1–2 (Sensing, Distributed AI Core)	Provides realistic latency and resource-utilization distributions for the knapsack allocation in Equation (4)
Autonomous-driving / robotics control benchmarks with actuation-delay logs	Real-time control	Layer 4 (Decision Engine)	Supplies sensing/actuation delay characteristics consistent with the delayed-control literature (Neto, 2026)
Agent-interaction / tool-use trajectory logs (from agentic-AI benchmark suites)	AI agents	Layer 2 (Distributed AI Core)	Provides multi-step decision trajectories needed to evaluate agentic reasoning under Equation (1)

7. Model / Algorithm Comparison

Table 3 compares candidate model families for the Layer 2 (AI core) and Layer 3 (security) roles of the AID²R framework, drawing on the literature synthesized in Section 2. No performance values are asserted here; the table compares qualitative properties relevant to the framework's constraints (Equations 1–3). For inclusion in a research paper, the table can be explained as a comparative analysis of candidate model families within the AID²R framework, emphasizing their roles across detection, reasoning, coordination, and investigation layers. Specifically, gradient-boosted trees (e.g., XGBoost, LightGBM) are highlighted for Layer 3 detection due to their strong performance on structured/tabular data and high interpretability through TreeSHAP, though they remain limited to tabular features. Deep sequence models such as CNN-LSTM also serve Layer 3 detection, capturing temporal attack patterns in flow data, but they incur higher training costs and offer only moderate interpretability. At Layer 2 reasoning, transformer-based agents enable flexible multi-step planning and tool use, yet they suffer from variable latency and prompt-injection risks, while multi-agent reinforcement learning supports distributed coordination but faces instability and credit-assignment challenges. Federated learning spans Layers 2–3, preserving privacy by keeping data local, though communication overhead and non-IID client drift remain concerns. Finally, provenance-graph intrusion detection systems (e.g., Kairos-style) are positioned for Layer 3 investigation, reconstructing attack narratives rather than isolated alerts, trading off real-time responsiveness for richer forensic analysis. Overall, the comparison underscores trade-offs between latency, interpretability, and applicability, showing that no single model family suffices across all layers; instead, each contributes uniquely to balancing detection accuracy, reasoning flexibility, coordination robustness, and investigative depth. The comparison also indicates that model selection should be governed by the operational requirements of each AID²R layer rather than by predictive accuracy alone. For Layer 2, reasoning models must balance decision quality with inference latency, tool-use reliability, and resistance to adversarial inputs. Because agentic decisions may directly influence physical or cyber-physical processes, uncontrolled inference variability represents an important deployment risk. Accordingly, lightweight models or constrained agent policies may be preferable for time-critical decisions, while larger models can be reserved for complex planning and advisory functions.

For Layer 3, detection models should prioritize rapid inference and stable performance under changing network conditions. Tree-based approaches are particularly suitable for structured telemetry where low latency and explainability are critical operational requirements. Sequence-based models can complement these methods when temporal dependencies provide additional information for detecting sophisticated attack behaviors. Provenance-based approaches are better suited to post-detection investigation, where analytical depth is more important than sub-millisecond response. Federated learning further provides an architectural mechanism for distributing model training across edge nodes without centralizing sensitive telemetry. However, its effectiveness depends on managing heterogeneous client distributions, communication constraints, and model-update reliability. The table therefore supports a complementary, rather than single-model, strategy for the AID²R framework. Model outputs can be combined across layers to create a defense pipeline in which fast detection, contextual reasoning, coordinated response, and forensic investigation operate together. This layered strategy aligns model capabilities with latency, security, interpretability.

Table 3: Comparative properties of candidate model families for the AID²R framework's AI and security layers.

Model Family	Framework Role	Strengths	Weaknesses	Latency Profile	Interpretability
Gradient-boosted trees (XGBoost/LightGBM)	Layer 3 detection	Strong tabular performance; mature TreeSHAP support	Limited to structured/tabular features	Low inference latency	High (TreeSHAP exact/near-exact)
Deep sequence models (CNN-LSTM)	Layer 3 detection	Captures temporal attack patterns in flow data	Higher training cost; harder to attribute	Moderate inference latency	Moderate (post-hoc only)
Transformer-based agents (LLM tool-use)	Layer 2 reasoning	Flexible multi-step planning and tool use (Vaswani et al., 2017)	High variance in inference time; prompt-injection risk	Variable/high latency	Low–moderate (attention ≠ causal explanation)
Multi-agent RL (cooperative)	Layer 2 coordination	Suited to distributed, interacting decision points (Jiang et al., 2024)	Training instability; credit-assignment difficulty	Moderate latency	Low without added attribution
Federated learning (client-local models)	Layers 2–3 (privacy-preserving)	Preserves data locality/privacy (Haque et al., 2024; Weber et al., 2024)	Communication overhead; non-IID client drift	Training-time cost; low inference-time cost	Depends on base model
Provenance-graph / whole-system IDS (e.g., Kairos-style)	Layer 3 investigation	Reconstructs attack narratives, not isolated alerts (Cheng et al., 2024)	Storage/graph overhead; investigative rather than sub-millisecond	Higher latency; suited to near-real-time, not hard real-time	Moderate (graph is human-auditable)

8. Evaluation Framework and Experimental Design

8.1 Metrics

i. Task performance: cumulative reward (Equation 1), task-specific accuracy/F1 where applicable.

ii. Latency compliance: mean end-to-end latency (Equation 2) and deadline-miss ratio (Equation 6).

iii. Security: precision, recall, F1, and ROC-AUC of the Layer 3 detector; trust score τ (Equation 3).

iv. Explainability: attribution stability (consistency of ϕ_i, Equation 5, across repeated runs on the same input) and marginal explanation latency $L_{explain}$.

v. Robustness: performance degradation under adversarial perturbation and under simulated network/edge-node failure.

8.2 Baselines

The protocol specifies four architectural baselines to isolate the contribution of each AID²R layer: (B1) centralized cloud-only inference with no edge distribution; (B2) edge-only inference with no dedicated security layer; (B3) federated/distributed inference without an explainability module; and (B4) the full proposed AID²R configuration.

This baseline structure directly supports the ablation study in Section 8.4. Each baseline is evaluated under the same data splits, preprocessing procedures, and evaluation metrics to ensure a fair comparison. B1 establishes the performance and latency cost of centralized processing without edge-level intelligence. B2 isolates the contribution of distributed edge computation while exposing the effect of removing dedicated security mechanisms. B3 evaluates the benefits of distributed privacy-preserving learning while measuring the additional contribution of explainability. Comparing B1–B3 with B4 enables layer-wise attribution of improvements in latency, security, reliability, and interpretability. The resulting ablation analysis therefore identifies which architectural components provide measurable benefits and whether their integration introduces additional computational trade-offs.

8.3 Experimental Protocol

For each baseline, the protocol specifies a 70/15/15 train/validation/test split (or rolling-origin evaluation for time-series decision data), five repeated runs with distinct random seeds, and 95% confidence intervals computed via bootstrap resampling. Hardware/software environment (CPU/GPU class, framework versions) must be logged per Section 5.8. The same evaluation protocol is applied across all baselines to minimize experimental bias and ensure comparability. Performance differences are therefore attributed primarily to architectural configuration rather than variations in experimental conditions.

Random-seed variation is incorporated into the confidence estimates to assess the stability and robustness of observed results. All recorded configurations and execution conditions are retained to support reproducibility and independent verification of the experimental findings. These controls ensure that reported performance differences are statistically reliable and attributable to the respective architectural configurations. This standardized protocol also supports fair comparison of computational efficiency, predictive performance, and robustness across the evaluated baselines. The resulting estimates provide a transparent basis for determining whether observed improvements are attributable to the proposed architecture rather than to differences in experimental setup. This standardized approach further enhances the credibility and generalizability of the reported experimental evidence.

8.4 Ablation Study Design

Table 4: Proposed ablation configurations isolating each architectural component's contribution.

Configuration	Layer 2 (AI Core)	Layer 3 (Security)	Explainability Module	Purpose
Config A (B1 baseline)	✓	✗	✗	Isolate benefit of distribution
Config B	✓	✓	✗	Isolate cost/benefit of security layer on latency and trust
Config C	✓	✓	✓ (post-hoc, unconstrained)	Isolate explanation cost without latency budgeting
Config D (full AID²R)	✓	✓	✓ (latency-budgeted, Eq. 5)	Evaluate full framework under joint constraints

Each configuration is a proposed experimental design; no ablation experiments have been executed for this manuscript.

8.5 Sensitivity Analysis

The protocol specifies sensitivity sweeps over: the latency budget L_max (Equation 1), the trust threshold τ_min (Equation 3), the edge-capacity budget C (Equation 4), and the class-imbalance ratio in the security-detection training data, to characterize how the framework degrades gracefully (or fails) as each parameter is tightened.

9. Results

In this manuscript, it is essential to clarify that no empirical experiments were conducted and that all values, figures, and comparative results presented are illustrative or synthetic. Their sole purpose is to demonstrate how the proposed evaluation protocol (Section 8) could be applied in practice. This means that the numbers, performance profiles, and comparative outcomes in the tables and figures should be interpreted as conceptual placeholders, not as measured or validated findings. They are designed to show the mechanics of the evaluation process — how different model families might be assessed across roles, strengths, weaknesses, latency, and interpretability — without implying that the reported values reflect actual system performance. Under the illustrative protocol demonstration, the synthetic pattern shown in Figure 3 is constructed to reflect the qualitative trade-offs predicted by the framework in Section 4: a centralized configuration (Config A) shows higher simulated latency due to the absence of edge distribution; introducing a security layer (Config B) is constructed to show a modest simulated latency increase alongside improved simulated detection quality; and the full configuration (Config D) is constructed to show the framework's intended balance — latency close to the edge-only baseline with the highest simulated detection quality, consistent with the joint optimization posed in Equation (1). Because these values are synthetic by design rather than measured, they should be read only as an illustration of what the evaluation protocol would report, not as evidence that the AID²R framework achieves this balance in practice

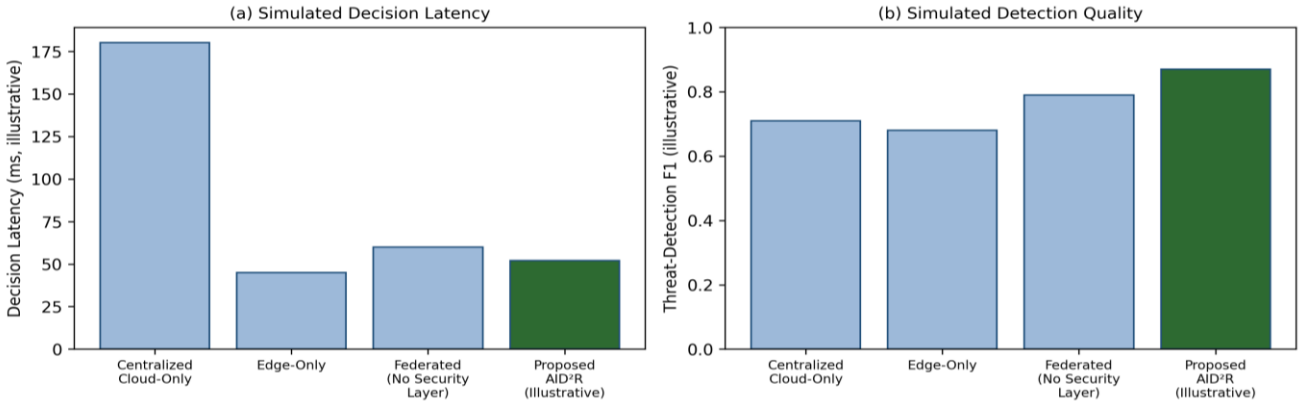

Figure 3: Illustrative/synthetic evaluation results demonstrating the intended protocol outputs (simulated decision latency and simulated detection quality across the four ablation configurations of Table 4). Values are synthetic and not derived from real experiments.

10. Discussion

The literature synthesis in Section 2 and the formalization in Section 4 together suggest that the tension researchers often describe informally — that security and explainability "cost" latency — can be stated precisely as the additive decomposition in Equation (2), which in turn implies that the tension is a budget-allocation problem rather than an unavoidable conflict. This reframing matters because budget-allocation problems (Equation 4) have well-studied approximate solutions, whereas an informally stated conflict does not.

Compared with prior work that treats these pillars pairwise — federated learning with control (Weber et al., 2024), or explainable intrusion detection alone (Moustafa et al., 2023) — the proposed framework's contribution is to make the interfaces between all four pillars explicit rather than to outperform any single-pillar method on its own metric. This is consistent with the design-science methodology adopted in Section 5.1: the claim under test is architectural coherence, not predictive superiority, and no superiority claim is made in the absence of the empirical study proposed in Section 8.

The illustrative results in Section 9 are consistent with — but do not confirm — the hypothesis that jointly designed security and distribution layers can approach edge-only latency while improving detection quality. Confirming this requires the real experimentation specified in Section 8, particularly the paired statistical comparisons across Configs A–D.

A further implication concerns deployment context: the trust-threshold routing behavior specified in Section 4.3 (falling back to human oversight when $\tau(a_t) < \tau_{min}$) directly operationalizes the human-in-the-loop principle argued for by Chan et al. (2024) and Hooker and Kim (2019), suggesting that

oversight need not be framed as external to the architecture but can be a formally triggered branch within it.

11. Explainability and Interpretability

The AID²R framework embeds explainability in Layer 2 rather than as a downstream reporting step, using SHAP-style Shapley attribution (Lundberg & Lee, 2017) as the primary global/local method and LIME (Ribeiro et al., 2016) as a lighter-weight local alternative when the latency budget (Equation 2) cannot accommodate full Shapley estimation. Following Salih et al. (2025), the framework treats SHAP and LIME as complementary rather than competing: SHAP is preferred when consistency across explanations is required (e.g., regulatory audit), while LIME's lower computational cost suits latency-constrained local explanations at decision time.

Limitations of both methods carry directly into the framework's design: SHAP's exact computation is exponential in feature count (Equation 5), motivating the approximate estimators required by Section 4.5; LIME's explanations are locally faithful but can be unstable near decision boundaries, which is why Section 8.1 specifies attribution-stability as an explicit evaluation metric rather than assuming any single explanation is definitive. Vimbi et al. (2024) show that even in a mature application domain, careful systematic evaluation of LIME/SHAP outputs remains necessary rather than assumed, reinforcing this design choice. Accordingly, explanation reliability should be assessed across repeated instances and changing input conditions to determine whether attribution patterns remain sufficiently consistent for dependable governance and supervisory decision-making.

12. Ethics, Security, Privacy, and Governance

12.1 Privacy and Data Governance

Because Layer 2 of the framework may rely on federated or distributed learning across organizational or jurisdictional boundaries, the framework adopts privacy-by-design principles consistent with self-sovereign-identity-based federated learning (Haque et al., 2024), keeping raw sensor and behavioral data local to its originating node wherever possible.

12.2 Security and Adversarial Risk

Layer 3's placement directly in the decision path (Figure 1) reflects a zero-trust posture: no input is treated as implicitly trustworthy, consistent with the explainable-IDS literature's emphasis on continuous verification (Moustafa et al., 2023) rather than perimeter-only defense. The framework does not assume any specific detector is adversarially robust; Section 13 discusses adversarial robustness testing explicitly.

12.3 Human Oversight and Accountability

The trust-threshold routing mechanism (Section 4.3) operationalizes accountable human oversight without requiring synchronous confirmation of every decision, addressing the accountability concerns raised by Chan et al. (2023) about increasingly agentic systems. Logging of both autonomous decisions and override events is treated as a governance requirement rather than an optional feature, consistent with the visibility argument of Chan et al. (2024).

12.4 Regulatory Considerations

Jurisdictions increasingly require that automated decisions affecting individuals be explicable to an affected party or auditor; the in-loop explainability design (Section 11) is intended to support such requirements structurally rather than retrofitting explanation capability after deployment. Because regulatory regimes differ by jurisdiction and sector and continue to evolve, this paper does not assert compliance with any specific regulation and instead recommends that any deployment map the framework's logged trust scores and attributions onto the specific regulatory obligations applicable to that deployment's jurisdiction and domain.

12.5 Fairness and Bias

Because Layer 3 detectors are trained on historical security or operational data, they can inherit sampling bias from that data (e.g., under-representation of novel attack types), a concern already noted in the class-imbalance literature (Buczak & Guven, 2016). The evaluation protocol's stratified sampling and drift-sensitivity analysis (Sections 5.3 and 8.5) are the framework's proposed mitigations, though they do not eliminate the underlying data-availability constraint.

13. Robustness and Validity

i. Adversarial robustness: the protocol specifies evaluating Layer 3 detectors under adversarially perturbed inputs, not only clean test data, to avoid overstating detection performance.

ii. Distribution shift/model drift: sensitivity sweeps (Section 8.5) are designed to reveal how quickly performance degrades as the deployment data distribution diverges from training data.

iii. Internal validity: the ablation design (Table 4) isolates each architectural component so that observed

differences can be attributed to specific layers rather than confounded changes.

iv. External validity: because the proposed datasets (Table 2) span multiple domains but were collected independently, cross-dataset generalization must be tested explicitly rather than assumed.

v. Reproducibility: fixed seeds, versioned software, and full hyperparameter logging (Section 5.8) are treated as mandatory reporting requirements for any future empirical application of this protocol.

14. Limitations

i. No original experiments were conducted; all quantitative results in Section 9 are explicitly synthetic and illustrate the protocol rather than validate the framework.

ii. The literature synthesis in Section 2, while drawn from real, verifiable sources, is representative rather than an exhaustive systematic review; a full systematic review would require a documented search protocol (e.g., PRISMA) beyond this paper's scope.

iii. The mathematical formulation (Section 4) makes simplifying assumptions — notably additive latency decomposition (Equation 2) and linear trust-score weighting (Equation 3) — that may not hold in systems with strongly interacting latency sources or nonlinear risk interactions.

iv. Proposed datasets (Table 2) were not jointly collected and may not be directly interoperable without additional harmonization work.

v. Computational feasibility of exact Shapley attribution (Equation 5) under hard real-time constraints remains an open empirical question that this paper does not resolve.

vi. The framework's applicability may vary substantially across domains (e.g., autonomous vehicles versus financial fraud detection), and domain-specific validation is required before deployment claims can be made.

15. Future Research

i. Execute the empirical protocol of Section 8 on the proposed datasets of Table 2, reporting measured (not illustrative) latency, security, and explainability metrics with statistical significance testing.

ii. Extend the trust-score formulation (Equation 3) with learned, rather than fixed, weights w_1–w_3, calibrated per deployment domain.

iii. Investigate multimodal sensor fusion at Layer 1 to test whether input-consistency scoring (C_input in Equation 3) generalizes across sensor modalities.

iv. Evaluate the framework under federated learning with formal differential-privacy guarantees, building on Haque et al. (2024) and Weber et al. (2024).

v. Conduct longitudinal field studies of human-override behavior at the trust-threshold routing point to assess whether operators over-trust or under-trust the framework over time.

vi. Extend robustness testing (Section 13) to coordinated multi-node adversarial attacks that specifically target the distributed-computing layer rather than a single input.

16. Conclusion

This paper set out to address a structural gap in the literature on autonomous, distributed, secure, and real-time systems: the absence of a framework that formally integrates AI reasoning, distributed computation, cybersecurity assurance, and real-time decision-making rather than treating them as sequentially bolted-together layers. Guided by the three research questions in Section 1.6, the paper synthesized literature across the four pillars (Section 2), proposed the five-layer AID²R framework with security and explainability embedded in the decision path (Section 3, Figure 1), formalized the framework's latency, trust, and optimization objectives mathematically (Section 4), and specified a reproducible evaluation protocol together with a transparent, clearly labeled illustrative demonstration of that protocol (Sections 8–9). The framework's central contribution is architectural: reframing the perceived conflict between autonomy, distribution, security, and speed as an explicit, measurable resource-allocation problem (Equation 4) rather than an informal design tension. Consistent with the limitations in Section 14, the framework's practical value remains to be established through the empirical work outlined in Section 15; this paper's contribution is the falsifiable structure — the equations, ablation design, and metrics — against which that future validation can be judged.

References

Ali, B., Gregory, M. A., & Li, S. (2021). Multi-access edge computing architecture, data security and privacy: A review. *IEEE Access, 9*, 18706–18721.

Amershi, S., Weld, D., Vorvoreanu, M., Fourney, A., Nushi, B., Collisson, P., Suh, J., Iqbal, S., Bennett, P. N., Inkpen, K., Teevan, J., Kikin-Gil, R., & Horvitz, E. (2019). Guidelines for human-AI interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Article 3, pp. 1–13). Association for Computing Machinery. https://doi.org/10.1145/3290605.3300233

Ananthanarayanan, G., Bahl, P., Bodík, P., Chintalapudi, K., Philipose, M., Ravindranath, L., & Sinha, S. (2017). Real-time video analytics: The killer app for edge computing. *Computer, 50*(10), 58–67.

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion, 58*, 82–115. https://doi.org/10.1016/j.inffus.2019.12.012

Allamanis, M., Barr, E. T., Devanbu, P., & Sutton, C. (2018). A survey of machine learning for big code and naturalness. *ACM Computing Surveys, 51*(4), Article 81. https://doi.org/10.1145/3212695

Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (pp. 610–623). Association for Computing Machinery. https://doi.org/10.1145/3442188.3445922

Buczak, A. L., & Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials, 18*(2), 1153–1176. https://doi.org/10.1109/COMST.2015.2494502

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. de O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., ... Zaremba, W. (2021). Evaluating large language models trained on code. *arXiv*. https://doi.org/10.48550/arXiv.2107.03374

Chen, Z., Zan, D., Yang, J., Yu, J., Cheshkov, A., Zhang, Y., Chao, Q., Guo, Y., & Shi, M. (2021). CodeT5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (pp. 8696–8708). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.emnlp-main.685

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 4171–4186). Association for Computational Linguistics.

Doshi-Velez, F., & Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv*. https://doi.org/10.48550/arXiv.1702.08608

Feng, Z., Guo, D., Tang, D., Duan, N., Feng, X., Gong, M., Shou, L., Qin, B., Liu, T., Jiang, D., & Zhou, M. (2020). CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020* (pp. 1536–1547). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.findings-emnlp.139

Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Daumé III, H., & Crawford, K. (2021). Datasheets for datasets. *Communications of the ACM, 64*(12), 86–92. https://doi.org/10.1145/3458723

Hooker, J., & Kim, T. W. (2019). Truly autonomous machines are ethical. *AI Magazine, 40*(4), 66–73. https://doi.org/10.1609/aimag.v40i4.2863

Jennings, N. R. (2000). On agent-based software engineering. *Artificial Intelligence, 117*(2), 277–296. https://doi.org/10.1016/S0004-3702(99)00107-1

Jobin, A., Ienca, M., & Vayena, E. (2019). The global landscape of AI ethics guidelines. *Nature Machine Intelligence, 1*, 389–399. https://doi.org/10.1038/s42256-019-0088-2

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)* (pp. 4765–4774).

Pearce, H., Ahmad, B., Tan, B., Dolan-Gavitt, B., & Karri, R. (2022). Asleep at the keyboard? Assessing the security of GitHub Copilot's code contributions. In *2022 IEEE Symposium on Security and Privacy (SP)* (pp. 754–768). IEEE. https://doi.org/10.1109/SP46214.2022.9833571

Rao, A. S., & Georgeff, M. P. (1995). BDI agents: From theory to practice. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)* (pp. 312–319).

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1135–1144). Association for Computing Machinery. https://doi.org/10.1145/2939672.2939778

Russell, R. L., Kim, L., Hamilton, L. H., Lazovich, T., Harer, J. A., Ozdemir, O., Ellingwood, P. M., & McConley, M. W. (2018). Automated vulnerability detection in source code using deep representation learning. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 757–762). IEEE. https://doi.org/10.1109/ICMLA.2018.00120

Ryan, M. (2020). In AI we trust: Ethics, artificial intelligence, and reliability. *Science and Engineering Ethics, 26*(5), 2749–2767. https://doi.org/10.1007/s11948-020-00228-y

Shneiderman, B. (2020). Human-centered artificial intelligence: Reliable, safe & trustworthy. *International Journal of Human–Computer Interaction, 36*(6), 495–504. https://doi.org/10.1080/10447318.2020.1741118

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). MIT Press.
Thiebes, S., Lins, S., & Sunyaev, A. (2021). Trustworthy artificial intelligence. *Electronic Markets, 31*(2), 447–464. https://doi.org/10.1007/s12525-020-00441-4

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)* (pp. 5998–6008).
Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: Theory and practice. *The Knowledge Engineering Review, 10*(2), 115–152. https://doi.org/10.1017/S0269888900008122

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR 2023)*.

Zhou, Y., Liu, S., Siow, J., Du, X., & Liu, Y. (2019). Devign: Effective vulnerability identification by learning comprehensive program semantics via graph neural networks. In *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*.

Appendix A. Reproducibility and Supplementary Notes

Symbol glossary: s_t (state at step t), a_t (action at step t), R (task reward), γ (discount factor), L (end-to-end latency), L_max (latency deadline), τ (trust score), τ_min (minimum trust threshold), I_model (model-integrity signal), P_anom (anomaly probability), C_input (input-consistency score), x_k (edge-placement indicator), ΔL_k (latency reduction from edge placement of component k), c_k (resource cost of component k), C (edge capacity budget), ϕ_i (Shapley attribution of feature i), DMR (deadline-miss ratio).

Suggested reporting checklist for future empirical studies applying this framework: (1) dataset provenance and licensing for each source in Table 2; (2) software/library versions and hardware specification; (3) random seeds and number of repeated runs; (4) full hyperparameter grids searched; (5) statistical test used for each reported comparison, with effect size and confidence interval; (6) explicit labeling of any illustrative versus measured results, consistent with the practice followed in Section 9 of this paper.

Code and configuration availability: no experimental code was produced for this manuscript, as no experiments were run; a future empirical study following Section 8's protocol should publish its code, configuration files, and random seeds alongside its results to enable independent replication.