

BAG-IDS: A Blockchain-Enabled Agentic AI Architecture for Trustworthy Intrusion Detection

Mohamed Ala Eddine BAHRI

`bahri.mohamedalaeddine@isitc.u-sousse.tn`

Institut Supérieur d'Informatique et des Technologies de Communication de Hammam Sousse

Farah JEMILI

Institut Supérieur d'Informatique et des Technologies de Communication de Hammam Sousse

Mohamed MOSBAH

Univ. Bordeaux, Bordeaux INP, CNRS

Research Article

Keywords: intrusion detection, blockchain, distributed consensus, multi-agent systems, ensemble learning, human-in-the-loop

Posted Date: October 30th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10843028/v1>

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

[Read Full License](#)

Additional Declarations: No competing interests reported.

BAG-IDS: A Blockchain-Enabled Agentic AI Architecture for Trustworthy Intrusion Detection

Mohamed Ala Eddine BAHRI

INSIGHT Lab, ISITCom, University of Sousse,
Tunisia

Email: bahri.mohamedalaeddine@isitc.u-sousse.tn

Farah JEMILI

INSIGHT Lab, ISITCom, University of
Sousse, Tunisia

Email: farah.jmili@isitc.u-sousse.tn

Mohamed MOSBAH

Univ. Bordeaux, Bordeaux INP, CNRS,
LaBRI, UMR5800, F-33400 Talence,
France.

Email: mohamed.mosbah@u-bordeaux.fr

Abstract

Learning-based intrusion detection produces verdicts operators must simply trust: the audit record is mutable and no independent party endorses the decision. We present BAG-IDS, a distributed detection architecture in which autonomous agents detect, independently re-check, and co-sign every remediation-triggering verdict on a Hyperledger Fabric ledger before it is acted upon, placing certification on the decision path rather than in an after-the-fact log. A calibrated, reliability-weighted ensemble with a reject option feeds the quorum; language-model and governed human-in-the-loop tiers handle escalation. The full stack is implemented and every number measured. Selective certification raises accuracy from 96.6% to 99.7% (0.04% false-positive rate) at 87.3% coverage on CICIDS2017, and reaches 99.98% on the independent GeNIS 2025 corpus while deferring 0.14% of flows. A 360-alert benchmark shows the ~2 s consensus latency is structural: one write per key per block forces K votes across at least K blocks, a bound no submission strategy removes.

Keywords: intrusion detection; blockchain; distributed consensus; multi-agent systems; ensemble learning; human-in-the-loop.

1. Introduction

Modern networks are defended by intrusion detection systems (IDS) that increasingly rely on deep learning to recognise attacks in high-volume traffic. These learned detectors are accurate but also opaque, brittle under adversarial pressure, and, most importantly for operations, they produce decisions that a security operations centre (SOC) must simply trust. In practice, a single model, running on a single host, emits a verdict that may trigger an automated response such as blocking a host or throttling a link. Three weaknesses follow from this arrangement, and they compound one another.

First, **the decision rests on a single point of trust**. If the model is wrong, poisoned, or the host on which it runs is compromised, there is nothing between a bad verdict and a harmful action. Second, **the record of the decision is mutable**. A conventional log can be altered or deleted, so after an incident there is no tamper-evident account of what was detected, who agreed, and what was done. This is an increasing liability as automated response and regulatory accountability grow. Third, **learned detectors are attackable**. Poisoning attacks that corrupt training data can silently degrade a deployed model, and defences that add an explicit poisoning detector merely move the problem to another fragile component. Each weakness has been studied in isolation; what is missing is an architecture in which detection, trust, and human oversight are designed together.

Two research currents each address part of the problem but not the whole. Blockchain-based

collaborative IDS bring decentralised, tamper-evident trust, yet they use the ledger mainly to share detection data or model updates and leave the detector itself a single, unhardened model. In parallel, large language models (LLMs) are being cast as autonomous agents for security operations, bringing contextual reasoning and human-readable rationale, yet an LLM can neither adjudicate every packet at line rate nor be trusted without oversight, and agentic proposals rarely provide any decentralised trust guarantee beneath the reasoning. Neither current, on its own, yields a detector that is simultaneously robust, trustworthy, auditable, and supervisable.

This paper presents **BAG-IDS**, a Blockchain-enabled Agentic AI architecture for intrusion detection that unifies these currents into a single, measured system. Its central claim is deliberately singular: **a trustworthy IDS can be built as an agentic architecture whose decisions are secured by a blockchain consensus**. Everything else in the system, ensemble calibration, weighted fusion, selective rejection, LLM deliberation, and governed human oversight, is a mechanism in the service of that goal rather than a standalone contribution. BAG-IDS is organised as four cooperating layers with deliberately asymmetric cost. A *reflex* detection layer classifies every flow in milliseconds through a calibrated ensemble of neural detectors fused by a reliability-weighted strategy we refer to as **Conscious Weighted Consensus** (a fusion of standard, well-understood components whose value here is architectural, Section 4.2.4), and gates low-confidence decisions with a reject option. A *deliberative* layer invokes an LLM only on salient events, namely a candidate attack or a burst of rejections, to reason about severity and remediation. A *consensus* layer converts a single agent's alert into a collectively certified, tamper-evident record: independent notary agents re-validate the alert for consistency and cast cryptographically signed votes to a permissioned-blockchain smart contract that certifies the decision only on reaching a quorum, before any remediation is authorised (Section 7.2 delimits what this does and does not prevent). Finally, an optional *human-in-the-loop* supervision layer gives an operator a governed, auditable point of control, and can be switched off to cede full autonomy to the agents, with every administrative action itself signed and anchored on-chain.

The design principle that makes an agentic, blockchain-backed IDS operationally viable is the asymmetry between the layers: thousands of flows per second traverse the reflex path, but only the small fraction that yield an attack verdict incur the cost of LLM deliberation and on-chain consensus. Detection therefore remains real-time while trust and oversight are applied precisely where they matter, on the decisions that trigger action.

Contributions. This paper makes four contributions:

- **An agentic detection architecture that places certification on the decision path**, realised as a Hyperledger Fabric smart contract: autonomous agents detect, independently re-validate and co-sign every remediation-triggering verdict through multi-party signed voting with quorum, anti-replay and tamper-evident audit, before any remediation is authorised, rather than logging alerts after the fact. Distinctively, the same ledger anchors human governance actions, so that even the administrator is auditable (Section 7.2 delimits what this does and does not prevent).
- **A cost-asymmetric layering that makes on-path consensus operationally viable**, combining a real-time calibrated ensemble with event-triggered LLM reasoning: every flow traverses the reflex path in milliseconds, but only the small fraction yielding an attack

verdict incurs the cost of deliberation and on-chain certification. Detection therefore remains real-time while the trust machinery runs at alert rate rather than at flow rate, and no language model sits on the per-flow critical path.

- **A reliability-weighted fusion that degrades gracefully under detector corruption**, in which detectors are weighted by their calibrated, recent reliability. Evaluated under simulated output corruption as a worst-case poisoning proxy, the ensemble loses 0.6 accuracy points (1.9 at worst) while a fully corrupted detector is auto-excluded, its weight collapsing from 0.332 to 0.014, without any explicit attack detection anywhere in the loop. Section 4.2.4 states precisely what this mechanism is and is not.
- **A complete measured evaluation, including a structural bound on consensus cost**, covering the full stack on commodity hardware: a component-by-component ablation (individual detectors, uniform ensemble, calibrated weighted fusion, fusion with rejection) that quantifies the contribution of each mechanism, the rejection-threshold trade-off, the reflex-path throughput, corruption robustness, a cross-corpus validation on an independently generated 2025 corpus under a leakage-audited natural-distribution protocol, and the faithfulness and cost of the human-facing explanation layer that supports analyst approvals. A 360-alert benchmark further establishes that consensus latency grows with the quorum for a structural reason (the chaincode read-modify-writes the alert's state key, and Fabric's optimistic concurrency commits at most one write per key per block, so K votes require at least K block commits whatever the submission strategy), and identifies the per-vote-key redesign that would lift the bound. Every number is produced by the running system rather than projected.

Our evaluation on CICIDS2017, on the class-balanced testbed of the detectors' original protocol (so that operational rates on raw, benign-dominated traffic will differ, Section 6.1), shows that the calibrated ensemble reaches 96.6% accuracy over all decisions, significantly above its best member, and 99.7% at 0.04% false-positive rate on the certified subset; that the consensus cost grows linearly with the quorum for a structural reason (Fabric commits at most one write per state key per block, so a quorum of K votes spans at least K blocks) and is dominated by ledger block formation rather than cryptography; and that, under a simulated output corruption that reduces the strongest detector to 0.6% accuracy, the ensemble loses only 0.6 accuracy points while the corrupted detector's influence is automatically suppressed. Retrained on GeNIS, an independently generated 2025 corpus evaluated under a leakage-audited protocol with the natural class distribution, the same stack reaches 99.98% accuracy while the gate defers only 0.14% of flows, against 12.7% on the harder 15-class task: selective certification adapts its cost to corpus difficulty (Section 6.9). Section 6.11 positions these figures against the results reported by recent blockchain-based, federated, and agentic IDS.

Paper organisation. Fig. 1 gives the architecture at a glance and fixes the step numbering used throughout the paper; Section 2 reviews related work and positions our contribution. Section 3 states the threat model and design goals. Section 4 presents the four-layer architecture. Section 5 describes the implementation. Section 6 reports the experimental evaluation. Section 7 analyses security threat by threat. Section 8 discusses limitations and Section 9 concludes.

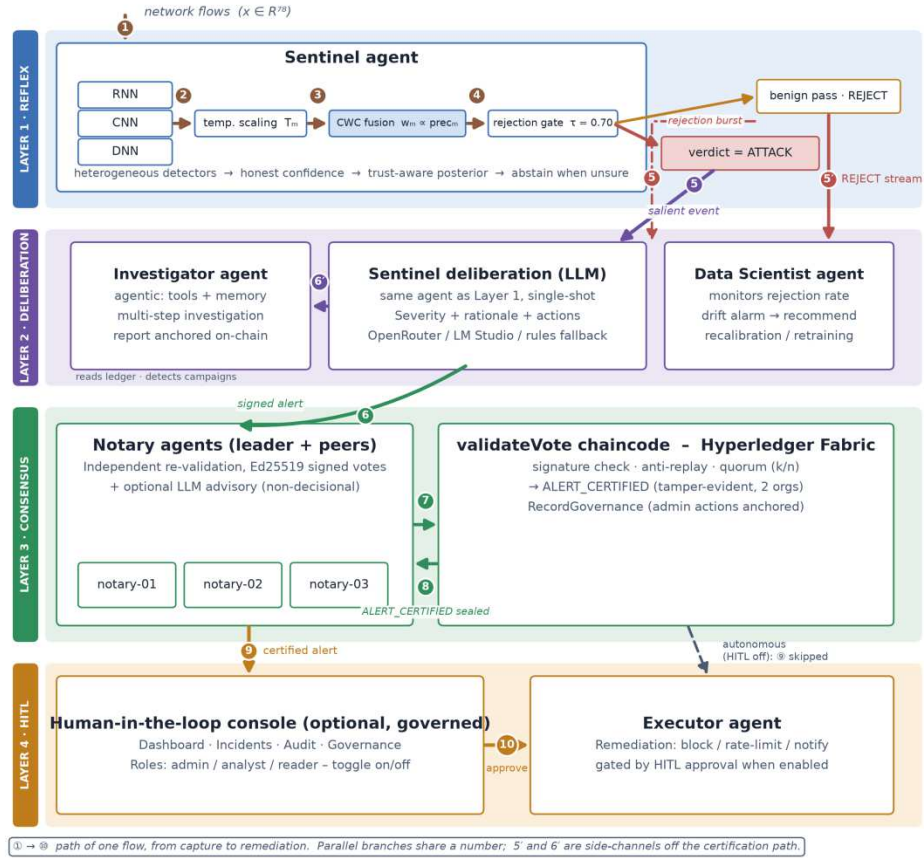


Fig. 1. Overview of the four-layer BAg-IDS architecture and the path of a certified attack (detailed in Section 4). The numbered badges mark the runtime path of a single flow, from capture to remediation; they are referred to as step 1 to step 10 throughout the paper and reappear, with the same meaning, in every architecture figure. Parallel branches share a number, and badges 5' and 6' mark the two side-channels that branch off the certification path.

2. Related Work

BAg-IDS lies at the intersection of five threads: deep-learning intrusion detection, confidence-aware and selective classification, adversarial robustness, multi-agent and blockchain-based collaborative security, and language-model agents in cyber defence. We review each critically rather than as a catalogue, with emphasis on the 2023–2026 literature, organising the agentic line along the three-phase evolution the recent surveys converge on [64, 76]. Two tables close the section: a bibliographic comparison of the closest systems (Table 1) and a capability matrix along eight criteria (Table 2).

2.1. Deep learning for network intrusion detection

Learning-based intrusion detection has moved decisively towards deep models: recurrent architectures capture temporal structure in flow sequences [1], convolutional and hybrid networks learn discriminative spatial features [2], and deep autoencoders cope with high-dimensional, imbalanced traffic [3, 4]. Systematic studies established deep networks as the state of the art across the standard benchmarks [5, 6], and surveys of learning-based detection, particularly for IoT-facing deployments, mapped the design space [85]. Progress rests largely on CICIDS2017

and CSE-CIC-IDS2018 [7, 8], one of the two corpora of our evaluation (the other being the independently generated 2025 GeNIS corpus [80], Section 6.9), whose tooling and labelling carry documented flaws [50–52] against which any result, including ours, must be read (Section 6.1). The most recent wave applies attention and Transformer architectures [9, 10] and treats detection as a bridge towards language-model analysis [11]. Our prior comparative study [56] belongs to this thread: it trained the RNN, CNN and DNN detectors reused here. The limitation common to this line is architectural rather than statistical: detection stops at the classifier and implicitly trusts its own output, saying nothing about how a decision should be calibrated, trusted, recorded, or acted upon. That silence is the gap this paper addresses.

2.2. Ensembles, calibration, and selective detection

Ensemble intrusion detection combines heterogeneous classifiers to offset the blind spots of any single model, with documented accuracy and robustness benefits [12, 13]. The tradeoff is known: ensembles add inference cost and, more subtly, most fuse votes with fixed or accuracy-agnostic weights over raw confidences, which modern neural networks miscalibrate [14, 15]. Two correctives have matured: temperature scaling [14] restores confidence reliability without changing the arg-max, and selective classification [16] adds a reject option that abstains on low-confidence inputs, an abstention whose operational value is increasingly recognised in the reject-option literature [17] and in uncertainty-aware NIDS [18]. Individually standard, these mechanisms are rarely combined into one detector whose fusion weights, calibration and reject threshold are co-designed. BAg-IDS makes that combination its reflex layer, tying the fusion weights to each detector's recent calibrated reliability so that the ensemble becomes a defence against a corrupted member rather than an accuracy aggregator (Section 6.7), while the reject region shapes the load offered to the deliberative layer (Section 6.5).

2.3. Adversarial and poisoning threats to learned detectors

Learned detectors are attackable at test time and at training time [19, 20]. Poisoning attacks, in which an adversary corrupts training data to degrade or backdoor the deployed model, are especially damaging for continuously retrained IDS [21, 22] and have been surveyed in depth [23]; recent work sharpens the operational picture with realistic-capability models [24], systematisations of attacks and defences [25], broad surveys [26], and demonstrations against federated IDS training [27]. The dominant defence adds an explicit poisoning detector, which itself becomes an attack surface and a source of false positives, trading one fragile component for another. Our approach is structural instead: rather than detecting poisoning, reliability-weighted fusion suppresses a corrupted member automatically, a resilience we quantify under simulated output corruption as a worst-case proxy and against a robust-aggregation baseline (Section 6.7).

2.4. Multi-agent and collaborative intrusion detection

Decomposing detection across cooperating agents dates back to early distributed IDS designs [28] and reference architectures for autonomous cyber defence [72], continuing through collaborative monitors that share evidence across nodes [29]. The paradigm scales detection but introduces a recurring trust problem: a single compromised participant can poison the shared view or forge alerts. The same difficulty arises wherever nodes act on peer-contributed evidence; in cooperative vehicular perception, trust-based mechanisms detect fabricated objects injected

into collective perception messages [86], structurally the problem a collaborative IDS faces when a participant asserts an attack that did not occur. Such mechanisms score plausibility but do not make the resulting decision non-repudiable, the complementary guarantee our consensus layer provides. Liang et al. [57] pair distributed deep-learning monitors with a blockchain ledger for tamper-proof alert sharing, but their agents remain reactive. Soltani et al. [58] add adaptivity through continual and federated learning, reporting detection rates above 95% on CICIDS2017 and CSE-CIC-IDS2018, yet provide no trust layer for the inter-agent exchange they depend on. Surveys of trust-based collaborative IDS [30] and of open multi-agent security challenges [75] identify insider threats and reliance on a trusted aggregation point as the central unresolved issues. BAg-IDS retains the multi-agent decomposition but removes the trusted aggregator, replacing reputation heuristics with a cryptographically enforced, quorum-based consensus on the decision itself.

2.5. Blockchain for trustworthy and collaborative security

Blockchain has been proposed as the missing trust substrate for collaborative detection: early position work argued for blockchain-based collaborative IDS [31], CloTA demonstrated blockchain-anchored anomaly detection for IoT [32], and reviews catalogued the design space [33]. Permissioned platforms, notably Hyperledger Fabric, whose execute–order–validate architecture underpins our deployment [34], made auditable, identity-bound ledgers practical.

The most active sub-line fuses federated learning with blockchain so that the chain secures shared model updates: FL-BCID validates federated updates through smart contracts [59]; BFLIDS combines CNN and BiLSTM classifiers with Ethereum contracts for decentralised aggregation, reporting accuracies above 97% [60]; Aydin and Aydin target DDoS with a federated, blockchain-backed multi-agent design [74]; consensus-driven variants add PBFT-verified alerting over gradient-boosted detectors [61]; and reputation-aware frameworks weight each participant's contribution by an on-chain reputation score [87]. That last idea, reliability recorded on-chain and used to discount a participant, is conceptually adjacent to our reliability-weighted fusion, except that we weight detectors within a single decision rather than clients across training rounds. Secure-logging IDS provide immutability but stop there [62]. A closely related line couples deep detectors with ledger primitives for data rather than decision integrity: Aljabri et al. secure the preprocessed traffic of a CNN-based IDS through SHA-256 block creation [83], and extend the design to cyber-physical systems with RSA-signed blocks feeding a deep belief network, reporting ledger latency alongside detection quality [84]. These works establish that hashing and signing the evidence is feasible and affordable, the premise our contribution builds on.

Two limitations are shared across this category and are decisive for our design. First, the ledger secures the collaboration, that is the sharing of data, alerts or model updates, while the detection decision itself is still produced by a single, implicitly trusted model and merely logged after the fact; certification of the verdict by several mutually distrusting parties, before it authorises an action, remains outside the scope of the category. Second, the multi-second cost of on-chain commitment is seldom reconciled with line-rate detection. BAg-IDS answers both: it places a smart contract on the decision path, so that a quorum of signed notary votes certifies each remediation-triggering verdict, and it confines that cost to the small certified fraction of traffic through the reflex/deliberative asymmetry (Sections 6.5–6.6).

2.6. Agentic AI in cyber defence: a three-phase evolution

The most recent thread reframes detectors as autonomous agents, and recent surveys converge on a three-phase reading [64, 76] we adopt as an organising device. In the **first phase** (learning-based agents), autonomy came from reinforcement and continual learning [73], at the cost of fragile reward design and unstable policies. The **second phase** (cognitive, LLM-based agents) added contextual reasoning over unstructured evidence, LLMs moving from text assistants to orchestrators able to plan investigations and summarise alerts [41–46], with the ReAct tool-use loop [45] supplying the perceive–reason–act pattern our Investigator instantiates. Concrete systems illustrate the range, on the offensive side with LLM-driven penetration testing [53, 54] and on the defensive side with analyst-facing triage: HuntGPT couples an anomaly detector with explainable AI and a conversational LLM [47]; H. Li et al. propose an LLM-enhanced multi-agent collaborative IDS reporting accuracy above 90%, but no blockchain trust layer for the alerts its agents exchange [63]; and NetMoniAI deploys per-node LLM agents with a central controller, acknowledging that it lacks on-chain immutability and that its orchestrator is a single point of failure [66]. The **third phase** (decentralised, resilient agents) makes trust and accountability first-class: blockchain provides tamper-proof records of agent actions and smart contracts encode coordination without a central authority. A conclusion recurs across all three phases, echoed by the SOC-automation literature [48, 55] and by agentic-security surveys [64]: an LLM should not adjudicate every low-level event at line rate, and its outputs require oversight. This motivates hybrid, human-supervised designs reserving language-model reasoning for high-level decisions, a modern instantiation of the classic reflex/deliberative separation [49]. BAg-IDS embodies that separation and supplies what phase-two agentic IDS lack: a decentralised trust layer beneath the agents and a governed human-in-the-loop layer above them.

2.7. Convergence of agentic AI and blockchain

The closest context for our work is the convergence of agentic AI with blockchain, developed largely outside intrusion detection. DeCoAgent lets LLM-based agents register on-chain, discover peers and negotiate task allocation through smart contracts [67], and Qi et al. extend this with reputation tracking and incentive-compatible matching [68]; both report significant contract overhead and are not validated at scale. BlockAgents introduces a Proof-of-Thought consensus combining reasoning-based evaluation with stake-based voting to blunt Byzantine behaviour among LLM agents, at the price of added latency [69], while DAO-AI evaluates agentic voting over more than 3,000 real governance proposals, finding strong alignment with human outcomes but no collaborative reasoning [70]. Closer to security, Lad et al. govern inter-agent communication and accountability through smart contracts [71], and Gosmar and Dahl's Sentinel Agents monitor inter-agent communication for coordinated threats, though validated only on synthetic scenarios and not integrated with an IDS [65]. The pattern is clear: blockchain and smart contracts coordinate, govern or incentivise the agents, and the common tradeoff is the latency of frequent on-chain interaction. What is missing is a system that turns this convergence back onto the detection decision itself, placing quorum-based certification on the per-verdict path while keeping that cost off the line-rate path. BAg-IDS is, to our knowledge, the first to do so.

2.8. Critical synthesis and positioning

Read as a whole, the literature exhibits a consistent division of labour and a consistent blind spot. The detection community (Sections 2.1–2.3) perfects the classifier but trusts its output; the collaborative and blockchain community (Sections 2.4–2.5) secures the sharing of alerts or model updates but leaves the decision to a single implicitly trusted model and rarely addresses line-rate cost; the agentic community (Section 2.6) adds reasoning and autonomy but, in its deployed IDS instances, provides neither decentralised trust nor a measured full-stack system; and the agentic-blockchain convergence (Section 2.7) demonstrates on-chain coordination and governance of agents but has not been brought to the intrusion-detection decision itself. Table 1 makes this concrete for the closest systems, and Table 2 abstracts it into eight capabilities that jointly define trustworthy autonomous detection.

(C1) Calibrated ensemble detection: heterogeneous detectors fused over honest confidences. **(C2) Selective classification:** an explicit reject region that withholds unreliable decisions from automation. **(C3) Corruption robustness:** graceful degradation when a member model is corrupted (evaluated under simulated output corruption, a worst-case poisoning proxy; Section 6.7). **(C4) Decentralised cryptographic consensus:** no single trusted aggregator may certify a decision. **(C5) Immutable decision audit:** every certified verdict and vote is tamper-evident. **(C6) Agentic LLM reasoning:** multi-step, tool-using investigation rather than single-shot classification. **(C7) Governed human oversight:** a human gate whose own administrative actions are signed and auditable. **(C8) Line-rate viability:** the expensive machinery is kept off the per-flow path.

Table 1. Bibliographic comparison of the closest agentic, blockchain and IDS systems. BC: uses blockchain; Ag: agentic/autonomous; SC: smart contracts. ✓ present, ± partial, ✗ absent. “n/r”: not reported in the source available to us; “qual.”: qualitative only. Reported results are stated by the cited authors under their own protocols and are not directly comparable across rows; Section 6.11 places our measured results next to these reported figures.

Ref. (year)	Approach	BC	Ag	SC	Reported result	Gap addressed by BAg-IDS
Liang [57] (2020)	DL monitors + blockchain alert log (IoT)	✓	±	✗	83.2% acc (NSL-KDD)	No autonomy or on-decision consensus
Soltani [58] (2023)	Multi-agent adaptive CNN/LSTM + FL	✗	✓	✗	>95% (CIC-IDS2017/18)	No trust layer for inter-agent exchange
Ali [59] (2025)	FL-BCID: federated IDS + SC validation	✓	✗	✓	97.3% acc (ToN-IoT)	No agent coordination; log-only ledger
Begum [60] (2024)	BFLIDS: FL + Ethereum aggregation (IoMT)	✓	✗	✓	97.4% acc (Edge-IIoTSet)	No autonomy; decision off-chain
Nandanwar [61] (2025)	PBFT consensus + XGBoost IDS	✓	✗	±	n/r (NSL-KDD)	No agent collaboration; dated data
Kumar [62] (2025)	Secure blockchain IDS logging (IoT)	✓	✗	✗	n/r (IoT traffic)	No automated mitigation or agency
H. Li [63] (2026)	LLM-enhanced agentic collaborative IDS	✗	✓	✗	>90% (benchmarks)	No blockchain trust for shared alerts
Gosmar [65] (2025)	Sentinel agents, behavioural MAS monitor	✗	✓	✗	n/r (synthetic)	Not integrated with an IDS
Zambare [66] (2025)	NetMoniAI: per-node LLM agents + controller	✗	✓	✗	n/r	Central SPoF; no on-chain immutability
Jin [67] (2024)	DeCoAgent: on-chain LLM-agent collaboration	✓	✓	✓	n/r (feasibility)	Not a detector; rigid contract logic
Qi [68] (2025)	Blockchain-driven LLM-agent matching (Web3)	✓	✓	✓	n/r (simulation)	High SC overhead; not security

Ref. (year)	Approach	BC	Ag	SC	Reported result	Gap addressed by BAg-IDS
X. Li [69] (2024)	BlockAgents: Proof-of-Thought consensus	✓	✓	✓	qual. (poison ↓)	Latency; not a detector
Capponi [70] (2025)	DAO-AI: agentic on-chain governance voting	✓	✓	✓	3k proposals	No collaborative reasoning; not IDS
BAg-IDS (this work)	Calibrated CWC ensemble + reject + LLM + Fabric quorum + HITL	✓	✓	✓	99.7% cert., 0.04% FPR (CICIDS2017); 99.98% cert. (GeNIS 2025)	this work

Table 2. Capability-matrix positioning of BAg-IDS. C1 calibrated ensemble; C2 selective classification (reject); C3 corruption robustness (poisoning proxy); C4 decentralised cryptographic consensus on the decision; C5 immutable decision audit; C6 agentic (multi-step, tool-using) LLM reasoning; C7 governed, auditable human oversight; C8 line-rate viability. ✓ intrinsic to the category; ± present in some works, not intrinsic; X absent.

Approach	C1	C2	C3	C4	C5	C6	C7	C8
Deep-learning / Transformer IDS [1–11]	±	X	X	X	X	X	X	✓
Ensemble IDS [12, 13]	±	X	±	X	X	X	X	✓
Selective / uncertainty-aware IDS [16–18]	±	✓	X	X	X	X	X	✓
Adversarially-defended IDS [24–27]	X	X	±	X	X	X	X	✓
Multi-agent / collaborative IDS [28–30, 57, 58]	±	X	±	X	±	X	±	✓
Blockchain / FL-blockchain IDS [31–40, 59–62, 74]	±	X	±	✓	✓	X	X	±
Agentic / LLM defence [41–47, 63, 65, 66]	X	X	X	X	X	✓	±	±
Agentic + blockchain (non-IDS) [67–71]	X	X	±	✓	✓	✓	±	X
BAg-IDS (this work)	✓	✓	✓	✓	✓	✓	✓	✓

Both tables converge on the same reading. Each capability has a mature literature behind it, yet no prior system exhibits all eight, and the agentic-blockchain convergence that comes closest (row 8 of Table 2) has never been applied to the detection decision nor shown to be line-rate viable. A new approach is needed not because a mechanism is missing, but because these mechanisms live in communities that rarely intersect and their naive composition fails: an LLM or a blockchain on the per-flow path would destroy real-time detection, while logging decisions after a single model has made them gives auditability without trust. **To our knowledge, BAg-IDS is the first measured system to satisfy all eight criteria simultaneously:** a calibrated, poisoning-resilient selective ensemble on the per-flow path; a smart-contract consensus certifying every remediation-triggering verdict through quorum-signed votes; an agentic investigator whose reasoning is anchored on the ledger; and a governed human-in-the-loop layer whose every administrative act, including switching autonomy on or off, is a signed on-chain transaction. It thereby resolves the single point of trust of single-model IDS (C1, C4), the after-the-fact role of the ledger in blockchain IDS (C4, C5), the absent trust layer of agentic IDS (C4–C5), and the line-rate infeasibility of naive designs (C8), as one implemented artefact whose every reported number is measured (Sections 4–6).

3. Threat Model and Design Goals

Before presenting the architecture we make explicit the adversary it is designed to withstand and the properties it aims to guarantee. A learning-based intrusion detector operating inside a security operations workflow is exposed on three fronts at once: its models can be corrupted, the software agents that run it can be subverted, and the record of what it decided can be forged or repudiated. BAg-IDS is designed against an adversary that operates on all three, under assumptions we state below.

3.1. System and trust assumptions

The system consists of five cooperating agents (Sentinel, Notary, Executor, Data Scientist, Investigator), a set of trained detectors, and a permissioned ledger operated by two or more organisations. We assume a **permissioned** setting: participating organisations are known and admitted through a membership authority, and each agent holds a long-lived key pair whose public half is registered before operation. We take the standard cryptographic assumptions (that signatures cannot be forged and hash functions are collision-resistant) as given, and we assume the initial registration of identities is performed correctly by an honest administrator. The consensus layer inherits the fault model of the underlying ledger; in the evaluated configuration a decision is certified only with the agreement of a **majority of independent notaries**, so the adversary is assumed unable to control that majority simultaneously.

3.2. Adversary capabilities

Within these assumptions, we grant the adversary a deliberately broad repertoire, organised by the front it attacks:

- **Model-level (poisoning).** The adversary can tamper with the training or update data of one detector, for example by flipping a fraction of its labels, so that the deployed model misclassifies, typically making attacks pass as benign (evasion). Corruption of a single detector is in scope; simultaneous corruption of a majority of the ensemble is not.
- **Agent-level (compromise).** The adversary can take over a single agent host: a Sentinel that then emits fabricated or manipulated alerts, one notary that votes dishonestly, or the supervision console from which it may attempt to weaken the system's posture. Compromise is limited to one agent of each kind at a time.
- **Record-level (forgery, replay, tampering).** The adversary can observe messages and attempt to inject a vote on behalf of a notary, replay a previously valid vote, or alter a decision already written to the ledger, in order to manufacture or erase a certification.
- **Governance-level (authority abuse).** Having reached the console, the adversary can attempt to lower the rejection threshold, reduce the quorum, or disable human oversight, so as to quietly degrade the guarantees the operator believes are in force.

3.3. Out of scope

We exclude threats that are orthogonal to the contribution or that no detection architecture of this kind can address alone. Denial-of-service against the network or the ledger infrastructure, side-channel and host-level exploits of the operating system, compromise of the membership authority

or of the initial key registration, and adversarial evasion crafted against the feature extractor itself are not treated here. We likewise assume the LLM used by the deliberative layer is not an attack vector for detection decisions, since it never sees raw traffic and its output only annotates already-certified alerts with a severity and a recommended action.

3.4. Design goals

From this threat model we derive the properties the architecture must provide. They double as the criteria against which Sections 6 and 7 evaluate the system:

- **G1. Poisoning resilience.** The detection decision must degrade gracefully when a single detector is corrupted, without relying on an explicit poisoning detector. (Addressed by the Conscious Weighted Consensus, Section 4.2.4; measured in Section 6.7.)
- **G2. No single point of trust.** No individual agent may unilaterally cause a remediation; certification must require agreement among independent parties. (Multi-notary quorum, Section 4.2.6.) We state the boundary of this goal up front: notaries re-validate the consistency of an alert, not the raw flow, so a *fully compromised* Sentinel can still produce an internally consistent alert that the quorum endorses. Against that specific adversary the property delivered is attributable, non-repudiable accountability rather than prevention; Section 7.2 analyses this boundary and the cost of closing it.
- **G3. Authenticity and non-repudiation.** Every alert, vote and certification must be attributable to a registered identity and impossible to forge or disown. (Signed votes verified on-chain, Section 4.2.6.)
- **G4. Tamper-evident auditability.** The decision record must be immutable and independently verifiable after the fact. (Hash-linked, replicated ledger, Section 4.2.6.)
- **G5. Governed, auditable oversight.** Human control must be possible and optional, and any exercise of administrative authority must itself be recorded immutably. (Human-in-the-loop layer, Section 4.2.7; verified in Section 7.5.)
- **G6. Operational viability.** The trust machinery must not compromise real-time detection; its cost must fall only on the small fraction of certified events. (Reflex/deliberative asymmetry, Section 4.1.1; measured in Section 6.6.)

The remainder of the paper is organised around these goals: Section 4 presents the architecture that realises them, Section 6 measures the extent to which they are met, and Section 7 argues, threat by threat, why each adversarial capability of Section 3.2 is neutralised.

4. The BAg-IDS Architecture

This section presents our contribution in two complementary parts. **Part I (Section 4.1)** gives the global architecture: the four layers, the five agents, and the end-to-end path of a network flow, at the level of abstraction needed to understand the design decisions. **Part II (Section 4.2)** then descends into a detailed, reproducible specification of every layer and every model (exact network architectures, calibration procedure and fitted parameters, fusion equations, message and vote formats, chaincode invariants, and deployment topology), so that the system can be re-implemented from the paper alone. Each subsystem specified here is evaluated in isolation in

Section 6, where we also measure how its output quality conditions the subsystem that consumes it.

4.1. Part I: global architecture

4.1.1. Four layers, one asymmetry

BAG-IDS is organised as four cooperating layers that separate the **speed of detection**, the **deliberation of judgement**, the **trust of the decision record**, and the **authority of human oversight** (Fig. 1). The *reflex detection layer* performs per-flow inference in the millisecond range through a calibrated ensemble of neural detectors and a confidence-gated rejection rule. The *deliberative layer* is invoked only on salient events (a candidate attack or a burst of rejections) and applies a large language model (LLM) to reason about severity and remediation. The *consensus layer* anchors every certified decision on a permissioned blockchain, where independent notary agents cross-validate and cryptographically endorse the outcome before any remediation is executed. The *human-in-the-loop supervision layer* gives a security operator a governed, auditable point of control, and can be switched off to cede full autonomy to the agents.

The design follows a **reflex/deliberative hybrid** that is deliberately asymmetric in cost. Thousands of flows per second traverse the reflex path, which never blocks on the network or on an external model; only the small fraction of flows that yield an attack verdict incur the cost of LLM deliberation and on-chain consensus. This asymmetry is what makes an agentic, blockchain-backed IDS operationally viable, and it is quantified subsystem by subsystem in Section 6.

4.1.2. Agent roles

BAG-IDS is realised as five autonomous agents, each an independent process exposing an HTTP interface and running its own perception, decision and action loop:

- **Sentinel:** the detection agent. It ingests network flows, runs the calibrated ensemble, applies the Conscious Weighted Consensus fusion, and, for attack verdicts, triggers deliberation before emitting a signed alert.
- **Notary:** the consensus agent, deployed as several independent instances. Each notary re-validates an incoming alert on its own and casts a cryptographically signed vote; the outcome is committed to the ledger only when a quorum of valid endorsements is reached.
- **Executor:** the remediation agent. It carries out the actions attached to a certified, quorum-approved alert (block, rate-limit, isolate, notify) and records their execution.
- **Data Scientist:** the maintenance agent. It continuously monitors the health of the Sentinel (chiefly the rolling rejection rate as a proxy for distribution drift) and proactively recommends recalibration or retraining when a threshold is crossed.
- **Investigator:** the triage agent. On an attack verdict, concurrently with the certification round, it conducts an autonomous, multi-step investigation (querying the ledger for related incidents, correlating them to detect campaigns, and reasoning to a prioritised incident report) with its conclusion anchored on-chain.

Agents are autonomous in the strong sense: each pursues its own objective and acts without being polled. The Sentinel decides on every flow; the Data Scientist decides, on its own initiative, when the system is drifting; the notaries decide independently whether to endorse. No single

agent is trusted to be correct, which is the premise the consensus layer enforces. Fig. 2 contrasts the five agents side by side (what each perceives, decides and does, how it uses or deliberately does not use the LLM, and what it writes on-chain), making the division of labour, and the deliberate asymmetry of LLM usage across the agents, directly comparable.

	Sentinel :8001	Notary x3 :8002 /12 /22	Executor :8003	Data Scientist :8004	Investigator :8005
Runs at	1 → 5	6 → 8	10	5'	6'
Mission	per-flow detection (reflex layer)	collective certification	remediation of certified attacks	model-health & drift guardian	triage & campaign detection
Perceives	network flows (78 features)	signed alerts from Sentinel	certified (+ approved) alerts	REJECT stream, verdict statistics	alerts, in parallel with certification, + ledger
Decides	CWC verdict + reject gate (τ)	consistency re-validation → vote	action mapping per severity	drift alarm (rolling reject rate)	agentic loop: reason → act (tools)
Acts	signed alert + deliberation summary	Ed25519-signed vote (quorum 2/3)	block / rate-limit / isolate / notify	recommend recalib. / retraining	incident report, triage severity
LLM use	event-triggered severity judgement	advisory opinion (non-decisional)	none (deterministic)	none (statistical)	multi-step reasoning (rules fallback)
On-chain	RecordAlert (every alert)	ValidateVote, ALERT_CERTIFIED	execution records	- (reports to supervision)	report anchored (auditable reasoning)

« Runs at » gives each agent's position in the numbered sequence of Fig. 1. 5' and 6' are side-channels off the certification path.

Fig. 2. The five BAg-IDS agents compared: mission, perception, decision, action, LLM usage and on-chain footprint. Only the Sentinel (event-triggered), the notaries (advisory only) and the Investigator (multi-step reasoning) touch the LLM; the Executor and Data Scientist are deliberately deterministic. The Runs at row places each agent in the numbered sequence of Fig. 1, so the division of labour can be read against the data path.

4.1.3. Life of a flow

The global data path is as follows. Each stage carries a number that is the same in every figure of this paper and in the text below, so that any step can be located at a glance. A flow reaches the three detectors in parallel (**step 1**), each posterior is temperature-scaled (**step 2**) and the calibrated posteriors are fused by Conscious Weighted Consensus (**step 3**). The fused posterior then passes the rejection gate (**step 4**), which returns one of three verdicts: **benign** (passed, no further cost), **rejected** (withheld from automation; its rolling rate is the Data Scientist's drift signal, **step 5'**, and a burst of rejections is an alternative trigger for the same deliberation as an attack verdict, hence **step 5** again), or **attack**. Only the attack verdict propagates: the Sentinel attaches an LLM deliberation (**step 5**), signs the alert, and submits it to the notaries (**step 6**). Each notary independently re-validates the decision and votes (**step 7**); when the chaincode observes a quorum of valid signed votes it seals an *ALERT_CERTIFIED* record on the replicated ledger and releases the certification event (**step 8**). The alert also opens an autonomous investigation, dispatched at submission time and therefore running concurrently with the certification round rather than after it (**step 6'**). Certification itself releases the remediation path, which either executes immediately (autonomous mode) or is held in the operator's approval queue (**step 9**) before the Executor acts (**step 10**). Every step (alert, votes, certification, investigation conclusion, human approval, and every governance action) lands on the same immutable ledger.

4.2. Part II: detailed, reproducible architecture

Fig. 3 expands Fig. 1 into the complete technical specification: the data plane, the exact detector architectures and their calibration temperatures, the fusion and rejection rules, the three deliberative agents, the consensus machinery with its chaincode invariants, and the supervision layer. The remainder of this section walks through Fig. 3 top to bottom; all numeric values are the ones used in the evaluation of Section 6.

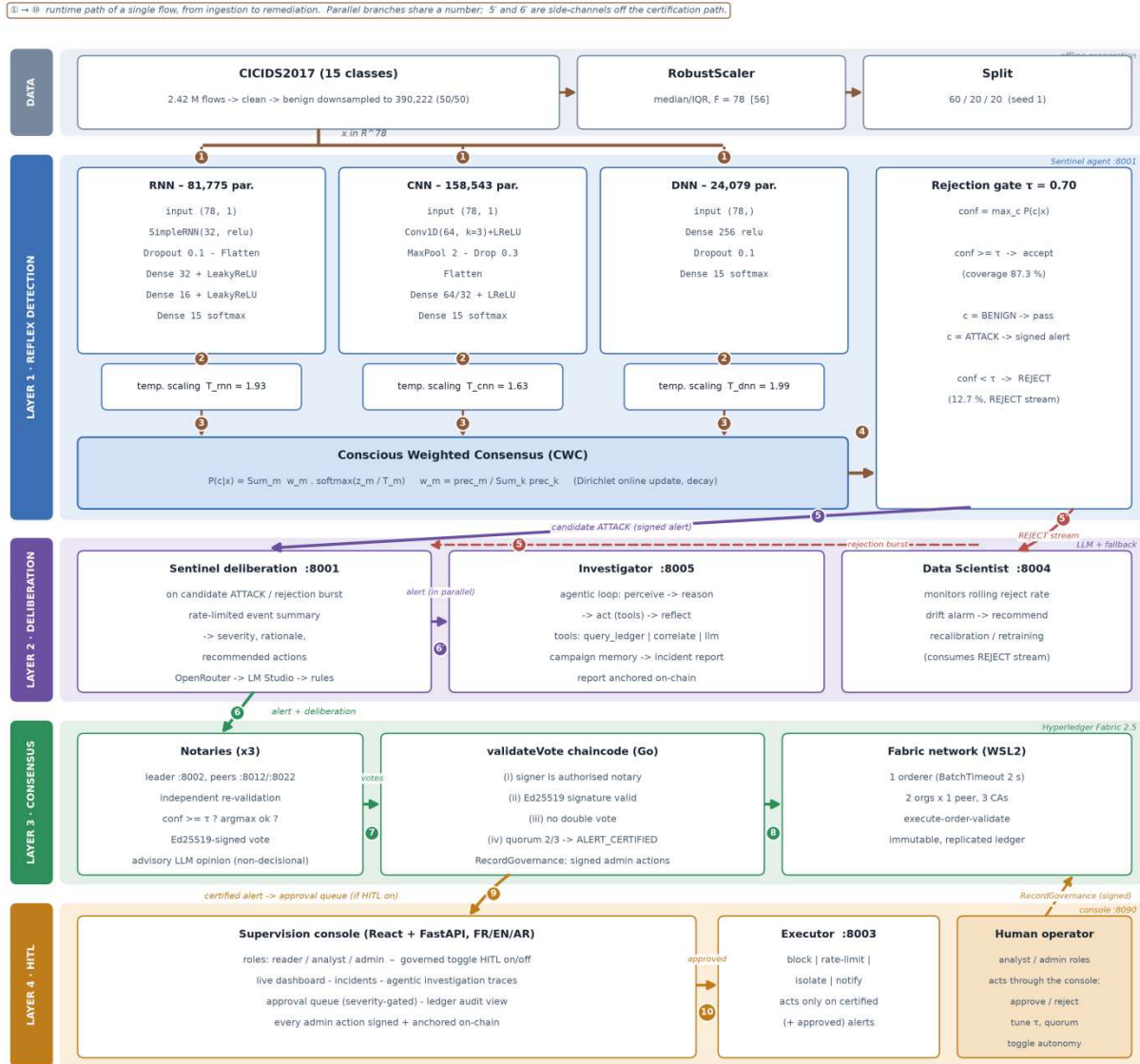


Fig. 3. Detailed BAG-IDS architecture: data plane, exact detector stacks and fitted calibration temperatures, Conscious Weighted Consensus, rejection gate, deliberative agents, consensus layer with the validateVote invariants, and governed human-in-the-loop supervision. Ports identify the concrete deployed services. Detector shapes and fitted parameters are those of the CICIDS2017 instantiation; the GeNIS instantiation of Section 6.9 differs only in the boxed reflex layer ($F = 59$, $C = 9$, Table 3) and leaves every downstream layer unchanged. The badges carry the same step numbering as Fig. 1; the data plane at the top is offline preparation and is therefore not numbered.

4.2.1. Data plane and preprocessing

All detectors consume the same flow representation, produced by the **exact preprocessing under which they were trained** [56], which we recovered from the original training notebook and reproduce seed for seed. The CICIDS2017 corpus (2.42 million flows after concatenating the daily captures and dropping duplicates, NaN and infinite values) yields $F = 78$ numeric flow features per record and 15 classes (benign plus fourteen attack families, 390,222 attack flows in total). The corpus is then class-balanced by down-sampling benign traffic to the attack count (sklearn resample, replace=true, seed 123), giving 780,444 flows; features are scaled with a **RobustScaler** (median/IQR, robust to the heavy-tailed flow statistics) fitted on the balanced corpus; labels are integer-encoded in lexicographic order; and the data are split 60/20/20 into training, validation and test partitions (test_size 0.2 then validation 0.25 of the remainder, seed 1). The test partition contains 156,089 flows. The validation partition serves three distinct purposes downstream (fitting the calibration temperatures, estimating the fusion weights, and monitoring drift) and is never used for training or testing. The recurrent and convolutional detectors receive the feature vector reshaped to $(78, 1)$; the dense detector receives the flat $(78,)$ vector.

The cross-corpus study of Section 6.9 instantiates this same data plane on a second corpus, GeNIS 2025 [80], and the differences are worth stating here because they define what is corpus-specific in the specification that follows. That corpus is produced by a different flow exporter, so the representation is $F = 59$ numeric features over 9 classes (benign plus eight attack families) after removing addresses, ports, identifiers and the five positional columns exposed by the leakage audit of Section 6.9. Because we are not bound there by fidelity to a published training pipeline, that instantiation uses the **clean protocol** instead: the RobustScaler is fitted on the training split only, no class rebalancing is applied, and the stratified 60/20/20 split preserves the natural class distribution. Everything downstream of the feature vector, calibration, fusion, gating, consensus and supervision, is unchanged between the two instantiations; only F , C and the fitted parameters differ.

4.2.2. Layer 1a: the three neural detectors

Fig. 4 summarises the whole reflex layer end to end, showing the role of each stage described in Sections 4.2.2 to 4.2.4. The reflex layer classifies each flow with an ensemble of $M = 3$ heterogeneous deep detectors with distinct inductive biases (recurrence, convolution and dense projection), so that their errors are only partially correlated. The detectors are not new to this paper: they were trained and evaluated on CICIDS2017 in our prior comparative study [56], and are reused here exactly as released, which both documents their provenance and lets this paper focus on the architecture built around them. Table 3 gives the exact architectures as deployed (layer types, dimensions, activations, dropout rates and parameter counts). All three end in a C -way softmax; given a flow x , detector M_m produces a posterior $P(c \mid x, M_m)$ over the C classes, with $C = 15$ on CICIDS2017.

The same three inductive biases are re-instantiated for the cross-corpus study of Section 6.9. Because GeNIS has a different feature schema, its detectors cannot be the released checkpoints of [56] and are retrained from scratch on that corpus, with $C = 9$; Table 3 reports both families side by side. The two sets are deliberately not identical in width or depth, since each was sized for its own corpus, and this is precisely the point of the comparison: what transfers unchanged between the two evaluations is not a particular network, it is the calibration, fusion, gating and

certification machinery built on top of whatever detectors the reflex layer holds. To avoid any ambiguity, the two sets are **alternative instantiations, never a single six-member ensemble**: a deployment loads one manifest and therefore one set of $M = 3$ detectors, and the two are not interchangeable at run time because their input widths and output cardinalities differ. Sections 6.2 to 6.8 report the CICIDS2017 instantiation and Section 6.9 the GeNIS one. Correspondingly, we claim portability of the architecture across corpora, not transfer of learned weights: the GeNIS detectors are trained on GeNIS, and no CICIDS2017 checkpoint is evaluated on GeNIS traffic.

Table 3. Exact architectures of the deployed detectors (Keras). The CICIDS2017 detectors are the released checkpoints of [56]; the GeNIS detectors are retrained for the cross-corpus study of Section 6.9, where the different feature schema ($F = 59$, $C = 9$) precludes reusing them. Together with the preprocessing of Section 4.2.1 and the parameters of Sections 4.2.3 to 4.2.4, this specification is sufficient to reproduce either reflex layer.

Corpus	Detector	Exact layer stack (as deployed)	Input	Params
CICIDS2017 (C = 15)	RNN	SimpleRNN(32, ReLU) → Dropout(0.1) → Flatten → Dense(32)+LeakyReLU → Dense(16)+LeakyReLU → Dense(15, softmax)	(78, 1)	81,775
CICIDS2017 (C = 15)	CNN	Conv1D(64, k=3)+LeakyReLU → MaxPool(2) → Dropout(0.3) → Flatten → Dense(64)+LeakyReLU → Dense(32)+LeakyReLU → Dense(15, softmax)	(78, 1)	158,543
CICIDS2017 (C = 15)	DNN	Dense(256, ReLU) → Dropout(0.1) → Dense(15, softmax)	(78,)	24,079
GeNIS 2025 (C = 9)	RNN	SimpleRNN(64, ReLU) → Dropout(0.3) → Dense(64)+ReLU → Dense(9, softmax)	(59, 1)	8,969
GeNIS 2025 (C = 9)	CNN	Conv1D(64, k=3)+ReLU → MaxPool(2) → Conv1D(32, k=3)+ReLU → Flatten → Dense(64)+ReLU → Dropout(0.3) → Dense(9, softmax)	(59, 1)	66,473
GeNIS 2025 (C = 9)	DNN	Dense(128, ReLU) → Dropout(0.3) → Dense(64, ReLU) → Dropout(0.2) → Dense(9, softmax)	(59,)	16,521

4.2.3. Layer 1b: confidence calibration (temperature scaling)

Raw neural posteriors are notoriously over-confident, which is damaging when a downstream consensus rule must weigh models against one another and reject low-confidence decisions. Every detector is therefore calibrated with **temperature scaling**: a single scalar $T_m > 0$ per model is fitted by minimising the negative log-likelihood on the validation partition (bounded scalar search on $T_m \in [0.05, 10]$), and posteriors are recomputed as $\text{softmax}(z_m / T_m)$ from the logits z_m . The fitted values on CICIDS2017 are $T_{\text{rnn}} = 1.93$, $T_{\text{cnn}} = 1.63$ and $T_{\text{dnn}} = 1.99$, moderate temperatures that quantify a mild in-distribution over-confidence (they grow dramatically when the same models are fed out-of-distribution inputs, Section 8.3.1). Temperature scaling preserves the arg-max, so per-model accuracy is unchanged; what it changes, measurably (Sections 6.3 and 6.4), is the comparability of the confidence scores that the fusion weights and the rejection rule depend on.

4.2.4. Layer 1c: Conscious Weighted Consensus and the rejection gate

The heart of the reflex layer is the fusion of the calibrated detectors into a single, trust-aware decision, which we call **Conscious Weighted Consensus (CWC)**: the ensemble is “conscious” of each member’s historical reliability and weighs its vote accordingly. Fusion is a linear opinion pool of the calibrated posteriors (Eq. 1) with weights proportional to each detector’s weighted

validation precision (Eq. 2), followed by a confidence-gated decision with an explicit rejection region (Eq. 3):

$$\tilde{P}(c | x) = \sum_{m=1}^M w_m \cdot \text{softmax}\left(\frac{z_m}{T_m}\right) \quad (1)$$

$$w_m = \frac{\text{Prec}_m}{\sum_{k=1}^M \text{Prec}_k} \quad (2)$$

$$\hat{c} = \begin{cases} \arg \max_c \tilde{P}(c | x), & \text{if } \max_c \tilde{P}(c | x) \geq \tau, \\ \text{REJECT}, & \text{otherwise.} \end{cases} \quad (3)$$

On clean CICIDS2017 validation data the three precisions are nearly equal, giving near-uniform weights ($w_{\text{rnn}} = 0.332$, $w_{\text{cnn}} = 0.323$, $w_{\text{dnn}} = 0.345$), so in the clean regime the CWC behaves close to a uniform pool, with a small measured margin over it (Section 6.4). The decisive value of Eq. 2 is a **security mechanism**. If a detector is corrupted, for instance by label-flipping poisoning, its validation precision collapses, its weight shrinks automatically, and its influence on the fused decision vanishes **without any explicit attack detection** (measured in Section 6.7). For continuous operation the weights follow a mini-batch Dirichlet update: starting from a uniform prior $\alpha m = 1$, each window of W certified flows increments αm by the number of correct certified detections attributed to M_m , weights are renormalised, and a decay factor down-weights ageing evidence so the “consciousness” tracks concept drift.

A remark on novelty is in order. Algorithmically, the CWC is deliberately assembled from standard, well-understood components: a precision-weighted linear opinion pool over temperature-scaled posteriors with a reject option. The name designates the *architectural role* this reliability-weighted calibrated fusion plays in BAg-IDS (the stage that makes downstream consensus and rejection meaningful and that absorbs a corrupted member), not an algorithmic invention. The contribution of this paper is architectural and empirical, namely the co-design and subsystem-by-subsystem measurement of the full stack, rather than a new fusion algorithm.

The rejection gate (Eq. 3, $\tau = 0.70$ by default) makes the layer’s output ternary: **benign**, **attack**, or **rejected**. Rejection is not a failure but a routing decision with two consumers: rejected flows are withheld from automated remediation, and their rate is streamed to the Data Scientist as the primary drift signal. The choice of τ therefore simultaneously sets the reliability of what Layer 3 will certify and the load offered to Layer 2, a coupling we measure explicitly in Section 6.5.

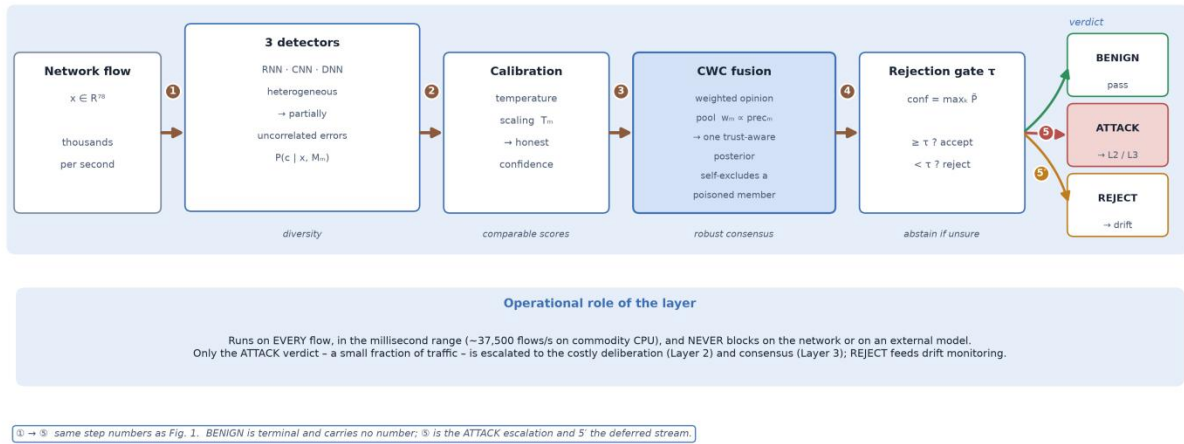


Fig. 4. The reflex detection layer at a glance. A raw flow passes through three heterogeneous detectors (diverse errors), temperature calibration (comparable confidences), the Conscious Weighted Consensus (one trust-aware posterior, self-excluding a poisoned member) and the rejection gate, yielding one of three verdicts. The layer runs on every flow at line rate and never blocks; only ATTACK is escalated to Layers 2 and 3, and REJECT feeds drift monitoring. The badges are steps 1 to 5 of Fig. 1.

4.2.5. Layer 2: deliberation, single-shot judgement and the agentic Investigator

An accepted attack verdict is necessary but not sufficient to act: the system must still decide *how severe* the event is and *what to do* about it. The Sentinel assembles a compact event summary (fused verdict and confidence, top pooled posteriors, per-model decisions, recent verdict counts) and submits it to an LLM under a strict instruction to return a single structured object: a *severity* (low/medium/high/critical), a technical *rationale*, and *recommended actions* from a fixed vocabulary (block, rate-limit, isolate, recalibrate, retrain, notify, no-op). Invocation is sparing (on a candidate attack or a rejection burst) and rate-limited; the deliberation is produced *before* the alert is signed and submitted, and travels with it to the notaries, so the word *certified* is reserved throughout for the post-quorum state. Two backends sit behind one interface (a cloud model via OpenRouter, a local model via LM Studio), and if no LLM is reachable a deterministic **rule-based fallback** produces a valid deliberation from the same summary: the layer degrades gracefully, never blocks autonomy, and remains reproducible.

Whereas the Sentinel's deliberation is single-shot, the **Investigator** embodies agency in the stronger sense: an explicit agent loop (*perceive* $\rightarrow$ *reason* $\rightarrow$ *act (tools)* $\rightarrow$ *reflect* $\rightarrow$ *conclude*) in which the agent itself decides which tools to invoke. Its tools query the ledger for recent incidents of the same class and correlate them in time to detect bursts indicative of a coordinated campaign; an internal memory tracks campaigns already observed. The output is a structured incident report plus the full trace of steps (exposed verbatim in the supervision console), and the conclusion is itself anchored on-chain, so even the agent's autonomous reasoning is auditable. Two boundaries are deliberate: the Investigator is dispatched when the alert is submitted and runs *in parallel with* the certification round, off the critical path (it never gates certification, and its report is anchored once the round has concluded), and no quantitative claim of Section 6 depends on this non-deterministic layer, a separation we regard as methodologically essential when an LLM is placed in an otherwise measurable system. Fig. 5 summarises the layer: its two entry points, the single-

shot and agentic paths, the backend fallback chain, and the two outputs.

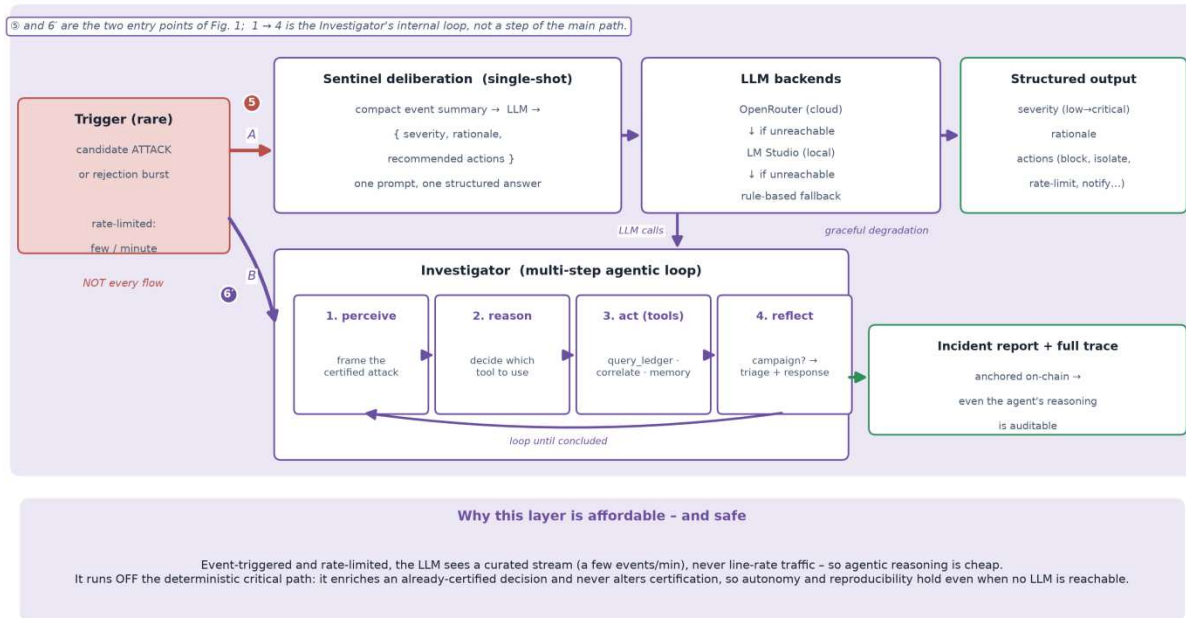


Fig. 5. The deliberative layer at a glance. A rare, rate-limited trigger (a candidate attack or a rejection burst) invokes either the Sentinel's single-shot deliberation (A) or the Investigator's multi-step agentic loop (B: perceive → reason → act with tools → reflect). Both target the same LLM behind a graceful OpenRouter → LM Studio → rules fallback, and their outputs (structured severity/actions, and an on-chain-anchored incident report) enrich but never alter the deterministic certification, so autonomy and reproducibility hold even with no LLM reachable. The layer has two distinct entry points, step 5 for (A) and step 6' for (B); the loop's own 1 to 4 are internal to the Investigator.

4.2.6. Layer 3: consensus, identities, votes, and the validateVote chaincode

Fig. 6 details the deployed Hyperledger Fabric network: two organisations, each with its certificate authority and one endorsing peer carrying the validateVote chaincode and a full ledger copy; an etcdraft ordering service (default 2 s BatchTimeout); and the HTTP-to-gRPC gateway through which the agents invoke the contract. The endorsement policy requires the signatures of both organisations on every transaction, so no single organisation can commit state alone. Fig. 7 then shows the complete consensus sequence for one alert, from the Sentinel's signed alert to the sealed certification, which is also the key to reading the latency measurements of Section 6.6.

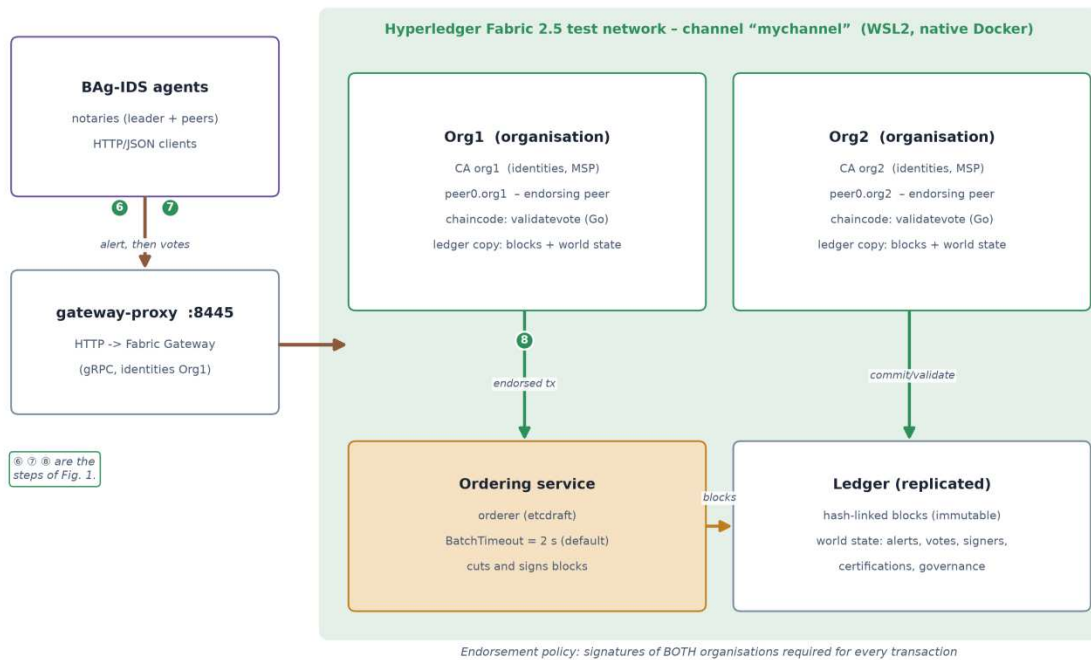
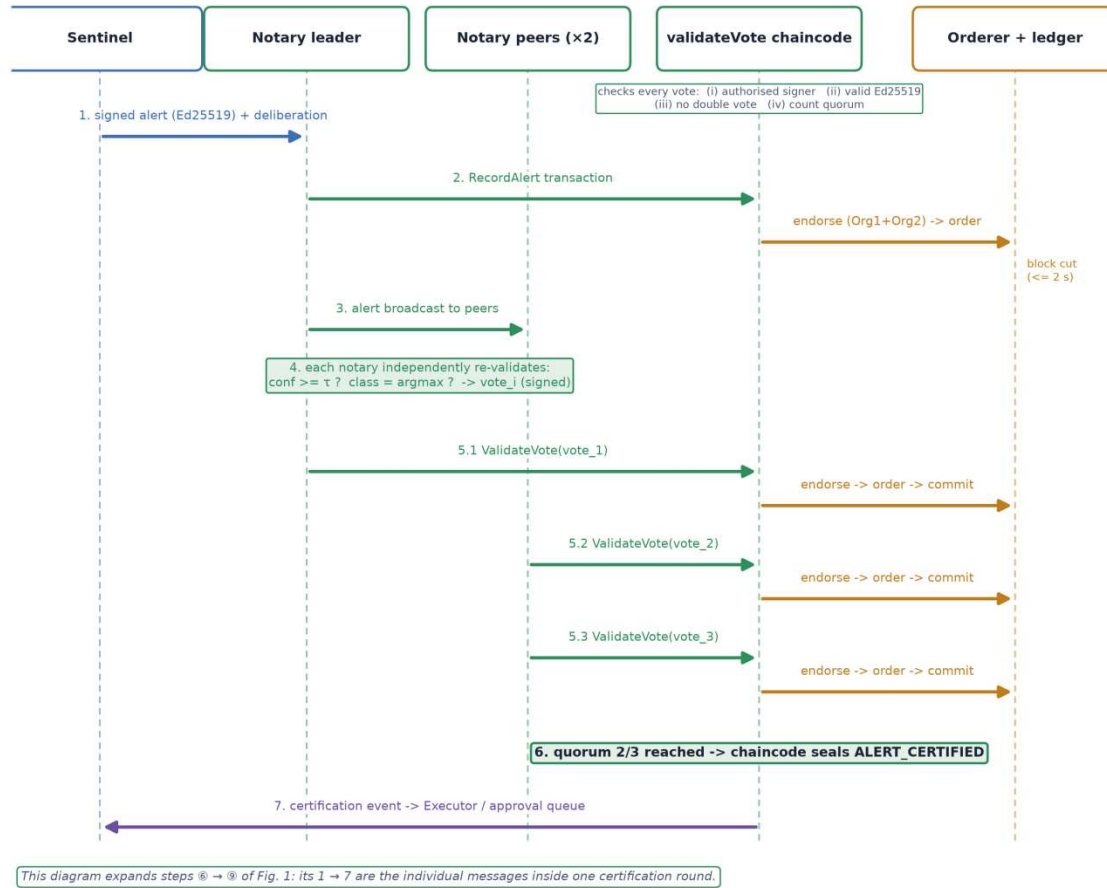


Fig. 6. Anatomy of the deployed Hyperledger Fabric network: organisations, certificate authorities, endorsing peers with the validateVote chaincode, ordering service, replicated ledger, and the agents' HTTP gateway. Both organisations must endorse every transaction. The badges are steps 7 and 8 of Fig. 1.



Measured: ~2.06 s per transaction (Table 8) – the (1 + quorum) scaling comes from SEQUENTIAL submission; concurrent votes would take ~2 block intervals regardless of quorum (§6.6).

Fig. 7. Consensus sequence for one alert: signed alert, RecordAlert transaction, independent notary re-validation, one ValidateVote transaction per vote (with the four chaincode invariants), quorum sealing of ALERT_CERTIFIED, and the certification event that releases the downstream agents. Each transaction follows Fabric’s execute–order–validate path (~2.06 s measured, Table 9). This diagram expands steps 6 to 9 of Fig. 1: its own 1 to 7 are the individual messages exchanged inside one certification round, a finer grain than the global path.

The consensus layer converts a single agent’s alert into a collectively certified, tamper-evident record. Every agent holds an Ed25519 key pair; the public keys of authorised notaries are registered in the ledger state. A notary endorses an alert only after re-validating it independently (checking that the verdict exceeds τ and that the claimed class is the arg-max of the pooled posteriors) and signs its vote over the canonical serialisation of the vote payload. The *validateVote* chaincode (Go) enforces four invariants on every submitted vote: (i) the signer is an authorised notary; (ii) the Ed25519 signature over the canonical payload verifies against the registered key; (iii) the notary has not already voted on this alert (defeating replay); and (iv) once valid approving votes reach the quorum (2 of 3 in our deployment), the contract itself seals an *ALERT_CERTIFIED* record. Because these checks run inside the chaincode and are endorsed by peers of two independent organisations, a compromised agent cannot forge an endorsement, cannot certify alone, and cannot tamper with the record afterwards. The identical logic is first unit-tested against a hash-chained simulated ledger, then deployed unchanged in semantics as Fabric chaincode, which is also what lets Section 6.6 isolate the infrastructure cost of Fabric from the intrinsic cost of the consensus rule.

The scope of this re-validation must be stated precisely. Each notary checks the *internal consistency* of the alert (threshold and arg-max over the posteriors carried in the alert itself) and the authenticity of every signature; it does not independently re-classify the underlying flow. The quorum therefore guarantees the **authenticity and consistency** of the certified record, and that no single agent can certify alone; it does not, by itself, guarantee the *veracity* of an alert whose posteriors were fabricated wholesale by a fully compromised Sentinel. Section 7.2 analyses this boundary and the compensating controls (behavioural rate monitoring, human review); notary-side independent re-classification is an explicit direction for future work, with a measurable latency cost.

A notary may optionally attach an LLM **second opinion** to its endorsement, a brief natural-language judgement of whether the alert looks genuine. This is deliberately **advisory only**: the endorsement decision remains the deterministic re-validation, the signed vote payload does not contain the opinion, and the opinion never changes whether the notary approves. The advice is logged separately and surfaced for audit, adding contextual judgement without introducing a non-deterministic factor into the cryptographic consensus.

4.2.7. Layer 4: governed human-in-the-loop supervision

Fig. 8 summarises the human's role in one view: **(a)** the capability ladder of the three operator roles (the reader observes, the analyst additionally decides, the administrator additionally governs); **(b)** the two operating modes side by side, showing exactly where the human sits on the decision path in supervised mode (the approval queue for severity-gated remediations) and that no human is on the path in autonomous mode; and **(c)** the accountability rule that makes the whole layer trustworthy: every human action is itself a signed, on-chain transaction.

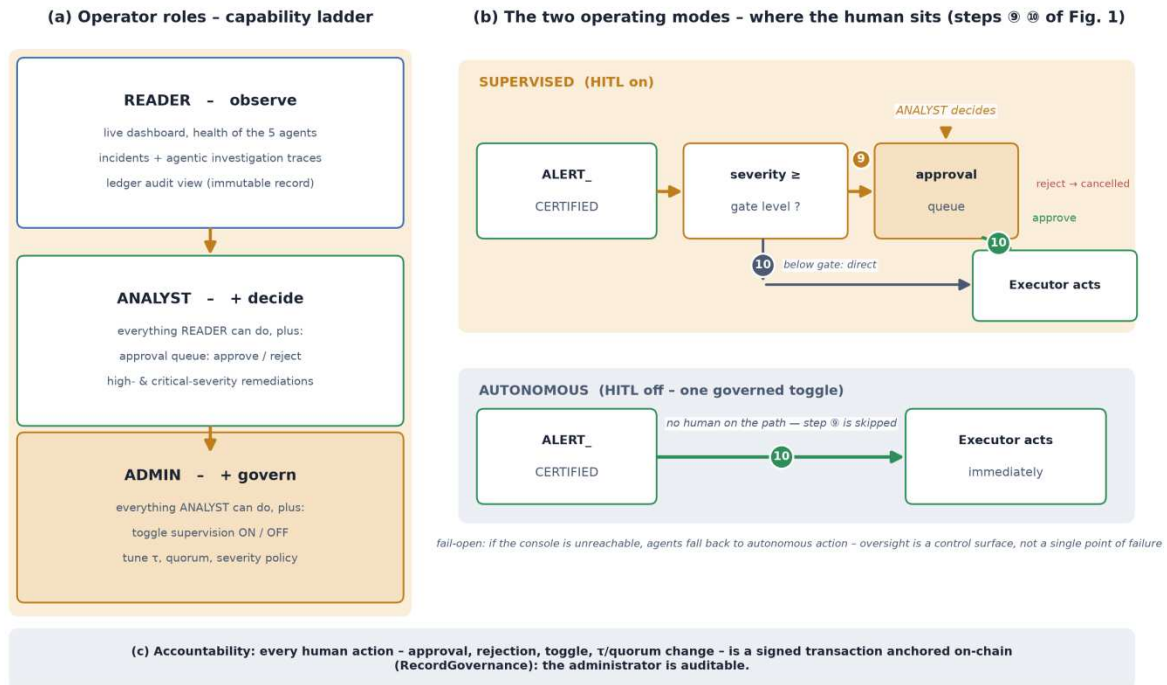


Fig. 8. Layer 4, the human's role: (a) the reader $\subset$ analyst $\subset$ admin capability ladder; (b) the supervised versus autonomous decision paths (one governed toggle apart, fail-open on console failure); (c) every human action is a

signed transaction anchored on-chain (RecordGovernance). The badges are steps 9 and 10 of Fig. 1; switching to autonomous mode is exactly what removes step 9.

The supervision console (React front-end, FastAPI back-end, trilingual FR/EN/AR) is the operator-facing face of the system, with three roles partitioning access: reading is open to all authenticated users, incident approval requires the analyst role, and governance the administrator role. It aggregates in real time the health of the agents, the live verdict stream, the incident list, the Investigator's reasoning traces, and exposes the blockchain as a first-class audit view: what the operator sees is the immutable record itself, not a mutable application log. When supervision is enabled, remediations above a configurable severity are held in an approval queue and the Executor acts only on analyst approval; when disabled, the queue is bypassed and the agents act autonomously. The gate is fail-open with respect to availability: if the supervision layer is unreachable, agents fall back to autonomous action rather than stalling the response.

Because a console that can lower τ , shrink the quorum, or disable oversight is itself a high-value target, every administrative action is a **first-class, signed transaction**: toggling supervision, changing a governed parameter, and every manual approval are serialised, signed by a registered governance identity, and anchored on-chain via a *RecordGovernance* contract call that verifies the signature before committing. The chaincode distinguishes the human actor (recorded as metadata) from the cryptographic signer, binding the human chain of responsibility to an unforgeable record: the ledger holds not only what the agents decided, but every human intervention that shaped their behaviour.

4.2.8. Layer 4b: explaining the decision at the approval surface

Supervision is only trustworthy if the analyst can see *why* a remediation is being proposed. Following the same asymmetric-cost principle that governs deliberation and consensus, BAg-IDS explains only the remediations that are actually held for human approval, a rare, non-benign population, so that a comparatively expensive explanation is paid exactly where a human must decide. For each queued flow the Sentinel computes model-agnostic Shapley-value attributions (KernelSHAP [77]) over the *fused* ensemble decision function itself, that is the calibrated, weight-pooled posterior rather than any single detector; the explanation is therefore faithful to the decision the system actually took, and robust to changes in the underlying Keras/TensorFlow models. The result is the signed contribution of every input feature to the predicted class, of which the eight most influential are surfaced.

The attributions feed a two-part decision aid whose trust property is a strict separation of concerns. First, a **deterministic recommendation** is derived from an analyst knowledge base covering the deployed CICIDS2017 classes: a remediation is recommended for direct approval only when the fused confidence is high (at least 0.90), the three detectors are unanimous, and the class is not known to be false-positive-prone; otherwise the system returns a *verify-first* recommendation that names the specific reasons (low confidence, detector disagreement, or a false-positive-prone class). This recommendation never originates from a language model. Second, an optional language-model narration reformulates the attributions and the knowledge-base entry into trilingual prose for the console, but is explicitly forbidden to introduce any feature outside the computed attribution set; it is confined to presentation and cannot alter or fabricate the decision. The decision-relevant outputs, what drove the verdict and what the analyst should check, are thus deterministic and grounded, while the generative component is strictly cosmetic.

Section 6.10 measures the faithfulness and cost of these explanations.

4.2.9. Deployment topology and reproducibility summary

The deployed topology (ports in Fig. 3) is: Sentinel :8001, Notary leader :8002 with peers :8012 and :8022, Executor :8003, Data Scientist :8004, Investigator :8005, and the supervision console :8090, all FastAPI services; the Fabric 2.5 test network (one orderer with the default 2 s BatchTimeout, two organisations with one peer each, three certificate authorities) runs under WSL2 with a native Docker engine. To reproduce the system end to end: preprocess as in Section 4.2.1 (down-sampling seed 123, split seed 1), train the three CICIDS2017 detectors of Table 3 (Adam, 30 epochs, batch sizes 64 to 512 as in [56]), fit temperatures and fusion weights on the validation partition (Sections 4.2.3 and 4.2.4), set $\tau = 0.70$ and quorum = 2/3, and deploy the validateVote chaincode with the invariants of Section 4.2.6. Every experimental artefact of Section 6 (metrics, figures, statistical tests) is regenerated by a single script from the released model checkpoints and the frozen test partition; the cross-corpus study of Section 6.9 ships its own bundle (retrained checkpoints, audited feature list, scaler parameters, and saved per-model probabilities) so that its calibration, fusion and gating are reproducible without retraining.

This layering yields the central operational trade-off of BAg-IDS: the reflex layer detects in milliseconds, deliberation adds seconds only for genuine attacks, the blockchain adds a bounded, quorum-dependent latency in exchange for decentralised, cryptographically verifiable trust, and the supervision layer adds optional human authority without surrendering auditability. Section 6 measures each of these costs subsystem by subsystem; Section 7 shows which adversarial behaviours each layer neutralises.

5. Implementation

BAg-IDS is implemented as a working system rather than a paper design, and every result in Section 6 is produced by the artefact described here. This section records the technology choices, the reproducibility provisions, and the engineering decisions that a reader would need to rebuild or extend the system.

5.1. Technology stack

The detection and agent layers are written in Python. The three detectors are Keras/TensorFlow models; calibration, fusion and metrics are implemented with NumPy, SciPy and scikit-learn. Each of the five agents is an independent service built on FastAPI and served by Uvicorn, communicating over HTTP with Pydantic-validated JSON messages. The deliberative layer talks to a large language model through the OpenAI-compatible chat-completions interface, which lets the same client address either a cloud model via OpenRouter or a local model served by LM Studio; a deterministic rule-based fallback keeps the system autonomous, and the measurements reproducible, when no model is reachable.

Agent identities are Ed25519 key pairs generated with the *cryptography* library. The consensus layer has two interchangeable back ends behind a single interface: a local, hash-linked JSONL ledger used for development and unit testing, and a Hyperledger Fabric 2.5 network used for the on-chain deployment. The *validateVote* chaincode is written in Go with the contract API; a small Go gateway process, built on the Fabric Gateway SDK and shown as gateway-proxy :8445 in Fig.

6, exposes the chaincode to the Python agents as two local HTTP endpoints (submit and evaluate), so that switching a notary from the simulated to the on-chain back end is a one-line configuration change with no modification to its logic. The supervision layer is a FastAPI backend with a React (Vite) single-page frontend; it is trilingual (English, French, Arabic with right-to-left layout) and role-based, and it is optional at run time through the governed toggle of Section 4.2.7.

5.2. Deployment environment

The evaluation runs on a single Windows workstation. The Python agents, the supervision application and the detectors execute natively on Windows; the Fabric network runs under WSL2 (Ubuntu) with a Docker engine installed directly in the Linux distribution rather than through Docker Desktop. localhost port forwarding between Windows and WSL2 lets the Python gateway client reach the Go gateway process transparently. This heterogeneous but commodity setup is deliberately modest: it demonstrates that the full stack (calibrated ensemble, agentic deliberation, permissioned blockchain and a supervision console) runs on a single developer machine, which matters for reproducibility and for adoption by teams without a cluster.

5.3. Reproducibility

The pipeline is designed so that a measurement can be regenerated from raw data. Data loading reproduces the detectors' original training preprocessing seed for seed (Section 4.2.1: balanced down-sampling with seed 123, RobustScaler, 60/20/20 split with seed 1); the fitted temperature scalers, fusion weights, rejection threshold, class list and the scaled test partition are exported together as a self-contained deployment bundle, which the Sentinel loads at start-up so that the agents reproduce exactly the configuration measured offline. Legacy model files are loaded through a compatibility shim that rewrites deprecated layer attributes, so the original trained detectors run unchanged under current TensorFlow. Each experiment of Section 6 is a single command that writes its results as JSON, from which the tables and the 300-dpi figures are regenerated by a synthesis script; the paper's numbers and plots are therefore traceable to those files.

We emphasise one provenance point because it is a common and consequential pitfall. The detectors are reused from our prior study [56], and reproducing their results required reconstructing the exact preprocessing under which they were trained, not merely the model weights. Released weights without their preprocessing are not a reproducible artefact: as Section 8.3.1 documents, evaluating the same checkpoints under a plausible but mismatched pipeline produced drastically worse and misleading figures. We therefore release the complete preprocessing specification (scaler type, balancing, seeds, split) alongside the deployment bundle.

5.4. The running system

Fig. 9 shows the deployed system in operation. Panel (a) is the supervision dashboard during a live run: the pipeline is active, all **five agents report online**, and the verdict distribution updates alongside a governance activity feed whose last entries record an administrator disabling oversight and changing τ and the quorum, each an on-chain governance transaction (Section 4.2.7). Panel (b) exposes the Investigator's **autonomous reasoning trace** for a certified DoS-

Hulk incident, showing the four steps *frame* → *query_ledger* → *correlate* → *llm_reason* with the thought, the tool invoked and the observation returned: the agent queries the ledger, finds 10 related on-chain incidents, correlates them in time (burst at ~2 incidents/min), reasons over the assembled evidence to escalate severity to **critical**, and recommends a graded response, with the conclusion anchored on the ledger. The view makes the perceive–reason–act–reflect loop, and its evidentiary basis, auditable rather than opaque.

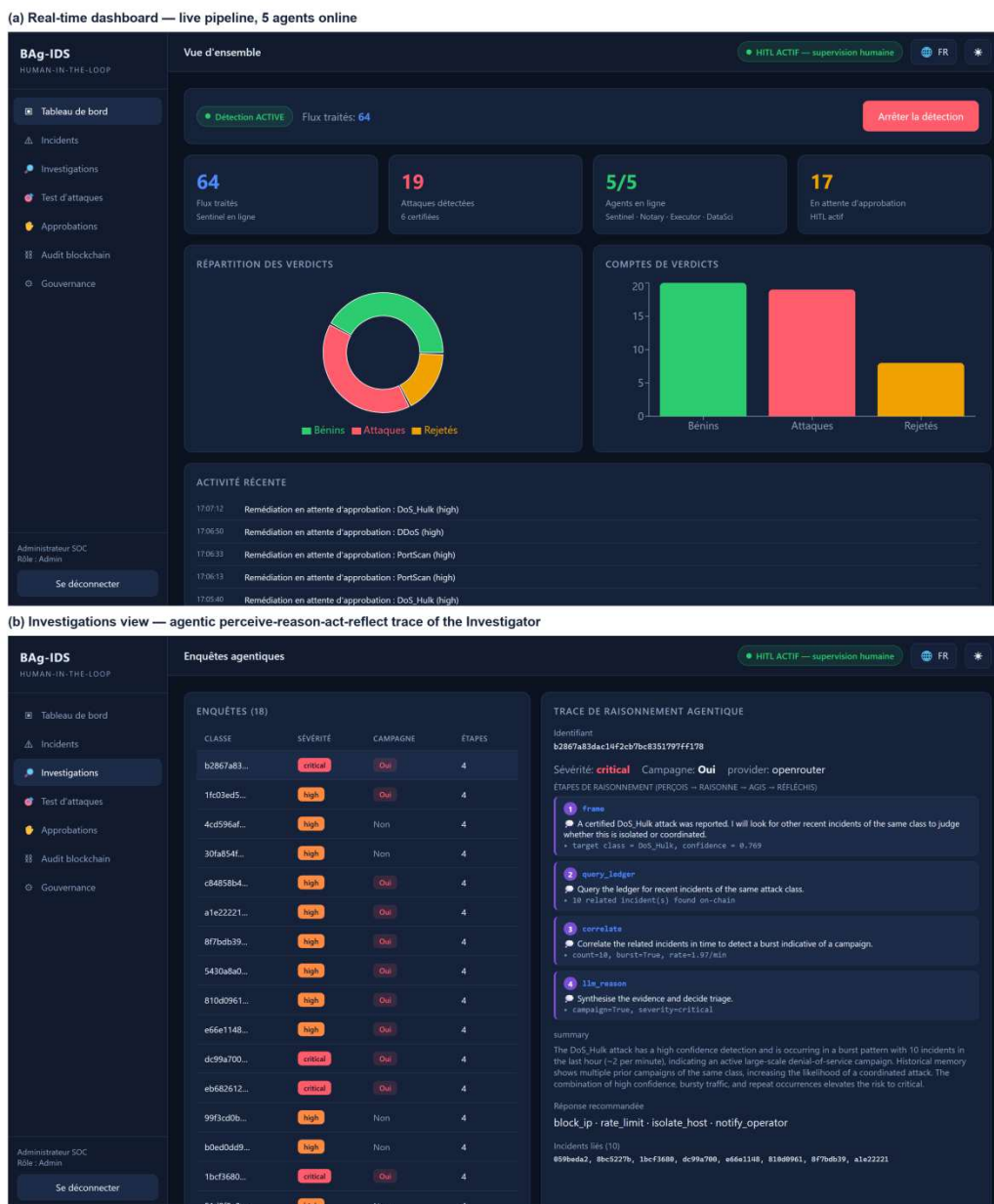


Fig. 9. The deployed BAg-IDS supervision console in operation. (a) Real-time dashboard: active pipeline, five agents online, live verdict distribution, and a governance-aware activity feed. (b) Investigations view: the Investigator’s autonomous perceive–reason–act–reflect trace for a certified DoS-Hulk incident, correlating 10 on-chain incidents into a detected campaign and reasoning through a cloud LLM (OpenRouter) to a critical-severity triage. The screenshots are from a live run, not a mock-up.

The console is trilingual and role-based (Section 4.2.7). In the run shown, human supervision is

enabled and the Investigator’s reasoning is served by a cloud language model through OpenRouter (the trace is labelled “openrouter”); the same loop degrades to a deterministic rule-based fallback when no model is reachable, so the agentic loop, campaign correlation and on-chain anchoring operate with or without an external model, exactly as the design requires.

5.5. Engineering notes

Two engineering decisions are worth recording because they shape the measurements. First, the reflex loop processes flows in micro-batches and performs all per-flow work (inference, temperature scaling, fusion, thresholding) with vectorised NumPy/TensorFlow operations, which is what keeps detection in the millisecond range while the consensus layer operates in seconds. Second, the deliberative layer is rate-limited and event-triggered: it is invoked only on a candidate attack or a burst of rejections, and a minimum interval between calls bounds both the load on the LLM and its influence on end-to-end latency, so that the language model annotates a curated stream of events rather than sitting on the per-flow critical path. The complete source, configurations and experiment scripts are organised to be run from a single repository, and an operational guide documents the cold-start sequence for both the simulated and the on-chain modes.

6. Experimental Evaluation

We evaluate BAg-IDS **subsystem by subsystem**, mirroring the pipeline of Fig. 3: the three individual detectors (Section 6.2), the calibration stage (6.3), the CWC fusion (6.4), the rejection gate and the load it offers to the deliberative layer (6.5), the consensus layer (6.6), the robustness to detector corruption that the fusion weights provide (6.7), a functional validation of the deliberative/agentic layer (6.8), a cross-corpus generalisation study on an independently generated 2025 corpus (6.9), and the faithfulness and cost of the explanation layer (6.10). Each subsection isolates one subsystem, reports its measured behaviour, and quantifies the **impact of its output on the subsystem that consumes it**. Section 6.12 assembles the chain. Every number and every curve below is measured on a real run of the implemented system (no value is projected or simulated), and the key comparisons are accompanied by a statistical analysis (95% bootstrap confidence intervals and McNemar tests).

6.1. Experimental setup and statistical methodology

Experiments use CICIDS2017 in its 15-class form under the **exact preprocessing of the detectors’ original training pipeline** [56], which we recovered from the training notebook and reproduce seed for seed (Section 4.2.1): the cleaned corpus is balanced by down-sampling benign traffic to the 390,222 attack flows, features are scaled with a RobustScaler, and the data are split 60/20/20 into training, validation and test partitions; the frozen test partition contains 156,089 flows. The detectors are those of Table 3, reused exactly as released by [56]; temperatures and fusion weights are fitted on the validation partition only. The agent layer runs as FastAPI services on a single Windows workstation; the consensus layer is evaluated both on a local hash-chained ledger and on a Hyperledger Fabric 2.5 test network (one orderer, two organisations with one peer each, three CAs) under WSL2 with a native Docker engine. Unless stated otherwise, $\tau = 0.70$ and the quorum is 2 of 3 notaries.

Two conventions apply throughout. First, besides the 15-class metrics we report **operational binary metrics** (BENIGN versus ATTACK, the decision that actually triggers remediation) with the attack class as positive; the operational FPR is the fraction of benign flows raised as attacks, and is measured on the balanced evaluation set of the original protocol. Second, all headline metrics carry a **95% bootstrap confidence interval** (1,000 resamples of the test partition, percentile method), and pairwise model differences are tested with **McNemar’s test** (χ^2 with continuity correction) on per-flow correctness, the appropriate paired test when two classifiers are evaluated on the same test set.

Why these two corpora. Our evaluation pairs CICIDS2017 with GeNIS 2025 (Section 6.9), and the pairing is deliberate. CICIDS2017 is kept because it is the community reference on which the reflex detectors were originally trained and published [56]: it preserves strict comparability with our prior work and with the systems of Table 12, and its 15-class taxonomy exercises the rejection gate under genuinely hard conditions. GeNIS is chosen as the second corpus over the more obvious candidates for three reasons. CSE-CIC-IDS2018 would not be an independent test: it shares CICIDS2017’s laboratory, generation tooling and flow exporter, and therefore the documented flaws of that tooling [50-52]. Older references such as UNSW-NB15 or NSL-KDD predate the current threat landscape by a decade or more, inviting the obsolescence objection they routinely receive. GeNIS, in contrast, is generated by a different laboratory with a different exporter, captured in February 2025, publicly archived [80], and, decisively, its attack families overlap CICIDS2017’s (FTP/SSH brute force, Hulk, slowloris), so cross-corpus differences measure generalisation rather than a change of task. Specialised IoT/IoMT corpora of the same generation were set aside because their traffic profile departs from the enterprise-network setting this paper targets.

Known limitations of the benchmarks and of the protocol. CICIDS2017 remains the community reference, but its generation tooling and labelling have documented flaws: Engelen et al. identified errors in the CICFlowMeter feature extractor and in the labels themselves [50], and subsequent studies quantified their material effect on reported performance [51, 52]. These flaws particularly affect the minority attack classes, and we interpret per-class figures accordingly. Two properties of the original protocol [56] are likewise stated openly: the scaler is fitted on the full balanced corpus before splitting (its statistics are label-free medians and quantiles over 780K flows, so the induced leakage is negligible but non-zero), and evaluation is on a class-balanced testbed, so operational rates on raw, benign-dominated traffic will differ. We keep both choices for strict fidelity to the published detectors. Consensus-latency results (Section 6.6) rest on 40 certified alerts for the per-quorum characterisation, complemented by a 360-alert benchmark comparing sequential and concurrent vote submission on the same network; both should be read as a characterisation of this deployment.

6.2. Subsystem 1: the individual detectors

We first evaluate each detector in isolation, exactly as it enters the pipeline. Table 4 reports the 15-class metrics, the operational binary metrics with confidence intervals, and the threshold-independent ROC-AUC and average precision; Fig. 10 shows the operational ROC and precision–recall curves; Fig. 11 the per-class F1; Fig. 12 the row-normalised confusion matrices.

Table 4. Measured performance on the CICIDS2017 test partition of the original protocol (156,089 flows, balanced).

Acc-15/F1-w: 15-class accuracy and weighted F1. Oper. F1, ROC-AUC, AP, FPR: operational BENIGN-vs-ATTACK metrics; CIs from 1,000 bootstrap resamples.

Model	Acc-15 (%)	F1-w (%)	Oper. F1 [95% CI]	ROC-AUC	AP	FPR (%)
RNN	90.68	90.50	0.903 [0.901, 0.905]	0.998	0.998	0.05
CNN	89.10	88.29	0.906 [0.904, 0.908]	0.985	0.977	1.01
DNN	95.27	94.74	0.952 [0.951, 0.953]	0.985	0.985	0.17
CWC ensemble (all decisions)	96.60	96.39	0.967 [0.966, 0.968]	0.999	0.999	0.09
CWC + reject, certified only ($\tau=0.70$)	99.65	99.62	0.996	n/a	n/a	0.04

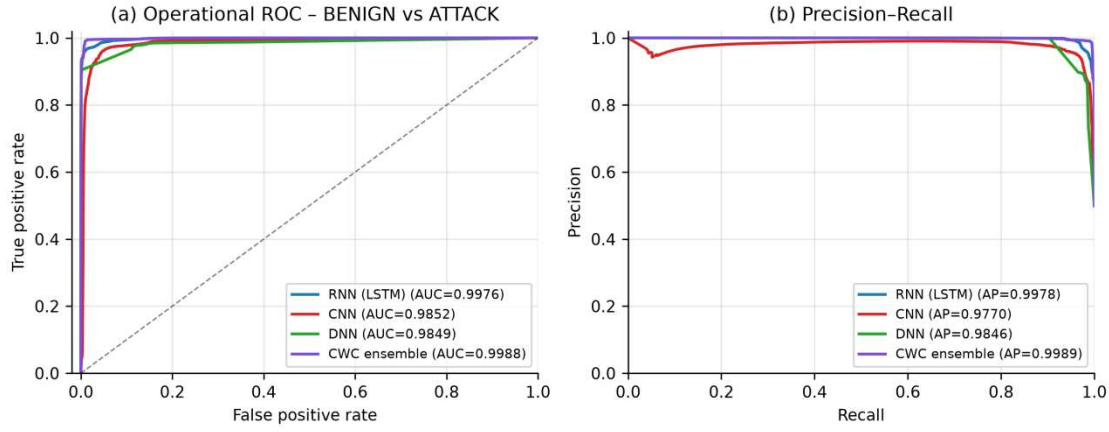


Fig. 10. Operational ROC (a) and precision–recall (b) curves of the three detectors and the CWC ensemble on the test partition.

Three findings emerge. **(i) All three detectors are operationally strong but complementary:** the DNN has the best 15-class accuracy (95.27%), the RNN the best ranking quality (AUC 0.998) and an extremely low FPR (0.05%), and the CNN, the weakest at the arg-max, still contributes high-quality ranking (AUC 0.985). Their error sets differ enough for fusion to profit, which Section 6.4 confirms. **(ii) The per-class picture is strong on the operationally dominant families** (Fig. 11): DDoS 0.975, DoS Hulk 0.973, PortScan 0.993, FTP/SSH-Patator 0.99/0.98 ensemble F1. **(iii) The residual weaknesses are the rare classes:** Bot (F1 0.52, 380 test flows), DoS slowloris (0.53), the web attacks, and the vanishing classes Heartbleed (1 flow) and Infiltration (9 flows) are missed or unstable, the region where the labelling flaws of the benchmark [50–52] are also concentrated. We report this candidly; it is exactly the low-confidence region that the rejection gate (Section 6.5) is designed to withhold from automated remediation.

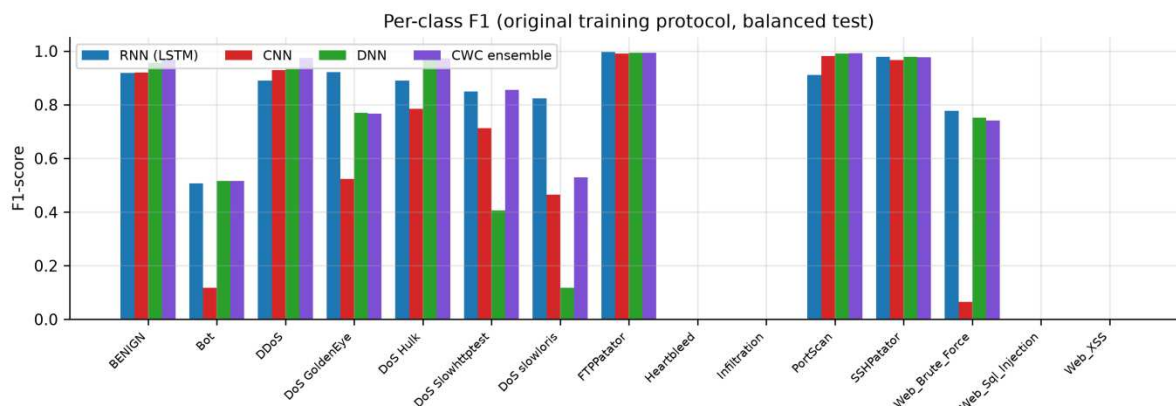


Fig. 11. Per-class F1 of the three detectors and the CWC ensemble (test partition, 15 classes).

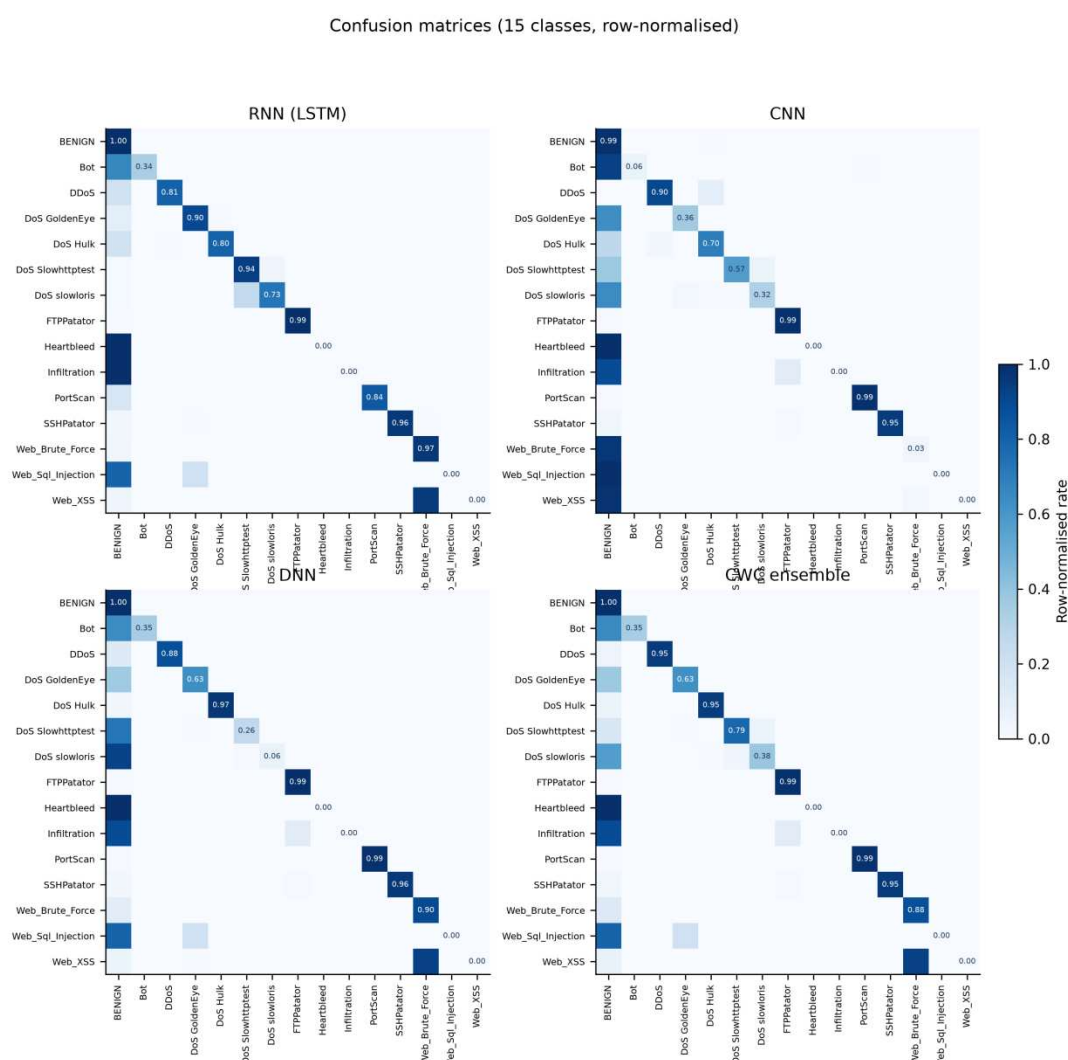


Fig. 12. Row-normalised confusion matrices of the three detectors and the ensemble.

6.2.1. Statistical analysis

Fig. 13 plots the operational F1 of each model with its 95% bootstrap interval; the intervals are

narrow ($\pm \sim 0.002$) and the ensemble’s interval is disjoint from every detector’s. Table 5 reports McNemar’s test on per-flow 15-class correctness for every pair: all differences are significant, and, unlike a weaker regime where fusion merely averages, **the ensemble is statistically superior to each of its members** ($\chi^2 = 709$ against the strong DNN, $p \approx 10^{-156}$). The model ranking of Table 4 is not a sampling accident, and every downstream comparison rests on statistically separated baselines.

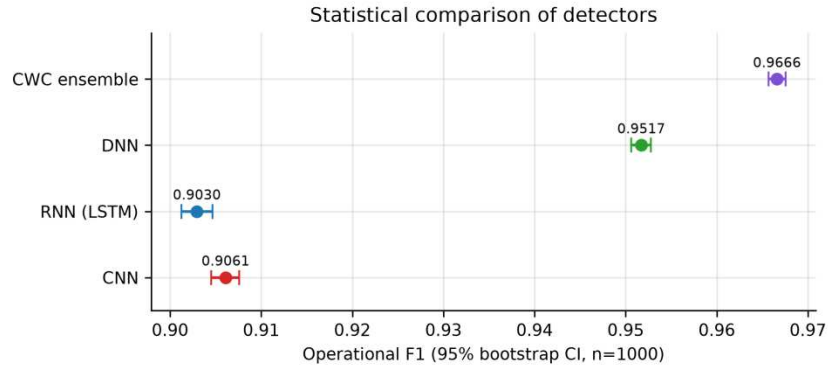


Fig. 13. Operational F1 with 95% bootstrap confidence intervals (1,000 resamples).

Table 5. McNemar tests (χ^2 , continuity-corrected) on per-flow correctness over the 156,089-flow test partition. The ensemble significantly outperforms every individual detector.

Pair	χ^2 (McNemar)	p-value	Interpretation
RNN vs CNN	296	2.2×10^{-66}	RNN > CNN (15-class)
RNN vs DNN	2,900	$< 10^{-300}$	DNN > RNN
CNN vs DNN	6,359	$< 10^{-300}$	DNN $\gg$ CNN
Ensemble vs RNN	7,113	$< 10^{-300}$	ensemble $\gg$ RNN
Ensemble vs DNN	709	2.9×10^{-156}	ensemble > DNN
Ensemble vs CNN	9,562	$< 10^{-300}$	ensemble $\gg$ CNN

6.2.2. Reflex-path cost and context with the state of the art

Because Table 1 uses line-rate viability (C8) as a discriminating criterion, we measure it rather than assert it. On the evaluation workstation (CPU only), the complete reflex pipeline (three detector inferences, temperature scaling, CWC fusion and the reject decision) sustains **37,524 flows/s** in batched operation (measured over 363,860 flows), i.e. an amortised cost of 0.027 ms per flow. Latency for an isolated single flow is framework-bound rather than model-bound: an eager single-flow pass measures 199 ms (median; p95 407 ms), dominated by per-call TensorFlow overhead, which is why the Sentinel processes flows in micro-batches. The reflex layer is comfortably line-rate for the flow volumes of the evaluated corpus on commodity hardware.

For context, recent deep and Transformer-based systems report 97–99.9% accuracy on CICIDS2017-family benchmarks [2, 5, 9, 10]. Under the original training protocol our measurements sit in the same regime: 96.60% for the ungated ensemble and 99.65–99.97% on the certified stream (Section 6.5). Cross-paper comparison nevertheless remains indicative at best (published protocols differ in class sets, rebalancing and splits [51]), which is why our claims

rest on within-pipeline comparisons and on the selective operating point evaluated against a selective-classification baseline in Section 6.5. We note for transparency that an earlier draft of this work evaluated the same released checkpoints under a reconstructed (mismatched) preprocessing and measured substantially lower absolute figures; recovering the original pipeline of [56] closed that gap entirely, a cautionary tale about preprocessing provenance that we discuss in Section 8.3.1.

Impact on the next subsystem. The detector stage hands three posterior vectors per flow to the calibration stage: one detector strong at ranking (RNN), one strong at the arg-max (DNN), and one weaker but complementary member (CNN). What the downstream fusion and rejection can achieve depends on making these three confidence scales comparable, measured next.

6.3. Subsystem 2: confidence calibration

Table 6 gives, per model, the fitted temperature and the Expected Calibration Error before and after scaling; Fig. 14 shows the reliability diagrams. In-distribution, the detectors are only mildly over-confident (the fitted temperatures are moderate, 1.63–1.99, and the raw ECEs small, 0.018–0.051), in sharp contrast with the pathological miscalibration they exhibit under a mismatched preprocessing (Section 8.3.1).

Table 6. Temperature scaling per detector: fitted temperature and Expected Calibration Error (15 bins) before/after, on the test partition.

Model	Fitted T	ECE before	ECE after	Effect
RNN	1.93	0.051	0.024	−53% (largest repair)
CNN	1.63	0.040	0.053	≈ neutral
DNN	1.99	0.018	0.039	slightly degraded
CWC ensemble	n/a	0.047	0.092	see discussion

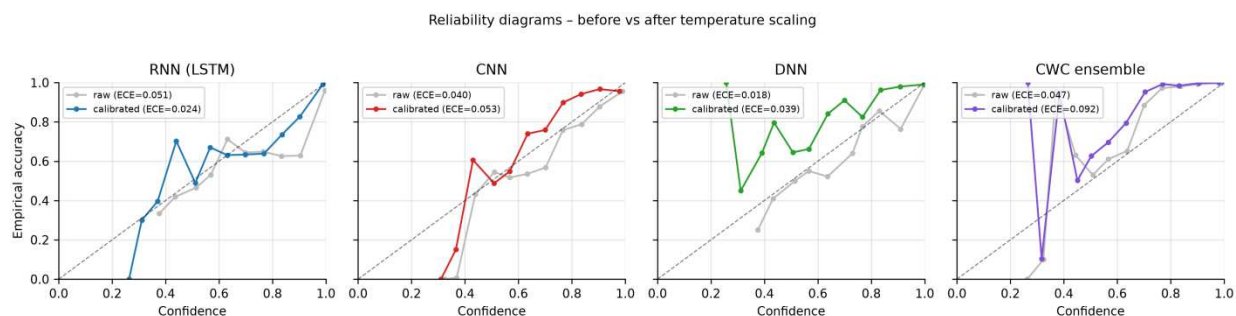


Fig. 14. Reliability diagrams of each detector and of the ensemble, before (grey) and after (colour) temperature scaling.

We report the calibration stage with complete candour, because its headline metric and its operational value diverge. Measured by ECE, temperature scaling only clearly repairs the RNN ($0.051 \rightarrow 0.024$); the CNN and DNN were already near-calibrated and a one-parameter correction not only fails to improve them but slightly degrades them ($0.040 \rightarrow 0.053$ and $0.018 \rightarrow 0.039$, both still small in absolute terms), and the fused ensemble’s ECE worsens ($0.047 \rightarrow 0.092$): averaging individually calibrated posteriors does not guarantee a calibrated average, a known property of opinion pools. What calibration measurably buys the pipeline is **better decisions downstream**: the ablation of Section 6.4 (Table 7) shows that removing it from the CWC costs

0.7 point of operational F1 (0.9666 $\rightarrow$ 0.9595) and raises the FPR by a fifth (0.09% $\rightarrow$ 0.11%), because the fusion weights and, above all, the rejection threshold consume the confidence ordering that scaling homogenises across members.

Impact on the next subsystem. Calibration hands the fusion three confidence-comparable posteriors. The measured effect is small in the clean regime precisely because the detectors are healthy in-distribution; its insurance value, repairing a member whose confidence scale drifts, is what the corruption experiment of Section 6.7 stresses.

6.4. Subsystem 3: Conscious Weighted Consensus fusion

We ablate the fusion rule itself: Table 7 and Fig. 15 compare, on identical inputs, the best single detector, a uniform-weight opinion pool, per-model majority voting, the CWC (precision weights + calibration), the CWC without calibration, and finally the CWC followed by the rejection gate.

Table 7. Ablation of the fusion strategy on identical detector outputs. The last row evaluates the accepted (certified) decisions only, at coverage 87.3%.

Fusion strategy	Oper. F1	Oper. FPR (%)	Acc-15 (%)
DNN alone (best single)	0.952	0.17	95.27
Uniform mean (calibrated)	0.965	0.09	96.48
Majority vote	0.950	0.11	95.08
CWC (precision weights, calibrated)	0.967	0.09	96.60
CWC without calibration	0.959	0.11	95.95
CWC + reject ($\tau=0.70$), certified only	0.996	0.04	99.65

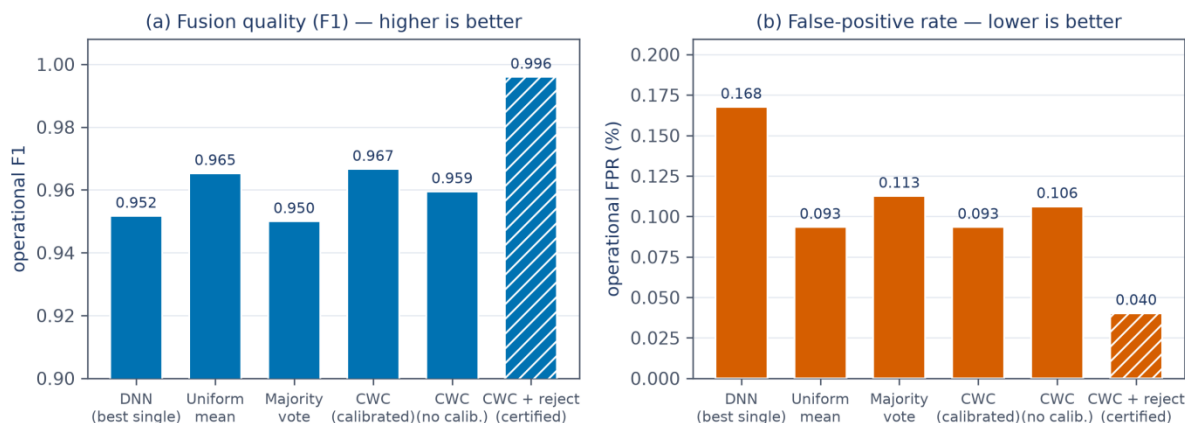


Fig. 15. Fusion-strategy ablation on identical detector outputs (Table 7): (a) operational F1 on a zoomed axis to expose the ordering, (b) operational false-positive rate. The hatched CWC + reject bar is evaluated on the certified population only (coverage 87.3%), not on all decisions like the other strategies.

The ordering is now unambiguous. **Fusion beats the best single model:** the CWC gains 1.3 accuracy points over the DNN (96.60% vs 95.27%), halves its FPR, and the McNemar test of Table 5 makes the difference decisive. In addition, **soft pooling beats hard voting** (majority vote loses the probabilistic information and lands below the best single model). Within soft pooling, the precision weights add a small but consistent margin over the uniform pool (96.60% vs 96.48%): with three healthy detectors the weights are nearly uniform by construction (0.332/0.323/0.345), so their clean-regime contribution is modest; their decisive role is adversarial, measured in

Section 6.7. Calibration inside the fusion is worth 0.65 accuracy points (Table 7, row 5), consistent with Section 6.3.

Impact on the next subsystem. The fusion emits one pooled posterior per flow whose confidence separates correct from incorrect decisions markedly better than any single member (AUC 0.999). That separation is the raw material of the rejection gate, quantified next by the risk–coverage curve.

6.5. Subsystem 4: the rejection gate and the load it routes to deliberation

The reject threshold τ turns the fused posterior into a ternary decision, and in doing so couples two quantities: the reliability of what is *certified* (and will therefore reach the consensus layer and possibly the Executor) and the volume of what is *deferred* (and will therefore load the Data Scientist’s drift monitor and the deliberative layer). Fig. 16(a) gives the risk–coverage curve of the CWC ensemble against the standard softmax-response selective baseline applied to the best single detector [16, 17]; Fig. 16(b) shows the two coupled fractions as τ sweeps; Table 8 tabulates the operating points.

Table 8. Sweep of the rejection threshold τ : coverage, deferred fraction (routed to drift monitoring/deliberation), and quality of the accepted decisions. $\tau = 0.70$ (bold) is the operating point.

τ	Coverage (%)	Deferred (%)	Acc. accepted (%)	F1 accepted	FPR accepted (%)
0.50	96.8	3.2	97.32	0.973	0.093
0.60	92.1	7.9	98.80	0.987	0.050
0.70	87.3	12.7	99.65	0.996	0.040
0.75	82.6	17.4	99.72	0.997	0.040
0.80	73.2	26.8	99.77	0.997	0.034
0.90	60.6	39.4	99.97	0.9995	0.000

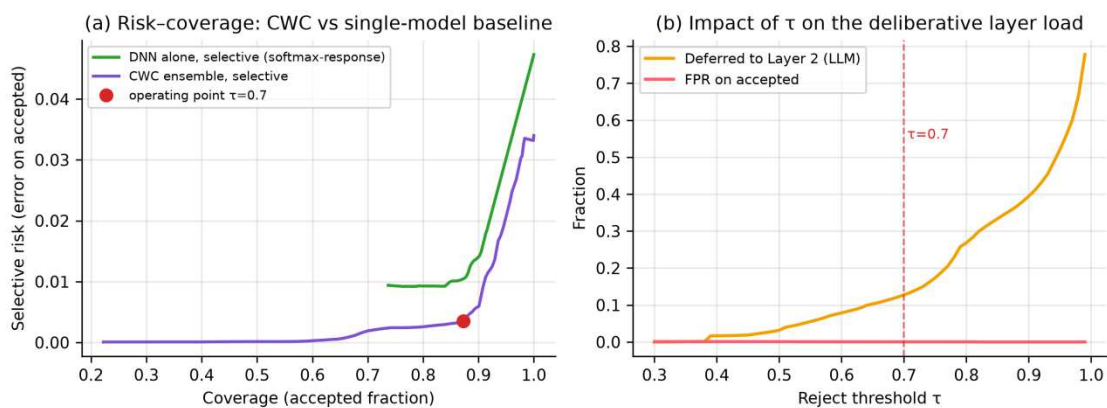


Fig. 16. (a) Risk–coverage curves of the CWC ensemble and of the best single detector under the softmax-response selective baseline, with the $\tau = 0.70$ operating point. (b) Fraction of flows deferred to the deliberative layer and FPR of the accepted decisions, as functions of τ .

The gate delivers the certified stream the architecture is built around. At $\tau = 0.70$ the ensemble certifies 87.3% of flows and, on those, reaches 99.65% 15-class accuracy, 0.996 operational F1 and 0.04% FPR, versus 96.60%, 0.967 and 0.09% ungated (Table 7). The trade-off degrades gracefully across the whole sweep: coverage falls smoothly (96.8% $\rightarrow$ 60.6% between $\tau = 0.5$

and 0.9) while the FPR of the accepted set decreases monotonically to zero at $\tau = 0.90$, an operating point that certifies 60.6% of traffic at 99.97% accuracy with no false positives on the test partition. We select $\tau = 0.70$ as the default because it is the knee where accepted accuracy saturates (99.65 $\rightarrow$ 99.77% costs 14 coverage points), but the whole curve of Fig. 16 is available to the operator as a governed, on-chain-audited parameter. Fig. 16(a) also situates the design in the selective-classification literature: at every coverage level the fused curve dominates the single-best-model softmax-response baseline [16, 17], so the gate’s benefit is not merely that of thresholding any confident model.

Impact on the next subsystems. Downstream, the gate is a load-shaping contract. To Layer 3 it promises that only 0.04%-FPR decisions are worth the cost of multi-party consensus, which is what keeps the multi-second Fabric latency of Section 6.6 off the reflex path. To Layer 2 it delivers the 12.7% deferred stream whose rolling rate is the Data Scientist’s drift signal, and whose bursts trigger deliberation. Raising τ trades certified reliability against deliberative load along the measured curve of Fig. 16(b), an explicit, tunable coupling between the reflex and deliberative subsystems.

6.6. Subsystem 5: the consensus layer, latency versus quorum

We now quantify the price of decentralised trust for exactly the decisions the gate certifies. For each of 40 certified attacks we measure the consensus time (recording the alert and collecting the signed votes up to certification) under both consensus backends and for quorums of one, two and three notaries (Table 9, Fig. 17).

Table 9. Consensus latency per certified alert (40 alerts), by backend and quorum. Overhead is the median Fabric-minus-simulated gap.

Quorum	Simulated (ms)	Fabric median (ms)	Fabric p95 (ms)	Overhead (ms)
1	1.8	4,130	4,151	4,128
2	2.2	6,185	6,395	6,183
3	2.8	8,212	8,229	8,209

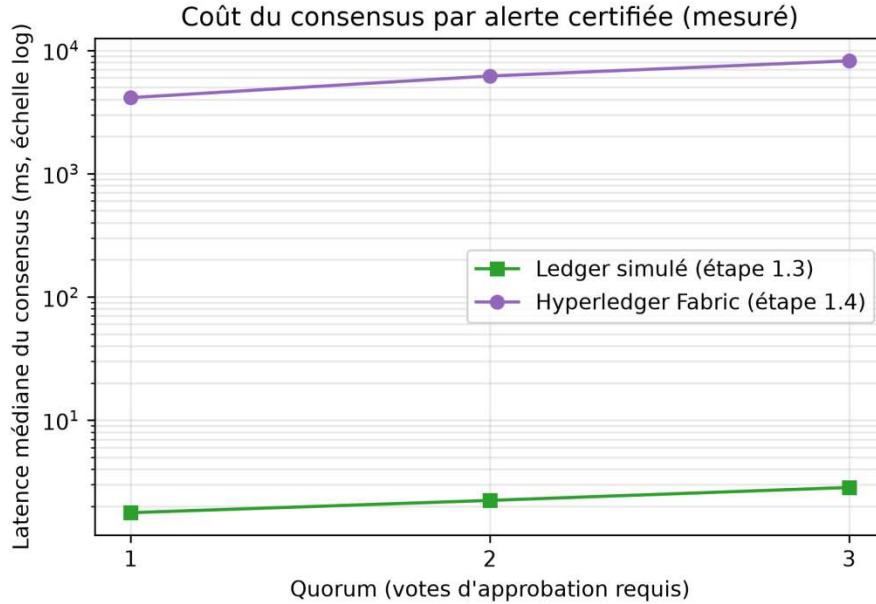


Fig. 17. Median consensus latency per certified alert versus quorum, simulated and Fabric backends (log scale).

(i) The linearity is a state-design property, and we measured why naive parallelism does not remove it: the leader submits the alert and the votes as one transaction each, each waiting for its block (the sequence of Fig. 7), hence the observed $(1 + \text{quorum}) \times 2.06$ s. To test whether concurrent submission removes the linearity, we ran a 360-alert benchmark on the same network (60 alerts per condition, quorums 1 to 3). Sequential vote collection costs $\text{quorum} \times 2.05$ s (median 2.05, 4.11, 6.15 s; adding the alert-recording transaction reproduces Table 9 within 35 ms). Concurrent submission collapses wall-clock time to a single block interval at every quorum (median 2.07 s), but commits *exactly one vote per alert*: at quorums 2 and 3, all 180 contending votes failed with `MVCC_READ_CONFLICT` and no alert reached its quorum, on all 120 alerts without exception. The cause is structural: `ValidateVote` read-modify-writes the alert's state key, and Fabric's optimistic concurrency commits at most one write per key per block, so K votes on one alert require at least K block commits whatever the submission strategy; a retry-on-conflict scheme would merely re-serialise the votes across successive blocks. The lever that would genuinely flatten the cost is a chaincode state redesign giving each vote its own key, with the quorum evaluated over the alert's key range; we scope that redesign and its re-measurement as revision work (Sections 7.7 and 8.1). **(ii) The cost is infrastructure, not cryptography:** the simulated backend performs the identical signature verification, anti-replay and quorum logic in ~ 2 ms, so the multi-second Fabric figure is dominated by the orderer's block-cutting interval (the default 2 s `BatchTimeout`, deliberately left untuned here), a deployment parameter rather than an intrinsic cost of the consensus rule. **(iii) The reflex path is unaffected:** detection continues at the throughput of Section 6.2.2; only certified attacks pay for consensus. With 40 alerts for the per-quorum characterisation and 360 for the concurrency benchmark, these figures characterise this deployment; they are not presented as population estimates.

Impact on the next subsystem. Certification is the event that awakens the remaining agents: it releases the Executor (directly, or through the operator's approval queue when supervision is on)

and opens an Investigator case. The ~ 2 s per vote is thus the end-to-end reaction-time floor of the autonomous remediation path, and the reason high-severity actions can afford a human gate: the human decides in parallel with, not in addition to, an already multi-second consensus.

6.7. Subsystem 6: robustness of the CWC weights to detector corruption (a worst-case poisoning proxy)

We finally test the security claim that motivates the CWC: that precision-weighted fusion neutralises a corrupted member. The experimental model must be stated precisely, because it is **not training-set label-flipping in the sense of the poisoning literature** [21, 22]: rather than corrupting training data and retraining, we directly invert a fraction p of one detector’s *output* decisions (attack into benign, i.e. evasion, and benign into a random attack) on both the validation and test data, so the corruption is reflected in the very statistics the weights are computed from. This **simulated output corruption** is a worst-case proxy: it upper-bounds the damage any training-time attack on that detector could inflict at rate p , while abstracting away the structured error patterns a genuinely retrained-on-poisoned-data model would exhibit. A companion experiment with true training-set poisoning and retraining is scheduled revision work (Section 8.2). We target the strongest ranking detector (RNN) and compare, as p grows, the corrupted model alone, the ensemble with static (clean) weights, the ensemble with adaptive precision weights (BAg-IDS), and, as a robust-aggregation baseline from the literature, the coordinate-wise **median pool**, which for $M = 3$ members coincides with the trimmed mean. (Krum and TRIM-style defences operate on gradient or training aggregation in federated settings and do not apply directly to posterior fusion.)

Table 10. Robustness to simulated output corruption of the RNN. The adaptive ensemble loses at most 1.9 accuracy points at any corruption level and only 0.6 at full corruption, while the corrupted detector’s weight decays $0.332 \rightarrow 0.014$.

p	RNN alone Acc (%)	Static ens. Acc (%)	Median pool Acc (%)	Adaptive ens. Acc (%)	RNN weight
0.0	90.7	96.6	95.1	96.6	0.332
0.3	63.6	94.9	93.9	94.7	0.286
0.5	45.6	93.8	93.1	95.0	0.247
0.9	9.6	91.5	91.5	95.9	0.098
1.0	0.6	90.9	91.1	96.0	0.014

The result is clear-cut, and stronger than under any static alternative. A fully corrupted RNN is destroyed as a standalone detector (0.6% accuracy at $p = 1.0$), yet the adaptive ensemble retains 96.0% of its clean 96.6%: the corrupted member’s weight decays from 0.332 to 0.014 (Fig. 19), so its influence is **automatically suppressed without any explicit attack detection**. The baselines isolate the mechanism (Fig. 18): static weights degrade steadily to 90.9% (a 5.1-point gap to adaptive at $p = 1.0$), and the median pool, which needs no validation statistics, tracks the static ensemble rather than the adaptive one, ending at 91.1%. Across the entire corruption range the adaptive weights dominate both baselines; their worst point is a shallow dip to 94.7% around $p \approx 0.3$, where the corrupted detector retains enough validation precision to keep a meaningful weight. This is where the fusion subsystem of Section 6.4 earns its place: the near-uniform clean-regime weights and the corrupted-regime exclusion are the same mechanism observed in its two regimes.

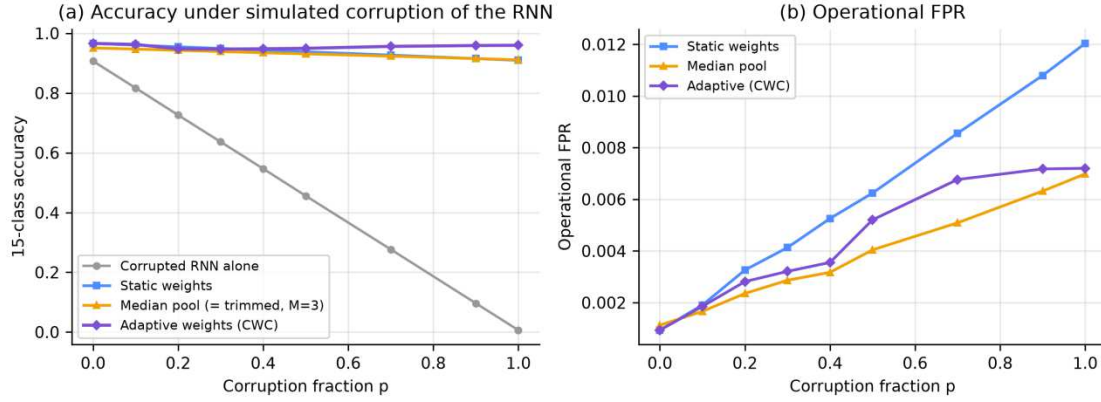


Fig. 18. Accuracy (a) and operational FPR (b) under simulated corruption of the RNN: corrupted model alone, static weights, median pool (= trimmed mean for $M = 3$), and adaptive CWC weights.

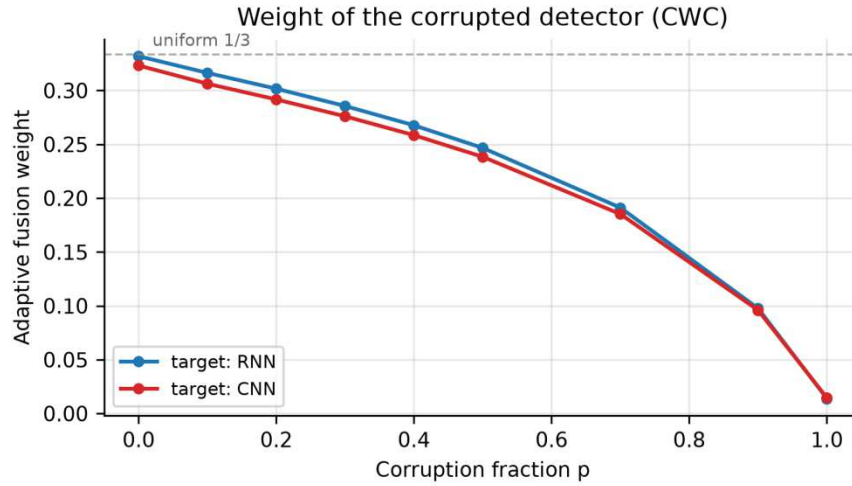


Fig. 19. Adaptive fusion weight of the corrupted detector versus corruption fraction, for the RNN and CNN targets.

6.7.1. The adaptive adversary

We state the adaptive-adversary limitation explicitly rather than leaving it implicit. Eq. 2 is public, so an informed attacker will not corrupt uniformly at random: a targeted backdoor that **preserves validation precision** (for instance one that only misclassifies a minority attack class contributing little to the weighted precision, or that activates on a trigger absent from the validation distribution) would keep the corrupted detector's weight intact and defeat the weighting entirely. The residual dip around $p \approx 0.3$ marks the region such an adversary would target. Within the architecture, the compensating controls are the median baseline (insensitive to validation statistics and only ~ 5 points behind in the worst case), the rejection-rate drift signal consumed by the Data Scientist, and the human audit of certified incidents; a systematic evaluation against precision-preserving poisoning is future work (Section 8.2).

6.8. Functional validation of the deliberative and agentic layer

The deliberative layer is deliberately excluded from the quantitative claims above (Section 4.2.5); what follows is a **functional validation** (evidence that the full five-agent system operates end to end as specified) and is presented as such, not as a statistical evaluation of deliberation quality

(assessing severity assignments against ground truth, inter-model agreement and cost is future work). In a live session with attack injection, the Sentinel processed 224 flows and emitted 8 signed alerts; all 8 were certified by the 2-of-3 notary quorum; the notaries attached 7 advisory second opinions; and the Investigator autonomously opened 7 investigations, correlating ledger history to flag 3 of them as multi-incident campaigns, each with its full perceive-reason-act-reflect trace recorded and displayed in the supervision console. Consistent with the design boundaries, disabling the LLM backends changed none of the certified verdicts: deliberation and advisories fell back to the deterministic rules, and detection, consensus and remediation continued unchanged. A failure mode we fixed and report for reproducibility: a synchronous advisory call on the endorsement path initially exceeded the 5 s endorsement timeout and stalled certification; advisory opinions must be strictly asynchronous to preserve the latency budget of Section 6.6.

6.9. Cross-corpus generalisation: an independently generated 2025 corpus

Every detection figure above is measured on CICIDS2017, a corpus generated in 2017 by the laboratory whose tooling flaws Section 6.1 documents. To test whether the architecture survives outside that corpus, we retrained the three detector families on **GeNIS**, a corpus generated in February 2025 by an independent laboratory (GECAD, Polytechnic of Porto) on an Airbus CyberRange enterprise simulation, with a different flow exporter (HERA) and a different attack execution [80, 81]. We use the 60-second flow interval, 338,820 flows in total, and nine classes after merging the three benign variants: FTP, SMB and SSH brute force, and five denial-of-service families (Hulk, ICMP, PSH-ACK, slowloris, UDP). Crucially, the corpus is evaluated under the **clean protocol** that CICIDS2017 fidelity forbids us (Section 8.3.1): the scaler is fitted on the training split only, no class rebalancing is applied, and the natural distribution is preserved (7.4% benign in the test partition of 67,764 flows), with a stratified 60/20/20 split.

Leakage audit. A first training run reached suspiciously high accuracy, and a per-feature probe located the cause: because GeNIS attacks execute in contiguous time blocks, the flow timestamp *alone* predicted the nine-class label at 99.3% (chance 11%). We therefore removed the five positional columns (start and end timestamps, rank, sequence and offset counters) along with addresses, ports and identifiers, keeping 59 behavioural features. After removal, the strongest single feature is the source inter-packet interval at 0.94, which is genuine volumetric signal, not position. We report this audit because it is the difference between a defensible 99% and a fabricated one, and we release the audited feature list. Fig. 20 shows why the shortcut existed: each class occupies a narrow, disjoint execution window, several of them lasting under two minutes for tens of thousands of flows.

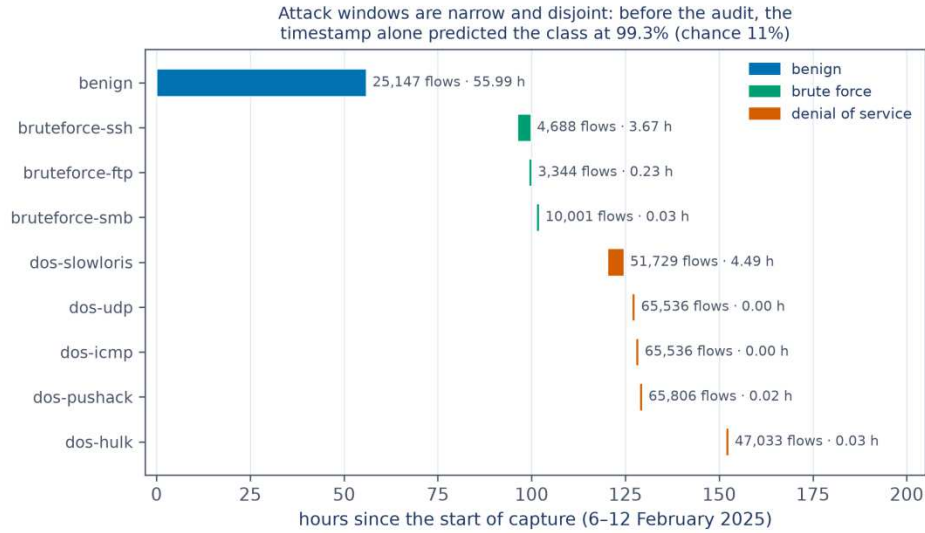


Fig. 20. Execution schedule of the GeNIS corpus. Benign traffic spans the first 56 hours while every attack class occupies a narrow, non-overlapping later window, which is why the flow timestamp alone separated the nine classes at 99.3% before the positional columns were removed.

Table 11 places the two corpora side by side. On GeNIS the three retrained detectors reach 99.77% to 99.92% accuracy, and the fitted temperatures are close to 1 (0.98 to 1.16, against 1.63 to 1.99 on CICIDS2017): the models are natively calibrated because the task is easier, and temperature scaling correctly leaves them almost untouched, which is the behaviour one wants from a calibration stage. The CWC ensemble reaches 99.96% accuracy (macro-F1 0.9994, operational FPR 0.12%), and per-class F1 is at least 0.997 for every class including the three brute-force families. At $\tau = 0.70$ the gate certifies 99.86% of flows at 99.98% accuracy with a single false positive among the 5,030 benign test flows (0.02%), and from $\tau = 0.75$ no false positive is observed at 99.3% coverage or more. We state the two honesty bounds explicitly: the 60-second GeNIS slice is dominated by volumetrically separable floods (87% of the test partition), which is why the absolute numbers are high; and with 5,030 benign flows the false-positive granularity is one flow per 0.02%, so a zero reads as none observed, not none possible.

Table 11. The same architecture, fusion code and certification path on two corpora generated eight years apart by independent laboratories with different extractors. Attack families overlap (FTP/SSH brute force, Hulk, slowloris), making the comparison meaningful; protocols differ as stated, so absolute numbers are not directly comparable across columns.

Property	CICIDS2017 (original protocol)	GeNIS 2025 (clean protocol)
Origin / extractor	CIC, UNB testbed / CICFlowMeter	GECAD, Airbus CyberRange / HERA
Capture year	2017	February 2025
Classes / features	15 / 78	9 / 59
Test flows / benign share	156,089 / 50% (balanced)	67,764 / 7.4% (natural)
Scaler fit	full corpus before split (kept for fidelity to [56])	training split only
Best single detector	DNN 95.27%	RNN 99.92%
CWC ensemble (all decisions)	96.60%, FPR 0.09%	99.96%, FPR 0.12%
Certified at $\tau = 0.70$	99.65%, FPR 0.04%, coverage	99.98%, FPR 0.02%, coverage

Property	CICIDS2017 (original protocol)	GeNIS 2025 (clean protocol)
	87.3%	99.86%
Zero-FPR operating point	$\tau = 0.90$ at 60.6% coverage	$\tau = 0.75$ at 99.8% coverage
Fitted temperatures	1.93 / 1.63 / 1.99	0.98 / 1.16 / 1.16
Deferred to Layer 2 at $\tau = 0.70$	12.7%	0.14%

The cross-corpus reading is the point of this section. The trust machinery (calibration, weighted fusion, rejection gate, consensus path) transfers unchanged: the same code produced both columns. And the rejection gate adapts its cost to corpus difficulty exactly as selective certification should: on the harder 15-class task it buys its certified accuracy by deferring 12.7% of flows to deliberation, while on the easier corpus it defers only 0.14%, certifying nearly everything at zero observed FPR. Because the overlapping attack families are captured by a different laboratory with a different extractor eight years apart, this supports generalisation within those families; it does not claim universality beyond them. The analyst knowledge base of Layer 4b was extended to the GeNIS taxonomy (four classes alias to their CICIDS2017 counterparts, four are new entries), so the explanation layer operates unchanged on the second corpus; its faithfulness evaluation (Section 6.10) remains scoped to CICIDS2017.

6.10. Faithfulness and cost of the explanation layer

We evaluate the explanation layer of Section 4.2.8 on 53 CICIDS2017 test flows stratified across attack classes; the operational population is the 34 flows the ensemble labels as attacks, since only these reach the approval queue. Two properties matter: do the attributions point at the features that actually carry the decision, and is their cost compatible with the approval-queue setting?

Explanations are faithful, in the ablation sense standard for security explanations [78,79]. Ablating the eight top-ranked features drops the predicted-class confidence by 0.70 on average, against 0.43 when eight *random* features are removed, and the top-ranked set is the more damaging in 100% of operational flows (Fig. 21(b)); per feature, the signed SHAP value and the measured ablation effect are positively rank-correlated in 33 of the 34 flows (mean Spearman $\rho = 0.32$). The recommendation is deterministic by construction; the sampled KernelSHAP ranking is stable in aggregate (full-vector rank correlation 0.72 across reruns) though the exact top-eight set varies moderately (Jaccard 0.58), an expected consequence of sampling-based attribution. Each explanation costs a median 3.1 s (95th percentile 5.0 s), affordable precisely because it is incurred only off the critical path, for queued remediations. Fig. 21(a) illustrates a representative case: an FTP-Patator flow certified at confidence 0.992 with unanimous detector agreement and an auto-approve recommendation, whose most influential features are backward inter-arrival-time statistics and packet-length descriptors, the very signature the knowledge base records for FTP brute-forcing (many short, regular flows to port 21 with small packets). We do not claim a human-utility result: faithfulness is a computational property, and whether these explanations measurably improve analyst decisions is left to a controlled operator study (Section 8.4).

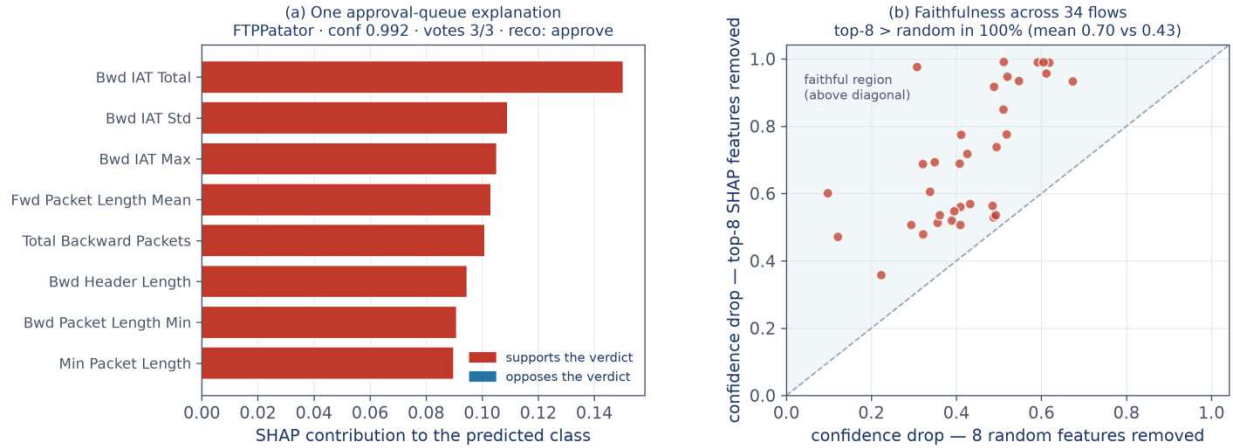


Fig. 21. Explanation layer on the approval-queue population. (a) SHAP attributions for one certified FTP-Patator flow (red supports the verdict, blue opposes it); (b) faithfulness across the 34 attack-predicted flows: removing the top-8 attributed features lowers confidence more than removing 8 random features for every flow (all points above the diagonal).

6.11. Positioning against recently reported blockchain, federated and agentic IDS results

A controlled quantitative comparison with the systems of Section 2 is not possible: no two of them share a dataset, a class granularity, a preprocessing protocol, or a notion of operating point, and none releases an artefact we could rerun under our protocol. What we can do honestly is place our measured figures next to the figures those papers report for themselves, under their own protocols, and be explicit about what is and is not comparable. Table 12 does this for the closest recent blockchain-based, federated, and agentic systems whose sources state concrete numbers.

Table 12. Our measured results on both corpora next to the self-reported results of recent blockchain, federated, and agentic IDS. Each row is measured by its own authors under their own protocol; the rows are indicative context, not a controlled benchmark. The GeNIS figures are high in part because that corpus is dominated by volumetrically separable floods (Section 6.9).

System (ref.)	Category	Dataset (protocol of the source)	Reported performance
Liang et al. [57]	Blockchain + multi-agent (IoT)	NSL-KDD, 2009 (binary)	83.2% acc, F1 0.839 (test set)
Soltani et al. [58]	Multi-agent adaptive DL	CIC-IDS2017/2018	detection rate > 95% (online, CNN)
Begum et al. [60]	Federated + blockchain (IoMT)	Edge-IIoTSet, 2022 / TON_IoT, 2020	97.4% acc, F1 0.97 / 98.2% acc, F1 0.98
Ali et al. [59]	Federated + blockchain (IIoT)	ToN-IoT, 2020; N-BalIoT, 2018	97.3% acc
H. Li et al. [63]	LLM-enhanced agentic IDS	unnamed benchmarks	> 90% acc
BAG-IDS (this work)	Agentic + blockchain consensus + HITL	CICIDS2017, 2017 (15 classes, balanced testbed)	96.60% acc, F1 0.967, FPR 0.09% (all decisions); 99.65% acc, FPR 0.04% at 87.3% coverage (certified)
BAG-IDS (this work)	Agentic + blockchain consensus + HITL	GeNIS, 2025 (9 classes, natural distribution)	99.96% acc, macro-F1 0.999, FPR 0.12% (all decisions); 99.98%

System (ref.)	Category	Dataset (protocol of the source)	Reported performance
			acc, FPR 0.02% at 99.86% coverage (certified)

Four observations survive the protocol differences. First, our all-decisions accuracy (96.60% over 15 classes) sits in the same band as the best federated-blockchain systems report on easier, coarser-grained corpora. The certified operating point (99.65% at 0.04% FPR) is numerically higher than every figure in the table, but we decline to read that as superiority: it is a selective point deferring 12.7% of flows on a class-balanced testbed, set against full-coverage figures from other corpora and protocols. Second, the compared systems report detection quality only; none reports a certified operating point, its false-positive rate, or a measured on-chain cost per decision, which are precisely the quantities an operator needs to act on a verdict. Third, in every blockchain-based row the ledger stores alerts or model updates after the fact, so the reported accuracy is that of an unhardened detector; BAg-IDS is, to our knowledge, the only system in the table whose figures are produced through the consensus path itself. Fourth, the corpora differ in age: the compared systems report on benchmarks generated between 2009 and 2022, whereas the GeNIS row is measured on traffic captured in February 2025. We do not claim this makes our detector better, since the two corpora are not equally hard, but the ageing of benchmarks is precisely the validity concern our second corpus addresses.

6.12. Synthesis: the measured chain of impacts

Reading Sections 6.2–6.8 as one chain: three strong but complementary detectors (90.7–95.3% accuracy) are made confidence-comparable by calibration (worth 0.65 accuracy points inside the fusion); the CWC pools them into one posterior that beats the best single member by 1.3 points (96.60%, McNemar $p \approx 10^{-156}$) and withstands the total corruption of its strongest-ranking member at a cost of 0.6 point (versus 5.7 for static weights and 5.5 for the median pool); the rejection gate converts that pooled confidence into a 99.65%-accurate, 0.04%-FPR certified stream covering 87.3% of traffic (at 37,500 flows/s, with a zero-FPR option at $\tau = 0.90$) while routing the deferred remainder to drift monitoring; the consensus layer certifies that stream at ~ 2 s per vote without touching the reflex path; and the agentic layer enriches certified incidents into campaign-aware, human-auditable reports without ever entering the deterministic decision chain. Each interface in this chain is backed by a measured curve (Figs. 10–19) rather than a design argument.

7. Security Analysis

This section revisits the adversary of Section 3 and argues, threat by threat, which architectural property neutralises it, grounding each claim in the mechanisms of Section 4 and, where available, the measurements of Section 6. The analysis assumes the threat model stated earlier: an adversary may poison a detector, may compromise a single agent (a Sentinel, one notary, or the console), and may attempt to forge, replay or tamper with decision records, but cannot break the underlying cryptographic primitives nor simultaneously control a quorum of independent notaries. Table 13 summarises the mapping; the subsections develop each entry.

Table 13. Threats and the BAg-IDS properties that neutralise them.

Threat	Neutralising property	Evidence
Detector corruption (simulated poisoning proxy)	Precision-weighted fusion auto-excludes the corrupted model	Sec. 6.7, Table 10
Compromised Sentinel (forged alerts)	Consistency re-validation + vote authenticity (partial; see §7.2)	Sec. 7.2
Malicious/forged vote	On-chain Ed25519 signature check against the signer registry	Sec. 7.2, measured
Vote replay / double vote	One-vote-per-notary invariant in the chaincode	Sec. 4.2.6
Single rogue notary	Quorum requirement (k of n) before any remediation	Sec. 7.3
Record tampering	Replicated, hash-linked ledger endorsed by two organisations	Sec. 7.4
Console/admin subversion	Governance actions signed and anchored on-chain (auditable admin)	Sec. 7.5, measured

7.1. Resilience to detector corruption

The most insidious attack against a learning-based IDS is one that corrupts the model itself. Section 6.7 measured BAg-IDS under simulated output corruption, a worst-case proxy for poisoning, that fully inverts the strongest detector: as a standalone classifier the corrupted RNN drops to 0.6% accuracy, yet the ensemble retains 96.0% of its clean 96.6% accuracy. The defence is structural rather than reactive. Because the fusion weight of each detector is proportional to its recent validation precision (Eq. 2), a detector whose reliability collapses sees its weight decay automatically, from 0.332 to 0.014 in the measured worst case, so its vote is discounted **without the system ever having to detect that an attack is underway**. This is a valuable property: it degrades gracefully against an adversary the system cannot see, and it requires no poisoning detector that could itself be evaded.

The analysis is honest about the attack’s residual leverage. The ensemble’s worst degradation is a shallow dip at partial corruption ($p \approx 0.3$, -1.9 accuracy points), where the corrupted detector keeps enough precision to retain a meaningful weight; and the guarantee is a majority-trust one: it holds while a majority of the ensemble remains honest, exactly as the consensus layer assumes a majority of honest notaries. Simultaneous poisoning of two of the three detectors is outside the threat model, and the complementary drift monitor (Section 7.6) is the intended second line of defence against slow, partial corruption.

7.2. Authenticity of votes and alerts

A compromised Sentinel could fabricate alerts, and a compromised host could attempt to inject votes on behalf of a notary. BAg-IDS blocks both at the consensus boundary. Every notary independently re-validates an alert before endorsing it (checking that an attack verdict truly exceeds τ and that the claimed class is the arg-max of the pooled posteriors), so a Sentinel cannot manufacture consensus merely by asserting a verdict. And every vote is an Ed25519 signature over the canonical vote payload, verified **inside the chaincode** against the public key registered for that notary. We verified this empirically: submitting a vote whose signature does not match the registered key is rejected at endorsement time by the peers of both organisations, with the contract returning an explicit *invalid signature* error. Because verification runs in the smart

contract and not in any single agent, subverting one host does not let an attacker forge an endorsement.

Scope of the re-validation: an honest boundary. The notaries' re-validation is a *consistency* check: it recomputes threshold and arg-max over the posteriors carried in the alert, but it does not independently re-classify the underlying flow. A fully compromised Sentinel can therefore fabricate an internally consistent alert (posteriors included) that all notaries will endorse. What the consensus layer guarantees against this adversary is accountability, not prevention: the forged alert is signed by the Sentinel's registered identity, immutably recorded, rate-visible to the supervision layer, and attributable in post-incident review: a compromised Sentinel cannot lie *quietly* or deniably. Prevention would require notaries to re-run detection on the raw flow themselves, which is architecturally straightforward (each notary can hold the model bundle) but multiplies inference cost and consensus payloads; we scope it as future work and, accordingly, temper the corresponding row of Table 13.

7.3. Anti-replay and the quorum barrier

Two further invariants defend the vote-counting itself. First, the chaincode records at most one vote per notary per alert, so a captured vote cannot be replayed to inflate an approval count. Second, and centrally, no alert is certified, and therefore no remediation is authorised, until a quorum of k valid, distinct approvals is reached. A single rogue notary can neither certify an alert alone nor block one that its peers approve; it is reduced to one vote among n . The quorum k is a tunable security/latency dial: Section 6.6 showed the consensus cost growing linearly with k , so an operator trades a higher collusion threshold against a few seconds of additional latency. In the evaluated configuration ($k = 2$ of $n = 3$), certification requires the cooperation of a majority of the notaries, which the threat model assumes an adversary cannot achieve.

7.4. Integrity and non-repudiation of the record

Once certified, a decision must be impossible to alter or disown. The simulated ledger already provides tamper-evidence through hash-linking (our unit tests confirm that altering any past entry breaks the chain) and the Fabric deployment strengthens this to tamper-resistance: the ledger is replicated across the peers of two independent organisations, and each transaction is endorsed and ordered before commitment, so rewriting history would require colluding across organisational boundaries. Every alert, every signed vote and every certification is therefore a non-repudiable record: the signatures bind each decision to the specific notaries that endorsed it, giving the audit trail evidentiary value that a single-host log cannot offer.

7.5. Auditability of human authority

Adding a human supervision layer creates a new target: an attacker who subverts the console could lower the rejection threshold, shrink the quorum, or simply switch off oversight, quietly weakening the whole system. BAg-IDS answers this by refusing to treat administrative power as implicit. Every governance action (toggling supervision, changing a governed parameter, and every manual approval) is serialised, signed by the governance identity, and anchored on the same blockchain through a dedicated contract call whose signature is verified before commitment. We confirmed on-chain that a governance record with an invalid signature is rejected, and that

valid records are readable back from the ledger in order. The contract deliberately separates the **human actor** (the operator account, stored as metadata) from the **cryptographic signer** (the registered governance identity), binding the human chain of responsibility to an unforgeable record. The consequence is a system in which the administrator is auditable: the ledger holds not only what the agents decided, but every human intervention that shaped their behaviour, including the decision to remove human oversight itself.

7.6. Defence in depth

The properties above are layered rather than independent, and their value lies in composition. A poisoning attack that the weighting absorbs also raises the rejection rate, which the Data Scientist agent surfaces as drift and can escalate to recalibration, a second line behind the first. An adversary who compromises a Sentinel still faces independent notary validation; one who compromises a notary still faces the quorum; one who reaches the ledger still faces cross-organisation replication; and one who reaches the console still faces on-chain, signed governance. No single compromise collapses the system, and the human-in-the-loop layer provides an out-of-band control surface that can tighten the posture (raising τ or the quorum, or re-enabling supervision) precisely when the autonomous layers report anomalies. The residual assumptions are explicit and conventional: soundness of the cryptographic primitives, an honest majority of notaries, and a correct initial registration of identities.

7.7. Why a distributed ledger rather than a co-signed append-only log

A permissioned ledger is an expensive way to store a decision (Section 6.6 measures the bill), so the design owes an answer to the obvious objection: could a simpler cryptographic construction not deliver the same guarantees? We take the objection seriously and answer it against the strongest simple alternative, a **hash-chained, append-only log whose entries carry a quorum of detached signatures**, the construction that transparency logs such as Certificate Transparency [82] have proven at Internet scale. It already provides a great deal: entries are attributable and non-repudiable, a quorum of notaries can be required before an entry counts as authorised, and any modification of a past entry breaks the chain and is therefore evident. Three properties, however, it does not provide without effectively rebuilding a ledger.

- **Agreed ordering, not merely detectable modification.** A signed log must be held by someone. If a single custodian holds it, that custodian is exactly the single point of trust the architecture set out to remove (goal G2). If instead each organisation keeps its own copy, nothing decides which copy is canonical when they diverge: two internally consistent, correctly signed histories can coexist, and equivocation becomes possible. Choosing between them is the ordering problem, and solving it is consensus.
- **Validation enforced by the infrastructure, not by convention.** In BAg-IDS the signature check, the signer-authorisation check, the one-vote-per-notary invariant and the quorum rule execute *inside* the chaincode and are endorsed by peers of two independent organisations before commitment (Sections 7.2 and 7.3). In a co-signed log those same rules run in whichever component assembles the log; a compromised assembler can append an entry that violates them, and a reader notices only if it independently re-applies

every rule to every entry. The difference is between an invariant that cannot be violated and one that is merely auditable after the fact.

- **Resistance to omission, not only to alteration.** Hash chaining makes an altered entry evident, but a custodian who simply never appends an inconvenient certification, or who truncates the log and re-signs a shorter valid chain, produces an artefact that verifies perfectly. Cross-organisation replication is what makes silent omission require collusion rather than a single decision.

The measurement of Section 6.6 should therefore be read as the **price of ordering**, not as an overhead one could optimise away. Section 6.6 measures this directly: submitting the votes of one alert concurrently does collapse wall-clock latency to a single block interval, but because every vote updates the same ledger key, all contending transactions but one are invalidated as read-write conflicts and the alert fails to reach quorum. Serialisation is not an implementation blemish; it is what buys the agreed order that the co-signed log lacks.

Where the simpler design is the right one. The argument has a boundary, and we state it rather than overclaim. When every notary belongs to a *single* administrative domain, and the operator is content to trust that domain's storage, a quorum-signed hash-chained log, periodically anchored to an external witness for timestamping, delivers attributability and tamper-evidence at millisecond cost and is the better engineering choice. The ledger earns its price precisely when the notaries sit in *distinct* trust domains, subsidiaries, security providers, regulators, that do not accept one another's storage as authoritative, and when the record is expected to carry evidentiary weight in a later dispute. That is the deployment BAg-IDS targets (Section 8.4), and it is the condition under which we claim the blockchain is warranted.

8. Discussion and Limitations

The results support the central thesis of the paper (that detection, decentralised trust, and human oversight can be co-designed into a single operational system), but they also delimit where the approach applies and where it must be extended. We discuss the main implications and state the limitations candidly, since several bound the interpretation of the numbers in Section 6.

8.1. The trust–latency trade-off in perspective

Section 6.6 measured a consensus cost of several seconds per certified alert, growing linearly with the quorum. This is orders of magnitude slower than the millisecond reflex path, and it would be prohibitive if it fell on every flow. The architecture's asymmetry is therefore not a convenience but a necessity: it is what makes decentralised trust affordable, by charging its cost only to the small fraction of traffic that triggers action. Two points temper the apparent slowness. First, the cost is dominated by the ledger's block-formation interval, an infrastructure parameter, rather than by the cryptographic verification (the identical logic runs in about two milliseconds on the simulated backend). An earlier draft conjectured that the linear growth with the quorum was merely an artefact of our sequential vote submission and that concurrent submission would flatten it; the 360-alert benchmark of Section 6.6 corrects us. Concurrency does collapse wall-clock time to one block interval, but because every vote updates the same state key, Fabric invalidates all but one vote per block and certification fails: the linearity is a property of the chaincode's state

design under optimistic concurrency, not of the consensus rule, and not one that parallelism alone removes. The lever that remains is a per-vote key redesign, which would let all votes of an alert commit in a single block and bound the cost near two block intervals irrespective of the quorum; implementing it, together with BatchTimeout tuning, is scheduled revision work, and neither lever changes the security argument (Section 7.7 reads the measured serialisation as the price of agreed ordering). Second, in a SOC the relevant deadline for a certified, audit-bearing decision is human-scale; a few seconds to obtain a tamper-evident, multi-party-endorsed record is a reasonable price for the accountability it buys, particularly as the alert is already queued for the operator while consensus proceeds.

8.2. Interpreting the corruption result

The corruption experiment shows a strong, structural defence, but its guarantees are majority-trust guarantees and should not be over-read. The ensemble absorbs the corruption of a single detector because the remaining detectors outweigh it; simultaneous corruption of a majority of the ensemble is outside the model and would defeat the mechanism, exactly as a colluding majority would defeat the notary quorum. The residual dip at partial corruption (Section 6.7) also matters operationally: a partially corrupted detector (around $p \approx 0.3$) is more damaging than a fully poisoned one, because it retains enough precision to keep a non-negligible weight. This argues against treating the weighting as a complete defence and in favour of the layered response we advocate: the Data Scientist agent’s drift monitoring is intended precisely to catch the elevated rejection rate that partial poisoning induces and to escalate to recalibration before the weights fully adapt. Quantifying that escalation loop end-to-end is left to future work.

8.3. Limitations

8.3.1. *Preprocessing provenance: a resolved threat to validity, and a cautionary tale*

The detectors originate from our prior published study [56], and all results in this paper are measured under that study’s exact preprocessing, which we recovered from the original training notebook and reproduce seed for seed (Section 4.2.1). We report the episode openly because it is instructive. An earlier draft evaluated the same released checkpoints under a plausible but mismatched reconstruction (a different scaler, no class rebalancing, a different split): every model degraded, one severely (the CNN fell to 31.8% accuracy with a 69% FPR), and the miscalibration induced by the out-of-distribution inputs mechanically inflated the apparent benefit of calibration and rejection. Recovering the original pipeline restored all three detectors (Table 3) and eliminated the artefact. The lesson generalises: released model weights without their exact preprocessing are not a reproducible artefact, and evaluation results on them can be dominated by the reconstruction rather than the models. We therefore release the full preprocessing specification alongside the models. Two properties of the original protocol remain as caveats, stated in Section 6.1: the scaler is fitted before splitting (negligible but non-zero leakage) and the evaluation testbed is class-balanced, so operational rates on raw traffic will differ.

8.3.2. *Single-host test network*

The blockchain is evaluated on a Fabric test network (one orderer, two organisations with one peer each) running on a single workstation. This is sufficient to measure the real cost of the execute–order–validate path and to demonstrate the full stack on commodity hardware, but it

does not capture wide-area network latency, geographic distribution of organisations, or the behaviour of a larger notary set under contention. A multi-site deployment would raise absolute latencies and is the natural next step for a scalability study; the linear quorum scaling we observe should, however, carry over.

8.3.3. Calibration of the fused ensemble

Temperature scaling improves per-model calibration substantially, yet the calibration error of the fused ensemble does not improve monotonically. Post-hoc scaling is applied per model before fusion, and the linear opinion pool does not preserve calibration in general; a joint or post-fusion calibration step would likely tighten the ensemble's reliability. We deliberately kept the simpler per-model scheme because the operational role of calibration here is the well-ordered confidence that the rejection rule and the fusion weights consume, which it provides; refining fused calibration is an orthogonal improvement.

8.3.4. Dependence on the deliberative model

The deliberative layer relies on an external LLM. Our design mitigates the dependence (the language model is off the per-flow path, is rate-limited, and degrades to a deterministic rule-based fallback when unavailable, so autonomy and reproducibility are preserved), but the quality of the severity assessments and remediation suggestions ultimately tracks the quality of the model used. A systematic study of how different LLMs affect downstream decisions, and of prompt-level robustness, is beyond this paper's scope. We note that the LLM never sees raw traffic and only annotates already-certified alerts, which bounds its influence on detection itself.

8.3.5. Scope of the empirical security evaluation

We measured robustness against label-flipping poisoning and verified the cryptographic invariants of the consensus and governance contracts on-chain. We did not empirically evaluate every adversary of Section 3 (for example, adaptive evasion crafted against the feature extractor, or coordinated multi-agent compromise), some of which are out of scope and others of which warrant dedicated study. The security analysis of Section 7 is therefore part argument, part measurement, and we have been explicit about which is which.

8.3.6. The cost of on-decision trust: latency and energy

Placing consensus on the decision path is not free, and we state the bill explicitly. Each certified alert pays roughly two seconds per signed vote (Section 6.6), a cost dominated by ledger block formation and growing linearly with the quorum for a structural reason: Fabric commits at most one write per state key per block, so a quorum of K votes spans at least K blocks. The architecture confines this cost to the rare certified decisions, so the per-flow reflex path is unaffected, but the price is real for any deployment whose certified-alert rate approaches the ledger's transaction budget, and it caps how large a notary set one can afford before the per-vote-key chaincode redesign of Section 6.6 becomes necessary. On energy, a permissioned Fabric network involves no proof-of-work: the overhead is that of a handful of CPU-bound endorsement, ordering and validation processes, orders of magnitude below public-chain mining and small next to the GPU cost of training the detectors. We did not meter consumption, however, so we make no quantitative energy claim; a joules-per-certified-alert measurement across quorum sizes is a straightforward and worthwhile extension.

8.3.7. Evaluation corpora: two testbeds, not yet live traffic

The detection figures in this paper are measured on two corpora: CICIDS2017 under its original protocol, and GeNIS 2025, an independently generated corpus evaluated under a leakage-audited clean protocol with the natural class distribution (Section 6.9). The second corpus delivers the cross-corpus validation an earlier draft could only promise: the same fusion code, gate and certification path transfer unchanged, and the overlapping attack families are captured eight years apart by a different laboratory with a different extractor. Three bounds remain. Both corpora are testbed-generated: CICIDS2017 carries the documented flaws of Section 6.1, and the 60-second GeNIS slice is dominated by volumetrically separable floods with only 7.4% benign traffic, so its false-positive granularity is coarse. The detectors and operating points remain corpus-specific, and the comparison of Section 6.11 shows how protocol differences dominate cross-paper deltas. Validation on live traffic behind a SOC therefore remains the single most valuable next measurement.

8.4. Generalisation and future work

Section 6.9 took the first step beyond the CICIDS family, retraining the detectors on an independently generated 2025 corpus with the trust machinery unchanged; the continuation is towards more specialised corpora, such as cloud-DDoS and encrypted-traffic (QUIC) datasets, and ultimately live traffic, at the sole cost of retraining on the new feature schemas. A different use of several corpora is also open: rather than instantiating the reflex layer on one corpus at a time, one could hold several domain-specialised detectors simultaneously and make the fusion weights depend on the input, trusting each member in proportion to how well the flow matches its training distribution. This generalises the reliability weighting of Section 4.2.4 from global to per-flow weights, and the rejection gate already covers the case where no member is competent; the open question is whether such routing captures genuine threat diversity or merely the idiosyncrasies of the feature exporters, which a leave-one-corpus-out protocol would settle.

On the systems side, a geographically distributed, multi-organisation Fabric deployment would turn the latency study into a genuine scalability study, and alternative finality protocols could shrink the consensus cost. On the learning side, closing the drift-to-recalibration loop through the Data Scientist agent, and studying online Dirichlet weight adaptation under sustained partial poisoning, would strengthen the poisoning defence into an adaptive one. Finally, Section 6.10 shows the explanation layer to be faithful and affordable, but not yet useful in the human sense: whether these explanations measurably improve the speed or accuracy of analyst approvals requires a controlled operator study pairing the attribution-and-recommendation panel against a no-explanation baseline. That study, together with evidence of how much oversight operators actually exercise and how the on-chain governance record supports post-incident review, is the natural next step towards usable, accountable autonomy.

On the deployment side, the natural insertion point for BAg-IDS in an operational environment is as a certification tier behind an existing SOC pipeline: the reflex ensemble subscribes to the same flow telemetry as the incumbent IDS, certified verdicts reach the SIEM as co-signed, replay-proof alerts rather than raw model outputs, and the supervision console becomes the analyst's approval surface for high-impact remediations. Because Fabric is permissioned, the notary set can be mapped onto trust domains an organisation already operates (subsidiaries, security providers,

regulators) without any public chain, and the measured costs of Sections 6.2.2 and 6.6 give an operator the two numbers needed to size such a deployment: reflex throughput per core and certified alerts per minute per quorum.

9. Conclusion

We presented BAg-IDS, a blockchain-enabled agentic architecture that treats intrusion detection, decentralised trust, and human oversight as a single design rather than three separate concerns. Its four layers are joined by a deliberate asymmetry of cost: a calibrated neural ensemble detects every flow in milliseconds and fuses the detectors through the reliability-weighted Conscious Weighted Consensus fusion; an LLM deliberates only on salient events; a permissioned-blockchain smart contract certifies each remediation-triggering verdict through multi-party signed voting with quorum, anti-replay and tamper-evident audit; and an optional, governed human-in-the-loop layer places a supervising operator above the autonomous machinery, grounding each approval in a faithful per-decision explanation and recording even its own administrative actions on-chain.

The system is implemented in full and evaluated on commodity hardware, so every reported figure is measured rather than projected. On CICIDS2017 the calibrated ensemble reaches 96.6% accuracy over all decisions and 99.7% on the certified subset at the selected operating threshold; retrained on the independently generated GeNIS 2025 corpus under a leakage-audited, natural-distribution protocol, the same stack reaches 99.98% while the gate defers only 0.14% of flows, evidence that the selective certification adapts its cost to corpus difficulty. The blockchain's cost is quantified and shown to grow linearly with the quorum while being dominated by ledger block formation rather than cryptography, which keeps it a tunable infrastructure parameter rather than an intrinsic barrier. Most importantly, the reliability-weighted fusion is shown to be a structural defence against detector corruption: when a simulated worst-case output corruption reduces the strongest detector to 0.6% accuracy, the ensemble loses only 0.6 accuracy points, and at most 1.9 anywhere on the corruption sweep, as the corrupted detector's weight is automatically suppressed, and the consensus and governance contracts reject forged signatures on-chain.

Taken together, these results support a practical claim: an intrusion detection system can be simultaneously accurate, robust to a corrupted component, decentralised in whom it trusts, tamper-evident in what it records, and answerable to a human, without sacrificing real-time detection. The limitations we stated, chiefly the class-balanced CICIDS2017 testbed of the original protocol, the volumetric bias of the GeNIS slice, and the single-host ledger deployment, bound the absolute numbers but not the architectural conclusions, and they map directly onto the future work we outlined: true training-set poisoning with retraining, a geographically distributed multi-organisation deployment, a closed-loop drift-to-recalibration defence, and validation on live traffic and on modern cloud and encrypted-traffic threat corpora.

Beyond these direct extensions, the architecture generalises past intrusion detection. The pattern it implements, autonomous agents whose action-triggering decisions are co-signed by a quorum and whose governance is itself on-chain, applies wherever an automated security decision must be trusted by parties who do not fully trust each other: certified malware triage verdicts, phishing takedown decisions, cross-organisation cyber-threat-intelligence sharing in which each indicator

carries a quorum signature, or SOAR playbooks whose every automated step is notarised. Two deployment horizons seem particularly promising. In a Zero Trust environment, where no verdict is accepted on the authority of a single component, the certified alerts of BAg-IDS are exactly the continuously verified, cryptographically attributable signals a policy engine expects, making the consensus layer a natural trust anchor for micro-segmentation and access decisions. And a validation campaign in a production SOC, measuring how certified, deferred and LLM-annotated alerts change analyst workload and time-to-respond, would test the human side of the design at the scale it was built for. We believe the co-design of learning, decentralised trust, and governed human oversight is a productive direction for trustworthy security automation, and we offer BAg-IDS as a concrete, measured step along it.

References

- [1] C. Yin, Y. Zhu, J. Fei, X. He, A deep learning approach for intrusion detection using recurrent neural networks, *IEEE Access* 5 (2017) 21954–21961.
- [2] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, S. Venkatraman, Deep learning approach for intelligent intrusion detection system, *IEEE Access* 7 (2019) 41525–41550.
- [3] N. Shone, T. N. Ngoc, V. D. Phai, Q. Shi, A deep learning approach to network intrusion detection, *IEEE Transactions on Emerging Topics in Computational Intelligence* 2 (1) (2018) 41–50.
- [4] A. Javaid, Q. Niyaz, W. Sun, M. Alam, A deep learning approach for network intrusion detection system, in: *EAI International Conference on Bio-inspired Information and Communications Technologies (BICT)*, 2016.
- [5] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, F. Ahmad, Network intrusion detection system: A systematic study of machine learning and deep learning approaches, *Transactions on Emerging Telecommunications Technologies* 32 (1) (2021) e4150.
- [6] H. Liu, B. Lang, Machine learning and deep learning methods for intrusion detection systems: A survey, *Applied Sciences* 9 (20) (2019) 4396.
- [7] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, Toward generating a new intrusion detection dataset and intrusion traffic characterization, in: *4th International Conference on Information Systems Security and Privacy (ICISSP)*, 2018, pp. 108–116.
- [8] J. L. Leevy, T. M. Khoshgoftaar, A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data, *Journal of Big Data* 7 (2020) 104.
- [9] Z. Long, H. Yan, G. Shen, X. Zhang, H. He, L. Cheng, A Transformer-based network intrusion detection approach for cloud security, *Journal of Cloud Computing* 13 (2024) 5.
- [10] F. Ullah, S. Ullah, G. Srivastava, J. C.-W. Lin, IDS-INT: Intrusion detection system using transformer-based transfer learning for imbalanced network traffic, *Digital Communications and Networks* 10 (1) (2024) 190–204.
- [11] H. Kheddar, Transformers and large language models for efficient intrusion detection systems: A comprehensive survey, *arXiv:2408.07583* (2024).
- [12] G. Folino, P. Sabatino, Ensemble based collaborative and distributed intrusion detection systems: A survey, *Journal of Network and Computer Applications* 66 (2016) 1–16.
- [13] B. A. Tama, M. Comuzzi, K.-H. Rhee, TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system, *IEEE Access* 7 (2019) 94497–94507.
- [14] C. Guo, G. Pleiss, Y. Sun, K. Q. Weinberger, On calibration of modern neural networks, in: *International Conference on Machine Learning (ICML)*, 2017, pp. 1321–1330.
- [15] M. Minderer, J. Djolonga, R. Romijnders, F. Hubis, X. Zhai, N. Houlsby, D. Tran, M. Lucic, Revisiting the calibration of modern neural networks, in: *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [16] Y. Geifman, R. El-Yaniv, Selective classification for deep neural networks, in: *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [17] K. Hendrickx, L. Perini, D. Van der Plas, W. Meert, J. Davis, Machine learning with a reject option: A survey, *Machine Learning* 113 (2024) 3073–3110.
- [18] J. Talpini, F. Sartori, M. Savi, Enhancing trustworthiness in ML-based network intrusion detection with uncertainty quantification, *Journal of Reliable Intelligent Environments* 10 (2024) 501–520.
- [19] B. Biggio, F. Roli, Wild patterns: Ten years after the rise of adversarial machine learning, *Pattern Recognition* 84

(2018) 317–331.

- [20] N. Papernot, P. McDaniel, A. Sinha, M. P. Wellman, SoK: Security and privacy in machine learning, in: IEEE European Symposium on Security and Privacy (EuroS&P), 2018, pp. 399–414.
- [21] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, B. Li, Manipulating machine learning: Poisoning attacks and countermeasures for regression learning, in: IEEE Symposium on Security and Privacy (S&P), 2018, pp. 19–35.
- [22] T. Gu, K. Liu, B. Dolan-Gavitt, S. Garg, BadNets: Evaluating backdooring attacks on deep neural networks, IEEE Access 7 (2019) 47230–47244.
- [23] O. Ibitoye, R. Abou-Khamis, A. Matrawy, M. O. Shafiq, The threat of adversarial attacks on machine learning in network security: A survey, arXiv:1911.02621 (2019).
- [24] G. Apruzzese, M. Andreolini, L. Ferretti, M. Marchetti, M. Colajanni, Modeling realistic adversarial attacks against network intrusion detection systems, ACM Digital Threats: Research and Practice 3 (3) (2022) 1–19.
- [25] J. Vitorino, I. Praça, E. Maia, SoK: Realistic adversarial attacks and defenses for intelligent network intrusion detection, Computers & Security 134 (2023) 103433.
- [26] K. He, D. D. Kim, M. R. Asghar, Adversarial machine learning for network intrusion detection systems: A comprehensive survey, IEEE Communications Surveys & Tutorials 25 (1) (2023) 538–566.
- [27] T. D. Nguyen, P. Rieger, M. Miettinen, A.-R. Sadeghi, Poisoning attacks on federated learning-based IoT intrusion detection system, in: Workshop on Decentralized IoT Systems and Security (DISS, co-located with NDSS), 2020.
- [28] E. H. Spafford, D. Zamboni, Intrusion detection using autonomous agents, Computer Networks 34 (4) (2000) 547–570.
- [29] Á. Herrero, E. Corchado, Multiagent systems for network intrusion detection: A review, in: Computational Intelligence in Security for Information Systems (CISIS), Springer, 2009, pp. 143–154.
- [30] W. Li, W. Meng, L. F. Kwok, Surveying trust-based collaborative intrusion detection: State-of-the-art, challenges and future directions, IEEE Communications Surveys & Tutorials 24 (1) (2022) 280–305.
- [31] N. Alexopoulos, E. Vasilomanolakis, N. R. Ivanko, M. Mühlhäuser, Towards blockchain-based collaborative intrusion detection systems, in: International Conference on Critical Information Infrastructures Security (CRITIS), Springer, 2017, pp. 107–118.
- [32] T. Golomb, Y. Mirsky, Y. Elovici, CloTA: Collaborative IoT anomaly detection via blockchain, in: Workshop on Decentralized IoT Security and Standards (DISS), 2018.
- [33] W. Meng, E. W. Tischhauser, Q. Wang, Y. Wang, J. Han, When intrusion detection meets blockchain technology: A review, IEEE Access 6 (2018) 10179–10188.
- [34] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, et al., Hyperledger Fabric: A distributed operating system for permissioned blockchains, in: EuroSys, 2018, pp. 30:1–30:15.
- [35] D. Preuveneers, V. Rimmer, I. Tsingenopoulos, J. Spooren, W. Joosen, E. Ilie-Zudor, Chained anomaly detection models for federated learning: An intrusion detection case study, Applied Sciences 8 (12) (2018) 2663.
- [36] S. Rathore, B. W. Kwon, J. H. Park, BlockSecIoTNet: Blockchain-based decentralized security architecture for IoT network, Journal of Network and Computer Applications 143 (2019) 167–177.
- [37] O. Alkadi, N. Moustafa, B. Turnbull, K.-K. R. Choo, A deep blockchain framework-enabled collaborative intrusion detection for protecting IoT and cloud networks, IEEE Internet of Things Journal 8 (12) (2021) 9463–9472.
- [38] W. Li, S. Tug, W. Meng, Y. Wang, Designing collaborative blockchain signature-based intrusion detection in IoT environments, Future Generation Computer Systems 96 (2019) 481–489.
- [39] W. Issa, N. Moustafa, B. Turnbull, N. Sohrabi, Z. Tari, Blockchain-based federated learning for securing Internet of Things: A comprehensive survey, ACM Computing Surveys 55 (9) (2023) 1–43.
- [40] S. Ali, Q. Li, A. Yousafzai, Blockchain and federated learning-based intrusion detection approaches for edge-enabled industrial IoT networks: A survey, Ad Hoc Networks 152 (2024) 103320.
- [41] M. A. Ferrag, F. Alwahedi, A. Battah, B. Cherif, A. Mechri, N. Tihanyi, Generative AI and large language models for cyber security: All insights you need, arXiv:2405.12750 (2024).
- [42] J. Zhang, H. Bu, H. Wen, Y. Liu, H. Fei, et al., When LLMs meet cybersecurity: A systematic literature review, Cybersecurity 8 (2025).
- [43] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, Y. Zhang, A survey on large language model (LLM) security and privacy: The good, the bad, and the ugly, High-Confidence Computing 4 (2) (2024) 100211.
- [44] F. N. Motlagh, M. Hajizadeh, M. Majd, P. Najafi, F. Cheng, C. Meinel, Large language models in cybersecurity: State-of-the-art, arXiv:2402.00891 (2024).
- [45] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, Y. Cao, ReAct: Synergizing reasoning and acting in language models, in: International Conference on Learning Representations (ICLR), 2023.

- [46] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, et al., A survey on large language model based autonomous agents, *Frontiers of Computer Science* 18 (6) (2024) 186345.
- [47] T. Ali, P. Kostakos, HuntGPT: Integrating machine learning-based anomaly detection and explainable AI with large language models (LLMs), *arXiv:2309.16021* (2023).
- [48] M. Vielberth, F. Böhm, I. Fichtinger, G. Pernul, Security Operations Center: A systematic study and open challenges, *IEEE Access* 8 (2020) 227756–227779.
- [49] R. C. Arkin, *Behavior-Based Robotics*, MIT Press, 1998.
- [50] G. Engelen, V. Rimmer, W. Joosen, Troubleshooting an intrusion detection dataset: the CICIDS2017 case study, in: *IEEE Security and Privacy Workshops (SPW)*, 2021, pp. 7–12.
- [51] L. Liu, G. Engelen, T. Lynar, D. Essam, W. Joosen, Error prevalence in NIDS datasets: A case study on CIC-IDS-2017 and CSE-CIC-IDS-2018, in: *IEEE Conference on Communications and Network Security (CNS)*, 2022, pp. 254–262.
- [52] M. Lanvin, P.-F. Gimenez, Y. Han, F. Majorczyk, L. Mé, E. Totel, Errors in the CICIDS2017 dataset and the significant differences in detection performances it makes, in: *International Conference on Risks and Security of Internet and Systems (CRISIS)*, Springer, 2022, pp. 18–33.
- [53] G. Deng, Y. Liu, V. Mayoral-Vilches, P. Liu, Y. Li, Y. Xu, T. Zhang, Y. Liu, M. Pinzger, S. Rass, PentestGPT: An LLM-empowered automatic penetration testing tool, in: *USENIX Security Symposium*, 2024.
- [54] A. Happe, J. Cito, Getting pwn'd by AI: Penetration testing with large language models, in: *ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, 2023, pp. 2082–2086.
- [55] S. Srinivas, B. Kirk, J. Zendejas, M. Espino, M. Boskovich, A. Bari, K. Dajani, N. Alzahrani, AI-augmented SOC: A survey of LLMs and agents for security automation, *Journal of Cybersecurity and Privacy* 5 (4) (2025) 95.
- [56] M. A. E. Bahri, F. Jemili, O. Korbbaa, Comparative study between ML approaches in intrusion detection context, in: *2023 IEEE Afro-Mediterranean Conference on Artificial Intelligence (AMCAI)*, Hammamet, Tunisia, 2023, pp. 1–6, doi:10.1109/AMCAI59331.2023.10431511.
- [57] C. Liang, B. Shanmugam, S. Azam, A. Karim, A. Islam, M. Zamani, N. B. Idris, Intrusion detection system for the Internet of Things based on blockchain and multi-agent systems, *Electronics* 9 (7) (2020) 1120.
- [58] M. Soltani, K. Khajavi, M. J. Siavoshani, A. H. Jahangir, A multi-agent adaptive deep learning framework for online intrusion detection, *Cybersecurity* 6 (2023) 44.
- [59] A. Ali, et al., Federated learning-enhanced blockchain framework for privacy-preserving intrusion detection in industrial IoT, *arXiv:2505.15376* (2025).
- [60] K. Begum, et al., BFLIDS: Blockchain-driven federated learning for intrusion detection in IoMT networks, *Sensors* 24 (14) (2024) 4591.
- [61] H. Nandanwar, R. Katarya, A hybrid blockchain-based framework for securing intrusion detection systems in the Internet of Things, *Cluster Computing* 28 (2025).
- [62] A. Kumar, B. Sharma, A. Noonina, Secure blockchain-based intrusion detection for IoT networks, *Discover Computing* 28 (2025).
- [63] H. Li, et al., Multi-agent collaborative intrusion detection for low-altitude economy IoT: an LLM-enhanced agentic AI framework, *arXiv:2601.17817* (2026).
- [64] S. J. Lazer, K. Aryal, M. Gupta, E. Bertino, A survey of agentic AI and cybersecurity: challenges, opportunities and use-case prototypes, *arXiv:2601.05293* (2026).
- [65] D. Gosmar, D. A. Dahl, Sentinel agents for secure and trustworthy agentic AI in multi-agent systems, *arXiv:2509.14956* (2025).
- [66] P. Zambare, V. N. Thanikella, N. P. Kottur, et al., NetMoniAI: an agentic AI framework for network security and monitoring, *arXiv:2508.10052* (2025).
- [67] A. Jin, Y. Ye, B. Lee, Y. Qiao, DeCoAgent: large language model empowered decentralized autonomous collaboration agents based on smart contracts, *IEEE Access* 12 (2024), doi:10.1109/ACCESS.2024.3481641.
- [68] M. Qi, T. Zhu, L. Zhang, N. Li, W. Zhou, Towards transparent and incentive-compatible collaboration in decentralized LLM multi-agent systems: a blockchain-driven approach, *arXiv:2509.16736* (2025).
- [69] X. Li, et al., BlockAgents: towards Byzantine-robust LLM-based multi-agent coordination via blockchain, in: *Proc. ACM TURC Conference*, 2024, doi:10.1145/3674399.3674445.
- [70] A. Capponi, A. Gliozzo, C. Han, J. Lee, DAO-AI: evaluating collective decision-making through agentic AI in decentralized governance, *arXiv:2510.21117* (2025).
- [71] K. Lad, M. A. A. Dewan, F. Lin, Trust management for multi-agent systems using smart contracts, Athabasca University, 2020.

- [72] A. Kott (Ed.), *Autonomous Intelligent Cyber Defense Agent (AICA): A Comprehensive Guide*, Springer, 2023.
- [73] T. T. Nguyen, V. J. Reddi, Deep reinforcement learning for cyber security, *IEEE Transactions on Neural Networks and Learning Systems* 34 (8) (2023) 3779–3795.
- [74] H. Aydin, M. A. Aydin, A multi-agent-based DDoS detection and defense system design with federated learning and blockchain in a public cloud network environment, *Cluster Computing* 28 (2025).
- [75] C. S. de Witt, Open challenges in multi-agent security: towards secure systems of interacting AI agents, *arXiv:2505.02077* (2025).
- [76] V. Vinay, The evolution of agentic AI in cybersecurity: from single LLM reasoners to multi-agent systems and autonomous pipelines, *arXiv:2512.06659* (2025).
- [77] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 30, 2017, pp. 4765–4774.
- [78] J. DeYoung, S. Jain, N. F. Rajani, E. Lehman, C. Xiong, R. Socher, B. C. Wallace, ERASER: A benchmark to evaluate rationalized NLP models, in: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020, pp. 4443–4458.
- [79] A. Warnecke, D. Arp, C. Wressnegger, K. Rieck, Evaluating explanation methods for deep learning in security, in: *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2020, pp. 158–174.
- [80] M. Silva, D. Pinto, J. Vitorino, J. Gonçalves, E. Maia, I. Praça, GeNIS: A modular dataset for network intrusion detection and classification, *Data in Brief* 60 (2025) 111487, doi:10.1016/j.dib.2025.111487 (dataset: doi:10.5281/zenodo.14919237).
- [81] M. Silva, D. Pinto, J. Vitorino, E. Maia, I. Praça, I. Amorim, M. J. Viamonte, Binary and multiclass cyberattack classification on the GeNIS dataset, in: *Foundations and Practice of Security (FPS 2025)*, 2025, arXiv:2511.08660.
- [82] B. Laurie, Certificate Transparency: public, verifiable, append-only logs, *Communications of the ACM* 57 (10) (2014) 40–46.
- [83] A. Aljabri, F. Jemili, O. Korbaa, Convolutional neural network for intrusion detection using blockchain technology, *International Journal of Computers and Applications* 46 (2) (2024) 67–77, doi:10.1080/1206212X.2023.2284443.
- [84] A. Aljabri, F. Jemili, O. Korbaa, Intrusion detection in cyber-physical system using RSA blockchain technology, *Multimedia Tools and Applications* 83 (16) (2024) 48119–48140, doi:10.1007/s11042-023-17576-z.
- [85] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, P. Faruki, Network intrusion detection for IoT security based on learning techniques, *IEEE Communications Surveys & Tutorials* 21 (3) (2019) 2671–2701.
- [86] O. Zanouni, A. Ben Chehida Douss, M. Mosbah, Cooperative trust based detection mechanism for fake objects in collective perception messages, in: *Advanced Information Networking and Applications (AINA 2025)*, *Lecture Notes on Data Engineering and Communications Technologies*, Vol. 249, Springer, 2025, pp. 189–200, doi:10.1007/978-3-031-87775-9_16.
- [87] M. Mazouzi, W. Jerbi, O. Cheikhrouhou, M. Mosbah, Block-FRL: A reputation-aware blockchain framework for secure federated reinforcement learning in Internet of Vehicles, in: *2026 International Wireless Communications and Mobile Computing (IWCMC)*, 2026, pp. 262–268.

Statements and Declarations

Funding

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Competing interests

The authors have no relevant financial or non-financial interests to disclose.

Ethics approval

Not applicable. This study involves no human participants, no animal subjects and no personal data. Both evaluation corpora consist of network traffic generated in controlled laboratory testbeds and released publicly by their originating institutions.

Author contributions

Mohamed Ala Eddine Bahri: Conceptualization, Methodology, Software, Investigation, Data curation, Visualization, Writing (original draft). **Farah Jemili:** Conceptualization, Methodology, Supervision, Writing (review and editing). **Mohamed Mosbah:** Supervision, Validation, Project administration, Writing (review and editing). All authors read and approved the final manuscript.

Data availability

Both evaluation corpora are publicly available. CICIDS2017 is distributed by the Canadian Institute for Cybersecurity, University of New Brunswick (<https://www.unb.ca/cic/datasets/ids-2017.html>), and GeNIS is archived on Zenodo (<https://doi.org/10.5281/zenodo.14919237>). The three CICIDS2017 detectors were trained and published in our prior study [56]; their preprocessing protocol, and the leakage-audited protocol used for GeNIS, are specified in Sections 4.2.1 and 6.9.

Code availability

The complete source code, including the agents, the chaincode, the supervision console, the retrained GeNIS detectors and every measurement script behind the reported tables and figures, is held in a version-controlled repository. It will be made publicly available under an archived release with a permanent identifier upon acceptance, and is available from the corresponding author on reasonable request before then.

Use of generative AI in the writing process

During the preparation of this work the authors used Claude (Anthropic) to support literature search and to check the internal consistency of the manuscript. The tool was not used to generate any scientific content: every model, measurement, figure and result reported here is produced by the implemented system. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article. AI-assisted copy editing is disclosed in the cover letter, as provided for by Springer Nature policy.