High-Confidence Computing 6 (2026) 100354

Contents lists available at ScienceDirect

High-Confidence Computing

journal homepage: www.sciencedirect.com/journal/high-confidence-computing

Research article

PureChain-enhanced federated learning for dynamic fault tolerance and attack detection in distributed systems

Love Allen Chijioke Ahakonye [a], Cosmas Ifeanyi Nwakanma [b], Jae Min Lee [c], Dong Seong Kim [c,d,*]

[a] ICT Convergence Research Centre, Kumoh National Institute of Technology, Gumi 39177, Republic of Korea
[b] Lane Department of Computer Science and Electrical Engineering, West Virginia University, WV 26506, Morgantown, United States
[c] Department of IT Convergence Engineering, Kumoh National Institute of Technology, Gumi 39177, Republic of Korea
[d] NSLab Co. Ltd., Kumoh National Institute of Technology, Gumi 39177, Republic of Korea

ARTICLE INFO

Article history:
Received 16 May 2025
Revised 4 September 2025
Accepted 10 September 2025
Available online 1 October 2025

Keywords:
Attack detection
Decentralized systems
Federated learning
IoT
PoA^2
PureChain

ABSTRACT

The growing complexity of distributed industrial IoT systems heightens cybersecurity risks, exposing the limitations of centralized ML-based intrusion detection. Federated Learning (FL) enables decentralized, privacy-preserving model training but remains susceptible to adversarial threats and system-level failures. This study introduces PureChain, a decentralized ledger using a proof-of-authority and association (PoA^2) consensus mechanism to enhance FL-based IDS security. The study offers insight into the mathematical model of the PureChain-enhanced FL, which integrates blockchain-inspired consensus protocols for collaborative intrusion detection across organizations, ensuring data privacy while providing tamper-proof logs and automated responses through smart contracts. It incorporates dynamic fault tolerance, poisoning resistance, and privacy preservation with FL, enhancing security and performance in decentralized systems. Experimentation with varying client subsets demonstrates its adaptability with a TPS range of 312.5–1178.3 and a low latency range of 0.0008484–0.0032. The framework ensures comprehensive security, reliability, and privacy, providing a scalable solution for decentralized, secure systems.

1. Introduction

The rapid proliferation of distributed systems has heightened the need to ensure their reliability, security, and resilience [1]. The industrial Internet of Things (IoT) is a crucial technology for intelligent industries, integrating various processes. As the level of connectivity and complexity in IoT continues to grow, cybersecurity risks have emerged as a significant concern [1]. Artificial intelligence (AI) has emerged as a crucial tool to mitigate cyberattacks [2]. Among these are machine learning (ML) and deep learning (DL)-based intrusion detection models, which have been proposed to protect highly connected networks from cyber-attacks [3]. However, these central learning models are inefficient in safeguarding distributed systems [4].

These networked systems, characterized by their multi-node operations and decentralized architectures, face several challenges [5]. This includes vulnerability to cyberattacks, concerns over data privacy, and the need for improved failure tolerance [5]. Enhancing the resilience of distributed systems to errors and malicious attacks has consequently emerged as a crucial area of research [6]. Recent intrusion detection systems (IDS) employ the federated learning (FL) paradigm to train the detection model in a distributed and privacy-preserving mode [7–9]. FL enables cooperative model training without requiring the transfer of raw data, thereby improving participant data privacy [7]. Standard FL is vulnerable to external attacks and server outages due to its centralized model aggregation, which can lead to training failures or erroneous updates to the detection model [10].

FL enables collaborative model training across distributed devices without exchanging raw data, offering inherent privacy benefits and reduced communication costs [11,12]. However, despite its decentralized design, FL commonly relies on a centralized aggregation server, which creates a single point of failure and limits its robustness in distributed environments [13]. It remains vulnerable to poisoning and Byzantine attacks, where adversaries can corrupt model updates to degrade performance [11]. These systems also struggle under network partitions or node failures, and model updates may still leak sensitive information, raising privacy concerns [14]. To mitigate these issues, researchers have proposed integrating blockchain with FL to improve security, trust, and data integrity [15,16]. While promising, many blockchain-based FL approaches incur scalability bottlenecks, high computational overheads, and still rely on centralized

* Corresponding author.
E-mail address: dskim@kumoh.ac.kr (D.S. Kim).

https://doi.org/10.1016/j.hcc.2025.100354

or post-training aggregation, reintroducing the vulnerabilities they aim to resolve [17]. Thus, securing FL for decentralized IDS requires new frameworks that ensure privacy, resilience, and fault tolerance without sacrificing scalability or decentralization.

This study proposes PureChain [18]. This decentralized intrusion detection framework enhances FL with a proof-of-authority and association (PoA^2) consensus mechanism to overcome persistent limitations in fault tolerance, privacy, and adversarial robustness [19]. FL remains vulnerable to model poisoning, Byzantine behavior, and system failures, especially under dynamic or partitioned network conditions [20]. Existing solutions often fail to ensure secure, real-time model aggregation or lack the resilience needed in hostile environments [21]. PureChain, a custom blockchain, addresses these issues by integrating consensus-verified model updates, anomaly detection techniques, and smart contract-based automation. Its architecture leveraging the PoA^2 consensus mechanism ensures tamper-proof, auditable security logs, facilitating decentralized collaboration without the need for raw data exchange. It enforces update quality through a mathematical model that evaluates statistical validity and resistance to manipulation. Key features include Byzantine fault tolerance, privacy leakage control, and automated threat response, enabling secure, scalable, and collaborative intrusion detection across organizational boundaries without centralizing sensitive data.

Specifically, this study focuses on the following:

(1) **PureChain and smart contract integration:** The framework implements secure system initialization via smart contracts with custom PureChain, ensuring tamper-proof contract management and the creation of an attack signature database.
(2) **Dynamic fault tolerance and poisoning resistance:** The system enhances fault tolerance in FL by using Byzantine-resistant aggregation techniques and includes poisoning resistance mechanisms to preserve the integrity of the global model against adversarial threats.
(3) **Privacy Preservation:** Scalable and collaborative framework based on PoA^2 consensus mechanism for secure aggregation to preserve privacy and security, minimizing the risk of sensitive data exposure during local training and model updates in FL.
(4) **Comprehensive Attack Detection:** The framework employs multi-layered attack detection, incorporating local detection at individual participants and global model detection for system-wide anomaly identification. This dual approach, with attack signature generation, enhances overall security effectiveness.
(5) Smart contracts trigger predefined actions upon detecting specific attack patterns, with response effectiveness tracked to enhance future actions, creating an auditable trail for compliance and forensic purposes.

2. Related works

2.1. Traditional centralized approach

Traditional IoT intrusion detection relies on centralized cloud-based learning, which poses limitations like single points of failure (SPoF), variable device environments, and privacy concerns [22]. ML-based IDS often involves transmitting data centrally for universal model training [22]. Centralized learning enables models to generalize based on group data, but it faces challenges in diverse IoT environments [23]. This approach presents scalability, privacy, and security concerns [4]. FL prototypes address these challenges, offering an alternative to traditional centralized models [12].

2.2. Limitations of federated learning approach

FL combines ML techniques for decentralized training across edge devices, ensuring data privacy and security by transmitting only model derivatives to the cloud and redistributing the global model [11,12,24]. The vastness of IoT devices makes sending all data to the cloud impractical, emphasizing the need for FL's privacy-preserving aggregation in industrial IoT [11,12,24]. With increasing security concerns, FL gains attention in cyberattack detection, particularly in cyber–physical systems [25,26]. Recent works focus on privacy-preserving FL, with variations like node-level training, differential privacy (DP), and secure multiparty computing [27,28]. Despite these advancements, the susceptibility of FL to backdoor attacks remains a concern [29].

Despite enabling collaborative model training across decentralized nodes without exchanging raw data, as seen in smart grid applications [24,30], FL faces significant limitations. Centralized aggregation in standard FL introduces vulnerabilities to server outages and malicious updates, with studies showing susceptibility to data poisoning and model inversion attacks [31]. The reliance on a central server also creates a single point of failure, undermining resilience in disconnected scenarios [32]. Non-identical and independent distribution of data across nodes further degrades model performance, with convergence issues reported in heterogeneous networks [33]. Trade-offs include improved privacy from local training versus increased communication overhead and potential privacy leakage through gradient updates, where attackers can reconstruct training data [34]. Decentralization is partially achieved, but the central aggregator limits true peer-to-peer autonomy, necessitating robust defense mechanisms, such as differential privacy, to mitigate these risks [35].

2.3. Limitations of privacy-preserving approach

Privacy-preserving techniques are crucial for distributed networks with DP, adding noise (e.g., $\eta_i^{(t)} \sim \mathcal{N}(0, \sigma^2 I)$ to model updates [36]. Providing (ϵ, δ) privacy guarantees and reducing leakage risks [37]. Homomorphic encryption enables computations on encrypted data, preserving privacy during aggregation, while secure multi-party computation allows collaborative training without exposing individual contributions [38]. However, DP's noise can degrade model accuracy, especially with high ϵ, while homomorphic encryption and SMPC incur significant computational and communication overheads [39]. Zero-Knowledge Proof (ZKP), although effective, requires substantial resources for proof generation, thereby limiting scalability on edge devices [24]. These methods improve privacy but trade off efficiency and decentralization, as complex protocols often require trusted setups or centralized coordination, partially undermining distributed autonomy.

2.4. Blockchain-based federated learning technique

The Fed-Trust cyberattack detection approach [40] integrates a reputation-based blockchain to ensure decentralized recording and verification of transactions, thereby enhancing data security and privacy, and leveraging fog computing-optimized block mining operations to improve overall computation and communication. Blockchain technology addressed SPoF and privacy concerns in FL systems [41,42]. At the same time, private blockchain implementations like the InterPlanetary File System (IPFS) reduce storage costs [41]. Others like [42] secure traffic prediction model aggregation. However, these solutions may incur higher bandwidth and computational complexities. Sun et al. [43] proposed a blockchain to mitigate adversarial training data effects in FL

without considering participant selection and scalability. The approach by [44] emphasized the reliability of IoT devices in model aggregation, although its efficiency in variable network topologies warrants further investigation.

Chen et al. [45] proposed a blockchain framework to address security and privacy concerns in data middle platforms, incorporating smart contracts and encryption technologies. Further research is needed to assess the practical application of cryptographic techniques, particularly in secure aggregation mechanisms. While addressing global model integrity in industrial IoT, Kalapaaking et al. [46] emphasized the need for more consideration of stragglers. Li et al. [47] introduced a blockchain-based decentralized FL to counter malicious attacks and preserve privacy, but it increased validation consumption. Miao et al. [48] introduced an FL privacy-preserving scheme using blockchain, ensuring transparency but lacking mechanisms for handling stragglers and dropouts. Mbonu et al. [49] reduced the computational burden and communication cost of aggregating the trained model using the blockchain network. However, the approach still retains the central server for aggregating post-trained models.

Orabi et al. [50] provide a comprehensive survey of FL integrated with blockchain, highlighting recurring bottlenecks such as communication overhead and reliance on centralized aggregation. Thennakoon et al. [51] present two blockchain–based FL models, centralized and decentralized, yet acknowledge that off-chain aggregation still leaves room for vulnerabilities and adds computational strain. A study reviewing blockchain-based FL approaches highlights unresolved risks, including SPoF and inadequate handling of malicious updates [52]. Despite addressing SPoF through a hierarchical blockchain design, it concedes that both consensus protocols and off-chain aggregation continue to limit efficiency and security. Yang et al. [20] explore reputation-based consensus as a way to strengthen trust in blockchain-based FL, while noting that dependence on off-chain aggregation remains a persistent structural weakness.

Summarizing the existing studies highlights the limitations, suggesting enhancements in security, efficiency, and accuracy. We propose PureChain-enhanced FL, a framework that integrates PureChain with FL to enhance model aggregation security, data privacy, and defense against adversarial attacks, thereby reducing error loss from model compromises. Existing fault tolerance mechanisms in FL, while addressing node dropouts and partial failures, fail to ensure integrity in severe conditions like network partitions or coordinated attacks. The proposed approach leverages PoA^2 consensus mechanism [18] and advanced anomaly detection to enhance robustness against faulty or malicious nodes, addressing gaps in dynamic fault tolerance and attack detection. This framework contributes a novel approach to improving the security and reliability of FL in real-world distributed systems.

2.5. Interplay among fault tolerance, attack detection, and privacy

Most blockchain–FL systems still suffer from several shortcomings related to off-chain aggregation, creating a single point of failure [50]. They accept model updates without quality-gated consensus, allowing poisoned parameters to be permanently recorded [51]. Additionally, they rely on heavy privacy mechanisms that strain resource-limited edge devices. This leaves three persistent gaps: dynamic fault tolerance under network churn, proactive attack containment through integrated screening and on-chain enforcement, and a practical privacy–utility balance with verifiable ε-DP at the update layer without excessive cryptographic overhead.

The growing deployment of *human digital twin* (HDT) systems in healthcare and IoT-driven environments further amplifies these challenges, since HDTs require continuous, multi-modal data streams with stringent guarantees of trust, resilience, and privacy. Recent surveys on HDT networking architectures and supporting technologies highlight the centrality of reliable synchronization and secure cross-domain data exchange for personalized healthcare [53,54]. In particular, the integration of mobile AI-generated content and generative models into HDT frameworks underlines the urgency of safeguarding against adversarial perturbations and maintaining trustworthy provenance of model updates [55,56].

At the same time, the HDT paradigm introduces new expectations for anticipatory coordination and personalized learning. For instance, prediction-enhanced physical–virtual connectivity enables latency-aware synchronization across distributed digital replicas. Federated multi-task learning with formal DP offers adaptive personalization under strict privacy budgets [57,58]. Importantly, HDTs demand not only privacy-preserving but also fault-tolerant infrastructures, given the sensitivity of physiological and behavioral telemetry, where disruptions could directly affect patient outcomes [59].

Our proposed framework closes these gaps by combining PoA^2's standby validators and idle redundant signer to ensure uninterrupted consensus, even during validator failures, with a validation pipeline that integrates a multi-dimensional contribution metric and resilient aggregation so that only high-quality, trustworthy updates are committed. Privacy is preserved through validator-enforced ε-DP checks before acceptance, guaranteeing bounded information leakage while maintaining low computational overhead via SAM$^+$ and off-chain IPFS storage. This tight integration of fault tolerance, secure aggregation, and privacy protection strengthens both the reliability and security of the system.

This integrated framework enhances decentralization, trust, and security by enabling immutable logging, reputation tracking, and PoA^2 consensus-based aggregation, thus eliminating central points of failure. However, scalability remains a challenge due to blockchain's low throughput, high energy demands, and the computational overhead, which may also risk metadata leakage. While removing central servers improves decentralization, dependence on blockchain nodes and smart contract execution introduces constraints, necessitating hybrid designs to balance privacy, autonomy, and performance.

2.6. Analysis of consensus mechanisms

Table 1 situates PoA^2 within the broader consensus design space; we contrast it with Proof-of-Work (PoW), classical Proof-of-Authority (PoA), and Proof-of-Stake (PoS). PoA^2 distinguishes itself from traditional consensus mechanisms like PoW, PoA, and PoS by offering a balance of scalability, energy efficiency, and security. Unlike PoW and PoS, which face high confirmation latency and require significant computational resources, PoA^2 maintains low consensus times through association-weighted validator selection and an idle redundant signer for seamless failover. It makes it particularly efficient, even as signer sets change. In terms of energy consumption, PoA^2 avoids the high energy demands of PoW and the constant stake management of PoS, leveraging SAM$^+$ to suspend sealing during idle periods, resulting in sub-millisecond commit times in our prototype. Security-wise, PoA^2 builds on PoA's identity-based trust but enhances it with dynamic signer selection using association/trust scores, automatic failover with standby validators, and quality-gated commits to ensure that only reliable updates are processed (See Fig. 2).

Table 1
Comparison of consensus mechanisms.

Mechanism	Scalability	Energy profile	Security assumptions	Suitability for FL-IDS
PoW	Medium TPS, high latency under load	Very high	Economic majority (hashrate)	Poor for edge/IIoT due to cost/latency
PoS	Medium–high TPS, moderate latency	Low–moderate	Economic majority (stake), randomness	Better than PoW; still open-network overhead
PBFT	Low–medium validator count ($\mathcal{O}(n^2)$)	Low	$f < n/3$ Byzantine	Strong finality; chatty at scale
PoA	High TPS in permissioned settings	Low	Identity trust; validator compromise risk	Good baseline; brittle to signer failure
PoA^2	**High TPS with fast failover**	**Low (idle auto-mining)**	Identity + association scores; standby replacement	**Best fit for IIoT FL-IDS**

3. System methodology

3.1. Summarized mathematical model of proposed approach

The system components consist of participants $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$ as the set of participating organizations. For local data and models, each participant P_i has a private dataset $D_i = \{(x_j, y_j)\}_{j=1}^{m_i}$, where x_j represents network features, $y_j \in \{0, 1\}$ is the attack or normal labels and m_i is the number of data points at organization P_i. Each P_i maintains a local model θ_i for attack detection. θ_G is the global consensus model, which is updated through federated aggregation of local models. $\mathcal{B} = (B_1, B_2, \ldots, B_t)$ represents the blockchain for each block B_k containing model updates $\Delta\theta_i$, consensus parameters, smart contracts $\mathcal{S} = \{S_1, S_2, \ldots, S_r\}$, attack signatures database $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_q\}$.

The FL process involves local training for each participant P_i, which initialize local model $\theta_i^{(0)} = \theta_G^{(t-1)}$. Training on local data minimizes the loss function in Eq. (1), which in turn enables computation of model updates in Eq. (2).

$$\theta_i^{(t)} = \arg\min_{\theta} \mathcal{L}(\theta, D_i) = \arg\min_{\theta} \frac{1}{|D_i|} \sum_{(x,y)\in D_i} \ell(\theta; x, y). \tag{1}$$

$$\Delta\theta_i^{(t)} = \theta_i^{(t)} - \theta_i^{(0)}. \tag{2}$$

Update verification defines the contribution quality metric $Q(\Delta\theta_i^{(t)})$ based on model improvement, compliance with expected statistical properties, and resistance to poisoning detection tests. Eq. (3) defines secure aggregation that is resistant to poisoning for weight updates, considering both quality and data size.

$$\theta_G^{(t)} = \theta_G^{(t-1)} + \sum_{i=1}^{n} w_i \cdot \Delta\theta_i^{(t)}, \tag{3}$$

for weight w_i given in Eq. (4),

$$w_i = \frac{|D_i| \cdot Q(\Delta\theta_i^{(t)})}{\sum_{j=1}^{n} |D_j| \cdot Q(\Delta\theta_j^{(t)})}. \tag{4}$$

The contribution metric plays a key role in the PureChain-enhanced FL system. It helps ensure that model updates are reliable, resilient to attacks, and come from trustworthy sources. This metric also supports blockchain consensus by serving as a quality check for each contribution, thereby strengthening the system's overall security and privacy. Rather than relying on simple similarity or distance measures, it combines statistical validity, performance improvement, and resistance to adversarial behavior to evaluate each update. This multi-dimensional approach ensures that only meaningful, high-quality updates influence the global model.

Byzantine-resistant aggregation rule $\mathcal{R}$, trimmed mean is applied in Eq. (5).

$$\theta_G^{(t)} = \mathcal{R}(\{\Delta\theta_1^{(t)}, \Delta\theta_2^{(t)}, \ldots, \Delta\theta_n^{(t)}\}). \tag{5}$$

The Byzantine-resistant aggregation rule employs a trimmed mean technique to filter out extreme model updates, eliminating potentially malicious contributions from compromised participants. This mechanism preserves global model integrity by discarding outlier updates that deviate significantly from consensus, ensuring FL resilience when participant trustworthiness cannot be guaranteed.

Integrating the custom PureChain with the PoA^2 consensus mechanism stores consensus on model updates through the function $\mathcal{C}$ in Eq. (6).

$$\mathcal{C}(\{\Delta\theta_1^{(t)}, \Delta\theta_2^{(t)}, \ldots, \Delta\theta_n^{(t)}\}) \to \{0, 1\}. \tag{6}$$

The PoA^2 algorithm, combined with smart auto-mining, addresses the challenges of decentralization, security, scalability, and gas costs when operating across interconnected networks [18,19,60]. It dynamically optimizes transaction processing based on node capacity, block size, and mining speed, reducing latency and energy consumption while enhancing sustainability. Concurrently, PoA^2 maintains network reliability through a standby validator system that monitors and replaces compromised validators, thereby preserving fault tolerance and stability [18].

Validators verify that $\Delta\theta_i^{(t)}$ satisfactorily contribute quality threshold $Q(\Delta\theta_i^{(t)}) > \tau_Q$, and guarantees differential privacy ϵ-DP with parameter $\epsilon < \epsilon_{\max}$. For each detected attack pattern a in the attack signature database, generate signature σ_a in Eq. (7).

$$\sigma_a = H(F_a, T_a, R_a, I_a), \tag{7}$$

where H is a cryptographic hash function, F_a is the attack feature vector, T_a is timestamp, R_a is the response action hash, and I_a is organization identifier. For each participant P_i, detection function f_{θ_i} is given in Eq. (8).

$$f_{\theta_i}(x) = \begin{cases} 1 & \text{if } p_{\theta_i}(y = 1|x) > \delta_i, \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

where $p_{\theta_i}(y = 1|x)$ is the probability of sample x being an attack, and δ_i is the detection threshold. Ensemble detection enhances resilience against individual model failures, as shown in Eq. (9).

$$f_{\text{ensemble}}(x) = g(f_{\theta_G}(x), f_{\theta_i}(x), \{f_{\sigma_j}(x)\}_{j=1}^{q}), \tag{9}$$

where f_{θ_G} is the global model detection, f_{θ_i} is local model detection, f_{σ_j} is signature-based detection, and g is an ensemble function (majority vote).

Automated response orchestration by the smart contract triggers enables the system's predefined, automated responses to detect cyber threats that consider not just the attack signature itself but also contextual information about the system's current state. For example, the same attack signature might trigger different responses depending on whether the system is under heavy load, what other attacks are active, or which systems have already been affected. Eq. (10) defines smart contract S_k

$$S_k : (\sigma_a, \Gamma) \to A_k, \tag{10}$$

for σ_a as the attack signature, Γ is the system state, and A_k is the response action. Smart contract automation enables immediate responses to threats without human intervention, significantly

reducing response time while ensuring the consistent application of security policies across the distributed IDS. Response effectiveness tracking for each response action A_k to attack σ_a is defined in the effectiveness metric Eq. (11).

$$E(A_k, \sigma_a) = h(T_{\text{resolution}}, C_{\text{impact}}, R_{\text{recurrence}}), \tag{11}$$

where $T_{\text{resolution}}$ is time to resolve incident, C_{impact} is impact cost, $R_{\text{recurrence}}$ is recurrence rate, and h is an effectiveness function. Response learning updates response policy π in Eq. (12).

$$\pi^{(t+1)}(\sigma_a) = \arg\max_{A_k} E(A_k, \sigma_a). \tag{12}$$

PureChain audit trail and compliance for an immutable audit record for each security incident I is defined in Eq. (13).

$$\mathcal{A}(I) = (T_d, P_i, \sigma_a, A_k, E, H_{\text{prev}}), \tag{13}$$

where T_d is detection timestamp, P_i is detecting organization, σ_a is attack signature, A_k is response action, E is effectiveness score, H_{prev} is hash of previous audit record. Compliance function V for regulatory framework $\mathcal{F}$ is verified in Eq. (14).

$$V(\mathcal{A}, \mathcal{F}) = \{0, 1\}. \tag{14}$$

Byzantine fault tolerance determines the system's resilience properties. The system maintains integrity if Byzantine nodes $f < n/3$ as in Eq. (15).

$$\forall f < \frac{n}{3} : \|\theta_G^{(t)} - \theta_{\text{G,honest}}^{(t)}\| < \epsilon_{\text{resilience}}, \tag{15}$$

where $\theta_{\text{G,honest}}^{(t)}$ is the global model with only honest participants. For any poisoning attack strategy $\mathcal{P}$, the impact on detection is bounded in Eq. (16).

$$\max_{\mathcal{P}} |F_1(f_{\theta_G}) - F_1(f_{\theta_G^{\mathcal{P}}})| < \delta_{\text{poison}}, \tag{16}$$

where F_1 is the F1-score evaluation metric, $f_{\theta_G^{\mathcal{P}}}$ is the detection function after poisoning. Information leakage from model updates is bounded by Eq. (17) for the preservation of privacy.

$$I(D_i; \Delta\theta_i^{(t)}) < \epsilon_{\text{privacy}}, \tag{17}$$

where $I(X; Y)$ is mutual information between X and Y.

The system combines PureChain and FL to address vulnerabilities such as forks, selfish mining, and weaknesses in smart contracts. Security is enhanced through reputation-weighted consensus, dual-incentive reward functions, and multi-dimensional validation, requiring attackers to compromise blockchain and FL components simultaneously. DP and homomorphic encryption address privacy concerns, while formal verification ensures the security of smart contracts. This integrated framework provides robust security and privacy guarantees for distributed attack detection systems. Fig. 1 outlines the process flow of the proposed PureChain and FL IDS methodology.

3.2. PureChain-based threat detection and response mechanism

PureChain is a customized private Ethereum-based blockchain network that integrates the PoA^2 consensus mechanism [18] and the SAM$^+$ protocol [60] to facilitate secure, scalable, and energy-efficient transaction validation in FL systems, as shown in Fig. 2. Unlike computationally intensive consensus models such as PoW, which are unsuitable for resource-constrained IoT devices, PoA^2 leverages a lightweight, permissioned approach involving a small pool of trusted validators, typically administrators, thereby enhancing scalability and operational efficiency [61]. PoA^2 extends traditional PoA by incorporating identity-based validator authentication, trust-driven association metrics, and an idle redundant signer mechanism to ensure uninterrupted consensus under dynamic network conditions. The prototype implementation configures the validator pool with five (5) and ten (10) active signers, respectively, supplemented by a single redundant signer for resilience testing. In parallel, the SAM$^+$ protocol optimizes energy consumption by dynamically suspending mining during idle periods, using an intelligent mining timer and auto-mining functions to adjust mining intervals based on system activity [60]. This combination enhances PureChain's suitability for IoT and edge-centric federated environments, while maintaining robust fault tolerance, demonstrated through experimental configurations with five and ten signers and an automatic failover mechanism via standby validators.

In addressing the broader blockchain quadrilemma, an extension of the trilemma that adds sustainability or interoperability as a fourth dimension [62,63], PureChain provides a balanced approach to decentralization, scalability, security, and sustainability. It achieves this through a hybrid on-chain or off-chain execution model in which computationally intensive tasks, such as threat inference and federated model aggregation, are offloaded to off-chain modules. In contrast, critical operations, smart contract execution, transaction validation, and threat logging remain anchored on-chain to preserve verifiability and auditability.

FL models are persistently stored on the IPFS, with associated Content Identifiers (CIDs) immutably recorded on the blockchain. Cyber threat mitigation is automated through parameterized smart contract templates, instantiated with structured metadata (e.g., threat type, severity, mitigation strategy), compiled into domain-specific bytecode, and deployed on-chain. These contracts continuously monitor telemetry through on-chain event listeners, autonomously triggering mitigation protocols, such as node isolation, credential revocation, or incident reporting, upon detection of predefined threat conditions. Moreover, PureChain supports cross-domain threat intelligence sharing through decentralized, privacy-preserving cryptographic protocols, enabling secure and tamper-evident data exchange among heterogeneous systems. This architecture establishes a resilient and verifiable cybersecurity layer, capable of autonomous defense within dynamic federated environments. Algorithm Pseudocode 1 illustrates the Smart contract for fault-tolerant trust and model logging.

Algorithm 1 Fault-tolerant trust and model logging smart contract

Input: Participant p, trust score S, model CID *CID*, round r
Output: Trust log, model log, blacklist

Init: Set thresholds θ, Δt, window k, fault limit $f_{\max}$;
Function `LogTrustScore(p, S, r)`**:**
  Trust[p][r] $\leftarrow S$;
  Compute $\bar{S}_k \leftarrow$ AvgTrust(p, r, k);
  if $|S - \bar{S}_k| > \theta$ **then**
    Faults[p] $\leftarrow$ Faults[p] $\cup\{r\}$;
    if $|Faults[p]| > f_{\max}$ **then**
      Blacklist[p] $\leftarrow$ now;
Function `LogGlobalModel(CID, r)`**:**
  Model[r] $\leftarrow$ *CID*;
Function *IsBlacklisted(p)*:
  return ($p \in$ Blacklist) **and** (now $-$ Blacklist[p] $< \Delta t$);
Function *Reinstate(p)*:
  if $p \in$ *Blacklist* ***and*** *expired* **then**
    delete Blacklist[p]; delete Faults[p];
Function *AvgTrust(p, r, k)*:
  return average of Trust[p][$r - k$ to $r - 1$];

The PureChain-based FL framework integrates decentralized security and privacy-preserving mechanisms for collaborative

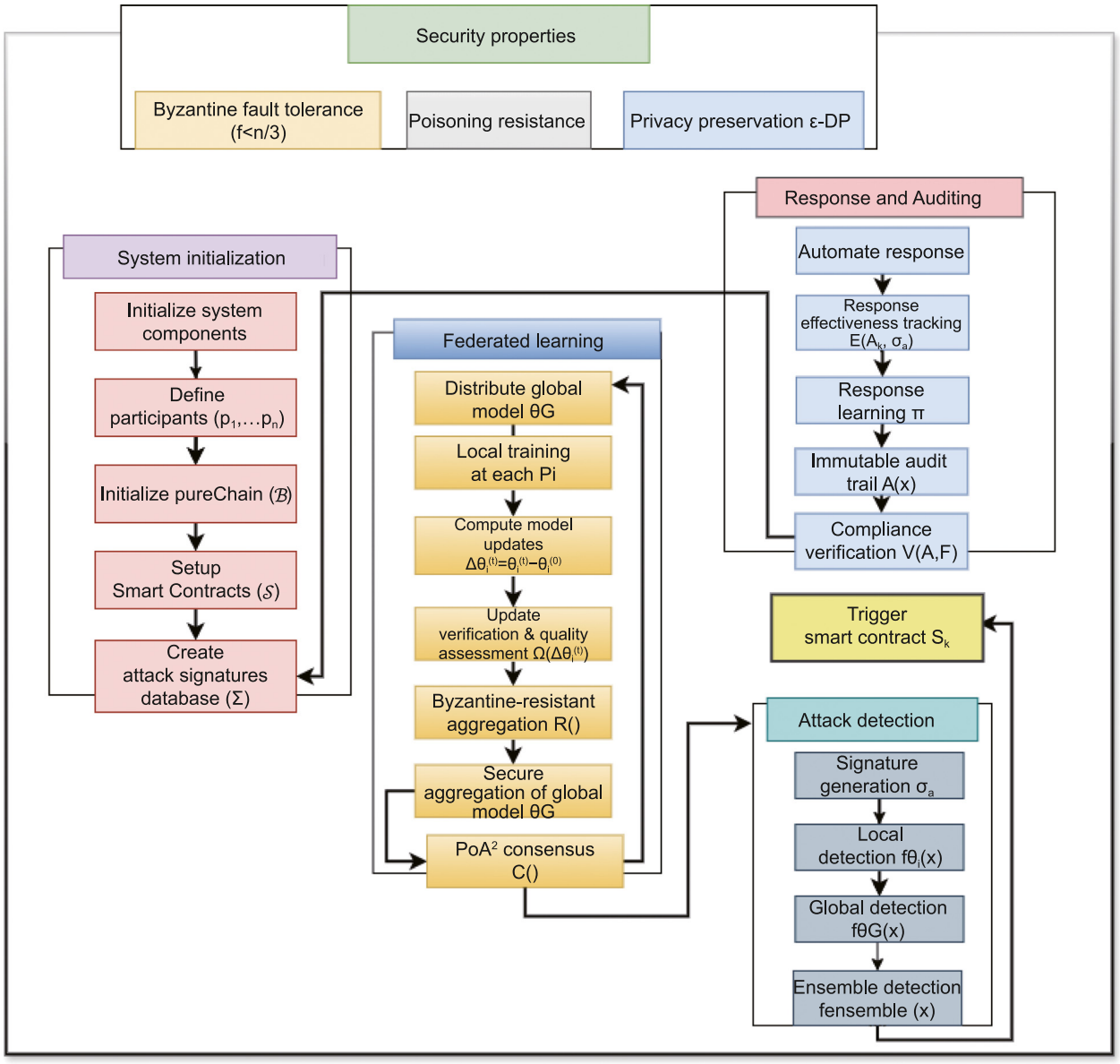

Fig. 1. Secure FL framework for industrial attack detection incorporating Byzantine fault tolerance, poisoning resistance, and DP. The system includes initialization, iterative Byzantine-resilient model aggregation, multi-level attack detection, and automated response with audit and compliance verification via smart contracts.

model training, ensuring data integrity and scalability. It utilizes PureChain-based smart contracts and IPFS for key processes, including node registration, training, verification, aggregation, and reward distribution. Interaction with the PureChain network is facilitated through the Web3.py Python library, which supports transactions, function calls, and contract deployment [64]. This setup, including an HTTP server and command-line interface, simplifies development and testing. IPFS stores training results, while smart contracts handle aggregation and verification on-chain. The process begins with client registration using the `register()` function, ensuring that only eligible nodes are included in the FL process. Clients are then randomly selected for training through the `selectClients()` function, ensuring fairness. Training cycles are tracked using the `verificationRound` and `round` counters.

During local model training, selected clients process IoT data locally, thereby preserving privacy by avoiding the transfer of raw data. Once training is complete, model updates are uploaded to IPFS and mapped to clients using the `update Client_Accuracy()` function, ensuring unique and tamper-proof submissions. The verification phase follows, where verifiers approve updates based on accuracy thresholds using the `verify Clients()` function. Majority voting is employed through the `acceptance_ Count()` function to ensure that only valid updates are accepted. Aggregated updates are stored in the `history` mapping, enabling transparency and auditability.

The global model, representing the collective intelligence of participating nodes, is redistributed for further local training in the next round. A reward mechanism incentivizes active participation, with verified clients claiming rewards via the `claimReward()` function. Clients failing to meet accuracy requirements are removed using the `deleteClient()` function, while non-participating verifiers are eliminated using the `deleteverifier()` function. At the end of each training round, the system is reset using the `Allresite()` and `Reset_Verified()` functions to ensure a fresh start for the next cycle.

To maintain transparency, the system stores a historical record of all model updates and training rounds using the

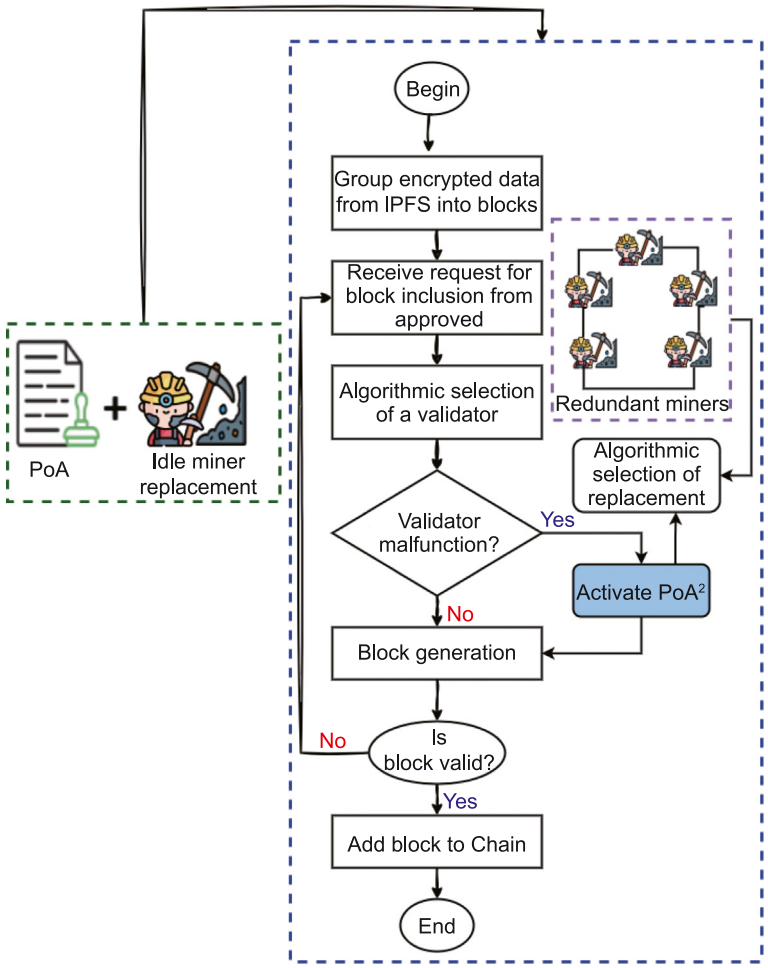

Fig. 2. Process pipeline of redundancy handling in PoA^2 consensus and validator selection in FL for dynamic fault tolerance and attack detection in distributed systems.

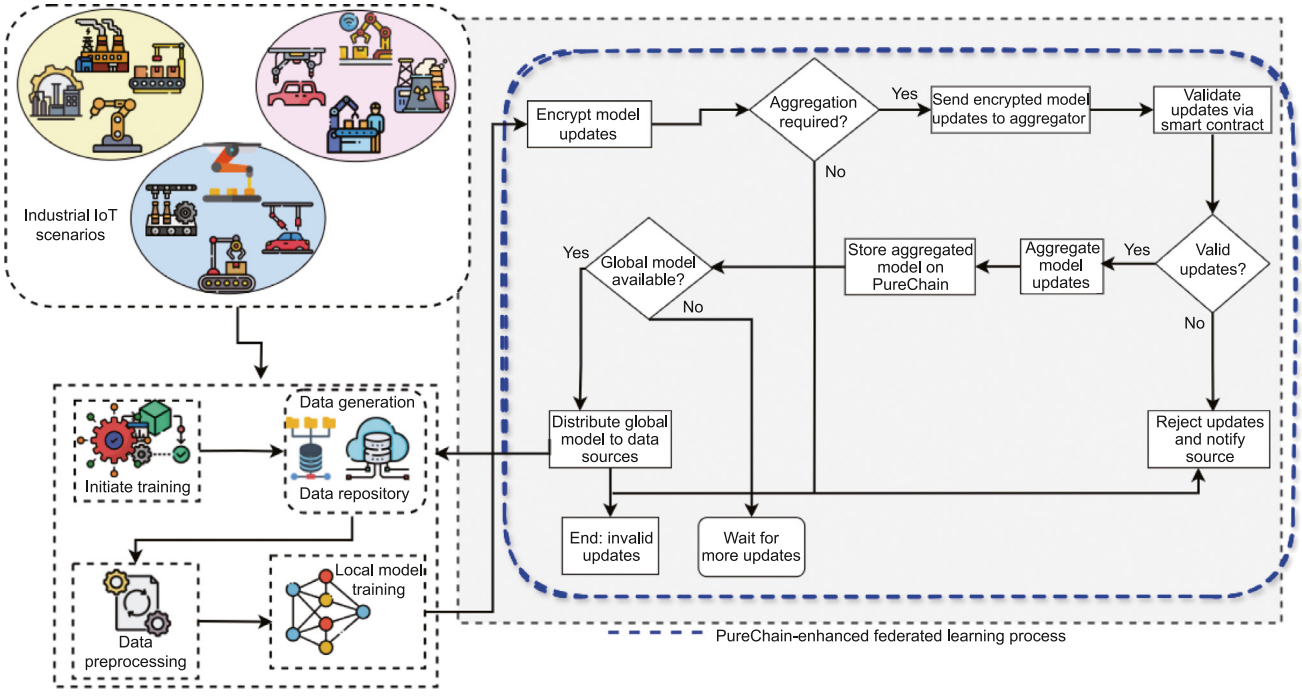

Fig. 3. Design details of the proposed system architecture illustrating the interaction of the involved entities.

`get_history()` function. This immutable record enables administrators to audit contributions, track outcomes, and assess individual clients, thereby fostering trust and accountability within the decentralized system. Algorithm Pseudocode 2 and Fig. 3 illustrate the design details of the system.

3.3. Smart contract design functionality

We implement *Registry* and *Aggregator* Smart contracts. The *Registry* tracks participant roles {Client, Verifier, Validator}. It manages trust scores via a sliding window $\bar{S}_k(c)$ with thresholds $(\theta, f_{\max})$, and enforces blacklists. The *Aggregator* governs update submission, verification, threshold-based aggregation, and reward distribution. Core state includes mappings for participant roles, global model CIDs, and per-round pending updates {`addr, cid,` n_i`, score, verified`}. Updates must satisfy DP and quality gates $Q(\Delta\theta_i) > \tau_Q$ and $\varepsilon_i \leq \varepsilon_{\max}$ before being marked as `verified`.

3.3.1. Submission → verification → aggregation commit process

Submission: Each participating client posts to the contract the IPFS CID of its encrypted model update and associated metadata $(n_i, \varepsilon_i, \text{hash}(\Delta\theta_i))$, where n_i denotes the local sample count, ε_i the claimed privacy budget, and $\text{hash}(\Delta\theta_i)$ a commitment to the model delta. The smart contract enforces *one* submission per round per client and rejects any transactions originating from blacklisted addresses.

Verification: Independent verifiers compute an off-chain quality score $Q(\Delta\theta_i)$ (Eqs. (3)–(4)) using statistical consistency tests and local hold-out evaluation. They then invoke `approve(`c, r, *q_score*, ε`)` to cast their verdict. Verification follows a majority-vote mechanism combined with score thresholds, marking updates as `verified=true` only when consensus is reached. This procedure aligns with the `verifyClients()` and `acceptance_Count()` routines.

Aggregation Commit: When the set of verified updates satisfies $|\{i : \text{verified}\}| \geq m_{\min}$, the lead validator finalizes aggregation according to Eq. (18).

$$\theta_G^{(t)} = R\left(\{\Delta\theta_i\}\right), \tag{18}$$

where $R(\cdot)$ implements a trimmed-mean aggregator (Eq. (5)). The resulting global model CID is persisted via `roundCID`$[r] = \text{cid}(\theta_G^{(t)})$, and an event is emitted anchoring the audit tuple $A(I)$ (Eq. (13)) for immutable traceability.

3.4. Machine learning model implementation

The intrusion detection model utilizes a Long Short-Term Memory (LSTM) network, which captures temporal dependencies through its gating mechanism and internal cell state, C_t. The LSTM structure consists of the following components:

(1) **Forget gate** in Eq. (19) determines the extent to which the previous cell state C_{t-1} is retained.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f). \tag{19}$$

At time t, x_t denotes the input vector and h_{t-1} the previous hidden state. The forget gate employs weight matrix W_f, bias b_f, and sigmoid activation $\sigma \in (0, 1)$ to regulate the retention of past information.

(2) **Input gate** controls the update of new information into the cell in Eq. (20).

$$\begin{aligned} i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i), \\ \tilde{C}_t &= \tanh(W_C[h_{t-1}, x_t] + b_C). \end{aligned} \tag{20}$$

The input gate i_t regulates cell state updates through the sigmoid activation function σ. The candidate cell state $\tilde{C}_t$, computed via the hyperbolic tangent function tanh, is based on the concatenated vector $[h_{t-1}, x_t]$. The parameters W_i, b_i and W_C, b_C are the respective weights and biases for i_t and $\tilde{C}_t$.

(3) **Cell state update** integrates past and new information to update the memory in Eq. (21).

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t, \tag{21}$$

where $\odot$ depicts element-wise multiplication and C_{t-1} is the previous cell state.

(4) **Output gate** produces the hidden state h_t, representing the LSTM output in Eq. (22).

$$\begin{aligned} o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o), \\ h_t &= o_t \odot \tanh(C_t), \end{aligned} \tag{22}$$

where W_o and b_o denote the output gate's weight matrix and bias, and h_t is the resulting hidden state, serving as the LSTM's output.

In the proposed framework, the LSTM-based intrusion detection model was trained locally on each client using a mini-batch gradient descent strategy with a batch size of 64. Each client employed the Adam optimizer, which adjusts the step size for each parameter based on the running average of both the gradients (mean) and the squared gradients (variance). This results in faster convergence and better handling of sparse or noisy gradients. A local learning rate of $\eta = 0.001$ was used, with the Adam hyperparameters set to the default values of $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Model training was performed with five (5) clients for five (5), ten (10), fifteen (15), and twenty (20) rounds. To prevent overfitting, early stopping was applied at the client level. Cross-entropy loss was used as the local objective function, appropriate for the multi-class classification task.

Considering edge-deployed systems utilizing LSTM architectures, a unified initialization strategy based on Xavier (Glorot) initialization is employed to ensure stable gradient flow, rapid convergence, and resilience against training instability [65]. Given that LSTM units typically use `tanh` or `sigmoid` activations, the weight initialization variance σ_ℓ^2 for each layer ℓ is in Eq. (23).

$$\sigma_\ell^2 = \frac{2}{n_{\text{in}} + n_{\text{out}}}, \tag{23}$$

where n_{in} and n_{out} denote the number of input and output units of the layer ℓ. The corresponding weights are initialized in Eq. (24).

$$W_\ell \sim \begin{cases} \mathcal{N}(0, \sigma_\ell^2) & \text{for Gaussian initialization,} \\ \mathcal{U}\left(-\sqrt{3\sigma_\ell^2}, \sqrt{3\sigma_\ell^2}\right) & \text{for uniform initialization.} \end{cases} \tag{24}$$

This strategy ensures stable gradient propagation and efficient convergence across diverse IIoT models. After local updates, each client transmits its model parameters to a blockchain-integrated aggregation layer. Rather than relying on a centralized server, the system employs the PoA^2 consensus mechanism to securely verify and record model updates on the PureChain ledger. This decentralized verification process facilitates dynamic fault tolerance by enabling real-time exclusion or quarantine of clients exhibiting abnormal behavior, such as erratic update patterns or missing transmissions. Additionally, an embedded anomaly detection module performs consistency and divergence checks on incoming model updates to identify potential poisoning or inference attacks. Once verified, Federated Averaging is applied across the authenticated models to iteratively update the global model. This blockchain-enhanced aggregation framework ensures transparent, tamper-resistant, and accountable learning across

heterogeneous IIoT nodes, while actively mitigating the impact of adversarial clients and unstable edge devices.

TensorFlow and Keras, for FL, utilize create_data_shard to divide the client's data into shards as shown in Eq. (25).

$$Shard_i = \{(x_i, y_i)\}\ \forall i \in \text{clients}, \tag{25}$$

for x_i as the input features of client i and y_i its corresponding labels. Splitting the data into mini-batches with batch_data as in Eq. (26).

$$\begin{aligned} &\text{Batch}_k = \{x_k, y_k\},\ \ k \in \{1, 2, \ldots, N_{\text{batches}}\}, \\ &\text{Batch Size} = b,\ \ N_{\text{batches}} = \frac{\text{Total Data Points}}{b}. \end{aligned} \tag{26}$$

Training the local models with train_Local_Model incorporates custom layers for improved performance as in Eq. (27).

$$L = \frac{1}{N}\sum_{i=1}^{N} \ell(y_i, f(x_i; w)), \tag{27}$$

where ℓ = Loss function, $f(x; w)$ is the model prediction function with weights w, and N is the number of data points. Local model weights are updated using gradient descent in Eq. (28).

$$w_{t+1} = w_t - \eta \nabla L. \tag{28}$$

Scaled weights from all clients are aggregated through scale_model_weights and sum_scaled_weights as in Eq. (29)

$$w_g = \frac{1}{C}\sum_{i=1}^{C} \alpha_i w_i, \tag{29}$$

for C as number of clients, $\alpha_i = \frac{n_i}{\sum_{j=1}^{C} n_j}$ is the proportion of data held by client i, w_i represents model weights from client i. Each client updates a local model using its data, and the global model's weights, with trn_points_count tracking each client's data contribution.

Adversarial training enhances robustness by utilizing the fast gradient sign method to generate adversarial examples, as shown in Eq. (30), by adding noise to the input data.

$$x_{\text{adv}} = x + \epsilon \cdot \text{sign}(\nabla_x L(y, f(x; w))), \tag{30}$$

where x_{adv} indicates the adversarial example, ϵ is the perturbation factor, and $\nabla_x L$ is the gradient loss for input x. Adversarial training combines clean and adversarial data in Eq. (31).

$$D_{\text{train}} = \{(x, y)\} \cup \{(x_{\text{adv}}, y)\}. \tag{31}$$

The model is retrained using the expanded dataset in Eq. (32).

$$L_{\text{adv}} = \frac{1}{|D_{\text{train}}|} \sum_{(x,y)\in D_{\text{train}}} \ell(y, f(x; w)). \tag{32}$$

Model evaluation is performed using test_model, providing metrics, such as accuracy, loss, and precision. The PureChain-enhanced FL model training architecture is in Algorithm 2.

Algorithm 2 PureChain-enhanced Federated learning

Input: Clients $\mathcal{C}$, registration map $\mathcal{R}$, hashes $\mathcal{H}$, selected $\mathcal{S}$, verifications $\mathcal{V}$, round r, data $\mathcal{D}$, global weights w_g, perturbation ϵ, epochs E
Output: Final model w_g^*, verified updates, rewards

Function `Init(r, c)`:
  if $r = 0$ ***and*** $c \notin \mathcal{R}$ **then**
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{c\}$;
    $\mathcal{R}[c] \leftarrow$ true;
Function `SelectAndHash(C, k)`:
  $\mathcal{S} \leftarrow$ Sample($\mathcal{C}, k$);
  $r \leftarrow r + 1$;
  foreach $c \in \mathcal{C}$ **do**
    if $h_c \notin \mathcal{H}$ **then**
      $\mathcal{H} \leftarrow \mathcal{H} \cup \{h_c\}$;
Function `Verify(S, V)`:
  foreach $v \in \mathcal{S}$ **do**
    if $Acc(v) \geq \theta$ **then**
      $\mathcal{V}[v] \leftarrow$ verify(u_c);
  if *majority*($\mathcal{V}$) **then**
    $\mathcal{H} \leftarrow \mathcal{H} \cup \{u_c\}$;
Function `Reward(c)`:
  if $c \in \mathcal{S}$ ***and*** *not rewarded* **then**
    reward(c);
Function `LocalTrain(S, w_g)`:
  foreach $c \in \mathcal{S}$ **do**
    Train $w_c \leftarrow w_g$ on $\mathcal{D}_c$ for E epochs;
    return w_c;
Function `AdversarialTrain(X, ε)`:
  for $e = 1$ **to** E **do**
    $X_{\text{adv}} \leftarrow X + \epsilon \cdot \text{sign}(\nabla_X \mathcal{L})$;
    Retrain on $X \cup X_{\text{adv}}$;
Function `EvaluateAndReset()`:
  Evaluate w_g on $(X_{\text{test}}, Y_{\text{test}})$;
  Clear $\mathcal{S}, \mathcal{H}$; reset counters;
  return Hash(timestamp, sender) mod $|\mathcal{C}|$;

3.4.1. Dataset description

The proposed concept was validated using benchmark datasets, Edge-IIoTset,[1] WUSTL-IIoT-2021.[2] These datasets, relevant to IIoT communication, capture vulnerabilities in heterogeneous networks.

Experiments were conducted using Visual Studio Code, Ganache v2.7.1, Solidity v0.8.22 for writing smart contracts, and Python 3.6.13 on a Windows 11 system with an Intel i5-8500 CPU and 8 GB RAM. The PureChain setup considered five (5) clients with 1000 data packets and evaluated transactions per second (TPS), throughput, consensus time, latency, and message overhead.

[1] https://ieee-dataport.org/documents/edge-iiotset-new-comprehensive-realistic-cyber-security-dataset-iot-and-iiot-applicatios

[2] https://ieee-dataport.org/documents/wustl-iiot-2021

4. Performance evaluation

4.1. Evaluation metrics

This section evaluates the integration of FL and blockchain for secure model aggregation. Precision minimizes false positives, reducing unnecessary alerts, while the Fscore balances precision and recall, addressing scenarios where both false positives and negatives are critical. Accuracy provides an overall correctness measure but may overlook imbalanced datasets, which are typical in intrusion detection. Cross-entropy loss tracks training convergence and refines prediction probabilities as in Eq. (33). Together, these metrics assess the model's robustness, scalability, and reliability for federated distributed IoT intrusion detection.

$$L = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i). \tag{33}$$

Table 2
Comparative analysis of the proposed method with and without blockchain across different clients and datasets for 10 communication rounds

Dataset	Clients	With blockchain						Without blockchain					
		Accuracy (%)	Precision (%)	F1 score (%)	Upload time (s)	Download time (s)	Aggregation time (s)	Accuracy (%)	Precision (%)	F1 score (%)	Upload time (s)	Download time (s)	Aggregation time (s)
Edge-IIoTset	5	95	94	75	0.000123	0.000066	0.000012	89	93	78	0.0128	0.0136	0.0015
	10	83	82	82	0.000255	0.000125	0.000012	89	93	78	0.0115	0.0132	0.0016
	15	90	89	76	0.000363	0.00019	0.00001	89	93	78	0.0114	0.0135	0.0018
	20	80	72	72	0.000515	0.000235	0.00001	89	93	78	0.0122	0.0147	0.0020
WUSTL	5	98	98	98	0.000076	0.000236	0.000007	99	99	99	0.0127	0.0134	0.0014
	10	98	98	98	0.000165	0.000378	0.000006	99	99	99	0.0125	0.0133	0.0015
	15	98	98	98	0.000521	0.001256	0.000014	99	99	99	0.0119	0.0139	0.0014
	20	99	98	98	0.000357	0.00118	0.000008	99	99	99	0.0120	0.0137	0.0018

The loss function quantified the difference between true labels y_i and predicted probabilities $\hat{y}_i$, optimizing model predictions of the true label y_i, the predicted probability $\hat{y}_i$ for i, and the total data sample N.

4.2. Federated learning evaluation

Table 2 shows the comparative analysis of the FL framework with and without blockchain. In cyber–physical intrusion detection using the Edge-IIoTset dataset, integrating blockchain results in a variable accuracy range of 80% to 95%, while the F1 score fluctuates between 72% and 82%. This variation appears to be influenced by the number of clients, with potential degradation in F1 scores at higher client counts due to stricter validation protocols and exclusion of noisy updates inherent in the blockchain mechanism. Conversely, the system without blockchain exhibits stable performance, maintaining a constant accuracy of 89% and an F1 score of 78%, indicating limited responsiveness to client diversity and a lack of trust-based adaptation. For sensor anomaly detection using the WUSTL dataset, both configurations, whether employing blockchain or not, achieve near-perfect results, with accuracy and F1 scores consistently ranging from 98% to 99%. It suggests that in environments characterized by clean and well-structured data, the advantages of blockchain integration are marginal, as the anomaly detection task is inherently less complex and more linearly separable.

4.2.1. Communication overhead

In terms of communication overhead, systems utilizing blockchain technology demonstrate significantly lower latency, with upload, download, and aggregation times operating in the sub-millisecond range, approximately between 10^{-4} and 10^{-3} s as shown in Table 2. Its efficiency is attributed to the PureChain-enabled parallelized verification mechanisms, PoA^2, which streamline the aggregation process. In contrast, without blockchain, communication delays are notably higher by approximately an order of magnitude, with upload and download times ranging from 0.011 to 0.014s and aggregation times between 0.0014 and 0.0020s. These findings indicate that while centralized approaches may be structurally simpler, they tend to incur greater latency compared to decentralized, blockchain-based alternatives.

4.2.2. System scalability

Table 2 shows that dataset characteristics, client heterogeneity, and the use of blockchain have a significant influence on FL scalability. In Edge-IIoTset, an increased client count leads to reduced accuracy under blockchain due to stricter anomaly filtering, which, while improving robustness, also amplifies rejection rates in noisy and heterogeneous settings. This reveals a trade-off between security and scalability. Conversely, in the more structured WUSTL dataset, performance remains stable as client count grows, suggesting minimal overhead from blockchain. These findings underscore the need to balance resilience and scalability in blockchain-enhanced FL, especially in diverse IIoT environments, and highlight the critical role of dataset structure in system performance.

Blockchain integration in FL enhances fault tolerance, attack resilience, and coordination efficiency, which are crucial for dynamic, latency-sensitive IIoT environments, such as Edge-IIoT. It outperforms traditional FL by adapting to varying client reliability and data quality, with minimal accuracy gains on clean datasets (e.g., WUSTL) but notable improvements in security and scalability. The approach proves particularly effective under adversarial or large-scale deployment conditions.

4.2.3. Resource utilization

Table 3 highlights critical trade-offs between computational efficiency and detection robustness in distributed systems. Edge-IIoT exhibits low to moderate CPU usage, ranging from 26.24% to 82.21%, supporting scalability on constrained devices and reducing the likelihood of client failures, thereby enhancing fault tolerance. WUSTL incurs significantly higher CPU demand exceeding 100%, up to 371.45%, indicating dependence on parallelism that may hinder deployment in edge environments. Detection performance also diverges notably. WUSTL maintains a consistently low false negative rate of 0.02 and false positive rate of 2.02%, reflecting high precision in isolating adversarial activity with minimal disruption to benign clients. Edge-IIoT, while more computationally efficient, exhibits higher variability in false negative rates, ranging from 0.06 to 0.28, and false positive rates of up to 30.56%, indicating a sensitivity–specificity trade-off that may impact stability in adversarial conditions. These findings highlight a core tension in the design of fault-tolerant, attack-resilient distributed systems: the need to balance computational feasibility with the accuracy and stability of detection. Edge-IIoT prioritizes responsive operation suited to a real-time, resource-constrained environment. WUSTL offers stronger adversarial resilience at a higher computational cost. Sustaining performance and robustness in dynamic settings requires adaptive strategies such as load-aware thresholding and hierarchical detection.

Fig. 4 illustrates the interplay between loss and accuracy for client weight sets A to E, identified by IPFS hashes. Weights B and E achieve maximum accuracy at 96% and 99.8%, respectively, indicating high performance. Weight D shows a drop in accuracy (95.7% and 98.8%) and increased loss, suggesting lower prediction reliability and suboptimal generalization. An inverse relationship between loss and accuracy is evident, with weight A having the lowest loss of 0.157, indicating high prediction reliability but slightly lower accuracy. Despite achieving peak accuracy, Weight B exhibits a higher loss of 0.163, suggesting a dependence on specific features. Weights C and D show increased loss and decreased accuracy, indicating reduced model efficiency. Weights B and E are optimal for accuracy, but B's higher loss may compromise reliability, while weight A strikes a balance between low loss

Table 3
Performance evaluation of CPU usage, False negative rates, False positive rate, train and inference times across different clients and datasets for 10 communication rounds.

Dataset	Clients	Train time (s)	Inference time (s)	Total time (s)	CPU usage (%)	FNR (1-P)	FPR (%)
Edge-IIoTset	5	0.261027	0.001327	0.262354	26.24	0.06	7.41
	10	0.461485	0.001475	0.462960	46.30	0.18	25.00
	15	0.594618	0.001501	0.596119	59.61	0.11	12.09
	20	0.820674	0.001469	0.822143	82.21	0.28	30.56
WUSTL	5	0.359945	0.001625	0.361570	36.16	0.02	2.02
	10	1.210861	0.001678	1.212539	121.25	0.02	2.02
	15	3.711897	0.002619	3.714516	371.45	0.02	2.02
	20	3.100867	0.002514	3.103381	310.34	0.02	1.01

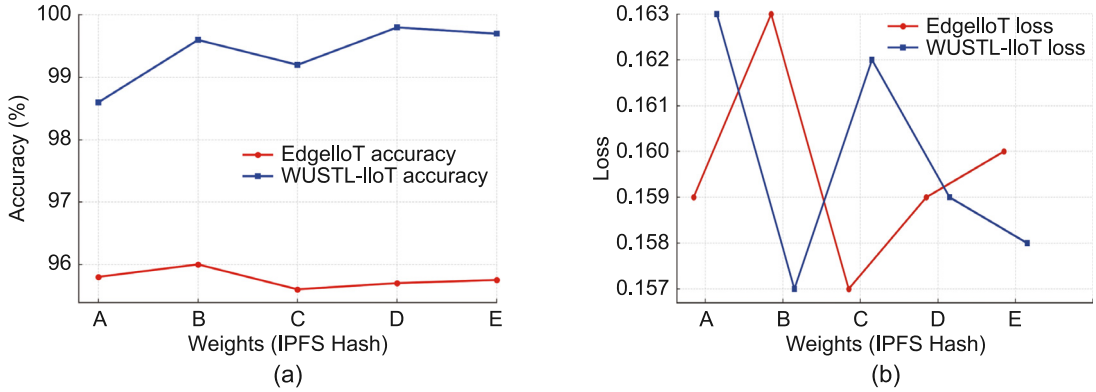

Fig. 4. Plots comparing the delay bound of five clients over ten communication rounds on the EdgeIIoT and IoT dataset scenarios. (a) EdgeIIoT dataset scenario. (b) WUSTL-IIoT-2021 data scenario.

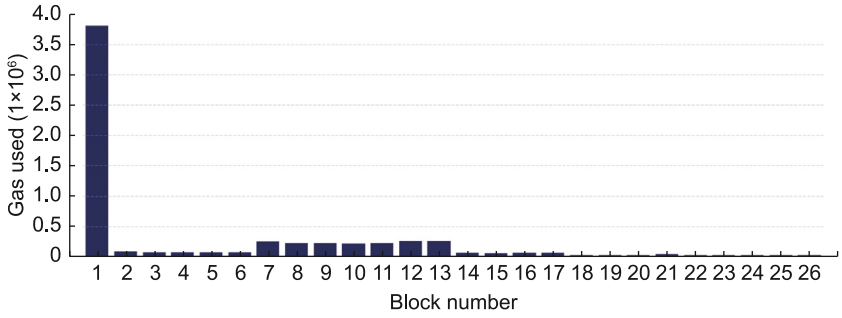

Fig. 5. Visualization of the proposed approach's gas usage distribution across the PureChain blocks.

and accuracy. These results emphasize the importance of weight selection in striking a balance between prediction reliability and performance.

4.3. PureChain evaluation

Fig. 5 illustrates gas usage across PureChain blocks, showing a significant spike of over 3.5×10^6 in the initial block due to the computational demands of smart contract deployment. Subsequent blocks 2 to 26 exhibit minimal and consistent gas usage, reflecting routine transactions like function calls, updates, data logging, and device authentication. This efficiency demonstrates the proposed system's suitability for resource-constrained IoT environments, where low ongoing costs offset initial deployment overhead. Such scalability and energy efficiency enable secure PureChain applications for intrusion detection and data aggregation without overburdening resource-constrained environments.

Further evaluations computed the total number of transactions T per round by multiplying the number of clients by each client's transactions. Transactions per second (TPS) are calculated by dividing the total number of transactions by the time required for each round. Table 4 highlights the system's ability to handle transactions efficiently with low latency (0.0008484 s – 0.0032 s) and variable throughput of 1.625 units/s and 6.127 units/s, with maximum values demonstrating its optimal performance. The system demonstrates high responsiveness and resilience, maintaining stable performance with low latency and moderate throughput variability, which is key for real-time monitoring, fault adaptation, and robust attack detection in secure, fault-tolerant distributed scenarios.

Table 4
PureChain enhanced federated learning latency, throughput, and transaction performance.

Metric	Minimum	Maximum
Transaction/s (TPS)	312.5	1178.3
Latency (s)	0.0008484	0.0032
Throughput (units/s)	1.625	6.127

Table 5 presents PureChain's performance under varying client loads. As the client count increases, consensus time and transactions per block rise, indicating more significant delays in processing and validation. Message overhead also increases, reflecting

Table 5
Detail evaluation outcome of the PureChain enhanced federated learning performance.

No of Clients	Transactions per block	Consensus (s)	Message overhead
40	100	16.000	24
70	120	0.790	84
100	200	0.494	180
120	250	1.518	312
160	330	3.632	480

the growing costs of communication. These trends highlight scalability bottlenecks in blockchain systems, suggesting that while moderate client growth can enhance throughput, surpassing a threshold leads to diminishing returns and higher latency. This highlights the need for scalable consensus protocols and efficient network design.

4.4. Consensus mechanisms performance analysis

Table 6 shows that PoA^2 consistently outperforms PoA, PoS, and PoW across both evaluated datasets, offering the best mix of speed, efficiency, and scalability. For Edge-IIoTset, PoA^2 delivers a throughput of 7.61 tps with a latency of 0.086 s, compared to PoA (4.94 tps, 0.210 s), PoS (8.45 tps, 0.341 s), and PoW (2.90 tps, 2.869 s). This low-latency, high-throughput combination enables the network to process more transactions while swiftly detecting and responding to faults or attacks. The low energy consumption of 5.14 compared to PoW's 157.06 and better than PoS 60.76 makes it viable for continuous operation on energy-constrained IoT nodes. Scalability reaching 1.47 indicates stable performance even as the validator set grows. On WUSTL-IIoT2021, PoA^2 maintains its combined advantage, achieving 6.71 tps and 0.055 s latency, outperforming PoA (4.03 tps, 0.126 s), PoS (9.30 tps, 0.362 s), and PoW (3.14 tps, 1.022 s). Energy use of 5.84, again drastically below PoW's 151.19, and scalability is 1.45, close to the Edge-IIoTset results, showing consistency across workloads. These metrics suggest that PoA^2 can sustain high transaction rates and rapid consensus finality while keeping resource usage low. This translates to quicker fault isolation, faster attack response propagation, and the ability to maintain service reliability under changing network conditions, whether from legitimate churn or adversarial disruptions.

4.5. PureChain vulnerability test

Transaction flooding tested the framework's resilience against denial-of-service attacks by simulating high transaction volumes [66]. Fig. 6 shows the number of successful transactions in the PureChain network, with a linear progression reaching 1000 transactions in 16 s. This consistent throughput demonstrates efficiency, reliability, and minimal latency, making PureChain a scalable and suitable solution for high-demand, distributed IoT environments.

Table 7 presents a comparative evaluation of secure model aggregation and cyberattack detection methods across key metrics, datasets, and computational demands [40,46,49]. While earlier studies demonstrate moderate accuracy, they often lack detailed evaluation metrics and impose substantial computational overhead. In contrast, the proposed blockchain-integrated FL approach offers significantly improved detection performance, achieving up to 98.25% accuracy and 98.00% F1 score on the WUSTL-IIoT-2021 dataset, alongside markedly lower training and aggregation times of 0.534 s and 0.000011 s, respectively. This reflects a substantial advancement in both model robustness and system efficiency, positioning the approach as a scalable and practical solution for deployment in resource-constrained IIoT environments.

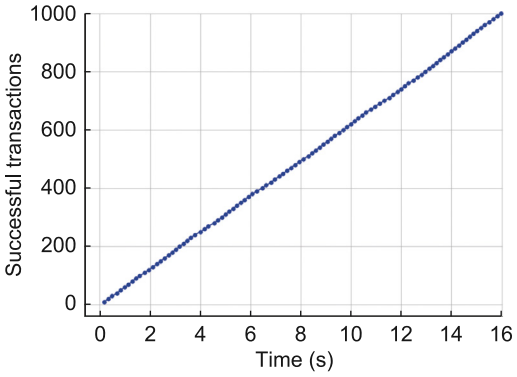

Fig. 6. Vulnerability analysis of the PureChain network.

5. Conclusion

Integrating PureChain with FL significantly enhances the security and resilience of decentralized IDSs. This combination addresses key vulnerabilities, such as model poisoning and Byzantine attacks, ensuring reliable performance in distributed environments. The system enables secure and energy-efficient FL by leveraging a private Ethereum network with the PoA^2 consensus mechanism. Decentered smart contracts guarantee model aggregation, data integrity, and incentivized participation without compromising privacy. IPFS integration for off-chain storage boosts scalability and minimizes blockchain overhead. In addition to achieving an experimental performance of 99.86% global accuracy, 100% precision, and an F1 score of 99%, it outperforms existing solutions in efficiency and computational cost. The PureChain demonstrates robustness and adaptability with a TPS range of 312.5–1178.3 and a low latency range of 0.0008484–0.0032.

The system is resilient to denial-of-service attacks and scalable in high-demand IoT environments, making it well-suited for critical industrial applications. Future work will focus on optimizing consensus mechanisms, expanding testing across various distributed settings, and adapting to evolving cyber threats, positioning this model as a robust, scalable, and secure solution for intrusion detection in distributed networks. High TPS, low latency, and high throughput improve performance but can increase consensus and smart contract vulnerabilities. Mitigating these risks requires strong fork-resolution protocols, anti-selfish mining strategies, and formal smart contract verification.

5.1. Limitations and future directions

Despite the strengths of the proposed PureChain-enhanced FL framework, observed limitations are highlighted thus:

(1) Scalability bottlenecks: While PoA^2 consensus improves efficiency, blockchain systems still face inherent throughput and latency constraints that affect large-scale deployments.
(2) Resource consumption and overhead: Although SAM$^+$ reduces idle costs, reliance on validators and smart contract execution introduces computational overhead and potential metadata leakage.
(3) Edge device constraints: Resource-limited IIoT devices may find it challenging to sustain privacy-preserving aggregation and DP operations under continuous workloads.
(4) Interoperability challenges: Current design does not fully address cross-chain operations or integration with heterogeneous industrial systems, which may hinder broader adoption.

Table 6
Performance of consensus mechanisms Datasets.

Dataset	Consensus	Throughput	Latency	Energy	Scalability
Edge-IIoTset	PoA	4.94	0.2100	15.95	1.01
	PoS	8.45	0.3415	60.76	0.82
	PoW	2.90	2.8697	195.95	0.57
	PoA^2	**7.61**	**0.08612**	**5.14**	**1.47**
WUSTL-IIoT-2021	PoA	4.03	0.1626	10.79	1.19
	PoS	9.30	0.3623	62.26	0.84
	PoW	3.14	1.2023	151.19	0.58
	PoA^2	**6.127**	**0.0032**	**5.84**	**1.45**

Table 7
Comparative analysis of the proposed technique with existing studies (Yes: $\surd$, No: χ).

Author	Approach	Dataset	Global accuracy (%)	Precision (%)	F1 score (%)	Average train time (s)	Average aggregation time (s)
[46]	Blockchain network for secure model aggregation	F-MNIST	93.20	χ	χ	660	1200
		CIFAR-10	73.40	χ	χ	240	1260
		CIFAR-100	84.80	χ	χ	390	1080
		MNIST	93.60	χ	χ	380	960
[49]	End-process blockchain-based secure aggregation mechanism	EMNIST	75	χ	χ	14	14
[40]	Fed-Trust for cyberattack detection in IIoT	ToN IoT	92.05	82.90	91.91	χ	1200
		LITNET-2020	93.13	84.33	92.47	χ	1200
This study	**This study**	**Edge-IIoTset**	**87.00**	**84.25**	**76.25**	**0.534**	**0.000011**
		WUSTL-IIoT-2021	**98.25**	**98.00**	**98.00**	**2.096**	**0.000007**

Future work will therefore focus on exploring hybrid consensus models and lightweight off-chain trust anchors to improve scalability and efficiency. Research into adaptive privacy budgets for personalized learning, quantum-resistant cryptographic primitives, and cross-chain interoperability mechanisms will also be valuable. Furthermore, extending our framework to integrate intent-driven orchestration and edge–cloud resource allocation could provide better adaptability to dynamic and adversarial environments.

CRediT authorship contribution statement

Love Allen Chijioke Ahakonye: Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Cosmas Ifeanyi Nwakanma:** Writing – review & editing, Visualization, Validation, Supervision, Investigation. **Jae Min Lee:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition. **Dong Seong Kim:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported 40% by Innovative Human Resource Development for Local Intellectualization program through the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (IITP-2025-RS-2020-II201612), 30% by the Priority Research Centers Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology, South Korea (2018R1A6A1A03024003), and 30% by the IITP (Institute of Information & Communications Technology Planning & Evaluation)-ITRC (Information Technology Research Center) grant funded by the Korea government (Ministry of Science and ICT) (IITP-2025-RS-2024-00438430).

References

[1] T.S. AlSalem, M.A. Almaiah, A. Lutfi, Cybersecurity risk analysis in the IoT: A systematic review, Electronics 12 (18) (2023) 3958, http://dx.doi.org/10.3390/electronics12183958.
[2] L.A.C. Ahakonye, C.I. Nwakanma, J.-M. Lee, D.-S. Kim, Trees bootstrap aggregation for detection and characterization of IoT-SCADA network traffic, IEEE Trans. Ind. Informatics (2023) 1–12, http://dx.doi.org/10.1109/TII.2023.3333438.
[3] L.A.C. Ahakonye, C.I. Nwakanma, J.-M. Lee, D.-S. Kim, Efficient classification of enciphered SCADA network traffic in smart factory using decision tree algorithm, IEEE Access 9 (2021) 154892–154901, http://dx.doi.org/10.1109/ACCESS.2021.3127560.
[4] G. Drainakis, P. Pantazopoulos, K.V. Katsaros, V. Sourlas, A. Amditis, D.I. Kaklamani, From centralized to federated learning: Exploring performance and end-to-end resource consumption, Comput. Netw. 225 (2023) 109657, http://dx.doi.org/10.1016/j.comnet.2023.109657.
[5] F. Louati, F.B. Ktata, I. Amous, Big-IDS: A decentralized multi-agent reinforcement learning approach for distributed intrusion detection in big data networks, Clust. Comput. 27 (5) (2024) 6823–6841, http://dx.doi.org/10.1007/s10586-024-04306-9.
[6] M. Yemini, A. Nedić, S. Gil, A.J. Goldsmith, Resilience to malicious activity in distributed optimization for cyberphysical systems, in: 2022 IEEE 61st Conference on Decision and Control, CDC, 2022, pp. 4185–4192, http://dx.doi.org/10.1109/CDC51059.2022.9992416.
[7] S. Agrawal, S. Sarkar, O. Aouedi, G. Yenduri, K. Piamrat, M. Alazab, S. Bhattacharya, P.K.R. Maddikunta, T.R. Gadekallu, Federated learning for intrusion detection system: Concepts, challenges and future directions, Comput. Commun. 195 (2022) 346–361, http://dx.doi.org/10.1016/j.comcom.2022.09.012.
[8] O. Alsad, Q. Abu Al-Haija, M. Al Nabhan, Federated learning for intrusion detection system: A systematic review, in: A. Ullah, S. Anwar (Eds.), Proceedings of International Conference on Information Technology and Applications, Springer Nature Singapore, Singapore, 2025, pp. 395–408.
[9] R. Zhao, Y. Wang, Z. Xue, T. Ohtsuki, B. Adebisi, G. Gui, Semisupervised federated-learning-based intrusion detection method for internet of things, IEEE Internet Things J. 10 (10) (2023) 8645–8657, http://dx.doi.org/10.1109/JIOT.2022.3175918.
[10] G. Shirvani, S. Ghasemshirazi, B. Beigzadeh, Federated learning: Attacks, defenses, opportunities and challenges, in: 2024 11th International Symposium on Telecommunications, IST, 2024, pp. 470–479, http://dx.doi.org/10.1109/IST64061.2024.10843494.
[11] J. Zhang, H. Zhu, F. Wang, J. Zhao, Q. Xu, H. Li, et al., Security and privacy threats to federated learning: Issues, methods, and challenges, Secur. Commun. Networks 2022 (2022) http://dx.doi.org/10.1155/2022/2886795.
[12] J. Wen, Z. Zhang, Y. Lan, Z. Cui, J. Cai, W. Zhang, A survey on federated learning: Challenges and applications, Int. J. Mach. Learn. Cybern. 14 (2) (2023) 513–535, http://dx.doi.org/10.1007/s13042-022-01647-y.

[13] C. Feng, A.H. Celdrán, J. Von der Assen, E.T.M. Beltrán, G. Bovet, B. Stiller, Dart: A solution for decentralized federated learning model robustness analysis, Array (2024) 100360, http://dx.doi.org/10.1016/j.array.2024.100360.
[14] X. Zhang, W. Chen, Theoretical analysis of privacy leakage in trustworthy federated learning: A perspective from linear algebra and optimization theory, 2024, http://dx.doi.org/10.48550/arXiv.2407.16735, arXiv preprint arXiv:2407.16735.
[15] V. Wylde, N. Rawindaran, J. Lawrence, R. Balasubramanian, E. Prakash, A. Jayal, I. Khan, C. Hewage, J. Platts, Cybersecurity, data privacy and blockchain: A review, SN Comput. Sci. 3 (2) (2022) 127, http://dx.doi.org/10.1007/s42979-022-01020-4.
[16] L.A.C. Ahakonye, C.I. Nwakanma, D.-S. Kim, Tides of blockchain in IoT cybersecurity, Sensors 24 (10) (2024) 3111, http://dx.doi.org/10.3390/s24103111.
[17] W. Ning, Y. Zhu, C. Song, H. Li, L. Zhu, J. Xie, T. Chen, T. Xu, X. Xu, J. Gao, Blockchain-based federated learning: A survey and new perspectives, Appl. Sci. (2076-3417) 14 (20) (2024) http://dx.doi.org/10.3390/app14209459.
[18] D.-S. Kim, I.S. Igboanusi, L.A. Chijioke Ahakonye, G.O. Anyanwu, Proof-of-authority-and-association consensus algorithm for IoT blockchain networks, in: 2025 IEEE International Conference on Consumer Electronics, ICCE, 2025, pp. 1–6, http://dx.doi.org/10.1109/ICCE63647.2025.10930052.
[19] D.-S. Kim, R. Syamsul, Integrating machine learning with proof-of-authority-and-association for dynamic signer selection in BlockchaiN-networks, ICT Express (2024) http://dx.doi.org/10.1016/j.icte.2024.10.008.
[20] G. Yan, H. Wang, X. Yuan, J. Li, Enhancing model poisoning attacks to Byzantine-robust federated learning via critical learning periods, in: Proceedings of the 27th International Symposium on Research in Attacks, Intrusions and Defenses, 2024, pp. 496–512, http://dx.doi.org/10.1145/3678890.3678915.
[21] J. Wu, F. Luo, T. Sun, H. Wang, W. Zhang, Privacy-preserving federated learning scheme with mitigating model poisoning attacks: Vulnerabilities and countermeasures, 2025, http://dx.doi.org/10.48550/arXiv.2506.23622, arXiv preprint arXiv:2506.23622.
[22] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, M. Guizani, A survey on federated learning: The journey from centralized to distributed on-site learning and beyond, IEEE Internet Things J. 8 (7) (2021) 5476–5497, http://dx.doi.org/10.1109/JIOT.2020.3030072.
[23] S. Peng, Y. Yang, M. Mao, D.-S. Park, Centralized machine learning versus federated averaging: A comparison using MNIST dataset, KSII Trans. Internet & Inf. Syst. 16 (2) (2022) http://dx.doi.org/10.3837/tiis.2022.02.020.
[24] V. Mothukuri, R.M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, G. Srivastava, A survey on security and privacy of federated learning, Future Gener. Comput. Syst. 115 (2021) 619–640, http://dx.doi.org/10.1016/j.future.2020.10.007.
[25] B. Li, Y. Wu, J. Song, R. Lu, T. Li, L. Zhao, DeepFed: Federated deep learning for intrusion detection in industrial cyber-physical systems, IEEE Trans. Ind. Informatics 17 (8) (2021) 5615–5624, http://dx.doi.org/10.1109/TII.2020.3023430.
[26] S.A. Rahman, H. Tout, C. Talhi, A. Mourad, Internet of things intrusion detection: Centralized, on-device, or federated learning? IEEE Netw. 34 (6) (2020) 310–317, http://dx.doi.org/10.1109/MNET.011.2000286.
[27] L. Yin, J. Feng, H. Xun, Z. Sun, X. Cheng, A privacy-preserving federated learning for multiparty data sharing in social IoTs, IEEE Trans. Netw. Sci. Eng. 8 (3) (2021) 2706–2718, http://dx.doi.org/10.1109/TNSE.2021.3074185.
[28] Y. Zhao, J. Zhao, M. Yang, T. Wang, N. Wang, L. Lyu, D. Niyato, K.-Y. Lam, Local differential privacy-based federated learning for internet of things, IEEE Internet Things J. 8 (11) (2021) 8836–8853, http://dx.doi.org/10.1109/JIOT.2020.3037194.
[29] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, V. Shmatikov, How to backdoor federated learning, in: S. Chiappa, R. Calandra (Eds.), Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, in: Proceedings of Machine Learning Research, 108, PMLR, 2020, pp. 2938–2948.
[30] B. Pang, H.-H. Liang, L.-H. Zhang, Y.-F. Teng, Z.-W. Chang, Z.-W. Liu, C.-Q. Hu, W.-H. Mou, An improved federated learning-assisted data aggregation scheme for smart grids, Appl. Sci. 13 (17) (2023) 9813, http://dx.doi.org/10.3390/app13179813.
[31] V. Tolpegin, S. Truex, M.E. Gursoy, L. Liu, Data poisoning attacks against federated learning systems, in: Computer Security–ESORICs 2020: 25th European Symposium on Research in Computer Security, ESORICs 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25, Springer, 2020, pp. 480–501, http://dx.doi.org/10.1007/978-3-030-58951-6_24.
[32] Y. Dong, L. Zhang, L. Xu, Privacy-preserving and reliable distributed federated learning, in: Z. Tari, K. Li, H. Wu (Eds.), Algorithms and Architectures for Parallel Processing, Springer Nature Singapore, Singapore, 2024, pp. 130–149.
[33] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, Y. Liu, Privacy-preserving blockchain-based federated learning for IoT devices, IEEE Internet Things J. 8 (3) (2021) 1817–1829, http://dx.doi.org/10.1109/JIOT.2020.3017377.
[34] M. Kim, O. Günlü, R.F. Schaefer, Federated learning with local differential privacy: Trade-offs between privacy, utility, and communication, in: ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, 2021, pp. 2650–2654, http://dx.doi.org/10.1109/ICASSP39728.2021.9413764.
[35] H. Kaur, V. Rani, M. Kumar, M. Sachdeva, A. Mittal, K. Kumar, Federated learning: A comprehensive review of recent advances and applications, Multimedia Tools Appl. 83 (18) (2024) 54165–54188, http://dx.doi.org/10.1007/s11042-023-17737-0.
[36] Y. Li, Y. Zhou, A. Jolfaei, D. Yu, G. Xu, X. Zheng, Privacy-preserving federated learning framework based on chained secure multiparty computing, IEEE Internet Things J. 8 (8) (2021) 6178–6186, http://dx.doi.org/10.1109/JIOT.2020.3022911.
[37] M. Wang, Q.-Y. Wang, Y.-H. Zhang, Z.-X. Zhang, Y.-N. Feng, Y.-F. Cao, Preserving differential privacy in neural networks for foreign object detection with heterogeneity-based noising among distributed devices, J. Supercomput. 80 (14) (2024) 21447–21474, http://dx.doi.org/10.1007/s11227-024-06243-1.
[38] S. Park, J. Lee, K. Harada, J. Chi, Masking and homomorphic encryption-combined secure aggregation for privacy-preserving federated learning, Electron. (2079-9292) 14 (1) (2025) http://dx.doi.org/10.3390/electronics14010177.
[39] R. Aziz, S. Banerjee, S. Bouzefrane, T. Le Vinh, Exploring homomorphic encryption and differential privacy techniques towards secure federated learning paradigm, Futur. Internet 15 (9) (2023) 310, http://dx.doi.org/10.3390/fi15090310.
[40] M. Abdel-Basset, N. Moustafa, H. Hawash, Privacy-preserved cyberattack detection in industrial edge of things (IEoT): A blockchain-orchestrated federated learning approach, IEEE Trans. Ind. Informatics 18 (11) (2022) 7920–7934, http://dx.doi.org/10.1109/TII.2022.3167663.
[41] P. Zhang, G. Liu, Z. Chen, J. Guo, P. Liu, A study of a federated learning framework based on the interstellar file system and blockchain: Private blockchain federated learning, in: 2022 3rd International Conference on Computer Vision, Image and Deep Learning & International Conference on Computer Engineering and Applications (CVIDL & ICCEA), 2022, pp. 267–273, http://dx.doi.org/10.1109/CVIDLICCEA56201.2022.9824166.
[42] Q. Zhang, P. Palacharla, M. Sekiya, J. Suga, T. Katagiri, Blockchain-based secure aggregation for federated learning with a traffic prediction use case, in: 2021 IEEE 7th International Conference on Network Softwarization (NetSoft), 2021, pp. 372–374, http://dx.doi.org/10.1109/NetSoft51509.2021.9492652.
[43] Y. Sun, H. Esaki, H. Ochiai, Blockchain-based federated learning against end-point adversarial data corruption, in: 2020 19th IEEE International Conference on Machine Learning and Applications, ICMLA, 2020, pp. 729–734, http://dx.doi.org/10.1109/ICMLA51294.2020.00119.
[44] R.A. Mallah, D. López, T. Halabi, Blockchain-enabled efficient and secure federated learning in IoT and edge computing networks, in: 2023 International Conference on Computing, Networking and Communications, ICNC, 2023, pp. 511–515, http://dx.doi.org/10.1109/ICNC57223.2023.10074277.
[45] C. Chen, S. Goyal, K. Ramaswamy, BSPPF: Blockchain-based security and privacy preventing framework for data middle platform in the era of IR 4.0, J. Nanomater. 2022 (2022) http://dx.doi.org/10.1155/2022/2219006.
[46] A.P. Kalapaaking, I. Khalil, M.S. Rahman, M. Atiquzzaman, X. Yi, M. Almashor, Blockchain-based federated learning with secure aggregation in trusted execution environment for internet-of-things, IEEE Trans. Ind. Informatics 19 (2) (2023) 1703–1714, http://dx.doi.org/10.1109/TII.2022.3170348.
[47] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng, Q. Yan, A blockchain-based decentralized federated learning framework with committee consensus, IEEE Netw. 35 (1) (2021) 234–241, http://dx.doi.org/10.1109/MNET.011.2000263.
[48] Y. Miao, Z. Liu, H. Li, K.-K.R. Choo, R.H. Deng, Privacy-preserving Byzantine-robust federated learning via blockchain systems, IEEE Trans. Inf. Forensics Secur. 17 (2022) 2848–2861, http://dx.doi.org/10.1109/TIFS.2022.3196274.
[49] W.E. Mbonu, C. Maple, G. Epiphaniou, An end-process blockchain-based secure aggregation mechanism using federated machine learning, Electronics 12 (21) (2023) http://dx.doi.org/10.3390/electronics12214543.
[50] M.M. Orabi, O. Emam, H. Fahmy, Adapting security and decentralized knowledge enhancement in federated learning using blockchain technology: Literature review, J. Big Data 12 (1) (2025) 55, http://dx.doi.org/10.1186/s40537-025-01099-5.
[51] R. Thennakoon, A. Wanigasundara, S. Weerasinghe, C. Seneviratne, Y. Siriwardhana, M. Liyanage, Decentralized defense: Leveraging blockchain against poisoning attacks in federated learning systems, in: 2024 IEEE 21st Consumer Communications & Networking Conference, CCNC, 2024, pp. 950–955, http://dx.doi.org/10.1109/CCNC51664.2024.10454688.

[52] S. Ren, C. Lee, A hierarchical blockchain architecture for federated learning in edge computing networks, J. Supercomput. 81 (7) (2025) 1–27, http://dx.doi.org/10.1007/s11227-025-07262-2.

[53] J. Chen, Y. Shi, C. Yi, H. Du, J. Kang, D. Niyato, Generative-AI-driven human digital twin in IoT healthcare: A comprehensive survey, IEEE Internet Things J. 11 (21) (2024) 34749–34773, http://dx.doi.org/10.1109/JIOT.2024.3421918.

[54] J. Chen, C. Yi, S.D. Okegbile, J. Cai, X. Shen, Networking architecture and key supporting technologies for human digital twin in personalized healthcare: A comprehensive survey, IEEE Commun. Surv. & Tutorials 26 (1) (2024) 706–746, http://dx.doi.org/10.1109/COMST.2023.3308717.

[55] S.D. Okegbile, J. Cai, J. Wu, J. Chen, C. Yi, A prediction-enhanced physical-to-virtual twin connectivity framework for human digital twin, IEEE Trans. Cogn. Commun. Netw. 11 (4) (2025) 2440–2455, http://dx.doi.org/10.1109/TCCN.2024.3519331.

[56] J. Chen, C. Yi, H. Du, D. Niyato, J. Kang, J. Cai, X. Shen, A revolution of personalized healthcare: Enabling human digital twin with mobile AIGC, IEEE Netw. 38 (6) (2024) 234–242, http://dx.doi.org/10.1109/MNET.2024.3366560.

[57] M. Turab, S. Jamil, A comprehensive survey of digital twins in healthcare in the era of metaverse, BioMedInformatics 3 (3) (2023) 563–584, http://dx.doi.org/10.3390/biomedinformatics3030039.

[58] S.D. Okegbile, J. Cai, H. Zheng, J. Chen, C. Yi, Differentially private federated multi-task learning framework for enhancing human-to-virtual connectivity in human digital twin, IEEE J. Sel. Areas Commun. 41 (11) (2023) 3533–3547, http://dx.doi.org/10.1109/JSAC.2023.3310106.

[59] P. Ruiu, M. Nitti, V. Pilloni, M. Cadoni, E. Grosso, M. Fadda, Metaverse & human digital twin: Digital identity, biometrics, and privacy in the future virtual worlds, Multimodal Technol. Interact. 8 (6) (2024) 48, http://dx.doi.org/10.3390/mti8060048.

[60] I.S. Igboanusi, A. Allwinnaldo, R.N. Alief, M.R.R. Ansori, J.-M. Lee, D.-S. Kim, Smart AutoMmining (SAM) for industrial IoT blockchain network, IET Commun. 16 (18) (2022) 2123–2132, http://dx.doi.org/10.1049/cmu2.12465.

[61] M.A. Manolache, S. Manolache, N. Tapus, Decision making using the blockchain proof of authority consensus, Procedia Comput. Sci. 199 (2022) 580–588, http://dx.doi.org/10.1016/j.procs.2022.01.071, The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 & 2021): Developing Global Digital Economy after COVID-19.

[62] V. Buterin, Ethereum: Platform review, Oppor. Challenges Priv. Consort. Blockchains 45 (2016) 1–45.

[63] F. Mogavero, I. Visconti, A. Vitaletti, M. Zecchini, The blockchain quadrilemma: When also computational effectiveness matters, in: 2021 IEEE Symposium on Computers and Communications, ISCC, 2021, pp. 1–6, http://dx.doi.org/10.1109/ISCC53001.2021.9631511.

[64] W.-M. Lee, Testing smart contracts using ganache, in: Beginning Ethereum Smart Contracts Programming: With Examples in Python, Solidity, and JavaScript, A Press, Berkeley, CA, 2019, pp. 147–167, http://dx.doi.org/10.1007/978-1-4842-5086-0_7.

[65] S.K. Jagatheesaperumal, M. Rahouti, A. Alfatemi, N. Ghani, V.K. Quy, A. Chehri, Enabling trustworthy federated learning in industrial IoT: Bridging the gap between interpretability and robustness, IEEE Internet Things Mag. 7 (5) (2024) 38–44, http://dx.doi.org/10.1109/IOTM.001.2300274.

[66] A. Hamdi, L. Fourati, S. Ayed, Vulnerabilities and attacks assessments in blockchain 1.0, 2.0 and 3.0: Tools, analysis and countermeasures, Int. J. Inf. Secur. 23 (2) (2024) 713–757, http://dx.doi.org/10.1007/s10207-023-00765-0.