Securing AI-as-a-Service Platform through Blockchain and Smart-Contracts

Ali Nadar, Jérôme Härri
EURECOM, 450 route des Chappes, 06904 Sophia-Antipolis, France
ali.nadar, jerome.haerri@eurecom.fr

***Abstract*—Vehicular AI-as-a-Service (AIaaS) enables connected vehicles to request and receive machine learning models over lightweight publish/subscribe connectivity like MQTT. However, this raises critical concerns about security, integrity, and accountability. This paper proposes a blockchain-enabled extension to our AIaaS platform that records model requests and deliveries on an immutable ledger and binds each delivered model to an on-chain cryptographic fingerprint. The receiving AI actor verifies model integrity and provenance by matching the computed hash against the blockchain record, enabling non-repudiation and back-tracing of *who provided which model and when*. The resulting audit trail strengthens trust in MQTT-based model distribution and supports provenance-driven explainability for connected-vehicle AI.**

***Index Terms*—AI-as-a-Service, blockchain, machine learning, MQTT.**

I. Introduction

Connected and automated vehicles increasingly rely on machine learning (ML) models to perceive their environment, predict risk, and decide safe maneuvers. Because the long tail of driving contexts cannot be covered by a single monolithic model, vehicles and infrastructure must exchange and reuse specialized models tailored to local spatio-temporal conditions. This motivates *AI-as-a-Service* (AIaaS) for vehicular systems, where *AI actors* (vehicles, roadside units, or services) can request, provide, train, and consume ML models on demand.

In our previous work [1], we introduced an AIaaS platform for the *Artificial Intelligence of Things* (AIoT), termed *Infrastructure-Assisted Knowledge Management* (IAKM). IAKM combines APIs for model request/storage, publish/subscribe messaging for decoupled interactions, and semantic context descriptions to match model supply and demand, and we demonstrated contextual model exchange between distributed vehicular actors in a driving simulator.

However, scaling AI model exchange raises fundamental *trust* and *accountability* challenges (Fig. 1). In safety-critical settings, actors must ensure that received models are authentic, intact, and not replayed out of context. Beyond runtime security, post-incident analysis requires *non-repudiation* and *traceability*: stakeholders must reconstruct *who provided which model, when, and under which context*, and verify that this history cannot be altered retroactively. These needs are closely tied to *provenance-driven explainability*, i.e., explaining not only model outputs but also a model's lifecycle across actors.

Distributed Ledger Technologies (DLTs) provide immutable, verifiable records among mutually distrustful parties; blockchains such as Ethereum further support programmable trust via smart contracts. In this paper, we leverage an Ethereum-compatible blockchain as an external trust anchor for AIaaS accountability and provenance across administrative domains.

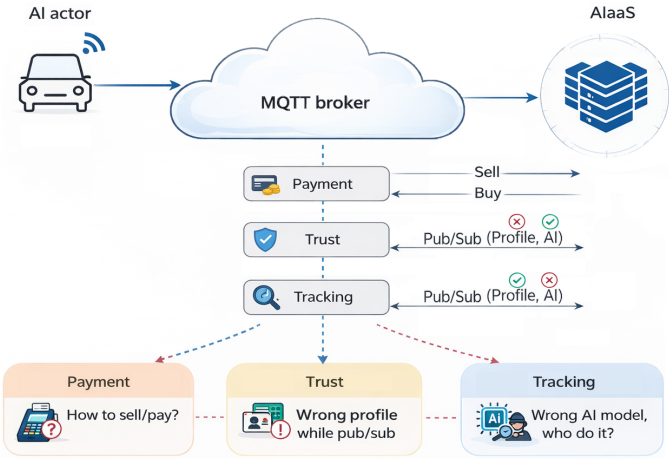


Figure 1: Challenges in AIaaS ecosystems

We integrate a blockchain-based provenance and verification layer into the AIaaS/IAKM workflow by binding each model exchange to an immutable audit trail: model requests and responses are recorded as ledger transactions, while cryptographic commitments (e.g., hashes and identifiers) enable recipients to verify that the delivered artifact matches what was announced. This yields an append-only history supporting end-to-end integrity, origin authentication/non-repudiation, and post-hoc provenance reconstruction. The integration remains lightweight, preserving MQTT-based delivery and requiring no redesign of the core platform; performance aspects are discussed via the prototype implementation.

In summary, this work makes the following contributions:

- We formalize threat and accountability requirements for vehicular AIaaS model exchange, with emphasis on authenticity, integrity, replay-resistance, and non-repudiation.
- We propose a blockchain-enabled AIaaS extension that logs model request/response events and binds model artifacts to verifiable cryptographic fingerprints.
- We provide an implementation blueprint (smart contract interface and agent-side verification flow) that integrates with the existing AIaaS connectivity layer and enables provenance-driven explainability.

The remainder of this paper is organized as follows: Section II reviews related work. Section III presents the architecture, and Section IV recalls the AIaaS platform. Section V details the blockchain layer, threat assumptions, and resulting security/accountability properties. Section VI concludes and outlines future work.

II. Literature Survey

Blockchain has been explored to support trustworthy exchange of AI/ML assets by turning trained models into tradable digital artifacts and by encoding exchange rules, settlement, and incentives in smart contracts. A representative example proposes a marketplace for trading IoT-derived AI models with contribution-aware rewarding, illustrating how on-chain logic can support *payment* and accountability for AI artifacts [2]. In parallel, a large body of work combines blockchain with federated learning (FL) in the Internet of Vehicles (IoV) to improve robustness and trust, e.g., via auditable aggregation and contribution tracking, and to mitigate poisoning and free-riding; a systematic review synthesizes the main architectures and security goals in this space [3]. While these studies are highly relevant for collaborative training, they primarily secure *training* and *update aggregation* and do not directly address the *model distribution* problem in vehicular AIaaS: verifying authenticity and provenance of a *delivered* model at the receiver and maintaining an immutable request/response history for post-hoc accountability. Beyond FL, blockchain has also been investigated for end-to-end provenance and tamper resistance in distributed ML pipelines [4], [5]. ProML proposes a decentralized provenance management platform for ML software systems that leverages blockchain and smart contracts to reduce insider tampering and strengthen accountability across pipeline stages [6]. More generally, provenance-driven *explainability* requires reliable lineage representations to answer questions such as *"who provided which model and when? "*. Complementary transparency artifacts further document intended use, evaluation, and limitations: *Model Cards* standardize reporting of model properties and performance across contexts [7], while *Datasheets for Datasets* advocate structured documentation of dataset motivation, composition, and recommended uses [8]. For vehicular AIaaS, these strands motivate coupling immutable, cryptographic provenance (non-repudiation) with optional descriptive documentation for responsible reuse. Finally, MQTT remains attractive for vehicular/IoT prototyping due to its lightweight publish/subscribe design, but practical security often relies on transport-layer protection and broker/client hardening. An empirical security assessment of common open-source MQTT brokers and clients reports deviations from standards and exploitable margins in implementations [9], underlining that channel security alone does not provide immutable accountability; recent MQTT+blockchain designs explicitly target this gap [10]–[12]. Motivated by these gaps, and building on our prior AIaaS platform architecture and experiments [1], this paper focuses on the *model delivery and consumption* stage. We deliver models off-chain (via MQTT), while the blockchain records commitments and events that enable recipient-side integrity and provenance verification. This creates an auditable trail that supports trust, traceability, and incentive-compatible settlement contrasting with prior blockchain-based AI systems that primarily secure training or aggregation. To the best of our knowledge, this is the first work to explicitly decouple AI model transport from cryptographic accountability in a vehicular AIaaS setting, enabling provenance-driven explainability at deployment time.

III. System Architecture

The proposed blockchain-secured AI-as-a-Service (AIaaS) platform extends our earlier Infrastructure-Assisted Knowledge Management (IAKM) system by integrating an immutable audit layer via a Ethereum-compatible blockchain (prototype on Ganache; target deployment can be permissioned/consortium). A permissioned blockchain is assumed to limit latency and transaction costs, with validators operated by infrastructure stakeholders (e.g., OEMs, road operators), while still ensuring immutability and non-repudiation. The architecture, as depicted on Fig. 1, is designed to ensure model integrity, provenance tracking, and non-repudiation while maintaining compatibility with lightweight vehicular communication protocols. The system comprises 2 core layers; AIaaS Application Layer and Blockchain Layer.

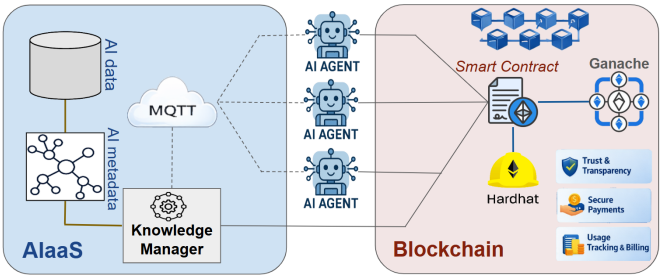


Figure 2: Secured AIaaS architecture

IV. AIaaS Application Layer

The AIaaS Application Layer represents the functional interface through which AI actors (vehicles, roadside units, or external services) interact with the platform to request, provide, or consume AI models. It leverages a lightweight publish/subscribe messaging protocol (e.g., MQTT) to support scalable and decoupled communication between actors and the central AIaaS server, while maintaining low latency suitable for vehicular environments. Key components of this layer include:

- **IAKM Agent:** Acts as a middleware for AI actors, exposing RESTful APIs to register availability, submit trained models, and search for contextually relevant models. The agent performs semantic interpretation of requests, aligns them with the IAKM ontology, and forwards them to the server via the MQTT-based message bus.
- **Knowledge Manager (KM):** Maintains a semantic knowledge graph describing the AI models, their context, capabilities, and spatio-temporal applicability. By enabling context-aware discovery, it ensures that AI models retrieved by actors are suitable for the current operational scenario.
- **AI Model Lifecycle Support:** The layer coordinates AI model registration, retrieval, and versioning. Producers can push models along with metadata describing model type, intended context, and training conditions. Consumers can request models based on the operational context and actor capabilities, while the system supports predictive caching to minimize retrieval latency.
- **Actor Role Management:** Each AI actor is assigned a role dynamically—consumer, producer, or worker—based

on its capabilities and context. For example, a vehicle currently in a training-eligible location may act as a producer, while another vehicle approaching the same context may act as a consumer. This dynamic role assignment aligns with AIoT principles where AI resources are orchestrated in space and time.

- **Security and Provenance Hooks:** While the AIaaS layer primarily handles communication and knowledge management, it is designed to integrate with the blockchain layer for model integrity and provenance verification. Each request and delivery is associated with a cryptographic fingerprint, enabling end-to-end verification and non-repudiation without altering the underlying lightweight MQTT communication.

By combining semantic context awareness, dynamic role management, and lightweight communication, the AIaaS Application Layer operationalizes the AI-as-a-Service paradigm: models can be discovered, shared, and consumed on demand, moving beyond the traditional "AI-as-a-Duty" approach toward a collaborative and service-oriented AI ecosystem.

V. Blockchain Layer

The blockchain layer acts as an external security and accountability layer between the AIaaS core and the distributed AI agents. Its role is to provide immutable logging of AI model lifecycle events, including model requests, deliveries, and publications, without interfering with the data plane used for model transport. To this end, a smart contract records these events as tamper-resistant ledger entries, enabling verifiable tracking of model provenance and integrity.

AI agents interact with the blockchain layer to register requests and responses and to retrieve cryptographic commitments associated with delivered models. Upon reception of a model via MQTT, the agent locally verifies its integrity by matching the computed hash against the on-chain record. This design yields an immutable audit trail that supports non-repudiation, traceability, and provenance-driven explainability, while preserving the lightweight nature of publish/subscribe communication.

The prototype implementation relies on a permissioned Ethereum-compatible blockchain deployed locally using Ganache[1], a local Ethereum blockchain emulator for smart contract development and testing. The AIaaS agent, MQTT broker, and blockchain node were executed on a workstation equipped with an Intel i7 CPU, 32 GB of RAM, and Ubuntu 22.04. Smart contracts were implemented in Solidity and deployed using the Hardhat[2] development framework.

1) End-to-End Model Exchange Workflow: Figures 3 illustrates the sequence of interactions involved in securely pushing an AI model to the AIaaS platform. The diagram emphasizes the decoupling between off-chain model transport via MQTT and on-chain accountability, where the model provider first commits a cryptographic fingerprint of the model to the blockchain before or alongside its publication. Figures 4 further shows how the immutable request–response history maintained on the blockchain enables provenance-driven explainability allowing an operator to later verify which provider supplied a model involved in a specific driving decision. By

[1]https://archive.trufflesuite.com/ganache

[2]https://hardhat.org/

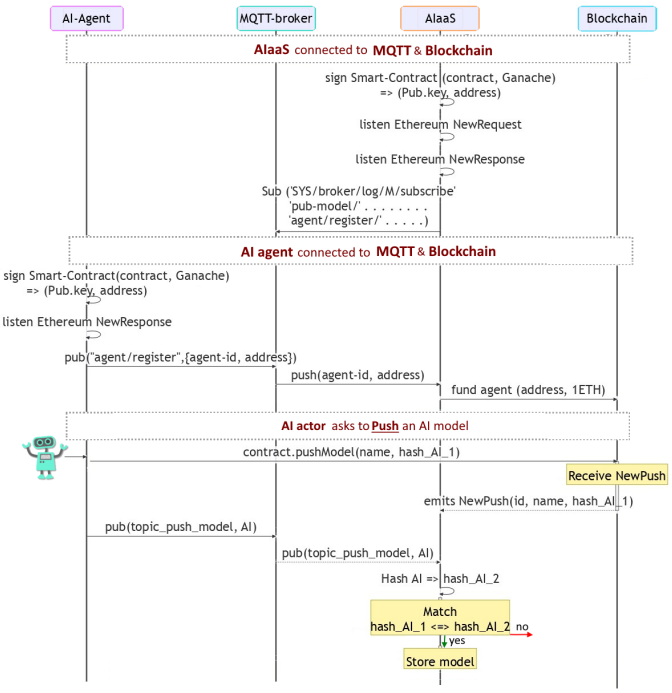


Figure 3: Sequence of messages on Blockchain-applied pushing of AI model to AIaaS platform

linking each delivered model to a specific request identifier and provider address, the agent can later reconstruct the origin and delivery context of any deployed model.

Together, the sequence diagrams in Figures 3 and 4 operationalize the security guarantees summarized in Table I. While the sequence diagrams detail the runtime enforcement of integrity and provenance during model push and retrieval, the threat mapping table provides a static view of how these mechanisms collectively address the identified attack surfaces.

The interaction between the AIaaS application layer and the blockchain layer follows a well-defined request–response workflow that decouples model transport from model accountability. When an AI actor requires a machine learning model adapted to a specific operational context (e.g., road topology, traffic density, or weather conditions), it issues a model request through its local agent. The request is forwarded to the AIaaS core using MQTT and REST interfaces, where the Knowledge Manager evaluates semantic compatibility against available models and providers. In parallel, the agent submits a corresponding transaction to the blockchain smart contract, registering the request as an immutable on-chain event. This event includes a unique request identifier, the requester address, and a semantic descriptor of the requested model.

Model providers, which may be infrastructure nodes or other AI actors, monitor the blockchain for new requests matching their capabilities. Upon selecting a suitable model, a provider responds by submitting a transaction to the smart contract that binds the request identifier to a cryptographic commitment of the model artifact (e.g., a SHA-256 hash). This on-chain response establishes an immutable association between the requester, the provider, the request, and the announced model fingerprint.

The actual model file is then delivered off-chain via MQTT, preserving bandwidth efficiency and latency requirements crit-

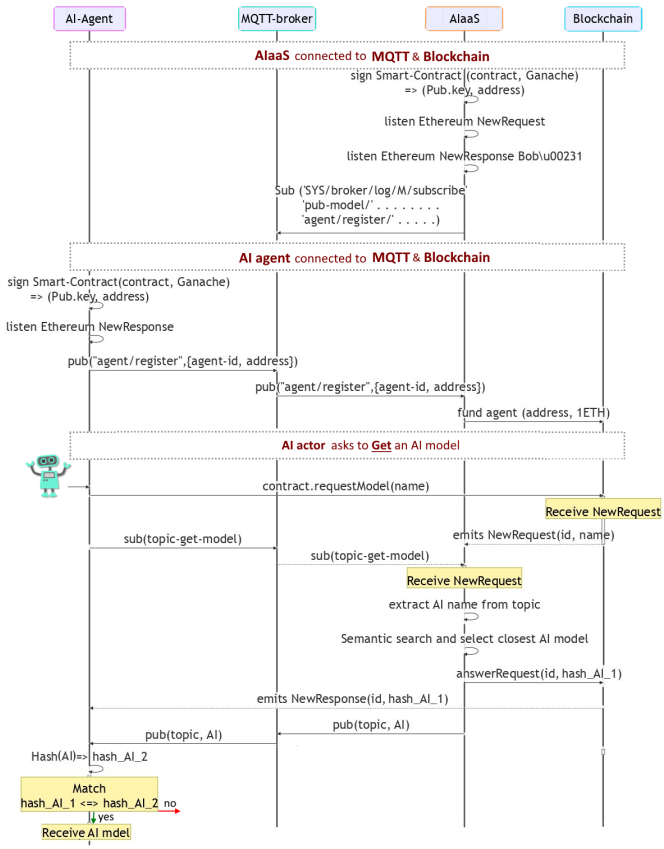


Figure 4: Sequence of messages on Blockchain-applied getting AI model from AIaaS platform

ical for vehicular systems. Upon reception, the requesting agent locally computes the cryptographic hash of the received model and verifies it against the on-chain record. Only if this verification succeeds is the model accepted and deployed by the AI actor; otherwise, the model is discarded and not exposed to downstream applications or control logic. This separation of concerns ensures that blockchain interaction remains lightweight while still enforcing strong integrity and provenance guarantees.

In our prototype using Ganache, blockchain-related operations add negligible overhead compared to model transfer latency. For a 10 MB model, SHA-256 hashing, request registration, block confirmation, and end-to-end verification each complete within 0–30 ms, whereas MQTT-based model transfer requires 500–1000 ms. Measurements were obtained on a single-node Ganache instance with automatic mining enabled over 50 runs (reported as mean $\pm$ standard deviation). Overall, accountability enforcement remains orders of magnitude faster than the data plane and does not impact model delivery performance.

2) Smart Contract Design and Auditability Guarantees: The smart contract acts as a trust anchor for AI model exchange, providing immutable logging rather than heavy on-chain computation or storage. Its primary role is to record events corresponding to model lifecycle actions—namely, model requests, responses, and optional model publications—without embedding the model data itself. This design choice avoids scalability bottlenecks and aligns with best practices for blockchain-based provenance systems.

Each request and response is indexed by a unique identifier, enabling deterministic linkage between off-chain model transfers and on-chain commitments. Because blockchain transactions are signed by their originators, the resulting ledger provides non-repudiable evidence of who requested or supplied a given model and when. In the event of a malfunction, incident, or dispute, stakeholders can reconstruct the exact sequence of model exchanges and verify that no tampering or substitution occurred after the fact. Importantly, the blockchain layer is agnostic to the internal semantics of the AI models themselves. It does not interpret model parameters or behavior, but instead guarantees the integrity and traceability of the artifacts exchanged. This makes the approach compatible with heterogeneous model types, training pipelines, and update strategies, while still enabling provenance-driven explainability at the system level. Smart contracts were implemented in Solidity and deployed using the Hardhat development framework on top of a local Ganache blockchain.

3) Security and Trust Properties by Design: By construction, the proposed architecture enforces several critical security and accountability properties:

- Integrity - Any modification of a model during transport is detected through cryptographic hash mismatch at the receiver.
- Authenticity - Blockchain identities bind requests and responses to verifiable entities, ensuring that model origin can be authenticated.
- Non-repudiation - Providers cannot deny having delivered a specific model once the corresponding transaction is recorded on-chain.
- Replay resistance - Each request is uniquely identified, preventing reuse of stale model announcements in new contexts.
- Auditability and explainability - The immutable ledger supports post-hoc analysis of model provenance, enabling stakeholders to trace deployed models back to their source and delivery time.

These guarantees complement, rather than replace, traditional communication security mechanisms such as TLS-protected MQTT channels. While transport-layer security protects data in transit, the blockchain layer provides persistent accountability, which remains verifiable long after the communication session has ended.

4) Threat Model Mapping: The proposed architecture directly addresses the threat and accountability requirements identified in Section I by mapping each threat class to a specific architectural control. Model tampering during delivery, a risk inherent to lightweight publish/subscribe systems, is mitigated by binding each off-chain model transfer to an on-chain cryptographic fingerprint; any modification of the model payload results in a hash mismatch at the receiving agent and leads to rejection. Model impersonation and unauthorized provisioning are countered through blockchain-based identity binding: only entities controlling a valid blockchain account can register requests or publish responses, and all actions are permanently attributed to their signers.

Replay and context-mismatch attacks, where a previously valid model is reused outside its intended operational context, are addressed by uniquely identifying each model request on-chain and linking responses to a specific request identifier

Table I: Threat Model Mapping in the Blockchain-Secured AIaaS Architecture

Threat / Attack Surface	**Architectural Mitigation**
Model tampering during delivery	Cryptographic hash of each model is recorded on-chain; receiving agent recomputes the hash and verifies integrity before deployment.
MQTT publish/subscribe channel	
Model impersonation / unauthorized provider	Smart-contract–mediated responses are signed by blockchain identities, binding each delivered model to a verifiable provider address.
AIaaS server or malicious actor	
Replay of stale or out-of-context models	Each model response is linked to a unique on-chain request identifier and semantic descriptor, preventing reuse across contexts.
Reuse of previously valid model artifacts	
Non-repudiation failure	Immutable blockchain ledger records who requested and who delivered each model, with timestamps, ensuring accountability.
Dispute after malfunction or incident	
Insider log manipulation	Audit trail is externalized to the blockchain layer, preventing retrospective modification or deletion of request/response records.
Centralized AIaaS server or database	
Lack of provenance explainability	On-chain request–response history enables reliable reconstruction of model provenance, supporting explainability beyond outputs.
Post-hoc analysis and certification	

and semantic description. This prevents the reuse of stale announcements and enables agents to verify that a delivered model corresponds to their own request. Non-repudiation and post-incident accountability, which are not achievable through transport-layer security alone, are ensured by the immutable blockchain ledger: once a request or response is recorded, neither the requester nor the provider can deny their participation or alter the historical record.

Finally, insider threats and retrospective tampering, such as modification or deletion of server-side logs, are mitigated by externalizing the audit trail to the blockchain layer, which operates independently of the AIaaS core services. As a result, even if the AIaaS server or MQTT broker is compromised, the integrity of the request–response history and model provenance remains verifiable. This explicit mapping between threats and architectural components demonstrates that the proposed design provides security and trust guarantees that are complementary to, and stronger than, traditional centralized AIaaS deployments.

To make the security implications of the proposed design explicit, Table I maps the main threat classes arising in AIaaS-based model exchange to the corresponding architectural mitigation mechanisms. This mapping highlights how blockchain-backed commitments complement lightweight MQTT-based delivery by enforcing integrity, provenance, and accountability guarantees that cannot be achieved through transport-layer protection alone.

VI. Conclusions

This paper presents a DLT-secured extension of our vehicular AIaaS platform that decouples MQTT-based model delivery from on-chain accountability. By logging requests and responses in a smart contract and anchoring model fingerprints on-chain, the system provides integrity verification, provenance tracking, and non-repudiation without altering model semantics or the core AIaaS workflow.

A Ganache-based prototype indicates only minor overhead: contract interactions and verification complete within milliseconds, while MQTT transfer dominates end-to-end latency. Future work will study scalability and extend the platform toward incentive-compatible AIaaS by embedding pricing, payment, and access control directly into smart contracts. The full implementation is available as open source on EURECOM GitLab: https://gitlab.eurecom.fr/a-team/aiaas-secured

VII. Acknowledgements

This work has benefited from a government grant managed by the "*Agence Nationale de la Recherche*" under the France 2030 program with NF-FITNESS, grant "ANR-22-PEFT-0007".

References

[1] A. Nadar and J. Härri, "An ai-as-a-service platform for an artificial intelligence of things (aiot)," in *IEEE Vehicular Technology Conference (VTC)*, 2024, final version provided by the authors.

[2] L. D. Nguyen *et al.*, "A marketplace for trading ai models based on blockchain and incentives for iot data," *CoRR*, vol. abs/2112.02870, 2021.

[3] M. Billah *et al.*, "A systematic literature review on blockchain enabled federated learning framework for internet of vehicles," *CoRR*, vol. abs/2203.05192, 2022.

[4] M. Spoczynski, M. S. Melara, and S. Szyller, "Atlas: A framework for ml lifecycle provenance transparency," 2025. [Online]. Available: https://arxiv.org/abs/2502.19567

[5] A. Jalbani, R. Weerawarna, and K. Al-Zubaidi, "Enhancing data provenance in ai with blockchain technology: a comprehensive quality model," *CSI Transactions on ICT*, vol. 13, no. 2–3, pp. 213–224, 2025.

[6] N. K. Tran *et al.*, "Proml: A decentralised platform for provenance management of machine learning software systems," *CoRR*, vol. abs/2206.10110, 2022.

[7] M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji, and T. Gebru, "Model cards for model reporting," *CoRR*, vol. abs/1810.03993, 2018.

[8] T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. M. Wallach, H. D. III, and K. Crawford, "Datasheets for datasets," *Commun. ACM*, vol. 64, no. 12, pp. 86–92, 2021. [Online]. Available: https://doi.org/10.1145/3458723

[9] E. D. Paolo, E. Bassetti, and A. Spognardi, "Security assessment of common open source mqtt brokers and clients," *CoRR*, vol. abs/2309.03547, 2023.

[10] S. Lazzaro and F. Buccafurri, "Achieving accountability and data integrity in message queuing telemetry transport using blockchain and interplanetary file system," *Future Internet*, vol. 16, no. 7, 2024. [Online]. Available: https://www.mdpi.com/1999-5903/16/7/246

[11] P. S. Akshatha and S. M. D. Kumar, "Mqtt and blockchain sharding: An approach to user-controlled data access with improved security and efficiency," *Blockchain: Research and Applications*, vol. 4, no. 4, p. 100158, 2023.

[12] T.-C. Hsu and H.-S. Lu, "Designing a secure and scalable service model using blockchain and mqtt for iot devices," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, 2024, pp. 645–653.