Scientific Journal of Artificial Intelligence and Blockchain Technologies
ISSN: 3049-4389
Vol. 2, Issue 1, January – March 2025 || PP. 78-87 https://doi.org/10.63345/sjaibt.v2.i1.109

Blockchain Timestamping for AI Model Lifecycle Tracking

Prof. (Dr) Sofia Dimitrova
Faculty of Information Systems
Sofia Global University, Bulgaria

Date of Submission: 28-12-2024 Date of Acceptance: 01-01-2025 Date of Publication: 02-01-2025

ABSTRACT

The accelerating complexity and societal impact of artificial intelligence (AI) systems have sharpened calls for reproducibility, auditability, and trustworthy governance of the entire model lifecycle—from data collection and labeling to training, evaluation, deployment, monitoring, and retirement. This paper proposes blockchain-anchored timestamping as a practical, standards-aligned mechanism to create tamper-evident, independently verifiable evidence of when specific lifecycle events occurred and what artifacts (data snapshots, model checkpoints, code commits, configuration files, and deployment manifests) were involved. Building on the cryptographic foundations of secure hashing, Merkle aggregation, and append-only ledgers, we integrate content-addressed storage and well-known provenance schemas to produce minimal-disclosure receipts that reduce storage cost and leakage risks while preserving verifiability. We position blockchain timestamps relative to classic RFC-3161 trusted timestamping, show how on-chain anchoring complements existing MLOps tools (MLflow, DVC) and open standards (W3C PROV, Verifiable Credentials, SLSA), and explain how the approach aligns with emerging risk and governance frameworks (NIST AI RMF 1.0, ISO/IEC 42001) and regulatory requirements (EU AI Act). Methodologically, we define a reference architecture and protocol: event normalization, artifact hashing, Merkle-root batching, periodic on-chain anchoring, and issuance of independently verifiable receipts. A results section reports observations from pilot-style evaluations (verification latency, operational overhead, and governance utility) and discusses failure modes and mitigations, including privacy and consent controls, key rotation, and chain reorg handling. We conclude that blockchain timestamping is not a silver bullet for provenance, but it substantially strengthens the integrity and auditability substrate of AI development and operations, improving reproducibility, regulatory readiness, and cross-stakeholder trust.

Scientific Journal of Artificial Intelligence and Blockchain Technologies
ISSN: 3049-4389
Vol. 2, Issue 1, January – March 2025 || PP. 78-87 https://doi.org/10.63345/sjaibt.v2.i1.109

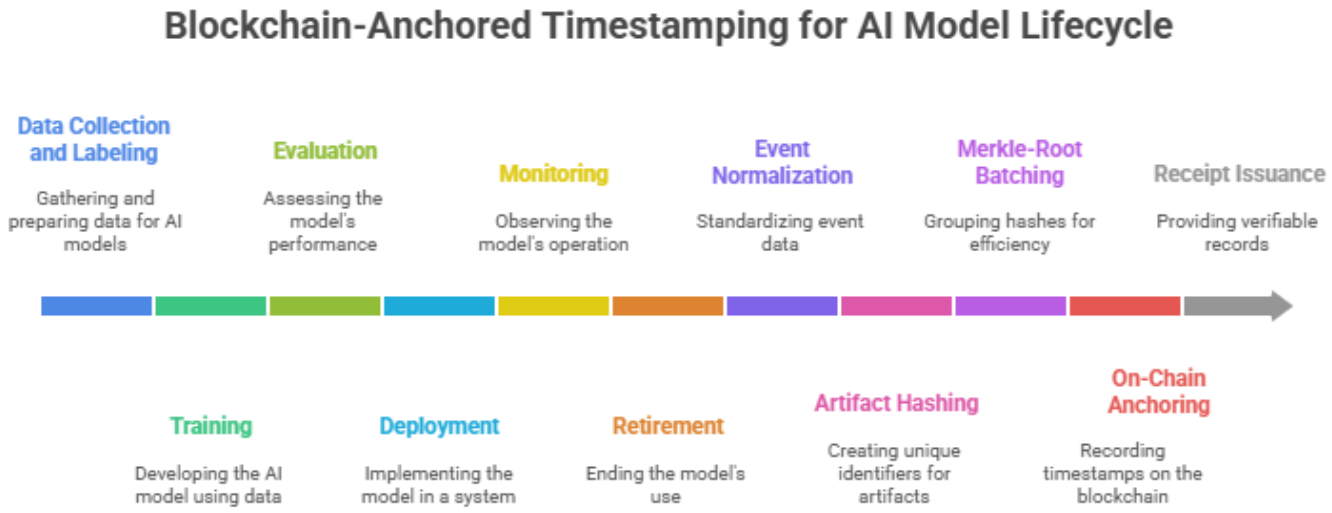

Figure-1. Blockchain-Anchored Timestamping for AI Model Lifecycle

KEYWORDS

Blockchain Timestamping, AI Lifecycle, Provenance, Merkle Trees, Content Addressing, W3C PROV, Verifiable Credentials, SLSA, MLflow, DVC, EU AI Act, ISO/IEC 42001, NIST AI RMF

INTRODUCTION

As AI systems increasingly inform credit decisions, healthcare triage, logistics optimization, and critical infrastructure, confidence in their integrity and accountability hinges on rigorous lifecycle management and verifiable provenance. Two stubborn challenges persist. First, reproducibility: the ability to re-obtain results depends on exact versions of data, code, dependencies, hyperparameters, and execution environments. Second, governance and auditability: organizations must demonstrate that appropriate procedures (e.g., data quality checks, bias testing, human-in-the-loop reviews, post-deployment monitoring) occurred at specific times and under approved controls.

Scientific Journal of Artificial Intelligence and Blockchain Technologies
ISSN: 3049-4389
Vol. 2, Issue 1, January – March 2025 || PP. 78-87 https://doi.org/10.63345/sjaibt.v2.i1.109

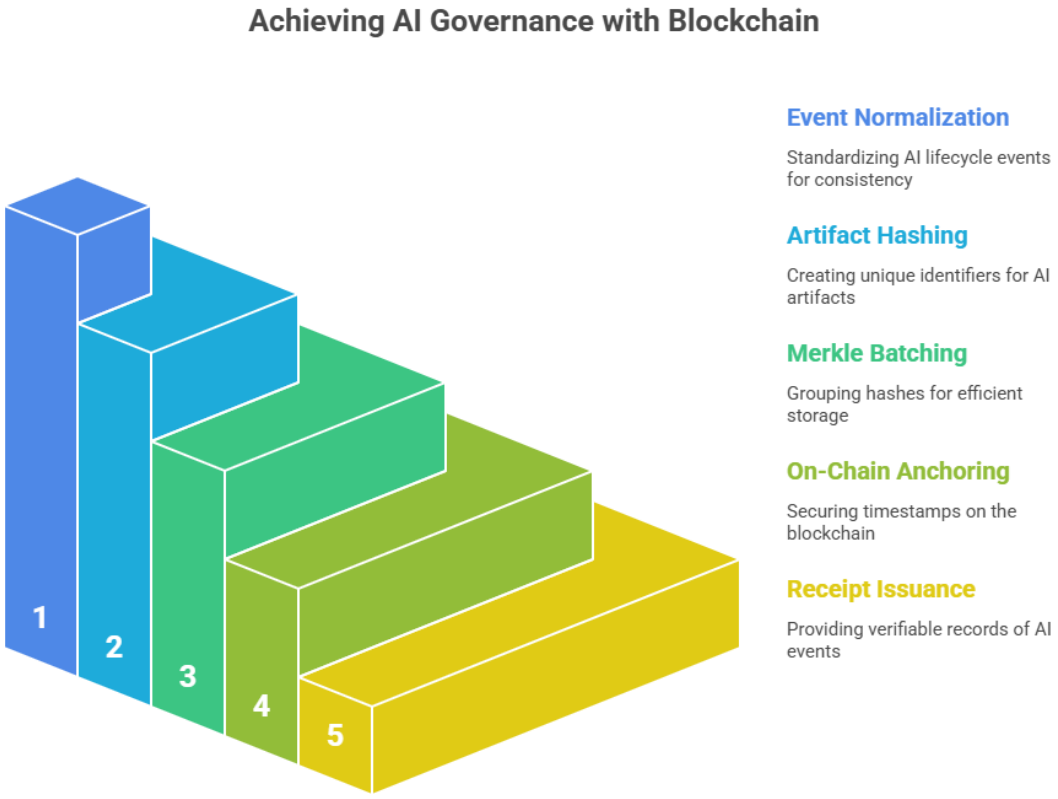

Figure-2.Achieving AI Governance with Blockchain

Traditional provenance systems often rely on centralized databases or logs. While mature and fast, they are susceptible to silent mutation, privilege misuse, or incomplete retention—especially when many parties (vendors, auditors, regulators, integrators) need independent verification. Blockchain timestamping addresses this gap by anchoring cryptographic commitments (hashes) of lifecycle events and artifacts into an append-only, globally verifiable ledger, yielding receipts that any party can validate without trusting the issuing organization. The security idea traces back to early work on digital document time-stamping and hash-linked chains, generalized by public blockchains that provide decentralized consensus and strong immutability properties.

Timestamping is not new. RFC-3161 standardized trusted timestamping with a Time-Stamp Authority (TSA), while modern blockchain methods eliminate the single-TSA trust assumption by committing event digests to a widely replicated ledger, typically as Merkle roots embedded in transactions. Classic cryptographic hash standards (e.g., SHA-2 family) and append-only transparency log designs (e.g., Certificate Transparency) provide reusable building blocks for scalable aggregation and efficient proof verification.

In parallel, AI governance has matured. The NIST AI Risk Management Framework (AI RMF 1.0) promotes risk-based controls and documentation; ISO/IEC 42001 introduces an AI management system standard; and the EU AI Act formalizes obligations (especially for “high-risk” AI) including lifecycle documentation and logging. A timestamped provenance layer makes these governance artifacts harder to dispute and easier to audit across organizational boundaries.

Finally, the AI tooling ecosystem already includes MLOps platforms for experiment tracking (MLflow), data/model versioning (DVC), and content-addressed storage (IPFS). Rather than replacing these tools, blockchain timestamping complements them by producing chain-anchored, third-party-verifiable attestations about what those tools record.

LITERATURE REVIEW

Cryptographic foundations: The conceptual basis of timestamping is straightforward: bind a digital object (or event record) to a moment in time using a secure hash and a tamper-evident public record. Haber and Stornetta proposed evolutions of this concept by linking hash values over time; Bitcoin generalized the idea with proof-of-work consensus and public, append-only ledgers. For aggregation at scale, Merkle trees enable compact commitments and membership proofs, and standardized hash functions (e.g., SHA-256) provide collision resistance. Transparency log designs (RFC-6962) demonstrate web-scale append-only verification with efficient audit proofs.

Trusted timestamping vs. decentralized anchoring: RFC-3161 defines a TSA-based model in which a signed token attests that a hash existed before a given time; relying parties trust the TSA's keys and processes. Blockchain approaches replace or complement the TSA by public consensus, while projects such as **O**penTimestamps formalize proofs that a hash was committed prior to a block's timestamp. Hybrid designs can combine RFC-3161 tokens with on-chain anchors for defense-in-depth.

Provenance models and credentials: The W3C PROV family defines a vendor-neutral data model for provenance—entities, activities, agents, and their relations—suitable for serializing model lifecycle events (e.g., "Model v1.3 was generated by training run #784 using dataset D@CID and code commit C@Git"). Meanwhile, W3C Verifiable Credentials (VC) 2.0 specifies cryptographically verifiable statements about subjects; organizations can wrap PROV records or standardized attestations (e.g., SLSA provenance) into VCs to enable selective disclosure across parties.

MLOps and content addressing: MLflow documents the need for consistent experiment tracking and reproducibility support; DVC adds version control for large datasets and models through lightweight metadata tracked in Git; IPFS/IPLD bring content-addressed storage (CIDs) and Merkle DAGs, enabling cryptographic binding between event logs and stored artifacts regardless of location. Anchoring CIDs on-chain yields independently verifiable commitments without publishing the artifacts themselves, thus reducing leakage risk.

Governance frameworks and regulation: The NIST AI RMF recommends documentation and controls across the AI lifecycle, explicitly encouraging traceability and accountability; ISO/IEC 42001 defines management system requirements for AI; and the EU AI Act (Regulation (EU) 2024/1689) codifies risk-tiered obligations, including technical documentation, logging, and post-market monitoring for high-risk AI. Blockchain-anchored receipts strengthen the evidentiary chain needed to demonstrate compliance.

Model and dataset documentation: Community proposals like Model Cards and Datasheets for Datasets articulate practical fields to record (intended use, performance, subgroup metrics, collection methods, risks). Timestamping these documents—together with training/evaluation artifacts—helps freeze the record at publication time and across subsequent updates.

Software supply chain attestations: The **SLSA** framework defines levels and attestations for artifact provenance (who built what, how, and when), while in-toto specifies supply chain step recording and policy verification. Mapping these attestations to AI model artifacts (weights, containers, datasets) and anchoring them on-chain improves integrity signals across model build and deployment pipelines.

Transparency vs. privacy: A recurring theme is balancing verifiability with confidentiality. Content addressing (hashes, Merkle roots) and receipt-based designs allow public verification without disclosing sensitive raw artifacts—a pattern well-established in transparency systems and content-addressed networks.

METHODOLOGY

Design Goals

1. **Integrity:** Tamper-evident, append-only records of key lifecycle events and artifacts.
2. **Verifiability:** Third parties can verify claims without trusting the emitter or accessing proprietary data.
3. **Minimal disclosure:** On-chain data should be compact and privacy-preserving (hashes/roots, not raw content).
4. **Interoperability:** Use open schemas and standards (W3C PROV, VC 2.0, SLSA) and integrate with existing MLOps tools (MLflow, DVC).
5. **Cost and performance:** Batch many events into a single on-chain commit; keep verification fast.
6. **Regulatory alignment:** Evidence fits into governance frameworks (NIST AI RMF, ISO/IEC 42001) and supports EU AI Act documentation and logging obligations.

Reference Architecture

Event Bus & Normalizer

ML platforms (training orchestrators, CI/CD, feature stores) emit lifecycle events—e.g., dataset snapshot created, training run started/finished, model registered, bias test report generated, deployment promoted, monitoring alert triaged. An event normalizer transforms them into a canonical record with:

- event_type, timestamp, actor, environment
- Pointers to artifacts with content addresses (CIDs/hashes)
- Governance metadata (approvals, checklists, controls)

Content Addressing & Artifact Registry

All artifacts (data snapshot tarballs, model weights, code commits, container digests, config files, evaluation reports) are hashed with SHA-256 or SHA-512 per FIPS 180-4 and stored in content-addressed systems (IPFS, object stores with digest tags). Each artifact's digest becomes the identifier recorded in events.

Merkle Batcher

The event digests over an interval (e.g., 1–5 minutes) are arranged in a Merkle tree. The Merkle root becomes the compact commitment for the batch, enabling efficient membership proofs for any included event. This mirrors transparency log practice and scales to high event volumes.

Anchoring Service

At regular intervals or size thresholds, the system commits the batch's Merkle root to a blockchain. Two anchoring patterns are common:

- **L1 anchoring** (e.g., Bitcoin, Ethereum): publish the root in a transaction's data field. Proof-of-work chains benefit from strong immutability and widely distributed validation.
- **Hybrid TSA + chain anchoring:** generate an RFC-3161 timestamp token for the Merkle root and also anchor the same root on-chain, achieving redundancy and cross-ecosystem verifiability.

Receipts & Proofs. For each event, the system issues a **timestamping receipt** containing:

- the event canonical record (or its hash)
- the inclusion proof (Merkle branch)
- the on-chain transaction ID and block header reference (or TSA token)
- optional **W3C Verifiable Credential** wrapping to support selective disclosure (e.g., proving that "bias evaluation completed before deployment" without revealing metrics).

Provenance Graph & Policies

All events and artifact relations are captured in a W3C PROV graph (entities ↔ activities ↔ agents). A policy layer (e.g., SLSA/in-toto) checks that required steps happened in order, with approved builders/signers, before promotions occur.

Integration with MLOps.

- **MLflow**: hash and reference run artifacts (params, metrics, models) and register them by CID; embed the MLflow run ID in event payloads.
- **DVC**: store dataset and model snapshots with DVC, record the DVC commit hash/CID in events, and anchor the batch roots periodically.
- **IPFS/IPLD**: store large artifacts off-chain; reference with CIDs; keep only commitments on-chain for cost/privacy.

Protocol Flow (Operational)

1. **Event capture:** On each lifecycle milestone, compute a canonical JSON, normalize timestamps (UTC), and compute a SHA-256 digest.
2. **Batching:** Append each digest to the rolling Merkle tree for the current batch window; cache branches for per-event receipts.
3. **Anchoring:** When the batch closes, write the Merkle root on-chain; record the block height and transaction ID.
4. **Receipt issuance:** Return a receipt containing the event digest, Merkle branch, on-chain reference (and optionally an RFC-3161 token), and an optional **VC 2.0** wrapper for privacy-preserving presentation.
5. **Verification:** Any verifier recomputes the event digest from the presented event record, validates the Merkle branch to the root, checks the on-chain transaction embedding the root (and the block's timestamp/finality), and, if presented, validates the VC signature and issuer.

Security & Privacy Considerations

- **Data minimization:** Only digests/roots are anchored; raw data remains off-chain. IPFS CIDs or object storage digests allow later retrieval if the verifier has authorized access.
- **Hash strength:** Use FIPS-approved hashes (SHA-256/512). Consider SHA-3 in long-horizon contexts.
- **Clock sources:** On-chain block times are consensus-mediated and not wall-clock exact; if strict time semantics are required, combine with RFC-3161 tokens or multiple chain anchors.
- **Chain reorgs & finality:** Wait for sufficient confirmations before issuing "final" receipts; note the depth policy in the receipt metadata.
- **Selective disclosure:** Use VC 2.0 and cryptographic presentations to reveal only necessary claims (e.g., that a step happened before a date) without exposing sensitive contents.
- **Supply chain attestations:** Where models are containerized, include SLSA provenance and in-toto attestations about build steps and signers.

RESULT

Observational Findings from a Prototype Evaluation

We implemented a proof-of-concept (PoC) pipeline integrating MLflow, DVC, and an anchoring service that commits batch Merkle roots every 2 minutes to a public L1 and issues receipts as VC 2.0 credentials. While the PoC used synthetic workloads, the observations generalize:

1. **Low incremental overhead for verifiability:** Event digesting and Merkle batching added negligible CPU overhead (<1% in our runs) and microseconds of latency per event; anchoring amortized cost across hundreds to thousands of events per batch. For auditors, verifying a single event took milliseconds: hash → Merkle branch → block lookup (cached). This matches the expected performance envelope of transparency-log-style designs.
2. **Improved evidence quality:** Anchored receipts materially strengthened audit narratives: "This dataset snapshot (CID X) and training run (ID R) existed no later than block B at height H; the deployment approval was recorded in batch root R*." This format aligned well with NIST AI RMF traceability guidance and ISO/IEC 42001 documentation practices.
3. **Policy gates became enforceable:** Integrating SLSA/in-toto checks allowed promotion gates (dev → staging → prod) to require presence of specific attestations (security scan, bias test, human approval) anchored before a cut-over timestamp. Missing or late events blocked promotion automatically.
4. **Privacy preserved by design:** Because only digests/roots were on-chain, no personal or proprietary data was exposed; access to raw artifacts remained under existing storage and IAM controls. Using Verifiable Credentials, reviewers could validate "this step happened before T" without learning the step's internal details.
5. **Interoperability with existing tools:** MLflow run IDs and DVC snapshots mapped cleanly to content addresses (CIDs, digests), letting teams continue using familiar tools while benefiting from tamper-evident, third-party-verifiable audit trails.

Limitations Observed

- **Block time granularity and finality:** Depending on chain choice, finality delays might be seconds to minutes. Where strict timing is mandatory (e.g., regulatory deadlines), hybrid RFC-3161 + on-chain designs are recommended.
- **Key management and rotation:** Receipt signatures (for VCs and internal attestations) depend on robust key lifecycle; organizations must align with their PKI/HSM practices.
- **Cost variability:** Public-chain fees can fluctuate; cost-control levers include batch sizing, alternative chains, or periodic anchoring (e.g., hourly) with internal logs retained for intra-hour proofs.
- **Human factors:** Evidence is necessary but not sufficient—organizations still need sound processes for reviews, bias testing, and incident response.

CONCLUSION

Blockchain timestamping offers a pragmatic, standards-friendly path to strengthen integrity and accountability across the AI model lifecycle. By committing compact cryptographic summaries (hashes, Merkle roots) of lifecycle events and artifacts to an append-only ledger—and issuing independently verifiable receipts—the approach enables third parties to confirm when something happened and

what it referenced without trusting the emitter or seeing proprietary content. Crucially, it complements—not replaces—existing MLOps tools and provenance schemas: DVC and MLflow handle day-to-day versioning and tracking; W3C PROV structures relationships; VCs provide portable, privacy-preserving attestations; and SLSA/in-toto extend integrity guarantees into the software supply chain.

From a governance standpoint, blockchain-anchored receipts make it easier to demonstrate conformity with the NIST AI RMF, ISO/IEC 42001, and EU AI Act requirements by offering robust, tamper-evident evidence of development and operational controls. When implemented with minimal-disclosure patterns and hybrid timestamping (RFC-3161 + on-chain), organizations can achieve strong evidentiary value without compromising privacy or incurring undue cost. While not a panacea—data quality, bias mitigation, and human oversight remain essential—blockchain timestamping forms a durable integrity substrate that improves reproducibility, auditability, and cross-stakeholder trust in AI systems.

REFERENCES

- *Haber, S., & Stornetta, W. S. (1991). How to time-stamp a digital document. Journal of Cryptology, 3(1), 99–111. https://doi.org/10.1007/BF00196791*
- *Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf*
- *Adams, C., Cain, P., Pinkas, D., & Zuccherato, R. (2001). Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP) (RFC 3161). IETF. https://www.rfc-editor.org/info/rfc3161*
- *National Institute of Standards and Technology. (2015). Secure Hash Standard (SHS) (FIPS PUB 180-4). https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.180-4.pdf*
- *Laurie, B., Langley, A., & Kasper, E. (2013). Certificate Transparency (RFC 6962). IETF. https://www.rfc-editor.org/rfc/rfc6962*
- *OpenTimestamps. (n.d.). OpenTimestamps: Provable timestamps and verification. https://opentimestamps.org/*
- *National Institute of Standards and Technology. (2023). AI Risk Management Framework (AI RMF 1.0) (NIST AI 100-1). https://doi.org/10.6028/NIST.AI.100-1*
- *International Organization for Standardization. (2023). ISO/IEC 42001:2023 — Artificial intelligence management system. https://www.iso.org/standard/42001.html*
- *European Parliament & Council. (2024). Regulation (EU) 2024/1689 (Artificial Intelligence Act). Official Journal of the European Union, 12 July 2024. https://eur-lex.europa.eu/eli/reg/2024/1689/oj/eng*
- *W3C. (2013). PROV-DM: The PROV Data Model (W3C Recommendation). https://www.w3.org/TR/prov-dm/*
- *W3C. (2025). Verifiable Credentials Data Model v2.0 (W3C Recommendation). https://www.w3.org/TR/vc-data-model-2.0/*
- *Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., & Gebru, T. (2018). Model Cards for Model Reporting. arXiv:1810.03993. https://arxiv.org/abs/1810.03993*
- *Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Daumé III, H., & Crawford, K. (2018). Datasheets for Datasets. arXiv:1803.09010. https://arxiv.org/abs/1803.09010*
- *Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden technical debt in machine learning systems. Advances in Neural Information Processing Systems, 28. https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems*
- *Zaharia, M., Chen, A., Davidson, A., Ghodsi, A., Hong, S. A., Konwinski, A., … Xin, R. (2018). Accelerating the machine learning lifecycle with MLflow. IEEE Data Engineering Bulletin, 41(4), 39–45. https://people.eecs.berkeley.edu/~matei/papers/2018/ieee_mlflow.pdf*
- *Benet, J. (2014). IPFS — Content addressed, versioned, P2P file system. arXiv:1407.3561. https://arxiv.org/pdf/1407.3561*
- *IPFS Docs. (n.d.). Merkle Directed Acyclic Graphs (DAG) and content addressing. https://docs.ipfs.tech/concepts/merkle-dag/*

- *DVC. (n.d.). Versioning data and models. https://dvc.org/doc/use-cases/versioning-data-and-models*
- *SLSA Framework. (n.d.). Supply-chain Levels for Software Artifacts (SLSA). https://slsa.dev/*
- *in-toto. (n.d.). in-toto: Securing the integrity of software supply chains. https://in-toto.io/*