Constructing AI Accountability with Decision Bills of Materials

Hanwei Zhang, Andreas Schmidt, Sarah Sterz, Holger Hermanns (Universität des Saarlandes)
Julius Wenzel, Christof Fetzer (Technische Universität Dresden)

DBOM White Paper V0.5

AI nowadays takes over ever more critical decisions across everyday life–from healthcare to finance–making accountability, transparency, and trust urgent priorities. Regulations such as the EU AI Act increasingly require organisations to ensure not only performance and safety, but also full traceability of decisions and clear attribution of responsibility. Yet in practice, this is extremely difficult. Modern AI systems are built and operated by multiple actors, and their probabilistic nature obscures root causes of failures. As a result, when something goes wrong, it is often unclear what happened, why, and who is accountable. At the same time, global initiatives such as the G7's call for *"SBOMs for AI"* highlight a growing consensus: AI systems need structured, verifiable traceability infrastructures.

> DBOM directly addresses this urgent gap—providing the technical foundation for provable AI accountability.

In detail, the DBOM solution has the following features:

End-to-End Traceability with Accountability Built In
DBOM extends the concept of SBOM to AI by capturing complete dependency and decision lineage, enriched with cryptographic signatures from all responsible stakeholders. This ensures that, when failures occur, responsibility can be clearly traced and provably attributed.

Tamper-Resistant by Design
Leveraging advanced confidential computing technologies, DBOM guarantees that both AI execution and its documentation remain secure, verifiable, and resistant to manipulation.

Full Lifecycle Coverage
DBOM spans the entire AI lifecycle—from data collection to end-user decisions. It addresses the real-world complexity of distributed AI systems and resolves the *"problem of many hands"* by ensuring continuous traceability across all participants.

Foundation for a Scalable Accountability Ecosystem
Built on trusted, verifiable data, DBOM enables a rich ecosystem of tools, including inspectors, integrity validators, compliance engines, and monitoring systems. This allows stakeholders to collaborate securely, share insights, and meet regulatory requirements without exposing sensitive information.

1 Context

The global market for artificial intelligence is expanding at an unprecedented pace. Valued at approximately $255 billion in 2025, it is projected to exceed $1.2 trillion by 2030, reflecting the accelerating adoption of AI across industries.[1]

However, this rapid growth is not matched by equivalent progress in accountability mechanisms. While AI capabilities continue to advance, the ability to trace, verify, and assign responsibility for AI decisions remains underdeveloped.

This gap is increasingly concerning as AI systems are already being deployed in high-risk and socially impactful domains, including autonomous driving, healthcare and medical diagnostics, education and assessment systems, employment and hiring decisions, law enforcement and judicial processes, *etc.*

In these contexts, AI outcomes can directly affect human safety, rights, and opportunities. Failures are not merely technical issues—they can lead to serious societal, legal, and economic consequences. Despite this, current AI systems lack end-to-end traceability, reliable attribution of responsibility, and verifiable records of the decision-making process. As a result, the faster AI is adopted, the more urgent it becomes to establish a robust accountability infrastructure.

Without such infrastructure, organizations face escalating risks of

- Inability to comply with emerging regulations;
- Exposure to legal and financial liabilities;
- Erosion of public trust.

[1] https://www.statista.com/topics/3104/artificial-intelligence-ai-worldwide/

2 Problem

Despite extensive research on AI accountability—including explainability, documentation, governance, and auditing—current approaches remain fragmented and incomplete.

They operate at different layers of the AI lifecycle:

- Model-level (interpretability of predictions) [1]
- System-level (documentation such as model cards and datasheets) [2], [3], [4], [5]
- Process-level (governance and policies) [6]
- Outcome-level (auditing and evaluation) [7], [8]

However, these methods fail to solve a core technical challenge:

> There is no end-to-end, tamper-resistant mechanism to trace AI decisions back to all contributing components and responsible parties.

Table 1: Limitations of Existing AI Accountability Approaches Compared to DBOM

Dimension	**Model-level**	**System-level**	**Process-level**	**Outcome-level**	**DBOM**
Scope coverage	Model	Partial	Organization-level	Outcome	End-to-end
Links training to decisions	×	×	×	×	✓
Decision-level traceability	×	×	×	Partial	✓
Stakeholder attribution	×	Partial	Partial	×	✓
Tamper resistance	×	×	×	×	✓
Verifiable integrity	×	×	×	×	✓
Cross-system / multi-actor	×	Limited	Limited	×	✓
Training–inference linkage	×	×	×	×	✓
Per-decision records	×	×	×	Partial	✓
Automation level	Low	Low	Low	Medium	High
Compliance readiness	Low	Medium	Medium	Medium	High

Specifically, as summarized in Table 1, existing methods cannot reliably map system behavior to specific actors, leaving responsibility distributed across multiple stakeholders—commonly known as the "problem of many hands." There is no unified mechanism that links training data, models, infrastructure, and individual decisions into a coherent trace. At the same time, logs and documentation can be altered, undermining their reliability for auditing and compliance. Training and inference processes are typically disconnected, lacking cryptographic or structural linkage. On top of that, many AI systems are deployed in cloud environments where execution integrity cannot be fully verified, further weakening trust in both outcomes and records.

On the other hand, confidential computing technologies provide important capabilities–such as protecting execution integrity, enabling remote attestation, and securing computation in untrusted environments. But they are not designed specifically for AI accountability, and thus do not offer structured lifecycle traceability, fail to capture decision-level provenance, and cannot serve for attribution of responsibility across stakeholders. As a result, no comprehensive, end-to-end accountability framework for AI systems currently exists, leaving critical governance gaps unaddressed.

3 Solution: Decision Bill of Materials

DBOM is a foundational accountability framework designed to provide end-to-end, verifiable, and tamper-resistant traceability across the entire AI lifecycle. DBOM extends traceability from software components to the data, models, systems, and stakeholders involved in building and operating AI systems.

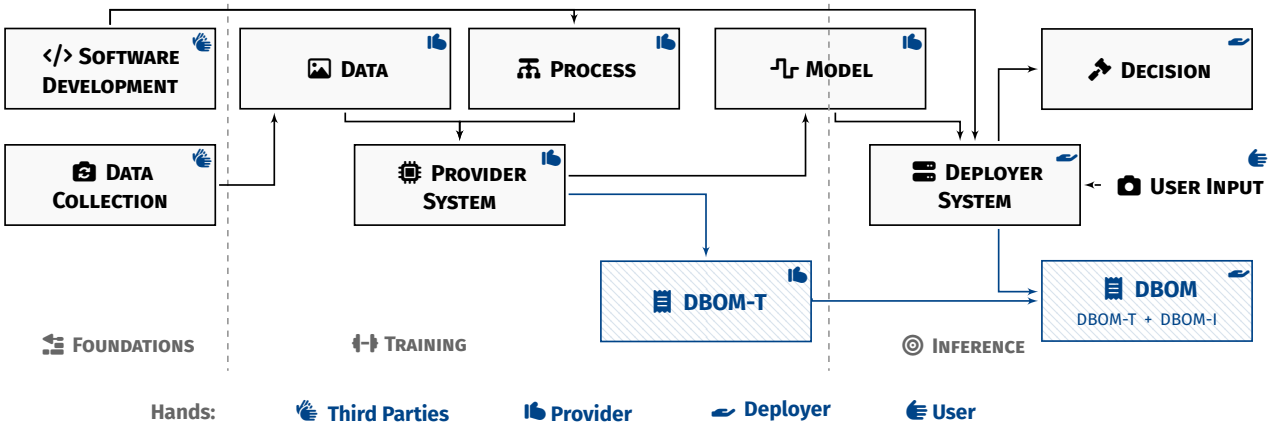

Figure 1: A generic AI system can be separated into components, and information flows in the direction of the arrows. Our contributions are marked in blue.

DBOM recognizes that modern AI systems are not monolithic artifacts but complex ecosystems involving multiple parties and processes. To capture this complexity, DBOM spans three major phases of the AI lifecycle (*cf.* Figure 1):

- **Foundations**: the software, data sources, and dependencies that form the basis of an AI system;
- **Training**: the processes, infrastructure, and stakeholders involved in creating a model; and
- **Inference** the deployment environment and runtime decisions delivered to end users.

By recording and linking evidence across these phases, DBOM creates a continuous chain of accountability from data collection to the final AI-assisted decision.

3.1 Two Complementary Records

Our approach introduces two core artifacts to enable full traceability of AI decisions:

DBOM-T captures all relevant information about the training process: dataset summaries, preprocessing steps, hyperparameters, cross-validation metrics, as well as hardware and software versions used. Model parameters and evaluation results are also documented in the D. This artifact enables reproducibility and auditing of the training workflow.

DBOM-I details the inference process for each decision: raw input features, encoding strategies, model predictions, decision logic, as well as hardware and software versions used. Addi-

tionally, it records timestamps and cryptographic signatures to ensure integrity and authenticity.

Together, DBOM-T and DBOM-I create a complete decision lineage that enables organizations to answer critical accountability questions:

- What data, software, and processes contributed to this decision?
- What type of model was used?
- Under what conditions was the decision generated?
- Which stakeholders were responsible for the relevant components?

3.2 Accountability by Design

Unlike existing documentation approaches, DBOM is not merely a reporting mechanism. It is designed to become the trust infrastructure for AI accountability.

By combining lifecycle-wide documentation, cryptographic integrity protections, stakeholder signatures, and verifiable links between training and inference, DBOM transforms accountability from a manual and fragmented process into a systematic and auditable capability.

As a result, organizations gain a reliable foundation for risk management, compliance, auditing, incident investigation, and trustworthy AI deployment, particularly in high-risk domains where understanding what happened, why it happened, and who is accountable is essential.

3.3 Contributions

DBOM makes the following contributions:

- A unified framework for end-to-end traceability of AI decisions across the full lifecycle;
- A cryptographically verifiable mechanism that links training processes to individual inference outcomes;
- A system architecture that integrates trusted execution environments (TEEs) into AI accountability pipelines;
- A prototype implementation demonstrating the practicality of the approach, including an empirical evaluation of its performance overhead.

The main hub for DBOM-related information is at `https://dbom.info`.

4 Approach

In this section, we present the detailed design of our solution. We begin by introducing the core artifacts of the framework, namely DBOM-T and DBOM-I, which capture training and inference provenance, respectively. We then describe the underlying security mechanisms, including trusted execution environments (TEEs) and cryptographic signatures, that ensure the integrity and authenticity of these records. Finally, we formalize the dependability aspects of DBOM by specifying the threat model for both training and inference and delineating the guarantees that DBOM provides under these threats.

4.1 DBOM-T/DBOM-I Design

Capturing Training Provenance with DBOM-T The first core artefact of the framework is the *Training Bill of Materials (DBOM-T)*. Generated during model development, DBOM-T records all information required to reproduce and audit the training process. This includes:

Project Metadata
Documents the high-level purpose of the task and versioning information.

Data Summary
Provides a complete overview of the data used, including, e.g., total sample counts or class distributions. For full reproducibility, this section also stores the exact indices used for the main data splits.

Model Architecture
A component-wise breakdown of the model.

Training Methodology
Details the evaluation approach and lists all hyperparameters used, such as learning rate, batch size, optimizer, and epochs per fold. It also specifies that a final model is trained on the full non-test dataset.

Performance Metrics
A comprehensive report of the model's performance. This includes detailed cross-validation statistics (mean accuracy, standard deviation, and per-fold results) as well as the final, unbiased performance metrics (accuracy, sensitivity, specificity, etc.) on the hold-out test set.

Environment and Dependencies
A manifest of the computational environment, including the hardware (e.g., CPU/GPU), Python version, and key library versions (e.g., PyTorch, scikit-learn).

Output Artifacts
Contains pointers to the files with the saved final model weights and the DBOM-T file itself.

Signature

A hash of the DBOM-T's contents to verify its integrity and ensure it has not been altered.

The resulting record serves as a comprehensive description of how a model was created. Because DBOM-T contains both technical and operational metadata, it allows auditors, operators, and regulators to reconstruct the conditions under which a model was trained and validated.

To guarantee integrity, each DBOM-T is cryptographically signed, creating a verifiable training record that cannot be altered undetectably.

Capturing Decision Provenance with DBOM-I While DBOM-T documents model creation, accountability also requires visibility into runtime behavior. For this purpose, DBOM generates an *Inference Bill of Materials (DBOM-I)* for each AI decision. A DBOM-I records:

Inference Identification
contains a unique inference ID, timestamp, and a hash linking to the specific DBOM-T used for this inference session.

Input Metadata
captures essential information about the inference input, including input identifier, input dimensions, preprocessing pipeline applied, and any input-specific transformations not covered in the DBOM-T training methodology.

Inference Results
documents the actual prediction process including:

- *Raw Model Output:* raw values, intermediate layer activations, and final probability scores before decision thresholding.
- *Decision Metrics:* final classification decision, confidence score, decision threshold used, distance from threshold, and certainty level assessment.
- *Feature Analysis* includes input-specific feature extraction results, concept similarity scores computed for this specific input, and any runtime feature modifications.

Decision Pathway Tracking
provides step-by-step documentation of the inference process from input processing through final classification, including any runtime optimizations or modifications applied during inference.

Temporal Inference Data
captures inference-specific timing information, computational resources used, and any runtime environmental factors that might affect reproducibility.

Signature
A hash of the DBOM-I's contents together with the link to DBOM-T to verify its integrity and to ensure it has not been altered.

Runtime Environment

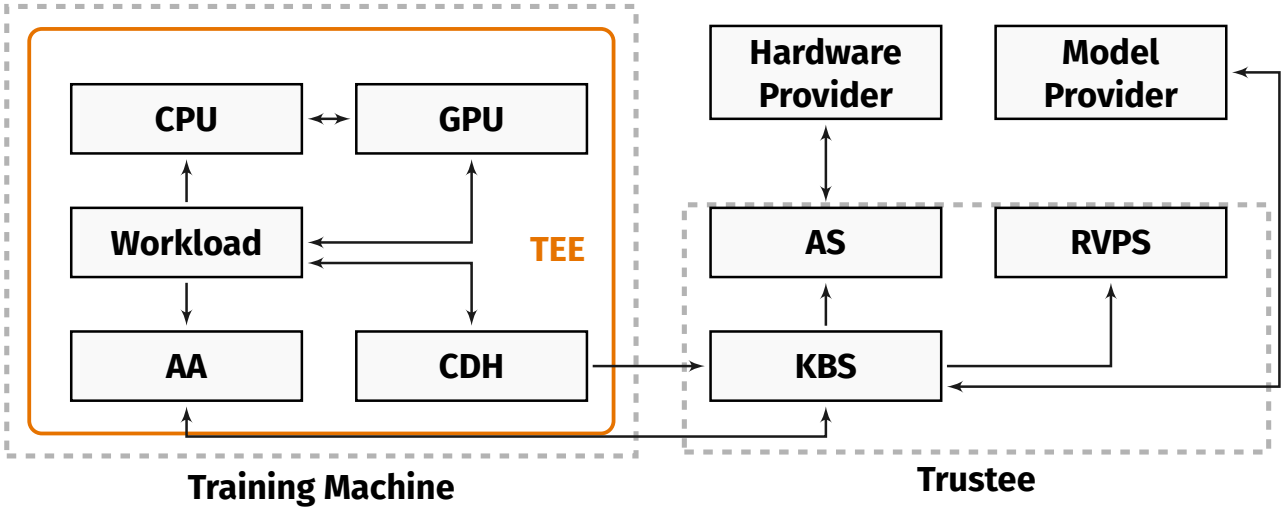

Figure 2: Component interaction for attestation during training.

documents hardware, software versions, or computational configurations. For models served in a distributed fashion (e.g. via web technology), this would include information about the serving system (which is not the same as the provider system in Figure 1).

Most importantly, every DBOM-I is linked to the exact DBOM-T from which the deployed model originated. This creates a traceable lineage between a specific decision and the training process that produced the corresponding model.

4.2 Security Mechanisms

4.2.1 Securing Training and Inference with Confidential Computing

To ensure that accountability records accurately represent what actually happened, DBOM integrates confidential computing technologies into both training and inference workflows.

Connecting the Hardware We use two different TEE technologies: Intel SGX, where isolation of the TEE happens on the process level (isolated processes run in so-called *memory enclaves* or simply *enclaves*) and Intel TDX, where entire virtual machines are isolated, thus creating a CVM. Only the latter one supports interaction with a confidential GPU.

By default, a TEE is restricted to one hardware component, so other hardware components cannot access the memory that a TEE device encrypts before writing. Confidential GPUs address this problem by creating special buffers shared between components. For the NVIDIA H100 and H200 GPUs, the CUDA platform has been extended to support transparent decryption and encryption.

We focus on the H200 GPU that supports *full pass-through*, a mode where the GPU needs to be entirely allocated to the TEE. The H100 will instead introduce a mode called *TEE I/O*, allowing for more fine-grained communication.

Attestation To protect the integrity of the code and the data inside a TEE, attestation establishes the trustworthiness of a TEE. We here use the terminology established in the RATS draft of the IETF [9]: A *Relying Party* uses a *Verifier* to establish trust in the *Attester*. The Verifier can use different sources: An *Evidence* provided by the Attester can be compared with a *Reference Value*, optionally from an external provider. *Endorsers* can act as trust anchors, establishing "ground truths". After successful attestation, the Relying Party can provide the Attester with secret resources, knowing that the Attester will not leak them.

We use this mechanism to deploy the signing key for the DBOM. Our Relying Party is a key distribution service (details in section 4.2.1). It provides the signing key to the TEE, which takes the role of the Attester. The evidence is a so-called *measurement* of the TEEs data that is signed using a key built into the CPU to form a so-called *quote*. This quote is verified using endorsements from the hardware manufacturer. If attestation is successful, the Relying Party is assured that the correct program is running on the right hardware. This allows to run the computing-intensive training and inference process in potentially malicious environments, such as cloud computers.

It is worth noting that, although the technology is usually named *Confidential* Computing, our usage is focused only on integrity protection (thus achieving tamper-resistance), not on confidentiality. Although we use encryption for data in-use, the training data, the model and the code could be eavesdropped by an attacker, but this would be detected by attestation, and the data-in-use encryption protects from manipulations.

Implementation For the Relying Party and the Verifier as well as parts of the Attester, we have been able to reuse the framework provided by the Confidential Containers[2] project. The tools are mainly used to ensure confidentiality, but we use them to inject our DBOM signing key into the enclave.

The parts we use and their relation are depicted in Figure 2. The services needed for attestation have to be mostly located on a separate machine. Since Confidential Containers currently has no support for running those services confidentially, they need to run on a trustworthy machine, the so-called *trustee*.

At the trustee, the KBS manages keys that are injected into TEEs. It communicates with the TEE in the role of a Relying Party. The role of the Verifier is taken by the AS that supports multiple hardware-specific remote attestation procedures, whereas the RVPS stores valid evidence values.

Inside the TEE, the AA takes care of generating the evidence and the CDH retrieves the secret signing key after attestation.

Attestation and Workflows for Training and Inference The attestation workflow is depicted in Figure 3. For training, we consider a human or organizational entity we call the *Model Provider* and who wants to train a model while generating a DBOM-T. We assume that they have sole control over a trustee.

[2] https://confidentialcontainers.org

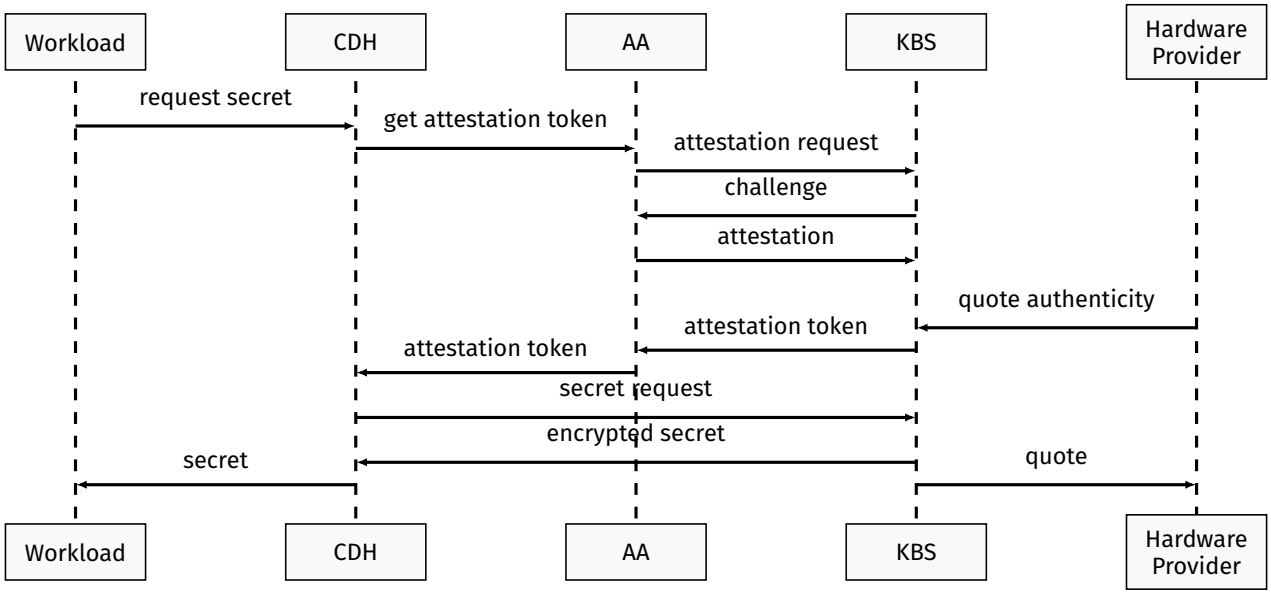

Figure 3: Attestation workflow.

1. The Model Provider builds the image used for training, including the training data, the training software and an authorization key K_a for KBS.
2. The Model Provider calculates the reference value for the image and uploads it to the KBS, which uses the RVPS to store it.
3. The Model Provider generates a signing key K_s for the DBOM and uploads it to the KBS together with a policy specifying optional additional restrictions.
4. The Model Provider deploys the image to the potentially malicious training machine. The TEE (i.e., either the TDX confidential VM or the SGX enclave) starts and initiates the attestation by authenticating to the KBS using K_a
5. After authentication, the KBS sends a challenge containing a nonce n for freshness. The AA creates an attestation report r. It contains, among other, hardware information, the nonce and the *measurement* m. The exact construction of the latter depends on the technology, but it always represents an inital state of the living memory.
6. After creating the attestation report $r = n||m||\ldots$, the AA uses the CPU to sign it with the CPUs secret key K_c and to create the quote $q = \texttt{sign}(r, K_c)$ and sends it back to the KBS.
7. The KBS uses the AS to verify the quote. The AS contacts an API provided by the hardware manufacturer to verify the signature of q and the RVPS to compare the measurement m. Optionally, if a policy has been defined, the KBS verifies its conditions.
8. If the attestation was successful, the CDH asks the KBS for the signing key K_s.
9. Once received the key, the training is started. After the training, the DBOM-T is signed using K_s.

The inference works similar, with the exception that the input contains the DBOM-T which includes a hash value of the model. This value has to be verified after attestation and before starting inference to ensure the integrity of the pipeline. It links the training and inference step and prevents tampering in-between. Since both training and inference are run in tamper-resistant environments, an attacker cannot modify the model without being noticed.

4.2.2 Ensuring Trust Through Cryptographic Linking

Cryptographic Technology The creation of a signed DBOM runs in the same TEE as the training or inference process. Thus, all cryptographically relevant operations have the same tamper-protection as the AI workflow itself. We use SHA-256 hashes to identify both data as well as artifacts. We further use digital signatures to bind the DBOM to the provider's identity (for training) and to the deployer's identity (for inference). Since digital signatures are always computed over hashed data, this also prevents the DBOM from being tampered with, as manipulation would lead to a mismatching signature. We chose DSSE [10] as our signature scheme to avoid canonicalization issues and because it is already used for software supply chain security.

Signing the DBOM After training, the resulting model is hashed and the hash is included in the DBOM-T, which is then signed with the provider's key. The DBOM-I also includes the model's hash to prevent the artifact from being modified at rest. The keys used for signing are given by the provider for DBOM-T and by the deployer for DBOM-I and can be linked to them. They are protected using the TEE's attestation mechanism that can verify both the integrity of the training code and data as well as the integrity of the hardware. Only if those verifications are successful, the key gets passed to the TEE through a secure channel, protecting it from being eavesdropped in a cloud environment. Provider and deployer can specify the code and data that is verified during attestation. This protects against valid DBOM signatures without the signer's consent. Other information in the DBOM, like training data provenance can be simply signed by a human signer who takes responsibility for the part they sign. The process stays clearly documented and we estimate that the humans signers will be held accountable, if they abuse their signing power to create corrupted yet valid DBOMs.

Machines Signing vs. Humans Signing So far, we have focused on a process in which processing steps are signed by machines, ideally using cryptographic hardware support. However, there is often a need for embracing signatures of natural persons. 1. At the edge of AI (e.g. when collecting data to learn from), certain properties might not be checkable in a fully automated process running on dedicated hardware. For instance, if some photographs to be used in learning were originally taken with a camera or smartphone not supporting TEE, then the trust chain were broken, unless we allow a person to vouch for the integrity of the photos when including them in the data to learn from. 2. Another meaningful example demonstrating the need for human signatures lies in the use of pre-trained LLMs. Since it is (by now) unrealistic to assume a DBOM to exist for a commercial LLM, the properties and quality guarantees of these LLMs cannot be scrutinized in their entirety. But they can be assumed to hold, provided a vendor representative signs for these guarantees to be present. 3. In a future DBOM adoption process, more and more parts of the AI pipeline will gradually become DBOM-enabled. The intermediate steps then require human signatures.

Our system provides full support for explicit human signers. Each part of the DBOM can be signed by a human signer instead of by a key linked to attestation. With their signature, they vouch for the correctness and integrity of what they sign. As mentioned, in practice the signer will usually be a natural person representing an organization, institution, or a company that acts as the legal person taking responsibility.

4.3 Dependability Concept

Within this worklfow and thanks to its implementation details, DBOM can protect against certain threats. DBOM provides full protection ( ) or only detection ( ), yet, in some cases, our approach can not help ( ). In the following, we outline specific threats to a generic system and how they can (or cannot) be handled. The component icon resembles the focus of the threat (multiple components can be threatened at the same time).

Threats during TRAINING

Model Manipulation (at rest) implies bypassing the training and influence decisions directly via the model. Since we protect the integrity of data-at-rest with cryptographic signatures, we can rule out this attack vector.

Data-in-use Manipulation may occur when an (advanced) attacker attempts to alter the RAM during the training process. Our use of TEEs effectively protects against such attacks by securing data during processing.

Training Data Tampering includes *poisoning and backdoor attacks*, which aim to corrupt model training by altering the training data. *Poisoning attacks* subtly modify inputs to degrade model performance, while *backdoor attacks* embed triggers that cause the model to behave maliciously only when activated. Although our approach cannot directly detect data manipulation, it facilitates forensic investigations by precisely documenting the datasets used and thus enabling traceability of problematic data points.

Training Process Poisoning targets the training pipeline's integrity by compromising the environment, altering training algorithms, or exploiting system vulnerabilities. DBOMs document the environment and the algorithms used. This cannot prevent, but will expose, an attack on this part of the training pipeline, as the attacker cannot prevent the alterations from being documented.

Threats during INFERENCE

Tampered Inference System is an attack on the inference process's RAM that is prevented by the usage of TEEs.

Decision Manipulation during Delivery replaces the actual AI decision with one chosen by the attacker. The DBOM is tied to the decision with a signature and can be verified, thus protecting the decision against tampering.

Illegitimate Inputs include OOD data and adversarial inputs. The latter involve subtle perturbations that mislead the model, while OOD inputs occur when inference data is not adequately represented in the training set. Although DBOM cannot prevent such errors, it enables retrospective analysis by identifying and tracing the relevant training data.

Information Leakage includes model extraction or inversion and membership inference. The aim is to reconstruct model parameters, reveal training data membership, or infer sensitive attributes learned by the model. We do not address this threat, as it is not possible to expose all kinds of information leakage in the DBOM and we cannot prevent a leakage.

5 Ecosystem

DBOM is designed to be more than a documentation standard. Similar to how SBOMs evolved from simple dependency manifests into a rich ecosystem of security, compliance, and supply-chain management tools, DBOM lays the foundation for a comprehensive accountability infrastructure for AI systems.

5.1 Legal Ecosystem

There is a variety of regulatory approaches around the globe with respect to AI, as well as software in general. DBOM's intellectual precursor SBOM is already explicitly mandated by legal frameworks (e.g., the US-American *Executive Order 14028 on Improving the Nation's Cybersecurity* (2021)) or it implicitly constitutes the state of the art in business-to-business contexts.

With respect to AI, a global perspective shows a heterogeneous landscape.[3] While many nations have formulated ethical principles or defined incentives for responsible usage, few have transparency rules or regulations with respect to liabilities.

Among the explicit normative frameworks for AI, the European AI Act [11] has a certain prominence. Europe is the second largest economy in the world, and the AI Act has extraterritorial reach, meaning that it also applies to providers and deployers of AI technology located outside the EU, if their AI systems are placed on the EU market or used in a way that affects people within the EU.

The AI Act distinguishes certain roles (among them providers, deployers, users) and risk categories (among them "high risk") for which we refer to [12]. It then includes a number of requirements; some of the most relevant ones are listed below, and are placed into a DBOM context.

Risk Management
Providers of high-risk AI systems are required to establish, implement, document, and continuously maintain an effective risk management framework [11, §9], and to guarantee that the system's accuracy, robustness, and cybersecurity are sufficient for its intended use [11, §15].
→ *A DBOM-T can be considered the conceptually central element of such a risk managment documentation.*

Data Governance
Providers of high-risk AI systems are required to develop and train their models using training and validation datasets that comply with established quality criteria, while also adhering to robust data governance requirements [11, §10].
→ *Again, a DBOM-T serves as a natural framework for documenting these quality and robustness requirements.*

Human Oversight
High-risk AI systems are to be designed and developed in such a way that a human overseer is in the position to monitor, control, revert AI decisions [11, §14].

[3] `https://www.brennancenter.org/our-work/research-reports/artificial-intelligence-legislation-tracker`

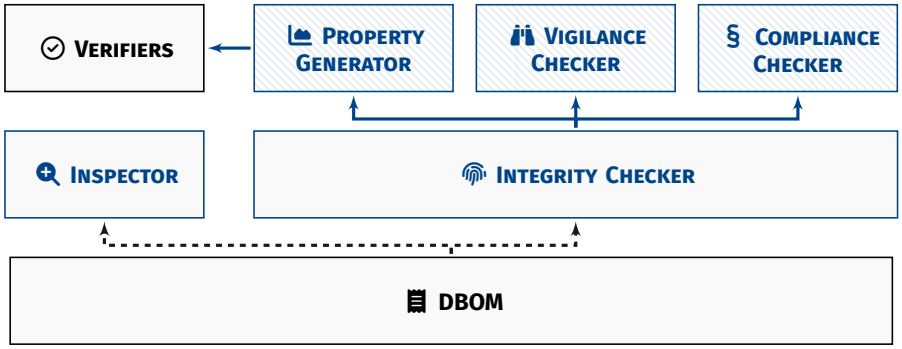

Figure 4: Tool chain based on DBOM. Solid blue lines show workflows based on DBOM that have been checked for integrity. Blue boxes mark ecosystem additions and black boxes represent existing external technology. The hatched components are envisioned future additions, the others are developed here.

→ *With adequate human-machine interfaces in place, the pair of DBOM-T and DBOM-I can become a pivotal vehicle for enabling human oversight.*

Post-Market Monitoring Duty

Providers of high-risk AI must provide post-market monitoring [11, §72]. They must support automated event logs during the full lifetime [11, §12]. The operation must enable deployers to interpret the system output [11, §13].

→ *Logging of decisions and their unfolding is the central functionality provided by DBOM-I.*

Live Monitoring Duty

Providers of high-risk AI systems must monitor the operation of their system [11, §26(5)].

→ *It does not appear far fetched to imagine that in-field DBOM-I instances are routinely and continually collected by AI providers for the purpose of live monitoring.*

Transparency Rights

When taking high-risk decisions, any person adversely impacted has the right to obtain explanations [11, §86], and providers must ensure transparency and deliver the necessary information to deployers [11, §13]. In addition, the European Court of Justice has (in C-203/22) confirmed that already the EU General Data Protection Regulation (GDPR) induces a right to explanations of algorithmic decisions that should be functional, important, relevant, and intelligible.

→ *With adequate interfaces in place, individuals can profit from getting access to DBOMs of the systems they are facing, and thus effectuate their transparency rights.*

5.2 Technical Ecosystem

Again, looking at SBOMs, we see that there is an increasing ecosystem of tools that use and extend the base format. We can conceive different tools that use DBOMs, each contributing to the dependability of high-risk AI decisions. Here, we propose a small, incomplete, set of tools to leverage DBOMs—also depicted in Figure 4. The middle two, blue components exist as prototypes, while the hatched blue boxes are future work.

DBOM Inspector

The DBOM inspector can incorporate a wide range of existing tools to evaluate the desired properties of an AI model, enabling a systematic assessment of how well the model satisfies key requirements such as security, fairness, explainability, and privacy. By integrating established methodologies and diagnostic frameworks, the inspector can support comprehensive inspections across multiple dimensions of model quality. For example, security analysis can leverage knowledge about existing adversarial attacks [13], [14], [15] and poisoning attacks [16], [17]. Fairness analysis may draw from dataset balancing [18], [19] or bias detection and mitigation [20], [21]. Explainability can be achieved by utilizing post hoc attribution explanation approaches [22], [23] or by selecting an interpretable model from the start [24], [25]. Finally, privacy verification can employ techniques such as differential privacy auditing [26], [27] or check for membership inference attacks [28], [29].
Collectively, these existing techniques with corresponding measurements allow the DBOM inspector to form a unified and extensible environment for inspecting AI systems against rigorous standards as needed. Unlike directly applying these techniques, DBOM ensures that the data and models used in such approaches are tamper-resistant.
As already shown by our exemplary user interface, one can use a DBOM to facilitate visualization and inspection of the training and inference processes. In our use case, explainability is a primary consideration; accordingly, we adopt interpretable models, namely concept bottleneck models [24], to ensure transparency. Bias is evaluated by visualizing and analyzing the underlying data distributions. During training, the system visualizes essential data insights, such as the training accuracy per epoch, together with all learned concepts along with their corresponding importance scores. For inference, given an input image, the interface retrieves concept correlation information from DBOM-I. Users can interactively modify concept correlations, enabling real-time intervention to observe how such changes affect the final prediction.

Integrity Checker

As the DBOM itself is just a file format, there will be instances of DBOMs that do not fulfill certain desirable properties. First, there is a syntactical check if the file is a conforming instance of a DBOM (i.e. adheres to the right structure). Second, there are crypthographic checks, to verify if a DBOM still has integrity, i.e. the entire cryptographic information (mostly related to confidential computing) is intact. Hence, an initial step in many DBOM related pipelines will be to run an integrity checker on a DBOM and react accordingly, in case it is invalid or damaged. As can be seen in Figure 4, most advanced checkers will typically be run only on integrity-checked DBOMs.

§ Compliance Checker

One level higher in the abstraction hierarchy is the check for compliance. In the paper, we have referred several times to current and upcoming legal requirements on the documentation and implementation of digital services. In the future, standardisation bodies (like ISO, IEC, CENELEC) will issue harmonised standards that enforce certain aspects of AI data quality, training, testing, and the like. Indeed, these standardisation processes are currently in full swing, induced mainly by the AI Act [11, §40]. It can be expected that, for high-risk AI applications, certification authorities and services will emerge that take over compliance checks with respect to these standards. Now, as the DBOM inspector incorporates existing techniques for evaluating security, privacy, fairness, and

explainability, while DBOM inherently offers transparency and traceability, these capabilities align closely with AI compliance requirements.

By putting these techniques behind the integrity check, it will be possible to collect certifiable evidences, and to construct a compliance checklist for review by a certification authority. As a simple example, one might imagine a compliance checklist to require that the model demonstrates valid and sufficient performance, is free of a set of predefined biases, and meets specified security, privacy, and explainability criteria. The DBOM can provide supporting evidence by enabling cross-validation with prescribed data splits and recording the resulting validation accuracy in the DBOM-T to demonstrate performance. The compliance checker can also apply adversarial attacks, membership-inference attacks, and post hoc attribution explanation approaches during inference, reporting the corresponding attack success rates and explainability measurements in the DBOM-I as evidence of security, privacy, and explainability. In consequence, a certification authority can use the DBOM to check if the corresponding model is compliant or not.

Vigilance Checker

Taking inspiration from the medical domain, where vigilance systems track medical products in the field, similar systems can be established for AI. Note that, e.g. the AI Act [11, Article §72] in the European Union includes post-market monitoring obligations. DBOM-producing applications could be forced to occasionally report DBOMs to a central authority. The vendor or (in highly critical cases) the authority would then check the information contained and act accordingly, possibly considering the temporal evolution across several DBOMs.

Property Generator

The property synthesizer is meant to wrap some of the properties discussed in the context of the compliance checker into formal language, so as to make them accessible to formal verification tools.

Verifiers

Owed to the properties made available by the property generator, it is possible to scrutinize the entire workflow with verification tools [30], [31]. This includes proof assistants (e.g. LEAN [32]), probabilistic statistical model checking (e.g. Storm[33] or PRISM [34]), but also tailored approaches for checking robustness [35], [36], [37] or safety [38]. This access to automated, high-reliability and state-of-research analysis on aspects of AI decisions is possible while being sure that the ground truth is tamper-resistant.

6 Use Case

FungAI is a lightweight and playful demonstrator of how DBOM supports AI system design and operation—sharing key architectural and assurance properties with typical safety-critical vision systems (in use, e.g., for skin-cancer recognition). FungAI is an AI-based application to determine whether a mushroom is poisonous or edible, based on an image uploaded by the app user.

The following table illustrates in what sense the task of mushroom classification can serve as a playful proxy for high-stake decision tasks such as those found in modern skin cancer recognition apps, while avoiding privacy issues.

	Skin cancer recognition	Mushroom classification
False positive	"cancer" despite no cancer	"poisonous" despite edible
False negative	"no cancer" despite cancer	"edible" despite poisonous
OOD risk	skin tone underrepresented	mushroom type underrepresented
Privacy	serious concerns & complications	none

Setup of FungAI. The entire AI system is implemented with tamper-resistant execution and integrated with DBOM support. The DBOM structure and its generation processes follow our methodology and thus capture all relevant artifacts and dependencies.

DATA We use a dataset[4] containing 8,468 labeled images. To arrive at a binary classification, we merge edible/conditionally edible and poisonous/deadly, yielding 2,895 *edible* and 5,573 *poisonous* samples.

MODEL Given the limited data, we adopt a few-shot model with built-in interpretability. Our binary classifier is based on Ph-CBM [39] with a pretrained CLIP (ViT-L/14) backbone.[5] The concept set is derived from the tabular Mushroom Classification dataset,[6] using attribute-value pairs (e.g., cap color: red). We freeze the CLIP backbone and add three fully connected layers for classification, tuning only their hyperparameters. Training uses class labels with a sparsity constraint on concept representations (no concept-level supervision).

PROCESS We split the data in 80% (training) and 20% (testing). We apply 5-fold stratified cross-validation for balanced class representation across folds in training. The model's performance is then evaluated on the held-out testing set.

PROVIDER SYSTEM Training is performed inside the TEE of an NVIDIA H200 NVL GPU with CC support, using Ubuntu (v25.04 host, v24.04 guest) and CUDA 12.0. We have developed a custom setup of the attestation components of the *Confidential Containers* project to protect the training process. The DBOM is cryptographically signed with a key released only after successful attestation and includes hashes of the model and training data to ensure integrity. Model encryption after training is a possible add-on, but not needed for tamper resistance and not implemented yet.

[4] https://www.kaggle.com/datasets/derekkunowilliams/mushrooms
[5] https://github.com/openai/CLIP
[6] https://www.kaggle.com/datasets/uciml/mushroom-classification

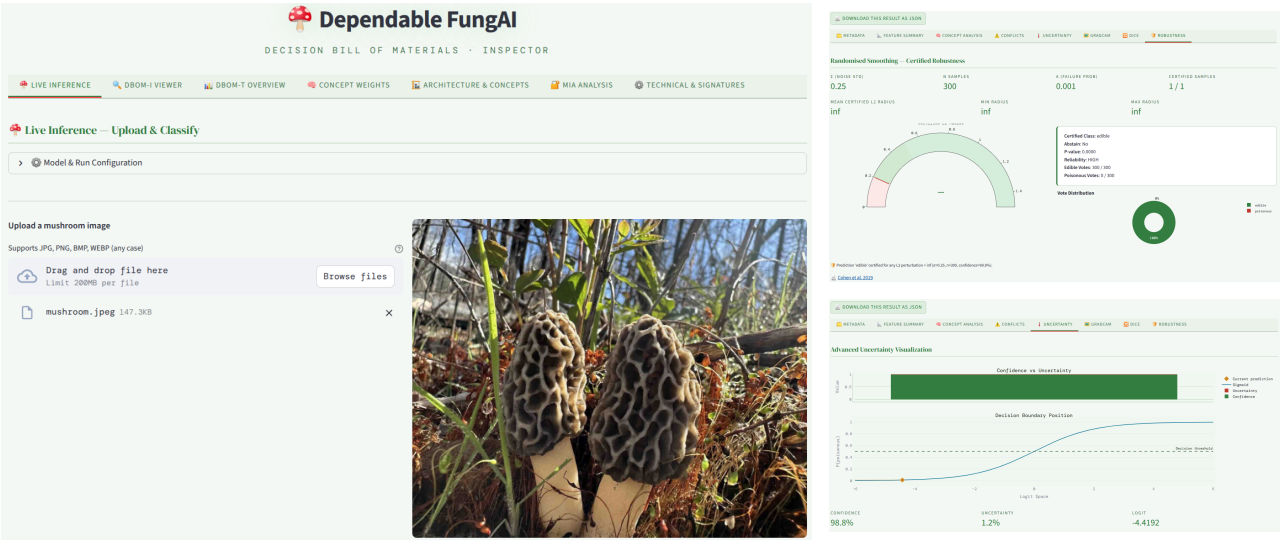

Figure 5: FungAI Inspector.

Deployer System The trained classifier runs inside the TEE of a CC-supporting server, while the user interface operates in the consumer's device. Users submit inputs (e.g., mushroom images) via the interface, which communicates with the server to perform inference inside the TEE, ensuring secure execution. Inference is lightweight and thus does not need GPU-support. Our setup uses a dual-CPU setup with two Intel Xeon 6952P processors providing SGX capability for CC.

Decision The user receives the classification ("edible" or "poisonous") and can inspect detailed DBOM information (e.g., concept contributions or predicted probability). This is supported by a dedicated tool component, the `FungAI` inspector (cf. Figure 5) that incorporates concept-based models [39] to promote transparency, leverages DiCE [40] to generate counterfactual explanations, and applies randomized smoothing [41] to provide certified robustness guarantees. Bias is evaluated by based on the underlying data distributions. Users can modify concept correlations, enabling real-time intervention to observe how such changes affect the final prediction. The DBOM can be downloaded, enabling extraction of the analysed image's signature (SHA-256 hash). This process enables comparison with the locally stored version of that image, thereby ensuring that, even on the last mile, no one tampered with the image upload.

A web version of the `FungAI` app, together with `FungAI`'s DBOM and the `FungAI` inspector is accessible at `https://fungai.cs.uni-saarland.de/`. Evaluating the classifier's accuracy on unseen images is not the topic of this paper, yet the DBOM and its inspector can support such analyses.

Empirical setup. We evaluate both training and inference for our `FungAI` system. All experiments have been carried out on our provider machine (cf. paragraph 6). Attestation was not measured because the necessary round-trip time can vary widely, depending on network connectivity and the response times of the hardware manufacturer's attestation infrastructure.

We measured the GPU-based training times needed for `FungAI` with and without CC. Each configuration was trained 40 times, and the full training was recorded (results in Figure 6). For CPU-based inference, we also include the `FungAI` inspector, which provides explanations using DiCE to gener-

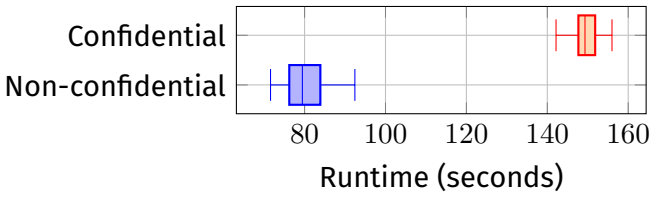

Figure 6: Runtimes over 35 runs for tamper-resistant configurations, with (Confidential) and without (Non-confidential) protection for GPU experiments of `FungAI`.

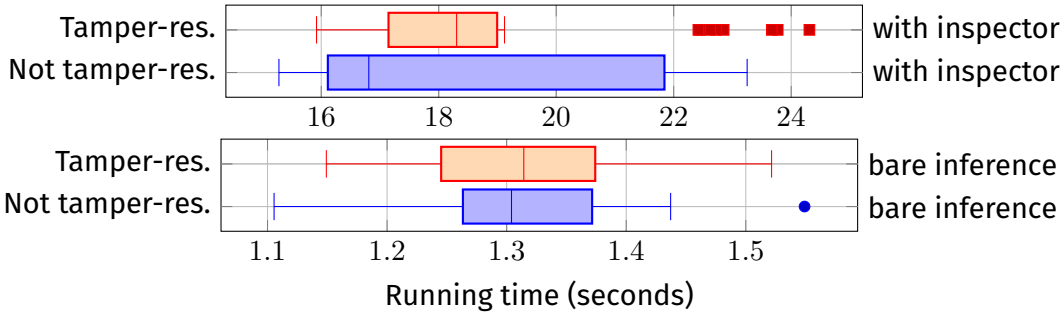

Figure 7: Running times of CPU-based inference over 40 runs of `FungAI` with/without tamper resistance, and with the inspector component/ only bare inference.

ate counterfactual examples and applies randomized smoothing to compute a certified robustness radius. Each variant has been run 40 times with and without CC. For each run, we report the full time the `FungAI` inspector process took, as well as the time it took to carry out the inference only, after having loaded the modules and the model. The resulting values are summarized in Figure 7.

Results. As evident in Figure 6, tamper-resistant training increases the running time by a factor well below two on average, a reasonable trade-off for the integrity assurance gain. With respect to inference (Figure 7), we observe that the difference in running time between the setting with and without tamper resistance is marginal. Running the entire inspector takes less than 20 seconds on average, while the bare inference is done within no more than two seconds. For a mushroom classification service that only once loads the modules and the model, just the bare inference time will be perceived on the user side.

The `FungAI` use case demonstrates the practical applicability of DBOM by providing a concrete, end-to-end example of its integration into a real-world AI system. It shows how DBOM can be systematically established within an application pipeline while incurring only moderate performance overhead. Furthermore, the use case illustrates how the DBOM ecosystem supports enhanced security and improved transparency through inspection and explanation capabilities.

7 Validation on Standard Benchmarks

Table 2: Total training time aggregated over five runs each.

Dataset	time unit	Not tamper-resistant		Tamper-resistant	
		average	standard deviation	average	standard deviation
MNIST	seconds	42.35	4.14	105.68	1.75
CIFAR-10	seconds	66.46	4.05	143.01	2.24
Tiny-ImageNet	minutes	35.42	0.66	43.28	0.14

Constructing a machine learning model is the most critical and resource-consuming part of building an AI system. To assess the feasibility of applying our tamper-resistant approach in general AI applications, we here provide a comprehensive analysis of DBOM performance through experiments on standard benchmark datasets and widely used model architectures on the provider GPU system.

Empirical setup We consider MNIST [42], CIFAR-10 [43], and Tiny-ImageNet [44], training the models from scratch within a DBOM each. These datasets, together with their standard architectures, span a range of scale and complexity, allowing us to evaluate how DBOM's tamper resistance affects efficiency across different computational profiles. For MNIST, we used a three-layer MLP; for CIFAR-10, a lightweight CNN with three convolutional and two fully connected layers; and for Tiny-ImageNet, ResNet-18 [45]. Models were trained with standard procedures for 3, 6, and 100 epochs, respectively, with each experiment repeated five times. Since the DBOM approach affects specifically the training time, this is reported as the primary metric.

Results Table 2 reports the average runtime and standard deviation for each of the entire training processes. The tamper-resistant configuration increases training time by $\sim 2.5\times$ for MNIST, $2.15\times$ for CIFAR-10, and $1.22\times$ for Tiny-ImageNet, with overhead decreasing as dataset size grows. This reflects a common pattern in confidential computing, where computation remains efficient but I/O adds overhead.

Altogether, our experiments indicate that DBOM with tamper resistance is a practical approach for real-world applications.

8 Limitations

While DBOM provides a foundation for end-to-end, verifiable traceability in AI systems, several limitations remain.

First, DBOM relies on the trust assumptions of the underlying hardware and cryptographic primitives. In particular, the security guarantees depend on the correctness of trusted execution environments (TEEs) and the integrity of the attestation infrastructure. Any compromise at this level may weaken the overall trust guarantees.

Second, DBOM focuses on ensuring the integrity and traceability of processes and artifacts, but it does not guarantee the correctness or quality of the underlying data, models, or decisions. For example, biased, incomplete, or low-quality training data may still lead to undesirable outcomes, even if fully traceable.

Third, while DBOM enables attribution of responsibility through cryptographic signatures, it does not prevent collusion or dishonest behavior among stakeholders who deliberately sign incorrect or misleading information.

Fourth, the integration of DBOM introduces additional system complexity and operational overhead. Although our evaluation shows that the performance impact is moderate, deploying such infrastructure in large-scale or latency-sensitive environments may require further optimization.

Fifth, the current implementation assumes partial coverage of the AI lifecycle. In practice, certain components—such as external data sources or third-party models—may not yet be DBOM-enabled, leading to gaps in the traceability chain that must be bridged through manual or organizational mechanisms.

Finally, DBOM primarily addresses technical aspects of accountability and does not fully capture broader legal, organizational, or ethical dimensions. Real-world accountability requires complementary governance frameworks and regulatory processes beyond the scope of this work.

Addressing these limitations represents an important direction for future research and development toward comprehensive AI accountability infrastructures.

9 The Value of DBOM

DBOM introduces a foundational trust layer for AI systems. By combining lifecycle-wide traceability, cryptographic integrity, and stakeholder accountability, DBOM transforms AI governance from a largely manual exercise into a verifiable and scalable capability.

9.1 Trustworthy, Verifiable Evidence and Accountability by Design

As AI becomes increasingly integrated into critical sectors such as healthcare, transportation, finance, education, and public administration, trust becomes a prerequisite for adoption. Stakeholders must be confident that AI decisions can be explained, traced, and audited when necessary. DBOM provides this confidence by establishing a trustworthy record of how AI systems are built and how decisions are made. By transforming accountability from an organizational promise into cryptographically verifiable evidence, DBOM helps bridge the trust gap between AI providers, users, regulators, and society.

Traditional documentation, logs, and reports can be incomplete, inconsistent, or modified over time. DBOM introduces cryptographically protected records that provide verifiable evidence of how AI systems were trained, deployed, and operated. This allows organizations to move from i) trust-based assertions to evidence-based accountability; ii) static documentation to living records; iii) manual audits to automated verification.

Current AI systems often operate as black boxes from an accountability perspective. When failures occur, organizations struggle to determine which data, model, process, infrastructure component, or stakeholder contributed to the outcome. DBOM addresses this challenge by creating an end-to-end chain of evidence that connects an AI decision to its origin. As a result, responsibility can be traced and justified rather than inferred after the fact.

9.2 Regulatory Readiness

AI regulation is rapidly evolving around the world, with increasing emphasis on transparency, risk management, human oversight, and post-market monitoring. Meeting these obligations today often requires significant manual effort and fragmented documentation processes. DBOM introduces a standardized mechanism for continuously collecting accountability evidence throughout the AI lifecycle. This allows organizations to move from reactive compliance activities to compliance-by-design, reducing regulatory burden while increasing confidence in audit and certification processes.

When an AI system produces unexpected or harmful outcomes, identifying the root cause can require weeks of investigation across multiple teams and systems. Because DBOM captures the complete provenance of training and inference activities, investigators can rapidly determine which model generated a decision, which data contributed to the model, which infrastructure was involved, and which stakeholders approved or supplied the relevant artifacts. This significantly reduces the time required for root-cause analysis and corrective actions.

9.3 Establishing a Common Accountability Language

Modern AI systems are built by complex ecosystems of data providers, model developers, platform vendors, cloud operators, integrators, and deployers. Each stakeholder typically maintains different records and documentation standards, creating information silos and accountability gaps. DBOM provides a common structure and shared source of truth that allows technical, organizational, and regulatory stakeholders to communicate using the same accountability framework. This harmonization is essential for large-scale AI governance across organizations and jurisdictions.

9.4 Enabling a New Ecosystem of Assurance Tools

Much like SBOMs enabled an ecosystem of software supply-chain security tools, DBOM creates the foundation for a new generation of AI assurance technologies. Integrity checkers, inspectors, compliance engines, monitoring services, certification platforms, and formal verification tools can all operate on a shared, trustworthy data source. The result is an extensible ecosystem that continuously improves the dependability of AI systems while reducing duplication of effort across industry and regulatory sectors.

Importantly, DBOM does not seek to slow AI innovation. Instead, it provides the infrastructure needed to innovate responsibly. By embedding accountability into the development and deployment process, organizations can adopt increasingly powerful AI systems while maintaining transparency, governance, and trust. This balance between innovation and accountability will become increasingly important as AI systems continue to grow in capability, autonomy, and societal impact.

In the long term, we envision DBOM becoming for AI what SBOM has become for software: a foundational infrastructure layer that enables trust across complex supply chains. As accountability requirements continue to expand, DBOM can provide the common framework through which AI systems are inspected, certified, monitored, and governed. By making accountability measurable, verifiable, and scalable, DBOM helps pave the way for a future in which AI systems are not only powerful, but also trustworthy by design.

Takeaway: DBOM transforms accountability from a policy aspiration into an operational capability, providing the trust infrastructure needed for the next generation of high-risk AI systems.

References

[1] G. Schwalbe and B. Finzel, "A comprehensive taxonomy for explainable artificial intelligence: A systematic survey of surveys on methods and concepts," *arXiv preprint arXiv:2105.07190*, 2021.

[2] M. Mitchell, S. Wu, A. Zaldivar, et al., "Model cards for model reporting," in *FAT**, 2019.

[3] T. Gebru, J. Morgenstern, B. Vecchione, et al., "Datasheets for datasets," *Communications of the ACM*, 2021.

[4] M. Schlegel and K.-U. Sattler, "Mlflow2prov: Extracting provenance from machine learning experiments," in *DEEM*, 2023.

[5] Google Cloud, *Vertex ai platform*, Accessed: 2025-07-28, 2025. [Online]. Available: `https://cloud.google.com/vertex-ai`.

[6] S. Mohseni, N. Zarei, and E. D. Ragan, "A multidisciplinary survey and framework for design and evaluation of explainable ai systems," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 11, no. 3-4, pp. 1–45, 2021.

[7] I. D. Raji, A. Smart, R. N. White, et al., "Closing the ai accountability gap: Defining an end-to-end framework for internal algorithmic auditing," in *FAT**, 2020.

[8] J. Metcalf, E. Moss, E. A. Watkins, R. Singh, and M. C. Elish, "Algorithmic impact assessments and accountability: The co-construction of impacts," in *FAccT*, 2021.

[9] H. Birkholz, D. Thaler, M. Richardson, N. Smith, and W. Pan, *Remote attestation procedures (rats) architecture*, Internet-Draft draft-ietf-rats-architecture-22, Work in progress, 2022. [Online]. Available: `https://datatracker.ietf.org/doc/draft-ietf-rats-architecture/22/`.

[10] Secure Systems Lab at NYU, *DSSE: Dead simple signing envelope*, 2024. Accessed: Mar. 12, 2026. [Online]. Available: `https://github.com/secure-systems-lab/dsse/blob/master/background.md`.

[11] European Parliament and Council of the EU. "Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence and amending Regulations (EC) No 300/2008, (EU) No 167/2013, (EU) No 168/2013, (EU) 2018/858, (EU) 2018/1139 and (EU) 2019/2144 and Directives 2014/90/EU, (EU) 2016/797 and (EU) 2020/1828 (Artificial Intelligence Act)," Accessed: Jul. 19, 2024. [Online]. Available: `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=OJ:L_202401689`.

[12] H. Hermanns, A. Lauber-Rönsberg, P. Meinel, et al., "AI act for the working programmer," in *AISoLA*, 2024.

[13] F. Croce, M. Andriushchenko, V. Sehwag, et al., "Robustbench: A standardized adversarial robustness benchmark," *arXiv preprint arXiv:2010.09670*, 2020.

[14] M. Zheng, X. Yan, Z. Zhu, H. Chen, and B. Wu, "Blackboxbench: A comprehensive benchmark of black-box adversarial attacks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[15] H. Zhang, T. Furon, L. Amsaleg, and Y. Avrithis, "Deep neural network attacks and defense: The case of image classification," *Multimedia Security*, 2022.

[16] Z. Tian, L. Cui, J. Liang, and S. Yu, "A comprehensive survey on poisoning attacks and countermeasures in machine learning," *ACM Computing Surveys*, 2022.

[17] C. Zhu, W. R. Huang, H. Li, et al., "Transferable clean-label poisoning attacks on deep neural nets," in *PMLR*, 2019.

[18] S. Biswas and H. Rajan, "Fair preprocessing: Towards understanding compositional fairness of data transformers in machine learning pipeline," in *SIGSOFT FSE*, 2021.

[19] A. Tawakuli and T. Engel, "Make your data fair: A survey of data preprocessing techniques that address biases in data towards fair ai," *Elsevier JER*, 2024.

[20] M. Hort, Z. Chen, J. M. Zhang, et al., "Bias mitigation for machine learning classifiers: A comprehensive survey," *Journal on Responsible Computing*, 2024.

[21] R. K. Bellamy, K. Dey, M. Hind, et al., "Ai fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias," *Jour. of Research and Dev.*, 2019.

[22] R. Dwivedi, D. Dave, H. Naik, S. Singhal, R. Omer, P. Patel, B. Qian, Z. Wen, T. Shah, G. Morgan, et al., "Explainable ai (xai): Core ideas, techniques, and solutions," *ACM computing surveys*, vol. 55, no. 9, pp. 1–33, 2023.

[23] O. Dijk, R. Bell, B. Serna, et al., "Oegedijk/explainerdashboard: V0. 3.8. 2: Reverses set_shap_values bug introduced in 0.3. 8.1," *Zenodo*, 2022.

[24] P. W. Koh, T. Nguyen, Y. S. Tang, et al., "Concept bottleneck models," in *International conference on machine learning*, PMLR, 2020, pp. 5338–5348.

[25] C. Chen, O. Li, D. Tao, et al., "This looks like that: Deep learning for interpretable image recognition," in *NeurIPS*, 2019.

[26] F. Tramer, A. Terzis, T. Steinke, et al., "Debugging differential privacy: A case study for privacy auditing," *arXiv preprint arXiv:2202.12219*, 2022.

[27] T. Steinke, M. Nasr, and M. Jagielski, "Privacy auditing with one (1) training run," *NeurIPS*, 2023.

[28] H. Hu, Z. Salcic, L. Sun, et al., "Membership inference attacks on machine learning: A survey," *ACM Computing Surveys*, 2022.

[29] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE SP*, 2017.

[30] Y. Dong, R. Mu, Y. Zhang, et al., "Safeguarding large language models: A survey," *Artif. Intell. Rev.*, 2025.

[31] X. Huang, D. Kroening, W. Ruan, et al., "A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability," *Comput. Sci. Rev.*, 2020.

[32] L. de Moura and S. Ullrich, "The lean 4 theorem prover and programming language," in *Automated Deduction - CADE 28 - 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings*, A. Platzer and G. Sutcliffe, Eds., ser. Lecture Notes in Computer Science, vol. 12699, Springer, 2021, pp. 625–635. DOI: 10.1007/978-3-030-79876-5_37. [Online]. Available: https://doi.org/10.1007/978-3-030-79876-5_37.

[33] C. Hensel, S. Junges, J. Katoen, T. Quatmann, and M. Volk, "The probabilistic model checker storm," *Int. J. Softw. Tools Technol. Transf.*, vol. 24, no. 4, pp. 589–610, 2022. DOI: 10.1007/S10009-021-00633-Z. [Online]. Available: https://doi.org/10.1007/s10009-021-00633-z.

[34] M. Z. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of probabilistic real-time systems," in *Computer Aided Verification (CAV)*, 2011. DOI: 10.1007/978-3-642-22110-1_47. [Online]. Available: https://doi.org/10.1007/978-3-642-22110-1_47.

[35] T. P. Gros, H. Hermanns, J. Hoffmann, et al., "Analyzing neural network behavior through deep statistical model checking," *Int. J. Softw. Tools Technol. Transf.*, 2023.

[36] Y. Dong, Y. Wang, M. Gama, et al., "Privacy-preserving distributed learning for residential short-term load forecasting," *IEEE Internet Things J.*, 2024.

[37] C. Huang, H. Zheng, X. Huang, and K. Pei, "Statistical certification of acceptable robustness for neural networks," in *ICANN*, 2021.

[38] Y. Dong, W. Huang, V. Bharti, et al., "Reliability assessment and safety arguments for machine learning components in system assurance," *ACM Trans. Embed. Comput. Syst.*, 2023.

[39] M. Yuksekgonul, M. Wang, and J. Zou, "Post-hoc concept bottleneck models," *arXiv preprint arXiv:2205.15480*, 2022.

[40] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual explanations," in *FAT**, 2020.

[41] J. Cohen, E. Rosenfeld, and Z. Kolter, "Certified adversarial robustness via randomized smoothing," in *international conference on machine learning*, PMLR, 2019.

[42] Y. LeCun, C. Cortes, and C. Burges, "Mnist handwritten digit database," *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, vol. 2, 2010.

[43] A. Krizhevsky, G. Hinton, et al., "Learning multiple layers of features from tiny images," 2009.

[44] Y. Le and X. S. Yang, *Tiny imagenet visual recognition challenge*, 2015. [Online]. Available: `https://cs231n.stanford.edu/reports/2015/pdfs/yle_project.pdf`.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE conference on computer vision and pattern recognition*, 2016.