

Why Auditability Fails in Current Generative AI Systems: A Governance Perspective

Monika Dvorackova
Independent Researcher, Czechia

DOI:10.37819/Jaike.26.0013

Article history:

Received : 28.02.2026
Revised : 15.07.2026
Accept : 16.07.2026
Published : 21.10.2026

Corresponding author:

E-mail:
monika.dvorack@gmail.com

Abstract: Regulatory frameworks increasingly assume that generative AI systems are auditable. Yet mainstream generative AI architectures do not structurally support ex post reconstruction of system behavior for enforceable legal or regulatory oversight. This review argues that auditability fails because prevailing deployment models lack the infrastructural preconditions necessary for verifiable accountability. Drawing on literature in algorithmic accountability, AI auditing, foundation model governance, and regulatory compliance, the paper identifies structural barriers including probabilistic non-determinism, continuous model updates, API-based black-box provisioning, distributed responsibility across AI supply chains, and insufficient prompt-response trace binding. The analysis distinguishes transparency, explainability, traceability, and auditability, arguing that enforceable auditability requires immutable and reconstructible system evidence. Using a structured narrative review, the paper conceptualizes auditability as a foundational computational primitive from which accountability, compliance, and regulatory oversight can be systematically derived. This concept is operationalized through the AIGov framework and the implemented AIGov Engine, a governance-native reference architecture based on append-only evidence logging, policy-constrained execution, and deterministic compliance-state derivation. The AIGov Engine further demonstrates that these principles can support runtime governance and oversight of autonomous and multi-agent systems.

Keywords: Generative AI, auditability, AI governance, algorithmic accountability, foundation models, regulatory compliance, AI architecture, risk management, governance by design

1. INTRODUCTION

Generative artificial intelligence systems have evolved from experimental research artifacts into infrastructure supporting legally, economically, and socially consequential activities. Large language models are increasingly used in legal services, healthcare, finance, public administration, and enterprise decision support, where system outputs may influence rights, obligations, financial interests, and regulatory exposure.

This expansion has been accompanied by growing governance expectations. Frameworks such as the OECD AI Principles (OECD, 2019), the NIST AI Risk Management Framework and Generative

AI Profile (NIST, 2023, 2024), and the European Union AI Act (European Union, 2024) emphasize accountability, transparency, traceability, oversight, and documentation. Despite differences in scope and legal force, these frameworks share a common assumption: that AI systems can be meaningfully audited.

Auditability requires more than visibility into system behavior. In governance contexts, it implies that relevant decisions and actions can be reconstructed and independently verified using preserved evidence. For contemporary generative AI systems, this includes not only prompts and outputs but also model versions, contextual information, retrieved data, delegated actions, tool invocations, governance

controls, and execution history. Auditability therefore serves as the operational foundation upon which accountability and regulatory oversight depend (Kroll et al., 2017).

This article argues that dominant generative AI deployment architectures often lack the structural conditions necessary to support such auditability. The challenge is not primarily regulatory. Rather, it reflects a mismatch between governance expectations and architectural design priorities. Contemporary systems are typically optimized for performance, scalability, and adaptability rather than evidentiary reconstruction. As generative AI deployments increasingly incorporate retrieval systems, external tools, autonomous execution, and multi-agent coordination, this mismatch becomes more pronounced (HAI, 2025).

The article therefore asks: under what technical and institutional conditions can generative AI systems be considered auditable in a manner consistent with enforceable governance?

To answer this question, the article develops a conceptual framework for auditability, examines the technical characteristics of generative AI systems that impede reconstruction, derives architectural requirements for governance-native auditability, and presents AIGov as an implemented reference architecture demonstrating the feasibility of these requirements in practice.

This article makes three contributions:

1. It defines auditability as reconstructibility grounded in preserved and verifiable evidence.
2. It identifies a structural mismatch between governance assumptions and dominant generative AI deployment architectures.
3. It introduces AIGov as a governance-native architectural model that operationalizes auditability through evidence preservation, policy-constrained execution, deterministic compliance-state derivation, and runtime governance mechanisms.

2. TRANSPARENCY, EXPLAINABILITY, TRACEABILITY, AND AUDITABILITY

Governance discourse frequently treats transparency, explainability, traceability, and auditability as closely related concepts. Although interconnected, they address distinct governance objectives and impose different technical requirements.

2.1. Transparency

Transparency concerns disclosure. It focuses on communicating system characteristics, intended uses, limitations, governance measures, and risk-management practices through mechanisms such as model cards, system documentation, and impact assessments (Mitchell et al., 2019). Transparency improves visibility but does not guarantee reconstruction of specific decisions or interactions.

2.2. Explainability

Explainability concerns interpretive understanding of model behavior (Doshi-Velez and Kim, 2017). Techniques such as feature attribution, surrogate models, and attention analysis may provide insight into how outputs are produced. However, explainability does not necessarily preserve the evidence required to reconstruct a specific execution history and therefore does not by itself satisfy audit requirements.

2.3. Traceability

Traceability concerns lifecycle provenance, including model versions, datasets, deployment changes, and system updates. While traceability is necessary for auditability, it often fails to capture runtime elements such as prompts, retrieved context, tool invocations, delegated actions, and execution state.

2.4. Auditability

Auditability requires independent reconstruction of governance-relevant system behavior using preserved and verifiable evidence (Raji et al., 2020). An auditable system must support verification of execution context, system state, governance controls, delegated actions, and compliance-relevant outcomes. Unlike transparency, explainability, or traceability, auditability requires evidentiary permanence, integrity verification, and reproducible reconstruction.

As argued by Kroll et al., (2017), accountability depends on the ability to contest and evaluate decisions.

Auditability therefore functions as the infrastructural prerequisite for enforceable accountability in generative AI systems.

Formal Definition: Let S denote an AI system, D the set of governance-relevant decisions, E_d the preserved evidence associated with decision d , $V(E_d)$ an integrity-verification function, and $P(E_d)$ a deterministic governance projection producing compliance state C_d .

A system is auditable if every governance-relevant decision is associated with verifiable evidence from which the corresponding governance state can be reproducibly derived:

$$A(S) = 1 \iff \forall d \in D, \exists E_d \text{ such that } V(E_d) = \text{true} \wedge P(E_d) = C_d.$$

This definition emphasizes operational reconstructibility rather than complete visibility into model internals. Auditability depends on preserved evidence sufficient to evaluate contested decisions and derive governance relevant conclusions through reproducible procedures.

3. RELATED WORK: ALGORITHMIC ACCOUNTABILITY AND GOVERNANCE

Prior research has examined accountability, transparency, documentation, and governance challenges in AI systems. A recurring finding is that meaningful accountability requires more than disclosure; it requires mechanisms enabling independent evaluation and reconstruction of system behavior (Kroll et al., 2017, Raji et al., 2020).

Research on foundation models further highlights challenges arising from scale, opacity, and distributed responsibility (Bommasani et al., 2021; Crawford, 2021). At the same time, the sociotechnical abstraction trap identified by Selbst et al. demonstrates how governance expectations may become detached from the technical realities of deployed systems (Selbst et al., 2019).

Existing tools and frameworks address important aspects of accountability infrastructure. Experiment tracking systems improve reproducibility, provenance frameworks support artifact relationships, and governance frameworks promote documentation and oversight. However, these approaches generally strengthen observability rather than providing a unified mechanism for independent reconstruction of system behavior.

From an audibility perspective, the central gap is the absence of architectures that combine evidence preservation, integrity verification, policy-aware

execution, and deterministic compliance-state derivation.

This gap motivates the governance-native approach developed in the remainder of this article.

4. TECHNICAL CHARACTERISTICS OF GENERATIVE AI THAT UNDERMINE AUDITABILITY

Several technical characteristics of contemporary generative AI systems complicate independent reconstruction of system behavior and therefore limit meaningful audibility.

4.1. Probabilistic Generation

Generative models produce outputs through probabilistic sampling rather than deterministic execution. Reconstructing a specific interaction therefore requires preservation of model versions, execution parameters, and contextual state associated with the original generation process.

4.2. Model Evolution and Alignment Updates

Foundation models are continuously updated through retraining, fine tuning, and alignment procedures. Without stable version binding and historical references, outputs cannot be reliably associated with the model state that generated them.

4.3. Hidden Instructions and Orchestration Logic

System prompts, routing logic, safety policies, and orchestration rules may materially influence outputs while remaining invisible to end users. Auditability therefore requires preservation of system-level instructions in addition to user inputs.

4.4. External Tools and Retrieval Systems

Modern AI deployments frequently rely on retrieval systems, external APIs, and tool invocation frameworks. Outputs may therefore depend on information and actions occurring outside the model itself, expanding the scope of evidence required for reconstruction.

4.5. Agentic and Multi-Agent Workflows

In agentic systems, execution is distributed across multiple models, tools, and delegated actors (Anthropic, 2024). Auditability consequently requires preservation of delegation chains, intervention events, tool usage, and governance controls across the complete workflow.

Together, these characteristics illustrate why auditability cannot be achieved through prompt and output logging alone. Meaningful reconstruction requires preservation of the broader execution environment in which system behavior occurs.

5. DISTRIBUTED RESPONSIBILITY AND AI SUPPLY CHAINS

Generative AI systems are deployed through multi-actor ecosystems involving model developers, infrastructure providers, API platforms, integrators, software vendors, and deploying organizations. Responsibility is therefore distributed across actors that control different parts of the technical and organizational stack.

This distribution creates a structural misalignment between evidentiary control, operational authority, and legal accountability. Organizations responsible for system outcomes may lack access to training data, model versions, alignment histories, or execution records, while upstream providers may possess critical technical artifacts without visibility into downstream deployment contexts.

As noted by Raji et al. (2020), accountability gaps emerge when governance expectations exceed available audit infrastructure. In generative AI deployments, these gaps are amplified by proprietary APIs, hosted services, retrieval systems, and external tool integrations that restrict access to governance-relevant evidence.

The challenge becomes more pronounced in agentic and multi-agent systems, where actions may be distributed across multiple models, tools, and organizations. In such environments, meaningful auditability requires interoperable evidence structures and cross-organizational audit interfaces capable of preserving causally linked actions across technical and institutional boundaries.

Auditability therefore cannot be achieved solely through local governance measures. It requires evidence preservation and accountability mechanisms that extend across the AI supply chain.

6. ARCHITECTURAL REQUIREMENTS FOR ENFORCEABLE AUDITABILITY

If auditability is to function as a meaningful governance mechanism rather than a declarative aspiration, it must be embedded directly into system architecture. The preceding analysis suggests that generative AI systems require structural modifications at the deployment layer to satisfy regulatory audit expectations.

The architectural requirements presented below are technology agnostic. They define the minimum infrastructural conditions under which generative AI systems can produce evidence sufficient for independent reconstruction and verifiable accountability.

6.1. Immutable Interaction Logging

Auditability requires preservation of governance-relevant execution evidence, including prompts, contextual inputs, model identifiers, generation parameters, governance controls, tool invocations, and delegated actions. To function as reliable evidence, these records must be protected through append-only and integrity preserving storage mechanisms.

6.2. Model Version Binding and State Preservation

Outputs must be linked to stable and verifiable model versions. Where direct access to model internals is unavailable, systems should at minimum maintain persistent identifiers that enable interactions to be associated with specific model revisions, execution environments, and delegated actions.

6.3. Contextual State Archiving

Because outputs frequently depend on conversational history, retrieved information, orchestration logic, and external tools, auditability requires preservation of the contextual state active during execution. Without such preservation, reconstruction remains incomplete even when prompts and outputs are retained.

6.4. Governance Control Pathway Logging

Governance mechanisms such as safety filters, approval gates, autonomy controls, and policy

engines should generate structured evidence describing policy evaluations, control outcomes, overrides, and intervention events. Auditability depends on visibility into decision pathways rather than final outputs alone.

6.5. Independent Audit Interfaces

Systems should expose standardized audit interfaces capable of providing verifiable evidence bundles, integrity verification, compliance-state inspection, and reconstruction of multi-step execution histories without requiring direct access to proprietary model internals.

Taken together, these requirements define auditability as a system-level property emerging from evidence preservation, version binding, contextual reconstruction, governance-event recording, and independent verification. They provide the architectural bridge between the conceptual analysis developed in earlier sections and the governance-native architecture presented below.

6.5.1. Audit Failure Scenarios

Auditability failures typically arise when systems cannot reliably reconstruct governance-relevant behavior. Common failure modes include unverifiable output origin, policy bypass, untracked model drift, supply-chain opacity, and autonomous actions that lack preserved evidence of delegation, intervention, or responsibility. These scenarios illustrate that auditability depends on maintaining reconstructible evidence across the full execution lifecycle.

7. AIGOV ENGINE: GOVERNANCE-NATIVE ARCHITECTURE

7.1. Design Objective

The objective of AIGov is to transform auditability from a documentation requirement into an operational system property. Rather than relying on post hoc reporting, the architecture embeds evidence capture, integrity verification, policy evaluation, and compliance-state derivation directly into system execution.

AIGov is designed for API-mediated and partially opaque environments in which direct access to model internals is unavailable. Auditability is therefore achieved through preservation of externally

observable interaction events, contextual metadata, governance decisions, and stable version identifiers rather than through inspection of model internals.

The architecture extends naturally to autonomous and multi-agent systems by treating delegated actions, tool invocations, autonomy controls, and intervention events as first-class governance artifacts. This enables reconstruction of complex workflows from preserved evidence rather than retrospective interpretation.

The implemented open-source prototype demonstrates the practical feasibility of this approach through append-only evidence logging, compliance-state derivation, audit verification, and runtime governance controls. The implementation is available at: <https://github.com/MonikaDvorackova/aigov-compliance-engine>

7.2. Architecture and Operational Flow

This section presents the AIGov Engine architecture as a governance-native execution model in which auditability is embedded directly into system operation. Rather than treating compliance as a post hoc assessment, the architecture derives compliance state from structured and verifiable events generated during execution.

As shown in Figure 1, system interaction produces both execution flow and evidence flow. All interactions are captured as append-only events that form the evidentiary basis for governance evaluation. Compliance state is derived from preserved artifacts rather than inferred system behavior, enabling independent verification without access to internal model implementation.

From a systems perspective, the architecture can be understood as a constrained state transition model in which permissible states are determined by the existence and validity of governance events. Auditability therefore becomes an execution property rather than an observational property.

The resulting architecture supports runtime governance, autonomous-action control, multi-agent traceability, and regulator-facing evidence export through a shared evidentiary substrate.

The architecture follows a fail-closed design philosophy. When required evidence is missing, integrity verification fails, policy evaluation blocks a transition, or mandatory governance steps are absent, the system does not infer compliance. Instead,

the resulting state is classified as **BLOCKED** or **INVALID**. Governance outcomes therefore remain directly coupled to preserved and verifiable evidence.

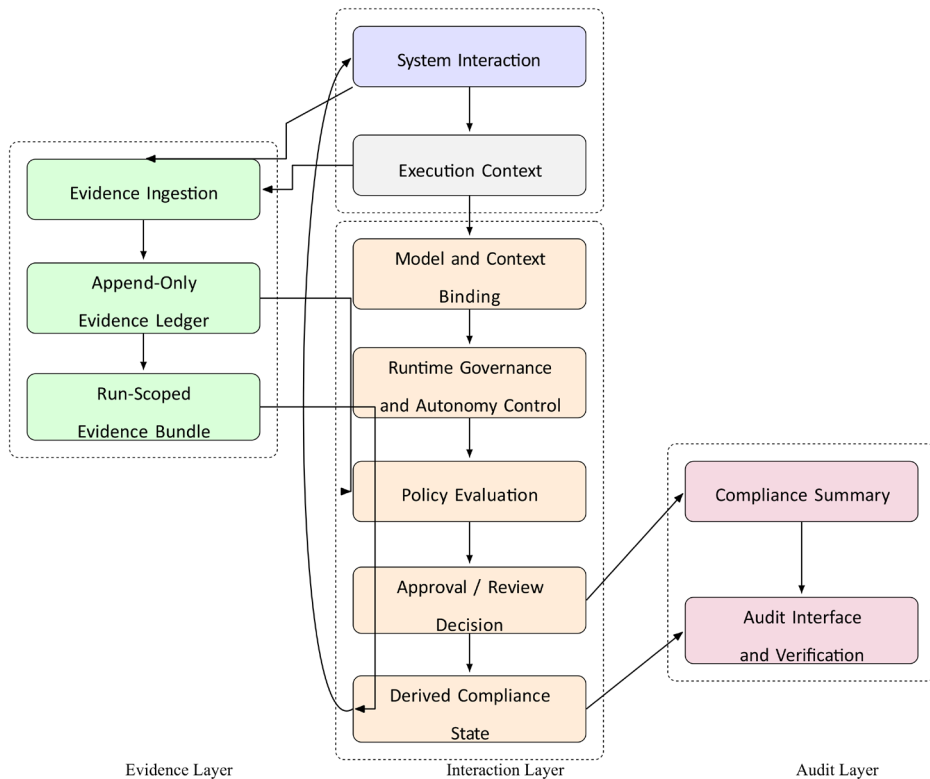


Figure 1. AIGov governance architecture showing the relationship between interaction handling, evidence capture, runtime governance, policy evaluation, compliance state derivation, and audit-facing outputs

Figure 1 shows the architecture as an evidence-first execution model. The evidence layer preserves the historical record, the governance layer derives

constraints and compliance outcomes from that record, and the audit layer exposes verifiable artifacts for review. The inclusion of runtime governance and

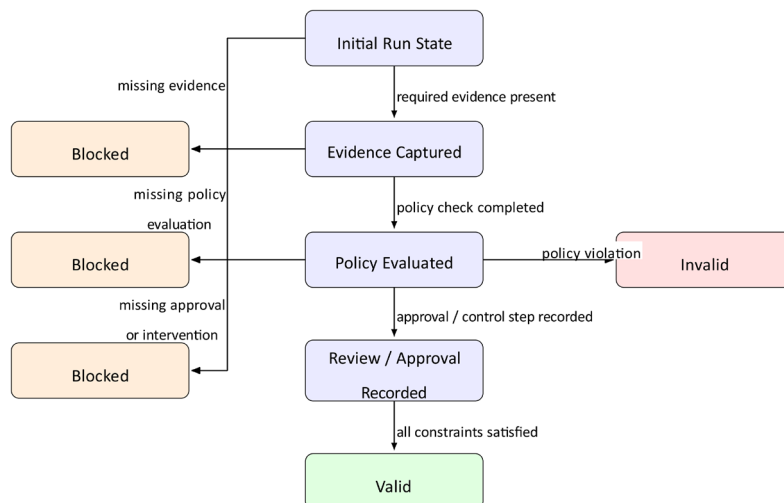


Figure 2. Simplified compliance state machine for the AIGov Engine

autonomy controls makes explicit that the same evidentiary substrate can support not only single-model inference but also tool-mediated and agentic workflows.

Governance-relevant events constrain permissible state transitions, while missing evidence, incomplete controls, or failed policy checks produce blocked or invalid outcomes under a fail-closed execution model.

Figure 2 formalizes the governance logic as a constrained state transition model. Valid states arise only when required evidence exists, policy evaluation completes, and required approval or control steps are recorded. Missing or incomplete governance steps result in blocked states, while explicit policy violations result in invalid states. This fail-closed model ensures that compliance cannot be inferred in the absence of reconstructible evidence.

7.3. Illustrative Technical Representation

The following excerpts illustrate the core auditability mechanisms implemented in the AIGov Engine. Auditability is operationalized through strongly typed event structures, append-only evidence storage, integrity verification, policy-constrained execution, and deterministic compliance projection over preserved event sequences.

The event model enforces structural consistency across recorded interactions. Each event is explicitly typed, associated with a run identifier, timestamped, and linked through integrity-preserving mechanisms. This enables deterministic reconstruction of system execution and governance-relevant decision pathways from preserved evidence.

Determinism is enforced at the level of governance evaluation rather than model generation. Generative outputs may remain probabilistic, but the compliance

```
#[derive(Serialize, Deserialize, Clone)]
pub struct Event {
    pub run_id: String,
    pub event_id: String,
    pub event_type: EventType,
    pub timestamp: DateTime<Utc>,
    pub model_version: String,
    pub environment: Environment,
    pub payload: EventPayload,
    pub previous_event_hash: String,
    pub event_hash: String,
}

#[derive(Serialize, Deserialize, Clone)]
pub enum EventType {
    InputReceived,
    PredictionGenerated,
    ToolInvoked,
    DelegationRecorded,
    EvaluationCompleted,
    ApprovalRecorded,
    PromotionDecided,
}

pub enum ComplianceState {
    Valid,
    Invalid,
    Blocked,
}

pub enum AuditError {
    BrokenChain,
    MissingEvaluation,
    ...
}
```

Listing 1. Illustrative event model and governance state types

state derived from a given evidence set is strictly deterministic. The same event sequence always yields the same governance outcome.

The evidence ledger therefore functions as an append-only, integrity-protected record of system execution. Rather than reproducing the internal numerical state of the underlying model, it preserves the externally observable artifacts required for operationally sufficient reconstruction of interactions, governance controls, and compliance outcomes.

The excerpts below correspond to mechanisms implemented in the open-source AIGov Engine reference implementation and illustrate how the architectural principles developed in this article can be realized in executable software.

This event model externalizes governance-relevant behavior into strongly typed evidence records linked through integrity-preserving mechanisms. The same structure supports reconstruction of both single-model interactions and multi-agent workflows from preserved evidence.

```
pub fn verify_hash_chain(events: &[Event]) -> Result<(), AuditError> {
    for window in events.windows(2) {
        let prev = &window[0];
        let curr = &window[1];

        if curr.previous_event_hash != prev.event_hash {
            return Err(AuditError::BrokenChain);
        }
    }
    Ok(())
}
```

Listing 2. Illustrative hash-chain linkage verification

```
pub fn evaluate_policy(events: &[Event]) -> Result<ComplianceState, AuditError> {
    let mut evaluation = None;
    let mut approval = None;
    let mut promotion = None;

    for event in events {
        match event.event_type {
            EventType::EvaluationCompleted => {
                evaluation = Some(extract_evaluation(event)?);
            }
            EventType::ApprovalRecorded => {
                approval = Some(extract_approval(event)?);
            }
            EventType::PromotionDecided => {
                promotion = Some(extract_promotion(event)?);
            }
            _ => {}
        }
    }

    let evaluation = evaluation.ok_or(AuditError::MissingEvaluation)?;
    let approval = approval.ok_or(AuditError::MissingApproval)?;
    let promotion = promotion.ok_or(AuditError::MissingPromotion)?;

    if !evaluation.passed {
        return Ok(ComplianceState::Invalid);
    }
    if !approval.granted || !promotion.allowed {
        return Ok(ComplianceState::Blocked);
    }
    Ok(ComplianceState::Valid)
}
```

Listing 3. Policy-aware compliance evaluation

Integrity verification ensures that historical evidence cannot be modified without detection. Once accepted into the ledger, any alteration of recorded events becomes observable through broken hash relationships.

Policy evaluation operates exclusively over recorded evidence rather than inferred system behavior. Each governance step is explicitly required, and missing or invalid transitions result in fail-closed outcomes. Compliance is therefore computed from observable execution rather than post hoc interpretation.

More broadly, governance rules become executable constraints over preserved evidence rather than textual declarations of intended practice. Because the same evidence sequence always yields the same outcome, auditors can independently reproduce governance conclusions without relying on institutional assertions.

The same principle extends to autonomous and multi-agent systems. Delegated actions, tool invocations, approval boundaries, and autonomy limits are treated as evidence-bearing events subject to deterministic evaluation. Auditability therefore scales from individual model calls to complex agentic workflows while preserving reconstructible and verifiable accountability.

7.3.1. Operational Policy Definition

Compliance is defined as a deterministic function over preserved evidence:

$$C = P(E), c \in \{\text{VALID}, \text{INVALID}, \text{BLOCKED}\}.$$

Governance outcomes are therefore derived from recorded evidence rather than inferred from system outputs or institutional assertions.

```
pub fn verify_and_derive(events: &[Event]) -> Result<ComplianceState, AuditError> {
    verify_hash_chain(events)?;
    evaluate_policy(events)
}
```

Listing 4. End-to-end audit verification and state derivation

The combined verification and evaluation pipeline illustrates the core architectural principle of the AIGov Engine: auditability emerges from the composition of evidentiary integrity and deterministic policy projection.

In the operational system, this logic is exposed through a single authoritative compliance summary interface and complemented by export and replay tooling. A run-scoped evidence bundle can therefore be transmitted to an independent reviewer, who can verify structural integrity, replay the policy projection, and confirm that the resulting verdict matches the original system output.

7.3.2. Failure Semantics

Failures in the AIGov Engine are categorized into two principal classes:

- **Integrity failures:** violations of structural assumptions, such as a broken hash chain or malformed evidence, which invalidate the execution trace as trustworthy audit evidence.
- **Policy failures:** absence or invalidity of required governance events, or explicit rule violations,

which result in BLOCKED or INVALID compliance states.

Integrity failures invalidate the evidentiary basis of the audit, whereas policy failures preserve the evidence while yielding non-compliant governance outcomes.

From a systems perspective, this evaluation model is equivalent to a constrained state machine in which compliance states are not manually assigned but deterministically derived from the presence and validity of preceding governance events.

Crucially, the system follows a fail-closed design philosophy. Missing, invalid, or inconsistent evidence prevents progression to compliant states, ensuring that auditability is enforced as a runtime property of system execution rather than a retrospective analytical exercise.

7.4. Quantitative Feasibility

The practical significance of governance-native auditability depends not only on conceptual coherence but also on whether the required evidentiary mechanisms can be implemented

using resources compatible with ordinary software engineering practice.

The AIGov Engine reference implementation includes benchmark tooling, storage analyses, reproducibility documentation, and deployment guidance designed to evaluate the operational feasibility of evidence-preserving governance. The objective is not to provide universal performance claims or production-scale cost estimates. Rather, the goal is to demonstrate that append-only evidence preservation, integrity verification, deterministic policy projection, and audit-bundle generation can be implemented using conventional infrastructure and engineering practices.

From a governance perspective, feasibility is primarily determined by whether evidentiary preservation scales proportionally with system activity and whether compliance-state derivation remains computationally tractable. The implemented prototype demonstrates both properties through deterministic validation, replay capability, integrity verification, and bounded storage growth.

7.4.1. Illustrative Storage Envelope

Evidence is represented as structured events appended to an append-only ledger. Let B denote the average serialized event size in bytes, E the average number of governance-relevant events per interaction, and I the number of interactions per month. A first-order estimate of monthly ledger growth is:

$$S = B \times E \times I.$$

Using representative planning assumptions documented in the reference implementation—approximately $B \approx 4\text{KiB}$, $E \approx 200$, and $I \approx 5,000$ interactions per month—the resulting primary ledger growth is on the order of:

$$S \approx 4\text{GiB/month}.$$

This estimate is illustrative rather than predictive and excludes indexing, replication, checkpoints, backup overhead, and deployment-specific retention policies. Its purpose is to demonstrate that evidence-preserving governance can be implemented with storage requirements that remain compatible with conventional deployment environments. The estimated storage requirement of approximately 4 GiB per month is intended solely as a planning estimate for moderate scale deployments and should

not be interpreted as a normative benchmark or universal deployment requirement.

7.5. Practical Constraints and Limitations

The AIGov Engine does not eliminate all auditability challenges. Several structural constraints remain:

- limited access to upstream model internals in API-based deployments,
- incomplete observability of proprietary alignment and safety processes,
- storage and retention costs associated with extensive evidence capture,
- privacy and confidentiality constraints affecting contextual preservation,
- dependence on external providers for stable version identifiers and metadata,
- jurisdiction-specific variation in evidentiary and regulatory requirements.

These limitations are not incidental. They reflect the reality that governance systems operate within partially observable and institutionally distributed environments. Auditability should therefore be understood as operational reconstructibility rather than complete epistemic transparency.

Accordingly, AIGov should be understood as a partial but operational approach to auditability. It does not claim full epistemic reconstruction of model internals, universal legal admissibility, or complete elimination of governance uncertainty. Its contribution is narrower and more practical: it demonstrates that operationally sufficient reconstruction, deterministic compliance derivation, verifiable evidence preservation, and structured governance of autonomous and multi-agent workflows can be implemented as first-class architectural properties.

In this sense, the AIGov Engine serves both as a software prototype and as an architectural argument. Auditability need not remain an aspirational governance principle or a documentation exercise. It can be implemented as executable infrastructure supported by explicit trust assumptions, measurable operational characteristics, and reproducible technical artifacts.

7.6. Illustrative End-to-End Governance Workflow

To illustrate the operational model, consider a representative governance workflow involving a generative AI system used to prepare a legally significant document.

1. A user submits a governance-relevant request.
2. Execution context, retrieved artifacts, model identifiers, and governance metadata are captured through the evidence-ingestion interface.
3. The audit service validates and stores the resulting events in the append-only evidence ledger.
4. Governance policies are evaluated and a deterministic compliance state is derived.
5. The resulting evidence bundle can be independently verified and replayed by an external auditor.

This workflow illustrates the central architectural principle of AIGov. Governance conclusions are derived from preserved evidence and can be independently reproduced by external reviewers. Auditability therefore becomes an operational property of system execution rather than a retrospective compliance exercise.

8. DISCUSSION AND IMPLICATIONS

The analysis suggests that many auditability failures in contemporary generative AI systems arise from a mismatch between governance expectations and system architecture. Governance frameworks frequently assume that contested decisions can be reconstructed and independently verified, while prevailing deployment architectures often fail to preserve the evidence necessary to support reconstruction.

The AIGov framework proposes an alternative approach in which auditability is embedded directly into the execution path through evidence preservation, integrity verification, policy enforcement, and deterministic compliance-state derivation. Under this model, governance is implemented as an operational system property rather than as a post hoc documentation activity.

At the same time, the article highlights important limitations. Audit systems cannot guarantee the truthfulness of model outputs, complete observability of proprietary model internals, or the elimination of institutional judgment. Auditability

remains constrained by provider opacity, privacy requirements, evidentiary limitations, and regulatory variation across jurisdictions. These constraints define the practical boundaries of reconstructibility in real-world deployments.

Nevertheless, the analysis demonstrates that auditability is technically realizable. The AIGov Engine reference implementation provides a concrete example of how evidence preservation, compliance projection, runtime governance controls, and audit verification can be integrated into a single governance-native architecture. While the implementation does not resolve all governance challenges, it establishes that accountability can be supported through reproducible technical mechanisms rather than relying solely on institutional representations.

More broadly, this work supports an emerging view of AI governance as an infrastructural discipline. Legal and policy requirements become operational only when translated into executable controls, preserved evidence, and verifiable system behavior. Under this perspective, auditability is a property to be engineered rather than merely documented.

The practical relevance of governance-native architectures is further reinforced by emerging regulatory frameworks. Requirements for record-keeping, traceability, incident investigation, and post-market monitoring, reflected for example in the European Union AI Act, increasingly emphasize the need for verifiable technical evidence throughout the AI system lifecycle. While compliance ultimately depends on organizational processes, legal interpretation, and deployment context, architectures that preserve immutable execution evidence and deterministic governance artifacts can substantially facilitate regulatory accountability and internal audit activities. Rather than implementing a specific legal framework, AIGov demonstrates how governance capabilities can be engineered as reusable technical infrastructure applicable across diverse regulatory environments.

Future research should evaluate governance-native architectures under large-scale production conditions, develop interoperable evidence and attestation standards, and further examine the relationship between technical audit artifacts and legal evidentiary frameworks. These questions will become increasingly important as autonomous and

multi-agent systems continue to expand the scope and complexity of AI governance.

9. CONCLUSION

This article has argued that auditability failures in contemporary generative AI systems arise from a structural mismatch between governance expectations and system architecture. While governance frameworks increasingly assume reconstructibility and independent verification, prevailing deployment models often fail to preserve the evidence required to support those objectives.

The analysis identified five core dimensions of auditability: reconstructibility, observability, institutional traceability, evidentiary integrity, and policy enforceability. Together, they demonstrate that auditability cannot be achieved solely through transparency measures, documentation, or governance reporting. Instead, it requires execution-level mechanisms that preserve evidence, verify integrity, enforce governance constraints, and support independent review.

To address this challenge, the article introduced the AIGov framework and the AIGov Engine as a governance native architecture in which compliance outcomes are derived from preserved evidence rather than institutional assertions. By combining append-only evidence preservation, deterministic compliance projection, runtime governance controls, and audit-oriented verification, the architecture demonstrates that auditability can be implemented as an operational system property.

Important limitations remain, including provider opacity, privacy constraints, and jurisdiction-specific regulatory requirements. Nevertheless, the presented architecture demonstrates that meaningful auditability can be achieved without requiring full visibility into underlying model implementations. Auditability should therefore be understood as operational reconstructibility rather than complete epistemic transparency.

Future progress will depend on interoperable evidence standards, provider attestation mechanisms, and governance-aware execution environments that support independent reconstruction and verification.

The central conclusion is straightforward: auditability should be treated as an architectural property rather than a reporting obligation. Accountability becomes operationally meaningful only when governance

claims can be supported by preserved, verifiable, and reproducible evidence.

Acknowledgements

The author received no external funding for this research.

REFERENCES

- Anthropic. (2023). *Constitutional AI: Harmlessness from AI Feedback*. arXiv preprint arXiv:2212.08073. <https://arxiv.org/abs/2212.08073>
- Anthropic. (2024). *Building Effective Agents*. Anthropic Engineering. <https://www.anthropic.com/engineering/building-effective-agents>
- Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., et al. (2021). *On the Opportunities and Risks of Foundation Models*. arXiv preprint arXiv:2108.07258. <https://arxiv.org/abs/2108.07258>
- Brundage, M., Avin, S., Wang, J., Belfield, H., Krueger, G., Hadfield, G., et al. (2020). *Toward Trustworthy AI Development: Mechanisms for Supporting Verifiable Claims*. arXiv preprint arXiv:2004.07213. <https://arxiv.org/abs/2004.07213>
- Crawford, K. (2021). *Atlas of AI: Power, Politics, and the Planetary Costs of Artificial Intelligence*. Yale University Press.
- Doshi-Velez, F., & Kim, B. (2017). *Towards a Rigorous Science of Interpretable Machine Learning*. arXiv preprint arXiv:1702.08608. <https://arxiv.org/abs/1702.08608>
- European Union. (2024). *Regulation (EU) 2024/1689 of the European Parliament and of the Council Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act)*. Official Journal of the European Union.
- Floridi, L., Cowls, J., Beltrametti, M., Chatila, R., Chazerand, P., Dignum, V., et al. (2018). AI4People: An Ethical Framework for a Good AI Society: Opportunities, Risks, Principles, and Recommendations. *Minds and Machines*, 28(4), 689–707.
- Google. (2023). *PaLM 2 Technical Report*. arXiv preprint arXiv:2305.10403. <https://arxiv.org/abs/2305.10403>

- Kroll, J. A., Huey, J., Barocas, S., Felten, E. W., Reidenberg, J. R., Robinson, D. G., & Yu, H. (2017). *Accountable Algorithms*. *University of Pennsylvania Law Review*, 165(3), 633–705.
- Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., et al. (2019). Model Cards for Model Reporting. In *Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*.
- National Institute of Standards and Technology (NIST). (2023). *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*. U.S. Department of Commerce.
- National Institute of Standards and Technology (NIST). (2024). *Generative AI Profile (NIST AI 600-1)*. U.S. Department of Commerce.
- OpenAI. (2023). *GPT-4 Technical Report*. arXiv preprint arXiv:2303.08774. <https://arxiv.org/abs/2303.08774>
- Organisation for Economic Co-operation and Development (OECD). (2019). *OECD Principles on Artificial Intelligence*. OECD Publishing.
- Raji, I. D., Smart, A., White, R. N., Mitchell, M., Gebru, T., Hutchinson, B., et al. (2020). Closing the AI Accountability Gap: Defining an End-to-End Framework for Internal Algorithmic Auditing. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*.
- Selbst, A. D., Boyd, D., Friedler, S. A., Venkatasubramanian, S., Vertesi, J. (2019). Fairness and Abstraction in Sociotechnical Systems. In *Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*.
- Stanford Institute for Human-Centered Artificial Intelligence (HAI). (2025). *AI Index Report 2025*. Stanford University. <https://aiindex.stanford.edu/report/>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention Is All You Need. In *Advances in Neural Information Processing Systems (NeurIPS)*.



Publisher's note: Eurasia Academic Publishing Group (EAPG) and AAKS remain neutral about jurisdictional claims in published maps and institutional affiliations.

Open Access. This article is licensed under a Creative Commons Attribution-NoDerivatives 4.0 International (CC BY-ND 4.0) licence, which permits copy and redistribute the material in any medium or format for any purpose, even commercially. The licensor cannot revoke these freedoms as long as you follow the licence terms. Under the following terms you must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorsed you or your use. If you remix, transform, or build upon the material, you may not distribute the modified material. To view a copy of this license, visit <https://creativecommons.org/licenses/by-nd/4.0/>.