A Formal Semantics of Governance History Validity in Encrypted Storage

Jesús F. Rodríguez-Aragón *, Carolina Zato , Fernando De la Prieta

Posted Date: 26 March 2026

doi: 10.20944/preprints202603.2102.v1

Keywords: governance validity; encrypted storage; formal semantics; state transition systems; admissibility; verifiable logs; auditability; security invariants; evidence-based verification

 doi:10.20944/preprints202603.2102.v1

Article

A Formal Semantics of Governance History Validity in Encrypted Storage

Jesús F. Rodríguez-Aragón [1,2,*], Carolina Zato [3] and Fernando De la Prieta [3]

[1] School of Engineering and Technology (ESIT), International University of La Rioja (UNIR), 26006 Logroño, Spain
[3] Department of Computer Science and Automation, University of Salamanca, 37008 Salamanca, Spain
[2] BISITE Research Group, University of Salamanca, 37008 Salamanca, Spain
* Correspondence: jesusfernando.rodriguez@unir.net

Abstract

Encrypted storage systems increasingly rely on governance mechanisms such as delegation, revocation, key updates, and policy evolution. While existing approaches provide strong guarantees for access enforcement, integrity, and transparency, they do not address a fundamental question: under which conditions can an observed sequence of governance events be accepted as a semantically valid evolution of authorization state? This work introduces a formal semantic framework for governance validity based on observable evidence. Governance is modeled as an admissibility-constrained state transition system in which events are accepted only if they satisfy explicit authorization, reference, temporal, revocation, and evidence conditions. The framework defines valid governance histories as sequences of admissible events, characterizes the conditions for deterministic state reconstruction, and establishes invariants capturing correctness properties such as revocation soundness, policy-constrained evolution, evidence completeness, non-equivocation, and temporal coherence. It also defines event-specific evidence obligations that support independent verification. The proposed approach is architecture-independent and does not prescribe specific enforcement or logging mechanisms, focusing instead on the semantic conditions required for accepting governance histories as valid from observable evidence.

Keywords: governance validity; encrypted storage; formal semantics; state transition systems; admissibility; verifiable logs; auditability; security invariants; evidence-based verification

1. Introduction

Encrypted storage systems have evolved from passive repositories into active infrastructures that enforce access control policies through cryptographic mechanisms. Modern approaches integrate techniques such as attribute-based encryption, proxy re-encryption, and policy-driven key distribution to ensure that only authorized entities can access protected data. However, beyond enforcing access restrictions, these systems must also support governance: the ability to define, modify, revoke, and audit authorization decisions over time.

Governance in encrypted environments introduces a fundamental challenge. Authorization is no longer a static property, but a dynamic process involving sequences of events such as delegations, revocations, policy updates, and evidence disclosures. These events may be distributed, partially observable, and produced by heterogeneous components. As a result, the correctness of governance cannot be determined solely by inspecting individual operations, but must instead be evaluated over entire histories of events.

Recent research has addressed different aspects of this problem. Cryptographic enforcement mechanisms ensure that unauthorized access is prevented. Transparency logs and verifiable data structures provide integrity, consistency, and non-equivocation guarantees over recorded events. Access control models define formal policy semantics. However, these approaches focus primarily on enforcement, integrity, or policy specification, rather than on the semantic interpretation of governance histories.

 doi:10.20944/preprints202603.2102.v1

This distinction gives rise to a fundamental question: given an observable sequence of governance-relevant events, under which conditions can this sequence be accepted as a valid evolution of authorization state? In other words, when can a governance history be considered semantically correct based solely on observable evidence?

Despite extensive work on access control, transparency, and verifiable systems, there is currently no formal criterion that determines when an observed sequence of governance events should be accepted as a semantically valid history based solely on observable evidence. This missing notion of acceptance is the central problem addressed in this work.

This perspective is intentionally orthogonal to architecture-centric proposals for verifiable governance in encrypted storage. System-level approaches address how governance-relevant events are generated, exposed, logged, or enforced within a concrete design. The present work addresses a different problem: under which semantic conditions an observed history can be accepted as valid, independently of the architecture or implementation mechanisms that produced it. Accordingly, the contribution of this article lies at the level of formal validity conditions and acceptance semantics rather than at the level of system construction.

In this sense, the paper should not be interpreted as a reformulation of a particular governance architecture, but as a formal account of the conditions under which governance histories can be semantically accepted. The framework is therefore purely semantic: it does not prescribe how governance events are generated, enforced, propagated, or logged in practice, but instead characterizes the conditions required for their valid interpretation once they become observable.

This work makes the following contributions:

- **Formalization of governance validity.** We introduce a formal semantic model in which governance is represented as an admissibility-constrained state transition system over observable event histories.
- **Admissibility and reconstruction.** We define admissibility predicates and state reconstruction semantics that determine how governance state evolves from observable events.
- **Global correctness conditions.** We establish invariants that characterize valid governance histories and ensure consistency, causality, and revocation correctness.
- **Evidence-based verification.** We introduce formal evidence obligations that allow independent verifiers to assess whether events can be accepted as valid.
- **Acceptance semantics for governance histories.** We define a decision framework that determines when an observed history can be accepted as semantically valid based solely on observable evidence.

To situate this problem within existing research, the following section reviews the main approaches to governance in encrypted and distributed systems and highlights the absence of a formal acceptance semantics for governance histories.

2. Related Work

The problem of governing access to protected data in encrypted and distributed systems has been studied across multiple research directions, including cryptographic enforcement, access control semantics, transparency infrastructures, and verifiable computation. While these approaches address key aspects of governance, they do not define a formal acceptance criterion for determining when an observed sequence of governance events constitutes a semantically valid history.

2.1. Cryptographic Enforcement and Encrypted Storage

Encrypted storage systems rely on cryptographic mechanisms to enforce confidentiality and access control, including early cloud storage protection models [1], encrypted query processing systems such as CryptDB [2], and Oblivious RAM constructions [3]. More expressive authorization mechanisms are provided by attribute-based encryption [4,5] and proxy re-encryption [6,7]. These approaches provide strong guarantees for access enforcement, but do not define when a sequence of governance-relevant

 doi:10.20944/preprints202603.2102.v1

operations constitutes a semantically valid evolution of authorization state from an external observer's perspective.

2.2. Access Control Models and Policy Semantics

Formal access control models such as Role-Based Access Control (RBAC) [8] and Attribute-Based Access Control (ABAC) [9] define how authorization policies are specified and evaluated. However, they typically assume a trusted policy evaluation mechanism and do not address how governance state can be reconstructed or validated from externally observable evidence. In particular, they lack a semantics of history validity that can be evaluated independently of a trusted enforcement point.

2.3. Transparency Logs and Verifiable Data Structures

Transparency mechanisms based on append-only logs and cryptographic commitments enable external verification of system behavior through inclusion and consistency proofs [10–12]. These ideas have been extended to key transparency systems [13–16], scalable log infrastructures such as Trillian [17], and auditable data structures including zero-knowledge sets [18]. While these approaches ensure integrity, consistency, and non-equivocation of recorded events, they do not define how such events should be interpreted as state transitions, nor do they provide a decision criterion for accepting a governance history as semantically valid.

2.4. Decentralized and Blockchain-Based Governance

Blockchain-based systems provide decentralized infrastructures for recording governance operations with strong guarantees of immutability and availability [19,20]. However, the presence of a transaction in a ledger does not guarantee that it corresponds to an admissible or semantically valid state transition under a formal governance model.

2.5. Verifiable Systems and Auditable Computation

Verifiable computation and auditable systems enable independent validation of computation results and system behavior [21–23]. However, these approaches focus on verifying individual computations or system executions, and do not define a global acceptance semantics over evolving governance histories.

2.6. Positioning of This Work

Existing approaches provide mechanisms for enforcing access control, exposing governance events, or verifying integrity and consistency properties, but they do not define a formal acceptance criterion that determines when an observed sequence of governance events constitutes a semantically valid history. As a result, the question of whether a governance history should be accepted as valid based solely on observable evidence remains undefined.

This work addresses this gap by introducing a formal semantic framework based on admissibility, state reconstruction, invariants, and evidence obligations. Rather than focusing on how governance mechanisms are implemented, it defines the conditions under which governance behavior—once observed—can be accepted as semantically valid by independent verifiers.

In this sense, the absence of an acceptance semantics for governance histories represents a fundamental gap that is not addressed by existing approaches.

2.7. Relationship to Architecture-Centric Governance Proposals

Beyond the general research directions discussed above, it is important to distinguish the present work from architecture-centric governance proposals. Such approaches specify how governance events are generated, enforced, logged, or made externally verifiable within a concrete system design.

By contrast, this work does not define a concrete governance architecture. Instead, it focuses on the semantic conditions under which an observed governance history can be accepted as valid, independently of the mechanisms that produced it. This distinction reflects a fundamental difference

 doi:10.20944/preprints202603.2102.v1

in objective: system-level approaches address how governance is realized, whereas the present work addresses when governance behavior can be accepted as semantically correct based on observable evidence.

3. Governance Validity Conditions

A formal model of governance validity must specify not only how governance state evolves, but also under which conditions an observed governance event is admissible and can be incorporated into a valid history. This requirement is essential for distinguishing between merely recorded events and semantically valid governance transitions.

In this framework, validity of governance histories depends on event-level admissibility. A governance event is admissible only if the following conditions hold:

- **Authorization validity.** A governance event must be issued by an actor authorized to perform the corresponding governance operation under the currently active policy state.
- **Reference validity.** Each event must refer to well-defined governance objects, such as users, resources, keys, or policies, that exist in the current or derivable governance state.
- **Temporal coherence.** The event must be consistent with the verifiable ordering of governance history. Events that contradict the observable temporal evolution of the log cannot be accepted as valid.
- **Revocation compatibility.** A governance event must not reactivate or preserve access rights that have been invalidated by prior revocation constraints unless a subsequent explicit authorization event exists.
- **Evidence sufficiency.** An event can only be accepted if it is accompanied by the minimal set of verifiable artifacts required to validate its authenticity, inclusion, ordering, and governance relevance.

These conditions define the local admissibility constraints that determine whether an event can induce a valid governance transition. The formal model introduced in the next section builds on these conditions by defining governance state, typed events, and state transition semantics.

4. Formal Governance Model

This section defines a formal model for governance validity over observable event histories. The model does not merely record governance events, but assigns them a precise semantic meaning in terms of state evolution, admissibility, and correctness-preserving reconstruction.

The model is intentionally architecture-agnostic. It does not assume any specific encrypted storage platform, logging infrastructure, transparency mechanism, or key-management design. Observable events and evidence artifacts are treated here as abstract semantic inputs to validation, not as elements tied to a particular implementation proposal.

4.1. Governance Universe

Let the governance layer operate over the following sets:

- U — the set of principals or users;
- O — the set of governed objects or encrypted resources;
- K — the set of cryptographic keys;
- P — the set of governance policies;
- E — the set of governance events;
- $\mathcal{V}$ — the set of verifiable evidence artifacts.

These sets define the domain over which governance operations are interpreted.

4.2. Governance State

The governance state at time t is defined as

 doi:10.20944/preprints202603.2102.v1

$$G_t = (A_t, P_t, K_t, R_t, M_t) \tag{1}$$

where:

- $A_t \subseteq U \times O$ is the access-assignment relation;
- P_t is the active policy state;
- K_t is the key state associated with governed objects;
- $R_t \subseteq U \times O$ is the revocation constraint set;
- M_t is the verification metadata required for governance interpretation, including log commitments, accepted evidence bindings, and freshness anchors.

The initial state G_0 is assumed to be a trusted baseline agreed upon by verifiers.

The inclusion of M_t is important because governance correctness is not determined exclusively by authorization relations, but also by the verification metadata required to justify the semantic interpretation of governance history. This metadata is modeled abstractly and is not intended to correspond to any particular implementation structure.

4.3. Typed Governance Events

A governance event is represented as a typed tuple

$$e = (\tau, \iota, s, o, \rho, \mu, \eta) \tag{2}$$

where:

- τ denotes the event type;
- $\iota \in U$ denotes the issuer of the event;
- $s \in U \cup \{\perp\}$ denotes the subject affected by the event;
- $o \in O \cup P \cup K$ denotes the governed object or policy target;
- ρ denotes an optional policy reference or authorization basis;
- μ denotes the event metadata, including ordering information;
- $\eta \subseteq \mathcal{V}$ denotes the evidence bundle associated with the event.

Typical event types include:

$$\tau \in \{grant, revoke, rotate, policy\_update, onboard\}.$$

This representation makes explicit that a governance event is not only an action record, but a typed and evidence-carrying semantic object.

4.4. Event Admissibility

Definition 1 (Event Admissibility). *Let G_t be a governance state and let $e \in E$ be a governance event. The admissibility predicate is defined as*

$$\mathrm{Adm}(G_t, e) \in \{\mathsf{true}, \mathsf{false}\}.$$

An event e is admissible in state G_t if and only if

$$\mathrm{Adm}(G_t, e) = \mathsf{true},$$

which holds if and only if the following conditions are satisfied:

- *the issuer ι is authorized under P_t;*
- *the referenced objects exist and are well-defined in G_t;*
- *the event is temporally coherent with the observed governance history;*
- *the event does not violate prior revocation constraints in R_t;*
- *the associated evidence bundle η satisfies the minimal evidence obligations required by the event type.*

 doi:10.20944/preprints202603.2102.v1

The admissibility predicate determines whether an observed event can induce a valid governance state transition.

4.5. Transition Semantics

Governance evolution is defined through a partial transition function

$$\delta : G \times E \rightharpoonup G \tag{3}$$

such that

$$G_{t+1} = \begin{cases} \delta(G_t, e_t), & \text{if } \mathrm{Adm}(G_t, e_t) = \mathsf{true}, \\ \text{undefined}, & \text{otherwise.} \end{cases} \tag{4}$$

This formulation makes governance evolution explicitly conditional on admissibility: only admissible events induce well-defined state transitions.

The intended effects of representative event classes are as follows:

- *grant*: updates A_t by introducing a new authorization relation, provided that no active revocation constraint forbids it;
- *revoke*: updates R_t and removes the corresponding authorization from A_t;
- *rotate*: updates K_t and invalidates superseded key relationships;
- *policy_update*: updates P_t subject to policy evolution constraints;
- *onboard*: extends the set of valid subjects or governance participants.

4.6. Valid Governance Histories

Definition 2 (Valid Governance History)**.** *Let G_0 be an initial governance state and let*

$$L = (e_1, e_2, \ldots, e_n)$$

be a sequence of governance events. The sequence L is a valid governance history from G_0 if and only if there exists a sequence of states

$$G_0, G_1, \ldots, G_n$$

such that for every $i \in \{1, \ldots, n\}$,

$$\mathrm{Adm}(G_{i-1}, e_i) = \mathsf{true} \quad \textit{and} \quad G_i = \delta(G_{i-1}, e_i).$$

Theorem 1 (Deterministic Reconstruction of Valid Governance Histories)**.** *Let G_0 be an initial governance state and let $L = (e_1, \ldots, e_n)$ be a valid governance history. Assume that:*

- *the admissibility predicate* Adm *is deterministic;*
- *the transition function δ is deterministic on admissible inputs.*

Then the reconstructed state G_n obtained by iteratively applying δ over L is unique.

Proof. Since L is a valid governance history, for each $i \in \{1, \ldots, n\}$ we have $\mathrm{Adm}(G_{i-1}, e_i) = \mathsf{true}$ and $G_i = \delta(G_{i-1}, e_i)$ is defined.

We prove by induction on i that each state G_i is uniquely determined.

Base case: G_0 is fixed by assumption.

Inductive step: assume G_{i-1} is uniquely determined. Since Adm is deterministic, the admissibility of e_i in G_{i-1} is uniquely determined. Since δ is deterministic on admissible inputs, the successor state $G_i = \delta(G_{i-1}, e_i)$ is uniquely determined.

By induction, the entire sequence $(G_0, \ldots, G_n)$ is uniquely determined, and therefore G_n is unique. □

 doi:10.20944/preprints202603.2102.v1

Theorem 1 establishes that governance reconstruction is well-defined for valid governance histories.

4.7. Illustrative Example of Governance State Reconstruction

Consider an initial governance state

$$G_0 = (A_0, P_0, K_0, R_0, M_0)$$

such that principal u_a is authorized under policy state P_0 to grant and revoke access to object o_1, and no revocation constraint is initially active for subject u_b. Assume also that the trusted evidence basis recorded in M_0 is valid.

Consider the following observed governance event sequence:

$$L = (e_1, e_2)$$

where

$$e_1 = (grant, u_a, u_b, o_1, \rho_1, \mu_1, \eta_1)$$

and

$$e_2 = (revoke, u_a, u_b, o_1, \rho_2, \mu_2, \eta_2).$$

Assume that both events satisfy the corresponding admissibility conditions. In particular, u_a is authorized to issue both events, the object reference o_1 is well defined in the reconstructed state, the ordering metadata in μ_1 and μ_2 is temporally coherent, and the evidence bundles satisfy

$$\eta_1 \models \Omega(grant) \quad \text{and} \quad \eta_2 \models \Omega(revoke).$$

where $\Omega(\tau)$ denotes the evidence obligation associated with event type τ, as defined in Section 6.2.

By admissibility, the first event yields

$$G_1 = \delta(G_0, e_1),$$

where the access relation (u_b, o_1) is introduced into A_1. The second event then yields

$$G_2 = \delta(G_1, e_2),$$

where the access relation is removed from A_2 and the corresponding revocation constraint is recorded in R_2.

Now consider a third observed event

$$e_3 = (grant, u_a, u_b, o_1, \rho_3, \mu_3, \eta_3)$$

presented after e_2 but without any intervening admissible re-authorization condition capable of overriding the revocation constraint in R_2. In that case,

$$\mathrm{Adm}(G_2, e_3) = \mathsf{false},$$

because accepting e_3 would violate revocation compatibility. Therefore, the transition

$$\delta(G_2, e_3)$$

is undefined, and the extended history (e_1, e_2, e_3) is not a valid governance history.

This example illustrates the admissibility-driven nature of governance state reconstruction. First, governance state is reconstructed through admissible transitions rather than by log presence alone. Second, evidence sufficiency and temporal coherence are necessary for event acceptance. Third,

 doi:10.20944/preprints202603.2102.v1

revocation soundness is enforced at the semantic level independently of log inclusion: once a valid revocation has been incorporated into reconstructed state, a subsequent grant cannot be accepted unless the model's admissibility conditions explicitly allow re-authorization.

5. Governance Invariants

The formal model introduced in the previous section defines when governance events are admissible and how governance state evolves through valid transitions. We now define the global invariants that characterize correct governance evolution.

The following invariants are defined over valid governance histories rather than isolated events. While admissibility enforces event-level validity, invariants characterize the structural correctness of reconstructed governance state evolution. Together, they specify the conditions under which reconstructed governance states can be accepted as semantically correct and independently verifiable.

5.1. State Validity

Definition 3 (State Validity)**.** *A governance state $G_t = (A_t, P_t, K_t, R_t, M_t)$ is valid if and only if it is reachable from an initial state G_0 through a valid governance history and all of its components are mutually consistent with the admissibility constraints of the model.*

This notion ensures that correctness is attached to semantically justified states rather than to arbitrary intermediate configurations.

5.2. Invariant I: Deterministic Reconstruction

Definition 4 (Deterministic Reconstruction)**.** *Deterministic reconstruction holds if and only if any two verifiers processing the same valid governance history from the same initial state reconstruct the same governance state.*

This invariant rules out semantic ambiguity in the interpretation of governance history.

5.3. Invariant II: Revocation Soundness

Definition 5 (Revocation Soundness)**.** *Revocation soundness holds if and only if, whenever a valid revocation event removes subject u's access to object o, no subsequent valid state permits the access relation (u, o) unless a later admissible re-authorization event explicitly restores it.*

This invariant captures the irreversibility of revocation in the absence of explicit subsequent authorization.

5.4. Invariant III: Policy-Constrained Evolution

Definition 6 (Policy-Constrained Evolution)**.** *Policy-constrained evolution holds if and only if every admissible governance transition is authorized by the active policy state in force at the point where the transition is applied.*

This invariant links event admissibility to policy semantics and prevents governance evolution from drifting outside the active authorization framework.

5.5. Invariant IV: Evidence Completeness

Definition 7 (Evidence Completeness)**.** *Evidence completeness holds if and only if every accepted transition in a valid governance history is supported by an evidence bundle sufficient to justify authenticity, inclusion, ordering, and governance relevance.*

This invariant distinguishes semantically accepted governance evolution from merely observed event traces.

 doi:10.20944/preprints202603.2102.v1

5.6. Invariant V: Non-Equivocation of Valid Histories

Definition 8 (Non-Equivocation). *Non-equivocation holds if and only if no two incompatible valid governance histories can be simultaneously accepted by honest verifiers while sharing a common trusted log continuity basis.*

This invariant expresses the impossibility of maintaining two divergent but simultaneously acceptable governance histories under the same verifiable continuity assumptions.

5.7. Invariant VI: Temporal Coherence

Definition 9 (Temporal Coherence). *Temporal coherence holds if and only if no valid governance history contains an event whose acceptance would contradict the verifiable ordering information required by the evidence model.*

This invariant prevents stale, replayed, or retroactively inconsistent governance events from being integrated into accepted state.

Proposition 1 (Invariant Preservation by Admissible Transitions). *Let G_t be a valid governance state satisfying Invariants I–VI, and let e_t be an event such that*

$$\mathrm{Adm}(G_t, e_t) = \mathsf{true} \quad \textit{and} \quad G_{t+1} = \delta(G_t, e_t)$$

is defined.

Assume that the admissibility predicate enforces all event-local conditions associated with authorization validity, reference validity, temporal coherence, revocation compatibility, and evidence sufficiency.

Then G_{t+1} also satisfies Invariants I–VI.

Proof. We show that each invariant is preserved under admissible transitions.

Deterministic reconstruction (Invariant I) is preserved because δ is applied to a uniquely determined predecessor state and admissibility is deterministic.

Revocation soundness (Invariant II) is preserved because admissibility enforces revocation compatibility, preventing the reintroduction of revoked access relations without explicit re-authorization.

Policy-constrained evolution (Invariant III) is preserved because admissibility requires that the issuer is authorized under the active policy state P_t.

Evidence completeness (Invariant IV) is preserved because admissibility requires that the evidence bundle satisfies the obligation $\eta \models \Omega(\tau)$.

Non-equivocation (Invariant V) is preserved because transitions are defined only over histories consistent with the trusted log continuity basis encoded in M_t.

Temporal coherence (Invariant VI) is preserved because admissibility requires consistency with the verifiable ordering of governance history.

Therefore, G_{t+1} satisfies Invariants I–VI. □

Theorem 2 (Invariant Satisfaction of Valid Histories). *Let $L = (e_1, \ldots, e_n)$ be a valid governance history from an initial state G_0, and let*

$$G_0, G_1, \ldots, G_n$$

be the corresponding reconstructed states.

Then every state G_i satisfies Invariants I–VI.

Proof. By Definition 2, each event e_i is admissible in G_{i-1} and induces a well-defined transition to G_i.

The base state G_0 is assumed to be a valid initial governance state. The result then follows by induction on i using Proposition 1. □

These invariants characterize governance correctness at the level of semantically valid state evolution, rather than at the level of event integrity alone. As illustrated in Section 4.7, this distinction is operationally

 doi:10.20944/preprints202603.2102.v1

relevant: an observed event may be present in the log and still fail to induce a valid transition if its acceptance would violate the admissibility conditions or the reconstructed governance state.

6. Governance Evidence Model and Verification Obligations

The formal semantics developed in this work requires not only that governance-relevant events be observable, but also that they be accompanied by sufficient evidence to justify their acceptance. This section defines the evidence model that supports admissibility checks, state reconstruction, and independent verification.

In this article, evidence is treated as a semantic requirement for acceptance rather than as a component of a particular system architecture. The purpose of the evidence model is therefore not to prescribe how evidence must be generated in practice, but to characterize the minimal verification conditions under which an observed event can be incorporated into a valid governance history.

6.1. Evidence Types

Let $\mathcal{V}$ denote the universe of evidence artifacts. The evidence model consists of the following classes:

- **Authenticity evidence**, such as digital signatures over governance payloads;
- **Acknowledgement evidence**, such as signed receipts linking accepted events to governance log states;
- **Inclusion evidence**, such as cryptographic proofs that an event is committed in the log;
- **Continuity evidence**, such as consistency proofs connecting successive trusted log states;
- **Ordering evidence**, such as timestamps, monotonic counters, or commitment anchors that support temporal coherence.

Each class corresponds to a distinct verification objective and serves a different role in validation.

6.2. Evidence Obligations

For each event type τ, the model defines an evidence obligation function

$$\Omega(\tau) \subseteq \mathcal{P}(\mathcal{V}) \tag{5}$$

which specifies the minimal evidence bundle required for an event of type τ to be admissible.

Intuitively, an event is evidence-complete only if its associated evidence bundle η satisfies the obligation induced by its type. This is written as

$$\eta \models \Omega(\tau).$$

These obligations are not tied to a specific system design, but represent minimal conditions required to support independent verification under standard cryptographic assumptions.

Representative obligations include:

- *grant*: authenticity + inclusion + continuity evidence;
- *revoke*: authenticity + inclusion + continuity + ordering evidence;
- *rotate*: authenticity + inclusion + continuity evidence;
- *policy_update*: authenticity + inclusion + continuity + ordering evidence.

Event types associated with stronger governance impact require correspondingly stronger evidence obligations for admissibility.

6.3. Evidence-to-Property Mapping

The purpose of explicit evidence obligations is to make verification guarantees traceable. The mapping between evidence classes and governance properties is summarized conceptually as follows:

- authenticity evidence supports issuer legitimacy;

 doi:10.20944/preprints202603.2102.v1

- inclusion evidence supports event existence in accepted history;
- continuity evidence supports append-only log evolution;
- ordering evidence supports temporal coherence;
- acknowledgement evidence supports accountability and accepted-state linkage.

This mapping clarifies that governance correctness depends on the combined sufficiency of multiple evidence classes rather than on any single artifact in isolation.

6.4. Verification Obligations of the Client

Let a verifier observe a governance event

$$e = (\tau, \iota, s, o, \rho, \mu, \eta)$$

while holding reconstructed governance state G_t. Client-side verification proceeds in two layers.

First, the verifier checks that the event is well formed and that its associated evidence can be validated relative to the trusted log basis. This includes syntactic well-formedness, validation of the relevant evidence obligations, and verification of inclusion and continuity artifacts.

Second, the verifier evaluates whether the event is admissible in the reconstructed state, that is, whether

$$\mathrm{Adm}(G_t, e) = \mathsf{true}.$$

If admissibility holds, the verifier applies the transition function and checks that the successor state is well defined. Accordingly, client-side verification corresponds to a local instance of the decision procedure later formalized for auditors.

6.5. Acceptance Semantics

Event acceptance by a verifier is defined in semantic terms through the admissibility predicate.

Operationally, a verifier first checks that the event and its associated evidence are well formed and verifiable relative to the trusted log basis. This includes syntactic well-formedness, validation of the relevant evidence obligations, and verification of inclusion and continuity artifacts.

Semantic acceptance then holds if and only if

$$\mathrm{Adm}(G_t, e) = \mathsf{true}.$$

Accordingly, admissibility captures the semantic condition for acceptance, while preliminary artifact validation provides the basis on which admissibility can be evaluated.

Theorem 3 (Soundness of Evidence-Based Reconstruction)**.** *Let $L = (e_1, \ldots, e_n)$ be a sequence of governance events processed from an initial state G_0 by a verifier that:*

- *validates that each event and its associated evidence are well formed and verifiable relative to a trusted log continuity basis;*
- *enforces the admissibility predicate* Adm*;*
- *applies the transition function δ to admissible events.*

Assume that:

- *the trusted log continuity basis is valid;*
- *the admissibility predicate* Adm *is correctly enforced;*
- *admissible transitions preserve governance invariants.*

If all events in L are accepted, then the reconstructed state G_n is a valid governance state in the sense of Definition 3.

 doi:10.20944/preprints202603.2102.v1

Proof. By assumption, each event e_i is processed only after its associated evidence has been validated relative to the trusted log basis. Semantic acceptance requires that $\mathrm{Adm}(G_{i-1}, e_i) = \mathsf{true}$, and therefore each transition $G_i = \delta(G_{i-1}, e_i)$ is well defined.

By Definition 1, admissibility ensures that all required conditions—including authorization validity, reference validity, temporal coherence, revocation compatibility, and evidence sufficiency—are satisfied at each step.

Since the trusted log continuity basis is valid, the reconstructed history is consistent with a single append-only evolution and does not arise from a forked or truncated log.

By assumption, admissible transitions preserve governance invariants. Therefore, by induction on i, each reconstructed state G_i is valid. In particular, G_n is reachable from G_0 through a valid governance history and satisfies all invariants.

Hence, G_n is a valid governance state in the sense of Definition 3. □

7. Semantic Failure Modes

The evidence model introduced in the previous section supports acceptance decisions over observable governance histories. In the formal setting developed in this work, adversarial behavior is not treated as an external collection of operational threat scenarios, but as a class of deviations from valid histories.

Accordingly, this section characterizes representative failure modes as manipulations that induce event sequences which cannot be extended to valid governance histories under the admissibility predicate and evidence model. The goal is not merely to enumerate examples, but to show that their detectability follows directly from the semantic structure of the model.

7.1. Adversarial Model

Definition 10 (Governance Manipulation)**.** *A governance manipulation is any deviation from a valid governance history that results in a sequence of observable events which cannot be extended to a valid governance history under the admissibility predicate* Adm *and the evidence model defined in Section 6.*

This definition reframes adversarial behavior in semantic rather than operational terms. Under the proposed model, an attack is considered successful only if a manipulated event sequence can still be accepted as a valid governance history.

We consider an adversary capable of manipulating the observable representation of governance evolution. The adversary may attempt to suppress governance events, present incompatible event views to different observers, reuse stale histories, or provide incomplete, inconsistent, or selectively disclosed evidence artifacts.

The adversary is assumed to have the following capabilities:

- modifying or suppressing governance events before they are observed by clients;
- presenting different governance log views to different observers;
- attempting to reuse outdated governance states;
- withholding, fragmenting, or selectively disclosing governance evidence.

However, the adversary is assumed to be unable to break standard cryptographic primitives such as digital signatures or hash functions. Clients are also assumed to perform verification procedures as defined in the governance evidence model.

Under this adversarial model, the central objective is to ensure that manipulations of governance history are semantically detectable by independent observers.

This adversarial model is consistent with those commonly adopted in transparency-based verification settings, where the observable history may be manipulated but standard cryptographic primitives remain sound. While more complex scenarios such as partial compromise, delayed disclosure, or collusion are possible, the model captures the minimal assumptions under which semantic detectability can be studied.

 doi:10.20944/preprints202603.2102.v1

7.2. Semantic Detection Result

Definition 11 (Governance Manipulation Classes). *Let L be an observed sequence of governance events. We define the following representative classes of governance manipulation:*

- *hidden revocation, where revocation events are suppressed;*
- *history forking, where incompatible event sequences are presented to different observers;*
- *log rollback, where a truncated history is presented as current;*
- *selective policy disclosure, where policy updates are inconsistently revealed.*

Each such class is understood as a representative instance of governance manipulation in the sense of Definition 10, because it induces an observed sequence that cannot be extended to a valid governance history.

Theorem 4 (Semantic Detectability of Governance Manipulation Classes). *Let L be an observed sequence of governance events with associated evidence artifacts, and let verifiers process it from an initial governance state G_0 while enforcing the admissibility predicate* Adm *and the evidence model of Section 6.*

Assume that the admissibility predicate is correctly enforced and that evidence verification is sound and that the available evidence is complete with respect to the defined obligations.

Then the following statements hold:

1. *(**Acceptance Equivalence**) L is accepted by a verifier if and only if L is a valid governance history from G_0.*
2. *(**Sound Rejection**) If L belongs to one of the governance manipulation classes of Definition 11, then there exists a minimal index i such that*

$$\mathrm{Adm}(G_{i-1}, e_i) = \mathsf{false},$$

and the verifier rejects L at step i.

Proof. (1) By Definition 2, a sequence is a valid governance history from G_0 if and only if all events are admissible and induce well-defined transitions. Since verifiers process L iteratively from G_0 and accept an event exactly when $\mathrm{Adm}(G_t, e) = \mathsf{true}$, acceptance is equivalent to validity.

(2) By Definition 11, each listed attack class induces an observed sequence that cannot be extended to a valid governance history. By Definition 2, this implies that there exists an index i for which $\mathrm{Adm}(G_{i-1}, e_i) = \mathsf{false}$. Since acceptance requires admissibility, the verifier rejects L at step i. □

Corollary 1 (Detectability of Hidden Revocation). *Any hidden revocation attack produces a sequence that violates revocation soundness and evidence completeness, and is therefore rejected by any verifier.*

Proof. Suppressing a revocation event prevents correct updates to R_t and violates Definition 5. The resulting sequence also lacks the required evidence-supported transition, conflicting with Definition 7. Hence it cannot be extended to a valid governance history and is rejected by Theorem 4. □

Corollary 2 (Detectability of History Forking). *Any governance history forking attack is rejected by at least one honest verifier.*

Proof. Forking induces incompatible histories under a shared continuity basis, violating non-equivocation (Definition 8). At least one branch cannot be extended to a valid governance history and is rejected by Theorem 4. □

Corollary 3 (Detectability of Log Rollback). *Any rollback attack is rejected by verifiers maintaining a consistent trusted continuity basis.*

Proof. Rollback produces a history inconsistent with previously validated continuity evidence, violating temporal coherence (Definition 9). The sequence is therefore invalid and rejected by Theorem 4. □

 doi:10.20944/preprints202603.2102.v1

Corollary 4 (Detectability of Selective Policy Disclosure)**.** *Selective disclosure of policy updates is rejected by verifiers.*

Proof. Inconsistent policy visibility violates policy-constrained evolution (Definition 6) and may induce divergent reconstructed states, contradicting non-equivocation. The resulting sequence is not a valid governance history and is rejected by Theorem 4. □

Remark 1 (Attacks as Violations of Validity)**.** *Theorem 4 shows that attack detection is not an additional mechanism layered on top of the governance model. Rather, it is a direct consequence of the semantic definition of validity: adversarial behavior is detectable because it induces sequences that fail admissibility and therefore cannot be extended to valid governance histories.*

8. Auditing as History Validation

The semantic model introduced in previous sections supports acceptance decisions over observable governance histories. Beyond event-by-event verification, it also supports independent auditing of governance evolution over time.

This section formalizes auditing as history validation by defining how an external observer reconstructs governance state and determines whether an evidence-supported event sequence is valid. The goal is to characterize auditing as a semantic decision process that does not require access to hidden internal state.

Definition 12 (Auditor as a Decision Procedure)**.** *An auditor is a deterministic decision procedure, whose operational realization is given in Algorithm 1, that, given an initial governance state G_0 and an observed sequence of governance events with associated evidence artifacts,*

$$L = (e_1, \ldots, e_n),$$

processes each event e_i by first checking that the event and its associated evidence are well formed and verifiable relative to the trusted log basis, then determining whether $\mathrm{Adm}(G_{i-1}, e_i) = \mathsf{true}$ *and, if so, computing the successor state*

$$G_i = \delta(G_{i-1}, e_i).$$

The auditor accepts the sequence L precisely when all events are accepted and the resulting reconstructed history is a valid governance history. Otherwise, the auditor rejects L at the first step for which preliminary artifact validation fails, admissibility fails, or the transition is undefined.

8.1. Auditing Objectives

An auditing procedure for governance histories must satisfy several objectives.

First, it must reconstruct governance evolution from the same observable events and evidence available to other verifiers, without relying on privileged internal state.

Second, it must determine whether reconstructed histories preserve the invariants defined in Section 5. In particular, it must be able to detect failures of revocation soundness, policy-constrained evolution, temporal coherence, and non-equivocation.

Third, it must provide a basis for independent validation using only observable artifacts, so that acceptance or rejection depends on explicit semantic conditions rather than on trust in the party that exposed the history.

These objectives define auditing as a semantic validation problem over evidence-supported histories.

8.2. Algorithmic History Validation

The history-validation process performed by the auditor can be expressed as a deterministic decision procedure over evidence-supported event sequences. Rather than relying on structural

 doi:10.20944/preprints202603.2102.v1

inspection of logs alone, validation requires a distinction between preliminary artifact validation and semantic acceptance relative to the reconstructed governance state.

Algorithm 1 summarizes the validation procedure used to determine whether an observed governance history is acceptable under the admissibility-constrained transition model.

Algorithm 1 Validation of an Evidence-Supported Governance History

Require: Initial governance state G_0, observed event history $L = (e_1, \ldots, e_n)$
Ensure: Accept/reject decision and reconstructed state if accepted

$G \leftarrow G_0$
for $i = 1$ to n **do**
  Let $e_i = (\tau, \iota, s, o, \rho, \mu, \eta)$
  if e_i is not well formed or its associated evidence cannot be validated relative to the trusted log basis **then**
    return reject
  end if
  if $\mathrm{Adm}(G, e_i) = \mathsf{false}$ **then**
    return reject
  end if
  if $\delta(G, e_i)$ is undefined **then**
    return reject
  end if
  $G \leftarrow \delta(G, e_i)$
end for
return accept, G

Algorithm 1 corresponds directly to the auditor defined in Definition 12. In particular, each iteration of the algorithm implements one step of the admissibility-constrained transition process. The preliminary validation step checks that the event and its associated artifacts are well formed and verifiable relative to the trusted log basis, thereby providing the conditions required to evaluate admissibility in a sound manner.

The semantic acceptance decision itself is determined by the predicate Adm, while state evolution is defined by the transition function δ. In this way, the algorithm separates artifact validation from semantic acceptance without introducing an additional layer of correctness conditions beyond those already captured by the formal model.

The correctness of this procedure follows from Proposition 2, which establishes that rejection by the auditor is equivalent to the presence of a governance violation. Accordingly, the algorithm rejects exactly those histories that cannot be extended to valid governance histories under the formal model.

More generally, Algorithm 1 provides an operational realization of Theorem 5. Under the stated assumptions on evidence verification, admissibility enforcement, and log continuity, acceptance by the algorithm implies that the processed sequence is a valid governance history and that the reconstructed state is semantically valid. Conversely, any governance manipulation induces a failure of admissibility or evidence verification at some step, leading to rejection.

This correspondence shows that the semantic notion of validity introduced in this work is not only declarative but also algorithmically checkable. The auditor therefore acts as a recognizer for the language of valid governance histories, with Algorithm 1 providing its concrete decision procedure.

8.3. Invariant Verification

After reconstructing the observable history, the auditor may inspect the reconstructed governance state in light of the invariants defined in Section 5. This step does not introduce an additional acceptance criterion beyond admissibility and well-defined transition semantics. Rather, it provides a global interpretation of the correctness guarantees already ensured by valid reconstruction.

 doi:10.20944/preprints202603.2102.v1

Let G_t denote the reconstructed governance state after processing events $(e_1, \ldots, e_t)$. By Proposition 1 and Theorem 2, if reconstruction proceeds through admissible transitions, the resulting states satisfy the governance invariants.

Accordingly, invariant verification can be understood as an explicit audit-level interpretation of deterministic reconstruction, revocation soundness, policy-constrained evolution, temporal coherence, and non-equivocation over the accepted history.

This perspective is useful because it makes the semantic consequences of acceptance directly inspectable by an external observer, even though invariant satisfaction is already entailed by the formal properties of the model.

8.4. Detection of Governance Violations

Definition 13 (Governance Violation). *Let $L = (e_1, \ldots, e_n)$ be an observed sequence of governance events. A governance violation occurs if and only if L is not a valid governance history in the sense of Definition 2.*

Proposition 2 (Auditor Correctness for Violation Detection). *Let $L = (e_1, \ldots, e_n)$ be a sequence of governance events processed by an auditor as defined in Definition 12.*

Then L contains a governance violation if and only if the auditor rejects L.

Proof. ($\Rightarrow$) If L contains a governance violation, then by Definition 13 it is not a valid governance history. By Definition 2, there exists an index i such that either $\mathrm{Adm}(G_{i-1}, e_i) = \mathrm{false}$ or the transition $\delta(G_{i-1}, e_i)$ is undefined. Therefore, the auditor rejects L at step i.

($\Leftarrow$) If the auditor rejects L at some step i, then either admissibility fails or the transition is undefined. In either case, the conditions of Definition 2 are violated, and therefore L is not a valid governance history. Hence, L contains a governance violation. □

Remark 2 (Auditor as a Recognizer of Valid Histories). *Proposition 2 admits the following interpretation. The auditing procedure induces a decision problem over event sequences, where the language of accepted sequences coincides exactly with the set of valid governance histories.*

Equivalently, the auditor acts as a recognizer for the language of valid histories defined by the admissibility-constrained transition system.

The auditing framework characterizes governance violations in semantic terms, by determining whether an observed event sequence can be accepted as a valid governance history under the formal model.

The following result operationalizes, in auditing terms, the semantic detection guarantees established in Section 7.

Theorem 5 (Soundness of Verifiable Auditing). *Let $L = (e_1, \ldots, e_n)$ be a sequence of governance events together with associated evidence artifacts, and let an auditor in the sense of Definition 12 reconstruct a sequence of states*

$$G_0, G_1, \ldots, G_n$$

by iteratively applying the transition function δ under the admissibility predicate Adm.

Assume that:

- *all evidence artifacts are verified according to the obligations defined in Section 6;*
- *the admissibility predicate* Adm *is correctly enforced;*
- *the trusted log continuity basis is valid.*

Then the following hold:

- *if the auditor accepts all events in L, then L is a valid governance history and G_n is a valid governance state;*
- *if L contains a governance manipulation, then the auditor rejects L.*

 doi:10.20944/preprints202603.2102.v1

Proof. If the auditor accepts all events in L, then by Definition 12 each event satisfies $\mathrm{Adm}(G_{i-1}, e_i) = \mathrm{true}$ and each successor state $G_i = \delta(G_{i-1}, e_i)$ is well-defined.

Therefore, by Definition 2, the sequence L is a valid governance history.

Since the trusted log continuity basis is valid and the admissibility predicate is correctly enforced, Theorem 3 implies that the reconstructed final state G_n is a valid governance state.

Conversely, if L contains a governance manipulation, then by Definition 10 it cannot be extended to a valid governance history. By Proposition 2, the auditor must therefore reject L at some step of the reconstruction procedure.

Hence, the auditing procedure is sound. □

Remark 3 (Auditing as Semantic Validation)**.** *The auditing framework validates more than the integrity of recorded events. Its function is to determine whether an observed evidence-supported event sequence can be accepted as a valid governance history under the admissibility-constrained transition system.*

Accordingly, successful auditing establishes semantic correctness of reconstructed governance evolution, whereas failed auditing identifies precisely the point at which validity breaks down.

9. Discussion

The framework developed in this work shifts the study of governance from implementation-oriented descriptions to semantic validity conditions over observable histories. Rather than focusing on how governance actions are produced, transmitted, or stored, the approach defines the conditions under which an observed sequence of events can be interpreted as a valid evolution of authorization state.

This shift of focus is essential for distinguishing the present contribution from architecture-level proposals. A system design may specify how governance is realized operationally, whereas the framework introduced here specifies how governance histories are to be interpreted and validated once they are observed. The contribution is therefore not a new encrypted storage architecture, but a semantic layer that can be used to evaluate whether the governance behavior exposed by any such architecture is semantically acceptable.

This distinction clarifies the difference between *event integrity* and *semantic correctness*. Integrity-oriented mechanisms may ensure that events are authentic, included, and consistently ordered, yet still leave open the question of whether those events correspond to admissible state transitions. By introducing explicit admissibility conditions and invariant-based validation, the proposed model offers a formal framework for distinguishing between observable event traces and valid governance evolution.

A central implication of this perspective is that verification becomes an acceptance problem over histories. An observed event is not accepted solely because it is well formed or cryptographically committed, but because it satisfies the semantic conditions required for admissibility in the reconstructed state. Likewise, a history is not considered correct merely because it is structurally consistent, but because it can be interpreted as a sequence of admissible transitions that preserves the invariants defined by the model.

The introduction of explicit evidence obligations also clarifies the role of observable artifacts in validation. Rather than treating evidence as auxiliary support for an operational mechanism, the framework defines the minimal conditions under which evidence is sufficient to justify event acceptance. This yields a direct relationship between evidence sufficiency and semantic validity, allowing correctness claims to be stated independently of any particular implementation design.

Another consequence concerns the interpretation of adversarial behavior. By defining manipulations as deviations from valid histories, the framework unifies detection and validation within a single semantic account. Under this formulation, detectability is not an external property provided by an additional monitoring layer, but a consequence of the fact that manipulated histories fail admissibility or violate invariant preservation.

 doi:10.20944/preprints202603.2102.v1

From an auditing perspective, the framework supports a principled understanding of validation as a decision problem over observable histories. Auditors do not merely inspect isolated artifacts; they determine whether an evidence-supported sequence belongs to the class of valid histories defined by the model. This yields a more precise notion of auditing as semantic recognition rather than procedural inspection.

Several limitations of the present framework remain. First, the model focuses on the semantic conditions required for validity and does not address how those conditions are guaranteed during event production. Second, the framework assumes that sufficient evidence is available to support admissibility checks; incomplete or delayed evidence may limit the ability to validate histories in practice. Third, the model abstracts away from efficiency considerations associated with long histories and repeated validation.

These limitations suggest several directions for future work. One direction involves connecting semantic validity conditions with mechanisms that constrain event production so that inadmissible histories cannot arise undetected. Another concerns the development of efficient techniques for validating long histories under realistic resource constraints. A further line of research involves automated procedures capable of continuously evaluating admissibility and invariant preservation over evolving histories.

By formalizing governance as a problem of semantic validity and acceptance, this work provides a foundation for reasoning about correctness through explicit conditions over observable histories, independently of how those histories are generated.

In practice, the applicability of the framework depends on the availability of sufficiently expressive logging and evidence mechanisms in real systems.

Importantly, the framework does not assume that governance validity is enforced at the time events are generated. Instead, it defines the conditions under which validity can be established retrospectively from observable evidence, which is a fundamentally different objective from system design.

10. Conclusions

This work introduces a formal semantic framework for reasoning about the validity of governance histories. The framework defines governance in terms of admissible events, state reconstruction, invariant preservation, and explicit evidence obligations, thereby characterizing the conditions under which an observed event sequence can be accepted as a correct evolution of authorization state.

A key contribution of this work is the distinction between observable events and valid governance evolution. Existing approaches can establish integrity and consistency properties of recorded events, but they do not by themselves define when those events induce semantically valid state transitions. By introducing admissibility conditions and invariant-based validation, the proposed model provides a formal basis for interpreting governance behavior as a well-defined semantic process.

The framework also clarifies the role of evidence in validation. By defining event-specific evidence obligations, it establishes the minimal conditions under which events can be accepted and states can be reconstructed. This enables independent observers to evaluate governance histories from observable artifacts alone, without relying on implicit trust assumptions.

In addition, the formulation of auditing as a decision procedure over histories provides a unified view of verification, reconstruction, and validation. Under this perspective, governance correctness corresponds to membership in the class of valid histories defined by the model, while deviations from correctness appear as failures of admissibility or invariant preservation.

Future work may extend this framework by connecting semantic validity conditions with event-generation constraints, by developing scalable validation techniques for long histories, and by exploring automated methods for continuous verification of governance properties.

This contribution should be understood as distinct from architecture-centric research on encrypted storage governance. Whereas system-level approaches explain how governance mechanisms can be

 doi:10.20944/preprints202603.2102.v1

built, the present work explains under which formal conditions the resulting observable histories can be accepted as semantically valid.

By establishing a formal notion of governance validity over observable histories, this work provides a formal basis for reasoning about correctness, verification, and auditability as semantic properties.

Author Contributions: Conceptualization, J.F.R.-A., C.Z. and F.D.l.P.; methodology, J.F.R.-A. and C.Z.; formal analysis, J.F.R.-A.; investigation, J.F.R.-A. and C.Z.; validation, C.Z. and F.D.l.P.; writing—original draft preparation, J.F.R.-A.; writing—review and editing, J.F.R.-A., C.Z. and F.D.l.P.; supervision, C.Z. and F.D.l.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by the NEOTEC program (CDTI, Spain) of the Spanish Ministry of Science and Innovation, under the Iberbox project (grant number SNEO-20201266).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study.

Acknowledgments: The authors acknowledge the Iberbox project for motivating the broader study of verifiable governance and evidence-based validation.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

ABAC	Attribute-Based Access Control
CDTI	Centro para el Desarrollo Tecnológico y la Innovación
RBAC	Role-Based Access Control

References

1. Kamara, S.; Lauter, K. Cryptographic Cloud Storage. In Proceedings of the Financial Cryptography and Data Security. Springer, 2010, Vol. 6054, *Lecture Notes in Computer Science*, pp. 136–149. https://doi.org/10.1007/978-3-642-14992-4_13.
2. Popa, R.A.; Redfield, C.M.S.; Zeldovich, N.; Balakrishnan, H. CryptDB: Protecting Confidentiality with Encrypted Query Processing. In Proceedings of the Proceedings of the 23rd ACM Symposium on Operating Systems Principles. ACM, 2011, pp. 85–100. https://doi.org/10.1145/2043556.2043566.
3. Stefanov, E.; van Dijk, M.; Shi, E.; Fletcher, C.W.; Ren, L.; Yu, X.; Devadas, S. Path ORAM: An Extremely Simple Oblivious RAM Protocol. In Proceedings of the Proceedings of the ACM SIGSAC Conference on Computer and Communications Security. ACM, 2013, pp. 299–310. https://doi.org/10.1145/2508859.2516660.
4. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-Policy Attribute-Based Encryption. In Proceedings of the IEEE Symposium on Security and Privacy. IEEE, 2007, pp. 321–334. https://doi.org/10.1109/SP.2007.11.
5. Yan, L.; Wang, G.; Yin, T.; Liu, P.; Hongxin, F.; Zhang, W.; Hu, H.; Pan, F. Attribute-Based Searchable Encryption: A Survey. *Electronics* **2024**, *13*. https://doi.org/10.3390/electronics13091621.
6. Green, M.; Ateniese, G. Identity-Based Proxy Re-Encryption. In Proceedings of the Applied Cryptography and Network Security. Springer, 2007, Vol. 4521, *Lecture Notes in Computer Science*, pp. 288–306. https://doi.org/10.1007/978-3-540-72738-5_19.
7. Mhiri, S.; Egio, A.; Compastié, M.; Cosio, P. Proxy Re-Encryption for Enhanced Data Security in Healthcare: A Practical Implementation. In Proceedings of the 19th International Conference on Availability, Reliability and Security, 2024, pp. 1–11. https://doi.org/10.1145/3664476.3670874.
8. Sandhu, R.S.; Coyne, E.J.; Feinstein, H.L.; Youman, C.E. Role-Based Access Control Models. *Computer* **1996**, *29*, 38–47. https://doi.org/10.1109/2.485845.

 doi:10.20944/preprints202603.2102.v1

9. Hu, V.C.; Ferraiolo, D.; Kuhn, R.; Schnitzer, A.; Sandlin, K.; Miller, R.; Scarfone, K. Guide to Attribute-Based Access Control (ABAC). Technical report, NIST, 2014. https://doi.org/10.6028/NIST.SP.800-162.
10. Merkle, R.C. A Digital Signature Based on a Conventional Encryption Function. In Proceedings of the Advances in Cryptology—CRYPTO '87. Springer, 1988, Vol. 293, *Lecture Notes in Computer Science*, pp. 369–378.
11. Crosby, S.A.; Wallach, D.S. Efficient Data Structures for Tamper-Evident Logging. In Proceedings of the USENIX Security Symposium, 2009.
12. Laurie, B.; Langley, A.; Kasper, E. Certificate Transparency. Technical report, RFC Editor, 2013. https://doi.org/10.17487/RFC6962.
13. Melara, M.S.; Blankstein, A.; Bonneau, J.; Felten, E.W.; Freedman, M.J. CONIKS: Bringing Key Transparency to End Users. In Proceedings of the 24th USENIX Security Symposium. USENIX, 2015, pp. 383–398.
14. Chase, M.; Meiklejohn, S. Transparency Overlays and Applications. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016, pp. 168–179.
15. Malvai, H.; Kokoris-Kogias, L.; Sonnino, A.; Ghosh, E.; Oztürk, E.; Lewi, K.; Lawlor, S. Parakeet: Practical Key Transparency for End-to-End Encrypted Messaging. In Proceedings of the Network and Distributed System Security Symposium, 2023.
16. Len, J.; Chase, M.; Ghosh, E.; Laine, K.; Cruz Moreno, R. OPTIKS: An Optimized Key Transparency System. In Proceedings of the USENIX Security Symposium. ACM, 2024, pp. 4355–4372.
17. Google. Trillian: A transparent, highly scalable and cryptographically verifiable data store, 2017. Available online at: https://github.com/google/trillian (accessed March 2026).
18. Chen, B.; Dodis, Y.; Ghosh, E.; Goldin, E.; Kesavan, B.; Marcedone, A.; Mou, M.E. Rotatable Zero Knowledge Sets: Post Compromise Secure Auditable Dictionaries. In Proceedings of the Advances in Cryptology—ASIACRYPT 2022. Springer, 2023, Vol. 13792, *Lecture Notes in Computer Science*, pp. 547–580. https://doi.org/10.1007/978-3-031-22969-5_19.
19. Zyskind, G.; Nathan, O.; Pentland, A. Decentralizing Privacy: Using Blockchain to Protect Personal Data. In Proceedings of the IEEE Security and Privacy Workshops. IEEE, 2015, pp. 180–184. https://doi.org/10.1109/SPW.2015.27.
20. Abdulrahman, E.; Alshehri, S.; Cherif, A. Blockchain-Based Access Control for IoT: A Survey. In Proceedings of the IEEE Asia-Pacific Conference on Computer Science and Data Engineering. IEEE, 2021, pp. 1–6. https://doi.org/10.1109/CSDE53843.2021.9718468.
21. Parno, B.; Howell, J.; Gentry, C.; Raykova, M. Pinocchio: Nearly Practical Verifiable Computation. In Proceedings of the IEEE Symposium on Security and Privacy, 2013, pp. 238–252. https://doi.org/10.1109/SP.2013.47.
22. Ben-Sasson, E.; Bentov, I.; Horesh, Y.; Riabzev, M. Scalable Zero Knowledge with No Trusted Setup. In Proceedings of the Advances in Cryptology – CRYPTO. Lecture Notes in Computer Science, 2019, Vol. 11694, pp. 701–732. https://doi.org/10.1007/978-3-030-26954-8_23.
23. Wilcox-O'Hearn, Z.; Warner, B. Tahoe: The Least-Authority Filesystem. In Proceedings of the ACM Workshop on Storage Security and Survivability. ACM, 2008, pp. 21–26. https://doi.org/10.1145/1456469.1456474.

Short Biography of Authors

J.F. Rodríguez-Aragón is an associate professor of computer science and cybersecurity at the International University of La Rioja (UNIR), Spain, and also an adjunct professor at the University of Salamanca (USAL), Spain. He has extensive experience in the design of secure cloud architectures, distributed systems, and encrypted storage platforms. He previously served as Chief Product Officer (CPO) at MEGA.nz, where he was involved in the development of large-scale client-controlled encrypted cloud storage systems. He is also the founder and former Chief Executive Officer (CEO) of Iberbox.com, a secure cloud storage platform focused on privacy-preserving data management.

 doi:10.20944/preprints202603.2102.v1

Carolina Zato is an associate professor of computer science at the University of Salamanca (USAL), Spain, and a member of the BISITE Research Group. She is involved in higher education programs in cybersecurity. She previously worked as Team Leader at MEGA.nz, contributing to the development and analysis of large-scale client-controlled encrypted cloud storage systems. Her research interests include distributed systems, multi-agent systems, secure software engineering, and the analysis of complex software infrastructures. She has participated in both academic research and industrial initiatives related to cloud platforms, software systems design, and secure distributed environments.

Fernando De la Prieta is a full professor of computer science at the University of Salamanca (USAL), Spain, and a member of the BISITE Research Group. His research focuses on distributed systems, multi-agent systems, and intelligent software architectures, with particular emphasis on the analysis, coordination, and validation of complex system behaviors. He has led and participated in numerous national and international research projects related to advanced software systems and data-driven infrastructures. His work also addresses issues of system consistency, reliability, and large-scale coordination, which are closely related to governance, verification, and the semantic interpretation of system evolution.