Spine: Forensic-Grade Audit Infrastructure for Regulated Systems

A Technical Whitepaper

Eul Bite — December 2025

Executive Summary

Regulated organizations face an unprecedented challenge: demonstrating to auditors, regulators, and courts that their operational records have not been tampered with. Traditional logging systems—designed for debugging, not evidence—fail this test completely.

New European regulations including **DORA**, **NIS2**, **MiCA**, and the **AI Act** explicitly require organizations to maintain audit trails with demonstrable integrity. The penalty for non-compliance is not just fines—it's the inability to operate.

Spine is a deployable forensic audit backbone that transforms ordinary application events into cryptographically verifiable evidence. Every record is hashed, chained, and signed—creating an immutable audit trail that can be independently verified without trusting the system that created it.

This whitepaper explains the auditability gap facing regulated organizations, why existing solutions fail, and how Spine provides the technical foundation for compliance.

Key Takeaways:

- Traditional logs offer no cryptographic proof of integrity
- DORA, NIS2, and MiCA mandate verifiable audit trails
- Spine provides tamper-evidence through hash chains and digital signatures
- Independent verification requires no trust in the running system
- Deployment is self-hosted with zero data egress

1. The Auditability Gap

1.1 The Question Organizations Cannot Answer

When a regulator, forensic investigator, or court asks: "*Prove this log entry existed at this time and has not been modified since*"—most organizations cannot provide a satisfactory answer.

They can show the log. They can explain their access controls. They can demonstrate backup procedures. But they cannot provide **cryptographic proof** that the record is authentic and unmodified.

This is the auditability gap.

1.2 Why Traditional Logs Fail

Traditional logging systems were designed for operational debugging, not forensic evidence. They have three fundamental weaknesses:

Mutable Storage Logs stored in files, databases, or SIEM platforms can be edited, deleted, or backdated by anyone with administrative access. There is no technical barrier to modification.

Unsigned Records Individual log entries carry no cryptographic signature. There is no proof of who created the record or when it was created.

Trust-Based Verification Verifying log integrity requires trusting the system that produced the logs. If that system is compromised—or if an insider has administrative access—verification is meaningless.

1.3 The Insider Threat

The most sophisticated attackers—and the most damaging compliance failures—often involve insiders. System administrators, database operators, and security personnel have legitimate access to modify logs.

When an organization says "we reviewed the logs and found no evidence of tampering," regulators increasingly ask: "*How do you know the logs weren't tampered with by someone with administrative access?*"

Traditional logging systems have no answer.

---

2. The Regulatory Landscape

2.1 DORA — Digital Operational Resilience Act

The Digital Operational Resilience Act (Regulation 2022/2554) applies to financial entities across the European Union, including banks, insurance companies, investment firms, and payment service providers.

Article 11 requires financial entities to establish policies for ICT-related incident management, including:

- Classification and recording of incidents
- Logging of ICT operations that enable detection
- Preservation of information for subsequent analysis

Article 6 mandates that ICT risk management frameworks include measures to ensure "*the integrity and availability of data*" with appropriate controls to "*prevent and detect manipulation.*"

DORA explicitly requires that audit trails be preserved in a manner that supports "*forensic analysis*" of ICT-related incidents.

2.2 NIS2 — Network and Information Security Directive

The NIS2 Directive (2022/2555) applies to essential and important entities across critical infrastructure sectors, including energy, transport, healthcare, and digital infrastructure.

Article 21 requires entities to implement risk management measures including:

- Incident handling and crisis management
- Security in acquisition, development, and maintenance of systems
- Policies and procedures to assess the effectiveness of measures

Essential entities must demonstrate to competent authorities that their security measures are effective. This requires audit trails that can be independently verified.

2.3 MiCA — Markets in Crypto-Assets Regulation

The Markets in Crypto-Assets Regulation (2023/1114) establishes requirements for crypto-asset service providers (CASPs) operating in the EU.

Article 68 requires CASPs to maintain "*orderly records of all their services, activities, orders and transactions*" sufficient to enable competent authorities to fulfill their supervisory tasks.

CASPs must preserve records for "*at least five years*" and ensure they are "*accessible and retrievable*" for regulatory review.

Given the inherently digital nature of crypto-assets and the history of exchange compromises, regulators expect CASPs to demonstrate technical controls that prevent record manipulation.

2.4 AI Act — Artificial Intelligence Act

The AI Act (2024/1689) establishes requirements for high-risk AI systems, which include systems used in critical infrastructure, employment, education, and law enforcement.

Article 12 requires high-risk AI systems to have "*logging capabilities*" that enable:

- Traceability of the system's functioning
- Monitoring of operation
- Post-market surveillance

Logs must be preserved for a "*period appropriate to the intended purpose*" and must be "*accessible*" to national competent authorities.

For AI systems making consequential decisions about individuals, the ability to demonstrate that decision records are authentic and unmodified is essential for accountability.

---

3. Why Existing Solutions Fail

3.1 SIEM Platforms

Security Information and Event Management (SIEM) platforms aggregate logs from multiple sources, apply rules, and generate alerts. They are valuable for security operations but fundamentally unsuited for forensic evidence.

The Problem: SIEMs are databases. Administrators can modify records. Even "immutable" storage tiers can typically be deleted or overwritten by privileged users.

The Gap: SIEM platforms provide no cryptographic proof of record integrity. They rely on access controls, which insider threats and compromised credentials can bypass.

3.2 Public Blockchains

Some organizations have experimented with anchoring log hashes to public blockchains (Bitcoin, Ethereum). This provides timestamping and immutability, but creates new problems.

The Problem: Public blockchains are slow (minutes to hours for confirmation), expensive (gas fees), and expose metadata publicly. They also require connectivity to external networks—problematic for air-gapped environments.

The Gap: Blockchain anchoring proves a hash existed at a point in time, but does not prove the underlying data is complete or correctly sequenced. An attacker who controls log generation can still manipulate records before anchoring.

3.3 Write-Once Storage

Write-once, read-many (WORM) storage prevents modification after writing. Some vendors offer "immutable" storage tiers with retention locks.

The Problem: WORM storage prevents modification of what was written, but does not prevent selective writing. An attacker can choose which events to log and which to suppress.

The Gap: WORM storage provides no chaining or sequencing. There is no way to detect if records are missing or out of order. There is also no signature proving authorship.

3.4 The Common Failure Mode

All existing solutions share a common failure mode: they require trusting the system that produces the logs.

If an attacker (or malicious insider) controls the logging system, they can manipulate records at the source—before they reach the SIEM, before they're anchored to a blockchain, before they're written to WORM storage.

Forensic-grade audit trails require a different approach: one that assumes the production system may be compromised and provides independent verification.

---

4. Introducing Spine

Spine is a forensic audit backbone designed to close the auditability gap. It transforms application events into cryptographically verifiable evidence that can be validated independently—without trusting the system that created the records.

4.1 Core Principles

Append-Only Storage Events are written to a Write-Ahead Log (WAL) that only supports appending. There is no update or delete operation. Modification requires rewriting the entire chain, which breaks cryptographic integrity.

Hash Chaining Every event contains a cryptographic hash (BLAKE3) of the previous event. This creates an unbreakable chain where any modification—insertion, deletion, or alteration—is mathematically detectable.

Digital Signatures Events are grouped into batches, and each batch is sealed with an Ed25519 digital signature. The signature proves authenticity and prevents repudiation.

Independent Verification A standalone verification tool can validate the entire chain by recomputing hashes and verifying signatures. This tool requires no connection to the running Spine system—only access to the data files.

4.2 What This Means for Compliance

For Auditors: Evidence that can be independently verified without trusting the organization's systems.

For Regulators: Technical controls that prevent manipulation, not just policies that prohibit it.

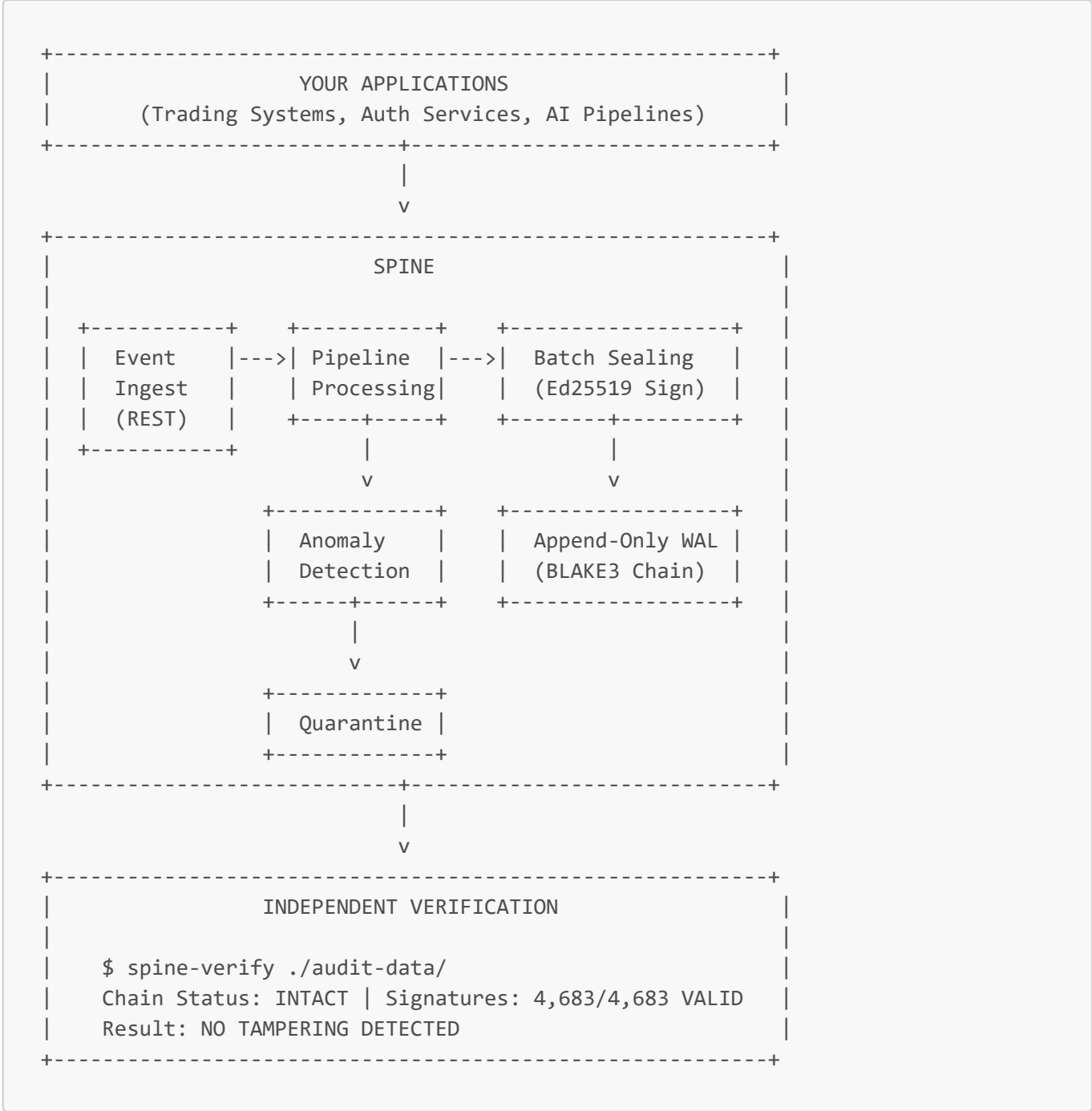
For Courts: Cryptographic proof of record integrity that meets evidentiary standards.

For Organizations: The ability to demonstrate compliance with specific, verifiable technical measures.

---

5. Architecture Overview

5.1 System Components



5.2 Event Flow

1. Ingest Applications send events to Spine via REST API. Each event includes a type, source identifier, timestamp, and payload. Spine assigns a monotonically increasing sequence number and records the precise timestamp.

2. Hash Chaining Before storage, each event is hashed using BLAKE3. The hash includes the previous event's hash, creating an unbreakable chain. Any modification to any event changes all subsequent hashes.

3. Batch Sealing Events are grouped into time-bounded batches (configurable interval, typically 30-60 seconds). Each batch is sealed with:

- A Merkle root hash (enabling efficient single-event verification)
- An Ed25519 digital signature (proving authenticity)

4. Anomaly Detection A rule-based engine evaluates events against configurable patterns. Anomalous events are automatically quarantined with full audit trail—enabling rapid incident response while preserving evidence.

5. Storage Sealed batches are written to append-only segments on the filesystem. Segments rotate based on size or time. fsync policies ensure durability.

5.3 Verification

The verification tool (`spine-verify`) reads data files directly and performs:

- **Hash Chain Validation:** Recomputes every hash and verifies linkage
- **Signature Verification:** Validates Ed25519 signatures on all batches
- **Merkle Root Verification:** Confirms batch roots match event contents
- **Sequence Verification:** Detects gaps or duplicates in sequence numbers

Verification requires only read access to data files. It does not connect to the running Spine instance. An auditor can verify records on an air-gapped machine.

---

6. Compliance Mapping

6.1 DORA Requirements

DORA Requirement	Spine Capability
Incident logging (Art. 11)	Event ingest with anomaly detection
Data integrity (Art. 6)	Hash chain prevents modification
Forensic analysis support	Independent verification tool
Record preservation	Append-only storage with retention policies

6.2 NIS2 Requirements

NIS2 Requirement	Spine Capability
Incident handling (Art. 21)	Real-time event capture and alerting
Security measure assessment	Cryptographically verifiable audit trails
Demonstrable effectiveness	Independent verification for auditors

6.3 MiCA Requirements

MiCA Requirement	Spine Capability
Orderly records (Art. 68)	Sequenced, timestamped events

MiCA Requirement	Spine Capability
5-year retention	Configurable retention policies
Accessible for review	REST API and verification tool

6.4 AI Act Requirements

AI Act Requirement	Spine Capability
Logging capabilities (Art. 12)	Event ingest from AI systems
Traceability	Hash chain with sequence numbers
Accessibility	Read API and export capabilities

---

7. Use Cases

7.1 Financial Services & Banking

Scenario: A bank's trading system executes orders based on algorithmic strategies. Regulators require proof that order execution followed the recorded sequence.

Spine Role: Every order event is captured, hashed, and signed. The hash chain proves sequencing. Signatures prove the trading system—not a later actor—generated the records.

7.2 Payment Service Providers

Scenario: A PSP processes millions of transactions daily. In case of dispute, they must prove transaction records are authentic.

Spine Role: Transaction events flow through Spine before settlement. The sealed audit trail provides forensic evidence that can be independently verified by dispute resolution bodies.

7.3 Crypto-Asset Service Providers

Scenario: A cryptocurrency exchange must demonstrate to MiCA regulators that order book and trade records are complete and unmodified.

Spine Role: Exchange events—order placement, matching, execution—are captured in Spine's hash chain. The verification tool allows regulators to independently confirm integrity.

7.4 Healthcare Systems

Scenario: A hospital's clinical decision support system recommends treatments. In case of adverse outcomes, the hospital must prove what information the system had and what it recommended.

Spine Role: Clinical events, AI recommendations, and physician overrides are captured with cryptographic integrity. The audit trail supports both patient safety reviews and legal defense.

7.5 Critical Infrastructure (OT/ICS)

Scenario: An energy grid operator must demonstrate that SCADA system commands were executed as recorded—not modified by an attacker.

Spine Role: Command and telemetry events are captured in Spine. The hash chain detects any manipulation of the operational record, supporting both incident response and regulatory reporting.

---

8. What Spine Does Not Do

Transparency about limitations builds trust. Spine is designed for a specific purpose and is not a replacement for:

8.1 Not a SIEM

Spine does not aggregate logs from multiple sources, correlate events, or generate security alerts in the traditional SIEM sense. It is designed to complement SIEMs by providing a forensic-grade evidence layer.

Recommendation: Feed critical events to both your SIEM (for operations) and Spine (for evidence).

8.2 Not a Backup Solution

Spine provides tamper-evident storage, not disaster recovery. While data is durable, organizations should maintain separate backup procedures.

Recommendation: Include Spine data in your backup strategy.

8.3 Not a Real-Time Monitoring Platform

Spine prioritizes evidence integrity over real-time alerting. While it includes anomaly detection, it is not optimized for sub-second security response.

Recommendation: Use Spine alongside your existing monitoring tools.

8.4 Not a Silver Bullet

Spine ensures that recorded events are tamper-evident. It cannot ensure that all relevant events are recorded. If your application doesn't send an event to Spine, Spine cannot capture it.

Recommendation: Audit your event coverage as part of compliance preparation.

---

9. Deployment Model

9.1 Self-Hosted Architecture

Spine deploys inside your infrastructure. There is no cloud dependency and no data egress.

Why This Matters:

- Sensitive data never leaves your perimeter
- Air-gapped deployments are fully supported
- No vendor lock-in or subscription dependency

- Full control over retention and access

9.2 Technical Requirements

Component	Specification
Runtime	Single binary (Rust, ~15MB)
OS	Linux (primary), Windows (supported)
Memory	2GB minimum, 8GB recommended
Storage	SSD recommended for write performance
Network	REST API (configurable port)

9.3 Integration Patterns

Direct API Integration Applications call Spine's REST API directly. Suitable for new systems or those undergoing compliance refactoring.

Sidecar Deployment Spine runs alongside applications in containerized environments, capturing events via local API calls.

Log Forwarding Existing log streams can be forwarded to Spine via standard protocols (syslog, HTTP). Suitable for brownfield deployments.

---

10. Next Steps

10.1 Proof of Concept

We offer structured proof-of-concept engagements to demonstrate Spine in your environment:

Duration: 2-3 weeks

Scope:

- Integration with one event source
- Basic anomaly rules configured
- Dashboard deployment
- Verification workflow demonstration

Deliverables:

- Working Spine instance in your environment
- Integration documentation
- Verification report template

No commitment required. The PoC is designed to validate fit before any commercial discussion.

10.2 Design Partnership

For organizations with complex compliance requirements, we offer design partnership arrangements to co-develop features and configurations specific to your regulatory context.

10.3 Contact

Email: contact@eulbite.com

Website: www.eulbite.com/spine

About Eul Bite

Eul Bite builds compliance-grade infrastructure for regulated systems. Our focus is on the intersection of cryptography, audit requirements, and operational reality.

Spine is our first product—designed to solve the auditability gap we observed across financial services, crypto, and critical infrastructure sectors.

We are currently operating in pre-incorporation phase, working with early design partners to refine our offering before formal company establishment.
