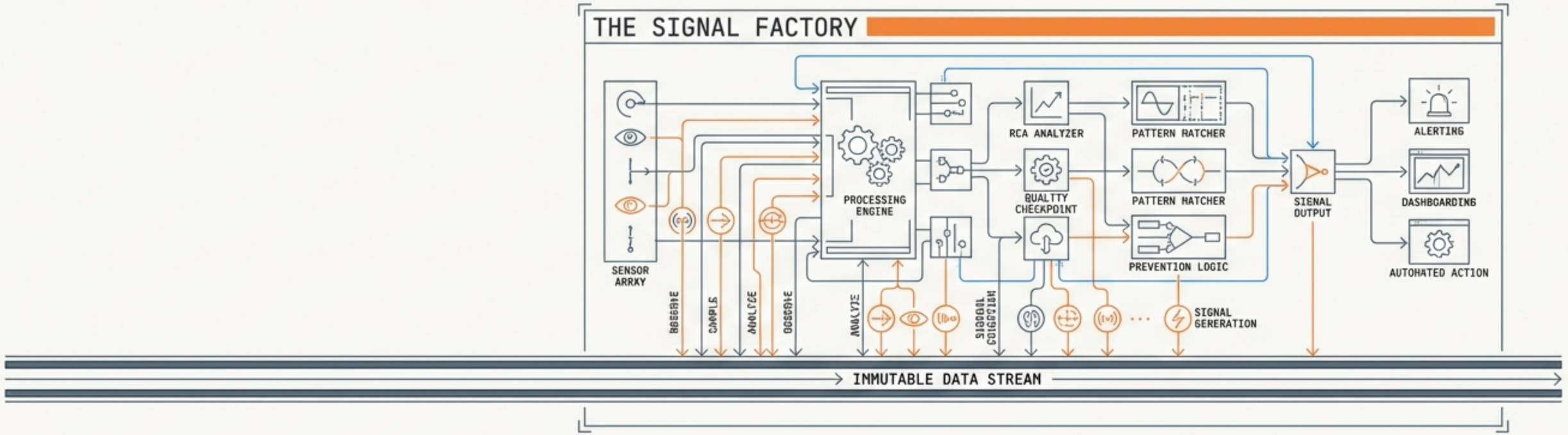
Out-of-Band Observability: The 'Signal Factory' Architecture

Achieving Prevention, Quality, and Root Cause Analysis (RCA) without Modifying Producers or the Central Platform.

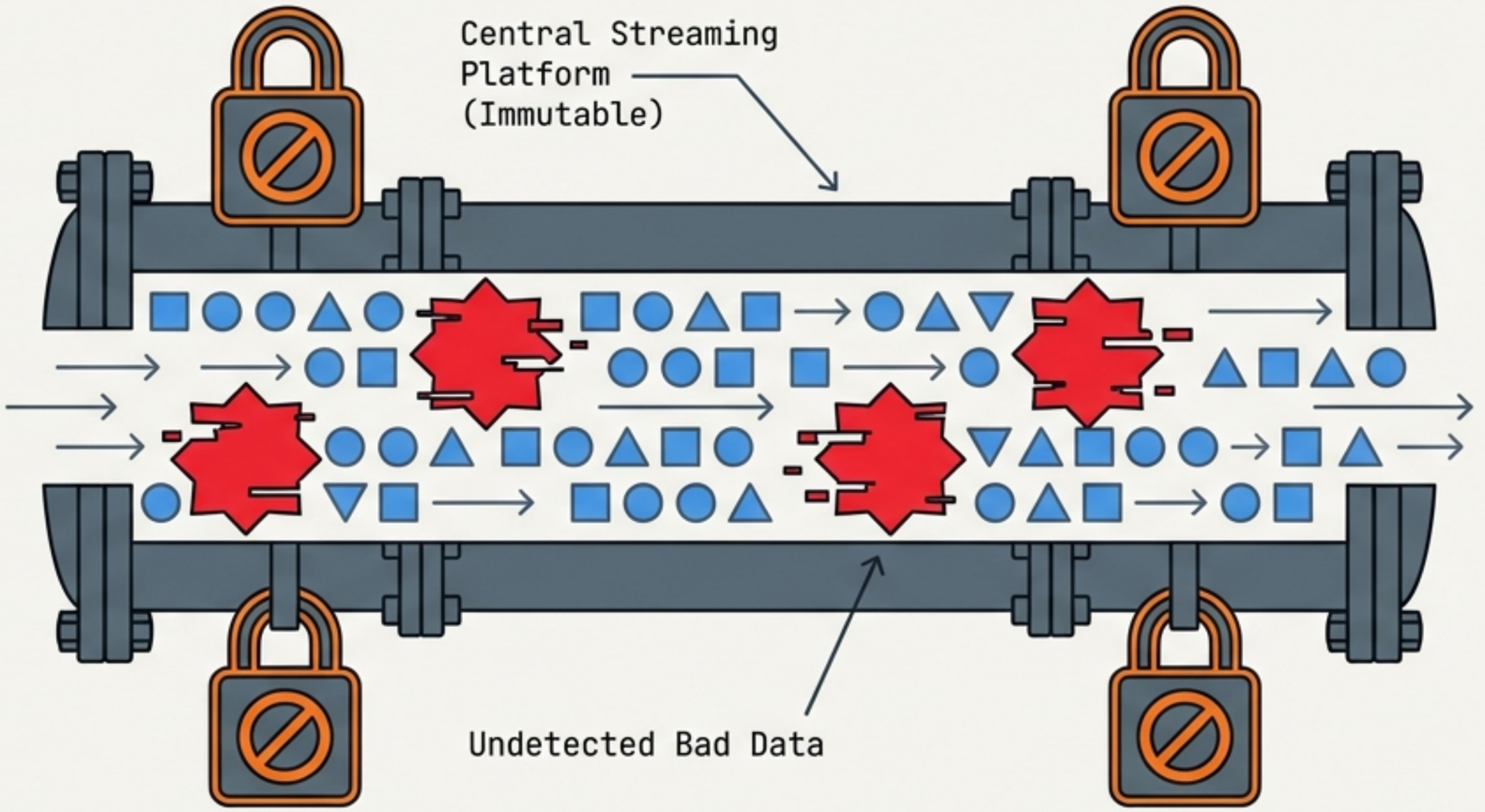


Technical Editorial slide illustration

The Challenge: Observability in an Immutable World

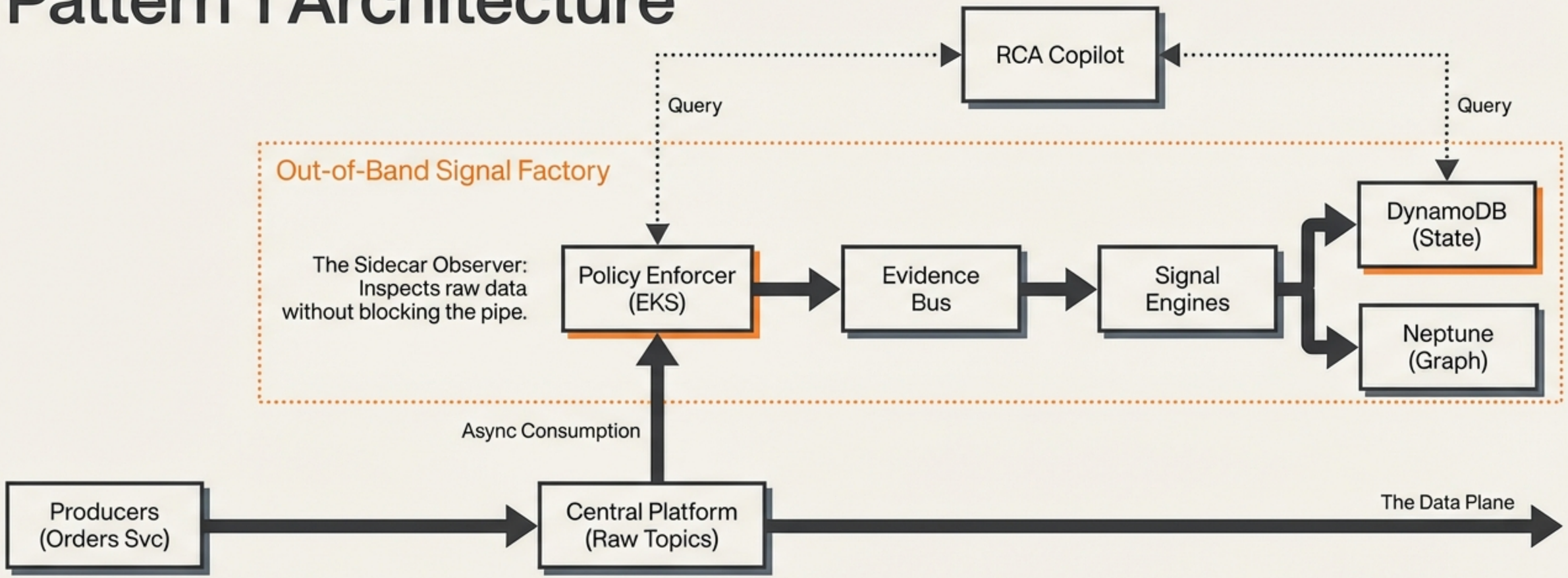
We operate in a constrained environment. The Central Streaming Platform (Kafka/MSK) acts as a rigid, immutable pipe. Producers stream business events directly into this pipe and cannot be modified to add observability SDKs. The Platform itself allows no ingress hooks or routing interception.

The Risk: Bad data (schema violations, PII leaks) flows undetected to consumers until downstream systems crash.



Technical Editorial

High-Level Solution: Pattern 1 Architecture



The Core Separation: Record Truth vs. System Health

Policy Enforcer

ESTABLISHING RECORD TRUTH

- **Scope**: Atomic. (One record at a time).
- **Responsibility**: Deterministic Validation.
- **Checks**: Schema, Contract, PII.
- **Output**: Immutable Evidence (PASS/FAIL).
- **Key Question:** “Is this specific record valid?”

Signal Engines

ESTABLISHING SYSTEM HEALTH

- **Scope**: Aggregated. (Time windows & datasets).
- **Responsibility**: Anomaly Detection.
- **Checks**: Thresholds, SLOs, Drift.
- **Output**: Signals & Incidents (SEV-1).
- **Key Question:** “Is the system healthy?”

The Journey Begins: A Flawed Event Enters the Stream

Scenario:
The Orders Service deploys a hotfix at 09:58 AM.

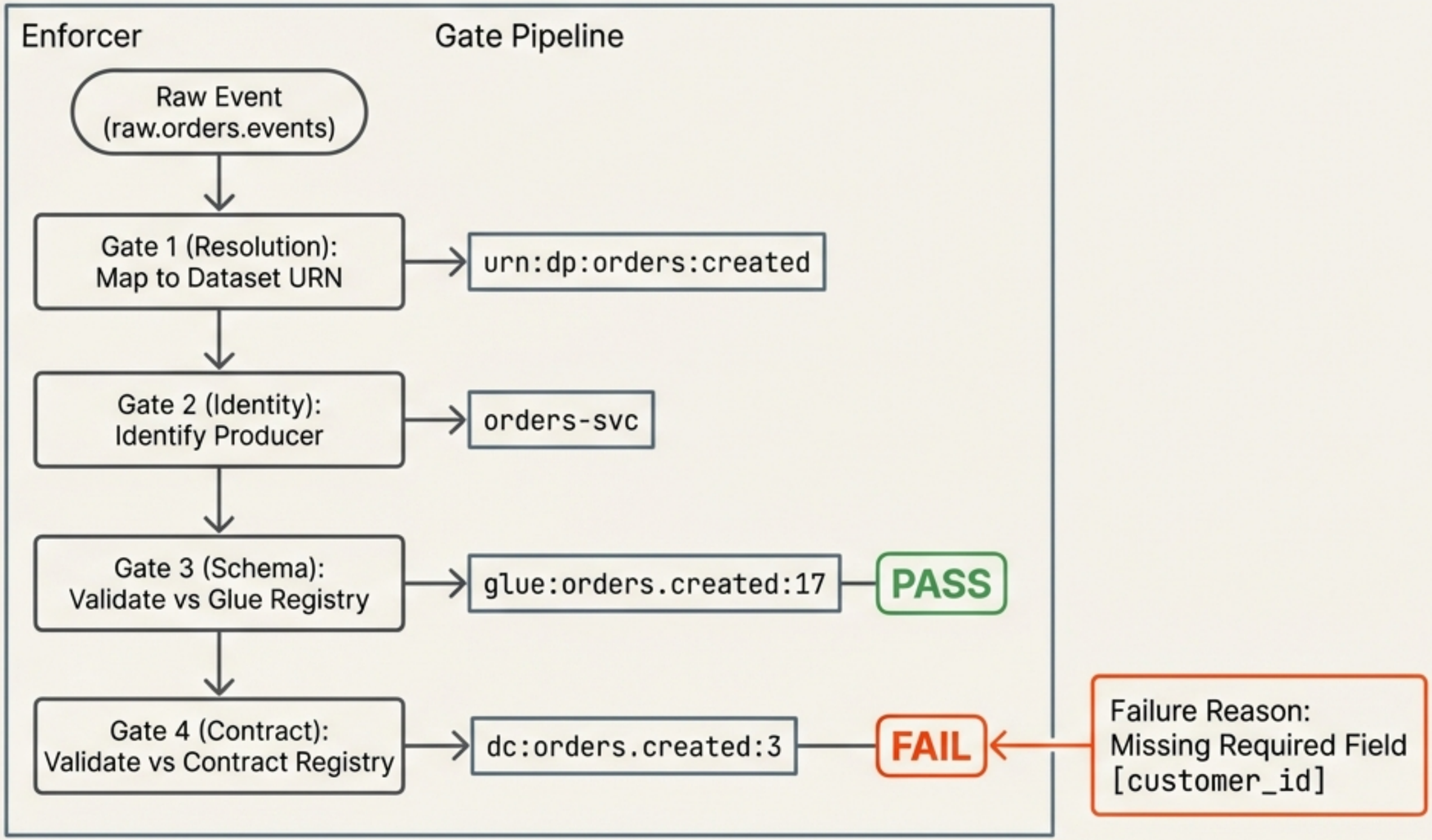
The Flaw:
The payload is missing the mandatory `customer_id` field.

Context:
The Central Platform accepts this message blindly because it is immutable. The bad data is now in the pipe.

```
{
  "topic": "raw.orders.events",
  "partition": 4,
  "offset": 9918273,
  "payload": {
    "order_id": "O-99123",
    "timestamp": "2023-10-27T09:58:01Z",
    "total_amount": 149.99,
    "currency": "USD",
    "items": [
      {"sku": "SKU-11", "qty": 1}
    ]
          <-- // MISSING: customer_id
  }
}
```

Technical Editorial

Step 2: The Policy Enforcer Establishes Truth



The Enforcer does not page on-call yet. It simply establishes the FACT of the failure.

Step 3: Canonical Evidence (The Interface)

This immutable object IS the observability product.

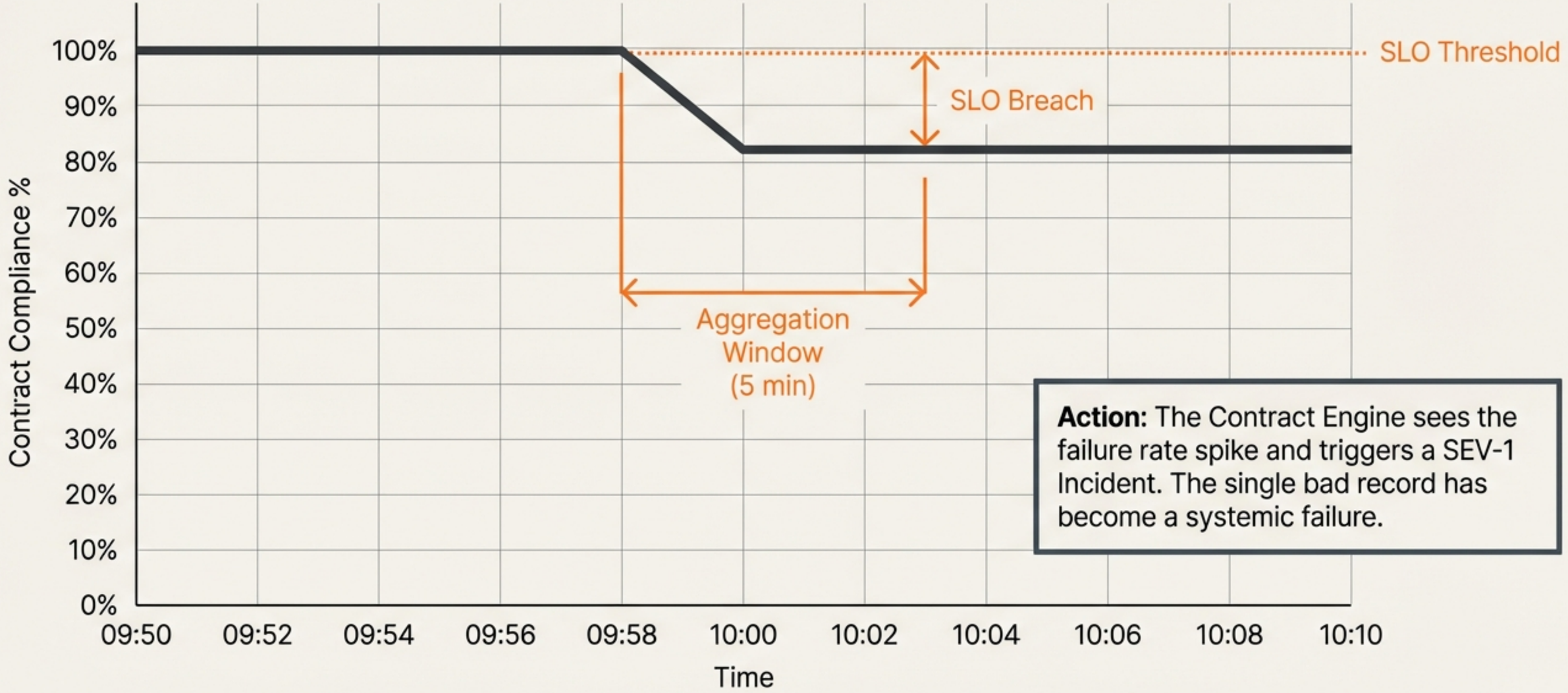
The Logical Context

The Deterministic Fact

The Join Key (Replayability)

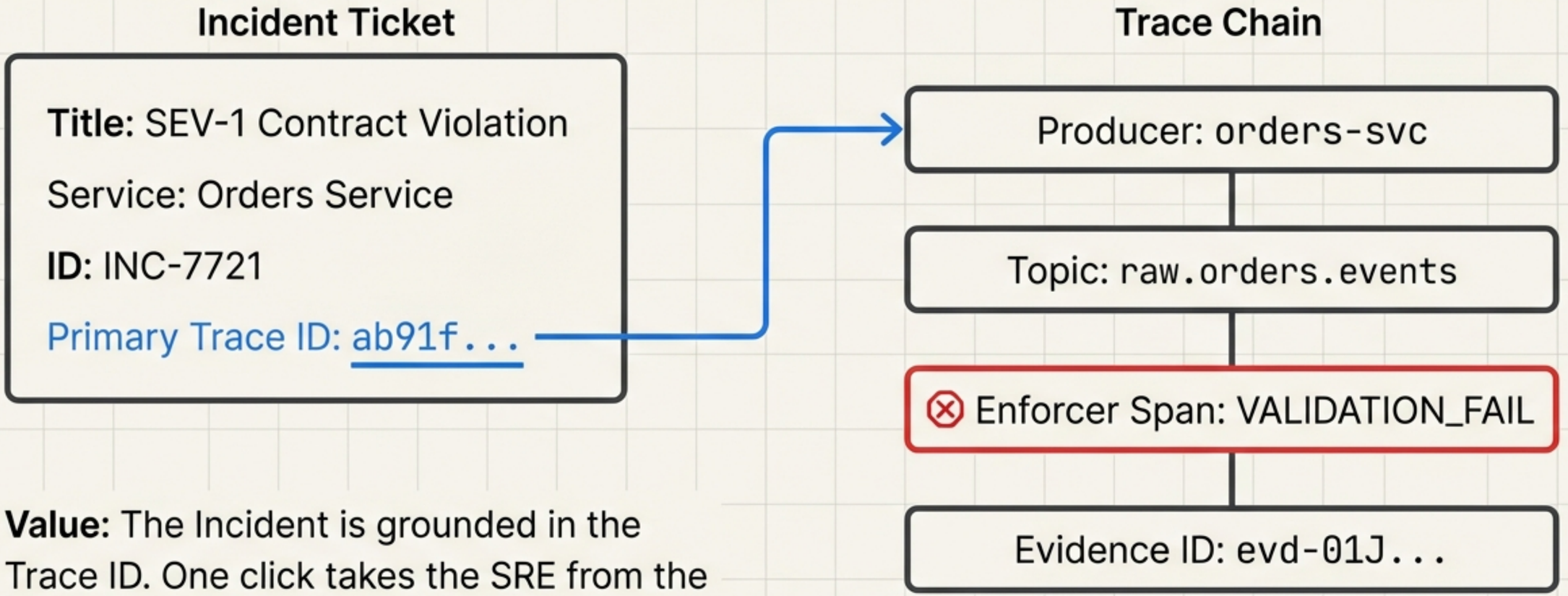
```
{
  "evidence_id": "evd-01J...",
  "timestamp": "2023-10-27T09:58:02Z",
  "dataset_urn": "urn:dp:orders:created",
  "producer": {
    "id": "orders-svc",
    "confidence": "HIGH"
  },
  "validation": {
    "result": "FAIL",
    "failed_gates": ["CONTRACT"],
    "reason_codes": ["MISSING_FIELD:customer_id"]
  },
  "source": {
    "topic": "raw.orders.events",
    "offset": 1882341
  },
  "otel": {
    "trace_id": "ab91f..."
  }
}
```

Step 4: Signal Processing Engines Establish Health



Action: The Contract Engine sees the failure rate spike and triggers a SEV-1 Incident. The single bad record has become a systemic failure.

Step 5: Incident Creation & Trace Anchoring

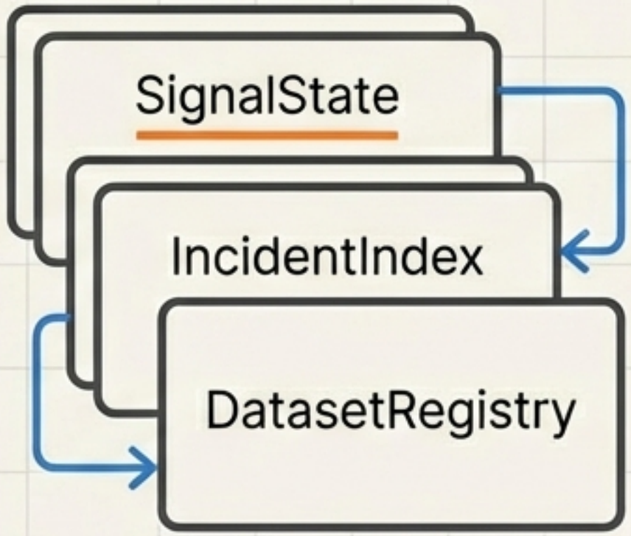


Value: The Incident is grounded in the Trace ID. One click takes the SRE from the alert to the exact sequence of events.

Data Store Architecture: State vs. Causality

DynamoDB

Operational Truth (State)

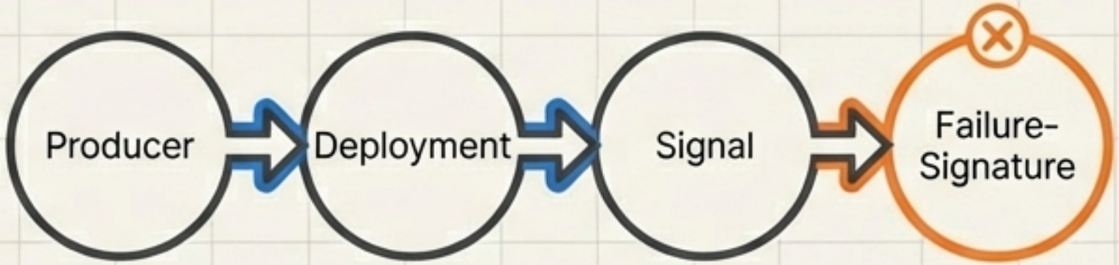


Use case: Fast Dashboards.

“What is the current health?”

Amazon Neptune

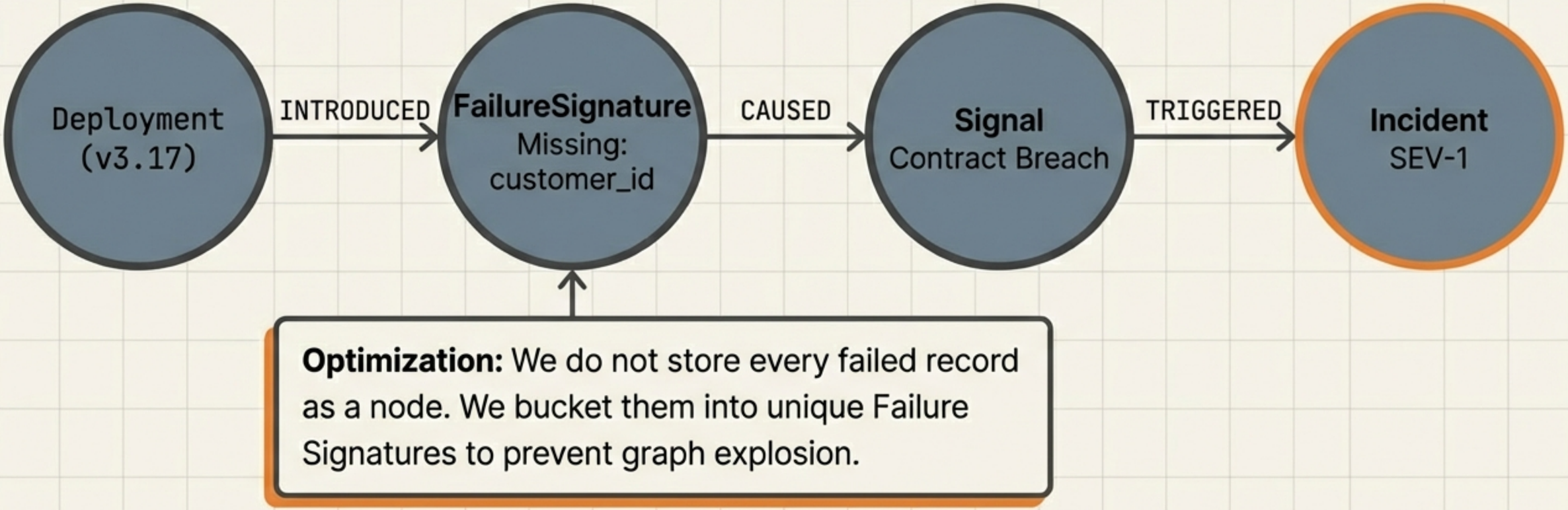
Causal Graph (Relationships)



Use case: Root Cause Analysis.

“Why did this happen?”

The Graph Model: Automatic RCA



Resilience & Failure Modes

Observability of Observability

Enforcer Down 	**Impact** Evidence Gap.	**Action** Signal Engines detect "Zero Evidence" vs Raw Lag. Trigger `OBSERVABILITY_PIPELINE_DOWN`.
Registry Down 	**Impact** Cannot validate schema.	**Action** Enforcer falls back to WARN mode. Emits evidence with `failed_gates=["REGISTRY_UNAVAILABLE"]`. Never crash the consumer.
Lag Spikes 	**Impact** Delayed signals.	**Action** HPA scales consumers. Signal Engine annotations mark data as “Stale”.

Summary of Responsibilities

Responsibility	Owner	Artifact
Record Validity	**Policy Enforcer**	Evidence (PASS/FAIL)
System Health	**Signal Engines**	Signals & Incidents
Alert Thresholds	**Signal Engines**	Dataset Policy Config
Schema Bindings	**Gateway Control Plane**	Registry Map
RCA Narrative	**RCA Copilot**	Causal Graph Path

The Result: Prevention-Grade Observability

Zero

Producer Changes Required

The Central Platform remains immutable. Enforcer runs entirely out-of-band.

< 2s

Latency

From raw ingest to Evidence emission and Signal update.

100%

Attribution

Failures are linked to producers via best-effort identity and confidence scores.

By moving enforcement to the sidecar, we gain the safety of a gateway with the flexibility of a decoupled observer.