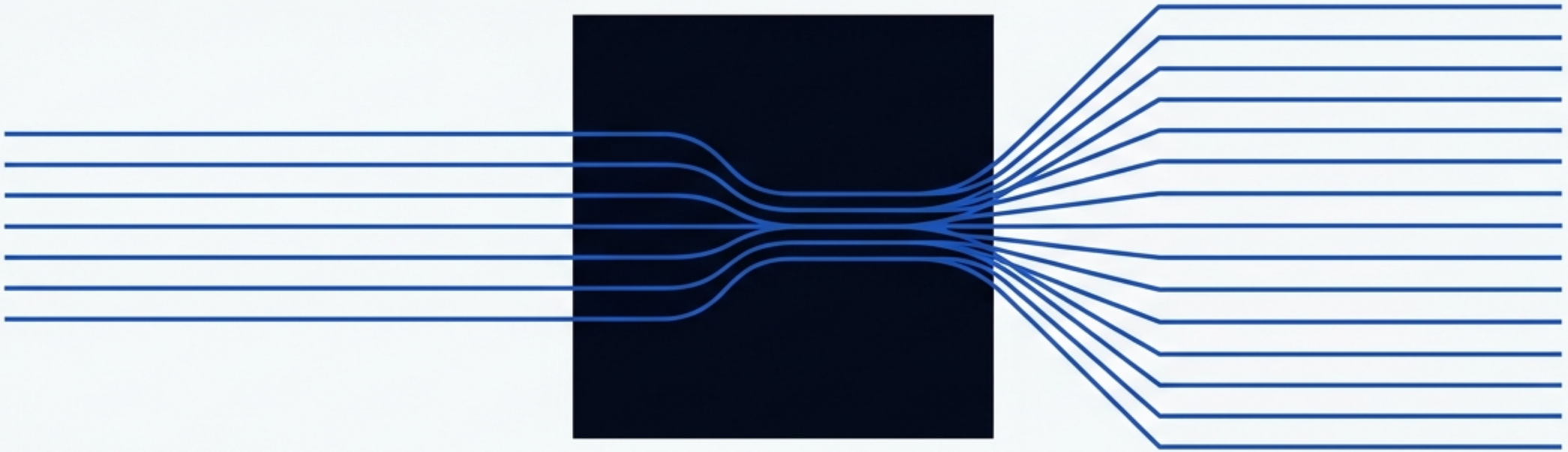
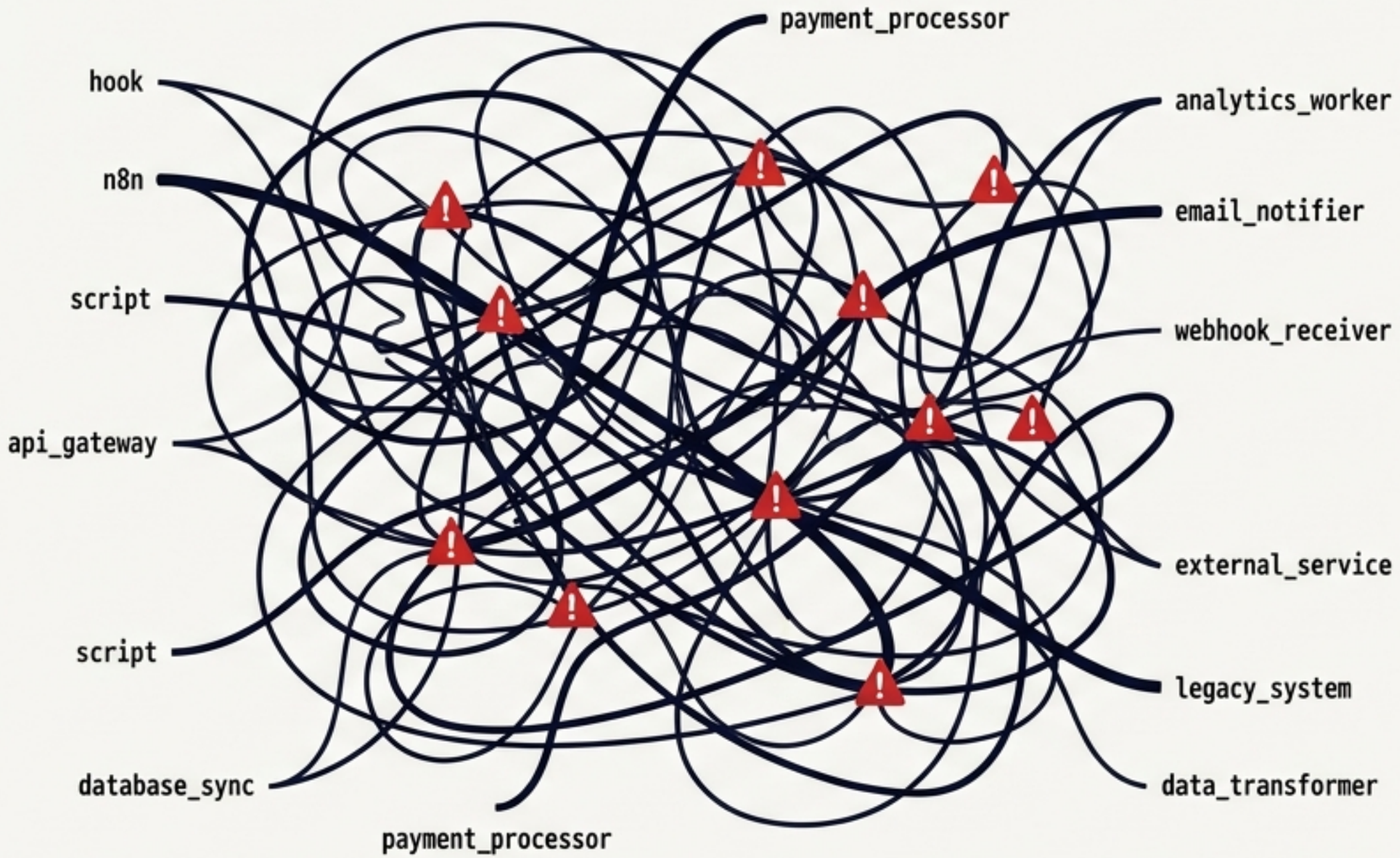


bulletproof-event-router

The central event backbone for multi-agent systems.

Point-to-point wiring creates brittle multi-agent systems



The Integration Trap

In a distributed system, every producer (a script, a hook) must know the URL, retry policy, and health state of every consumer.

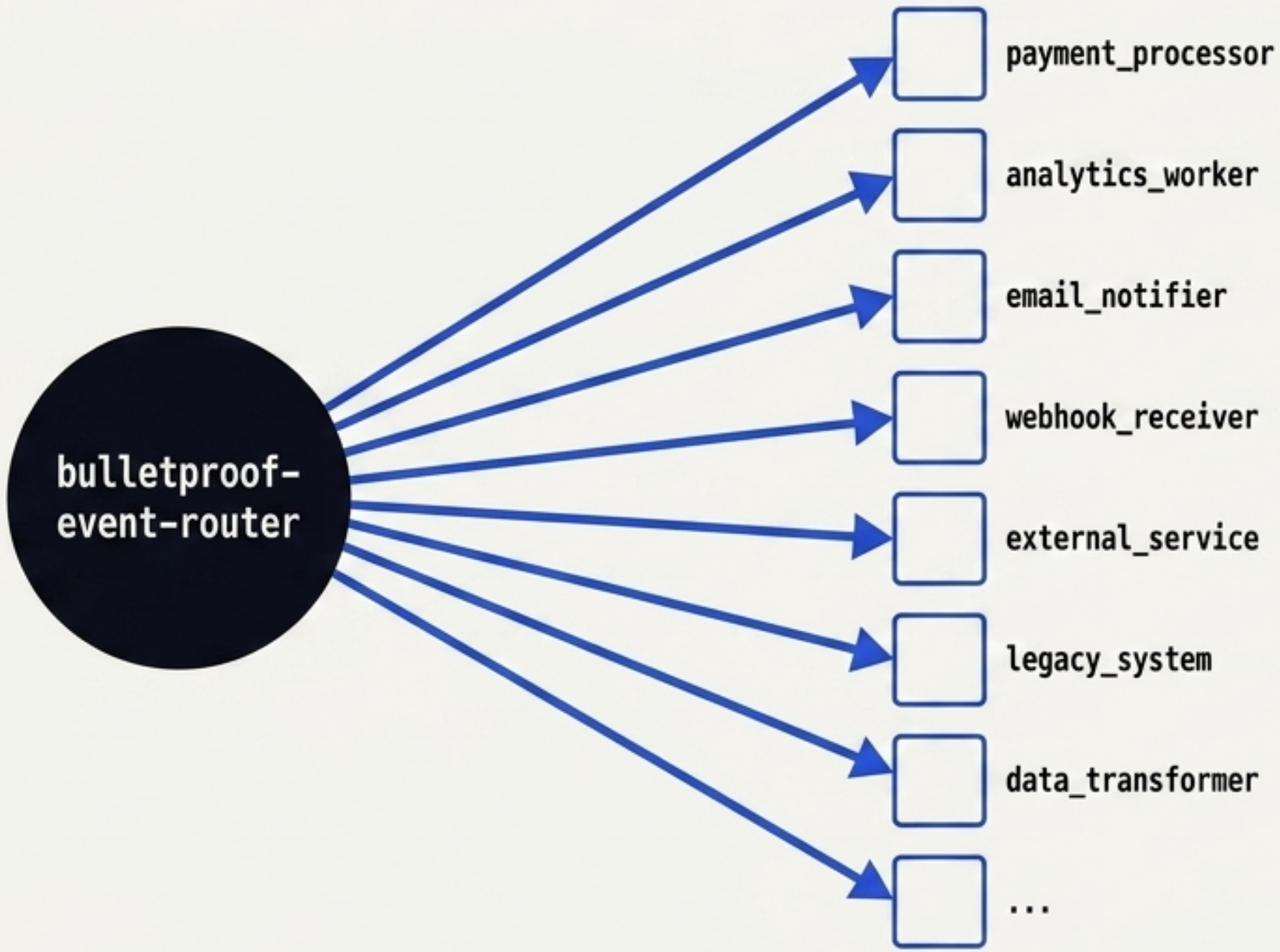
Cascading Failures

A downed downstream service blocks the upstream producer.

The Visibility Void

No single source of truth for system-wide events.

Invert the architecture: emit once, route anywhere



Decoupled Producers

Emit a typed event once. The router handles the rest.

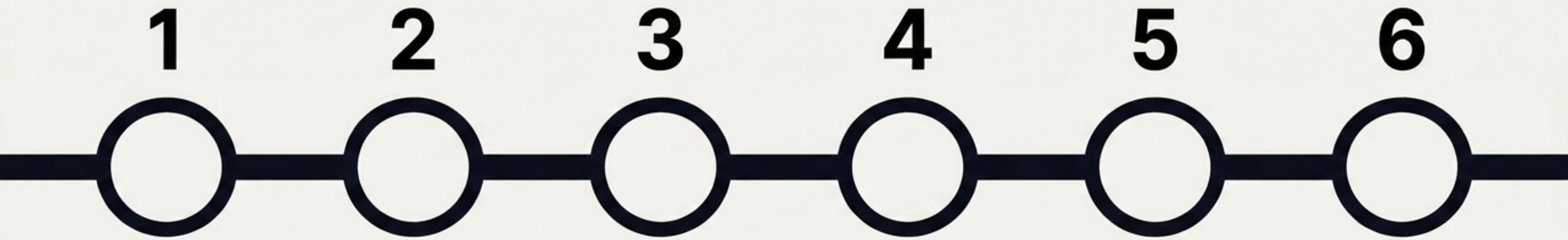
Declarative Rules

Fan out to consumers based on hot-reloaded YAML configurations.

Resilient Delivery

Built-in dead-letter queues, retries, and asynchronous dispatch protect producers from consumer latency.

The six-stage event lifecycle pipeline



Receive
Accept event via POST /events

Validate
Filter through versioned taxonomy

Route
Match glob-pattern configurations

Dispatch
Deliver to designated consumers

Recover
Catch failures in SQLite Dead-Letter Queue

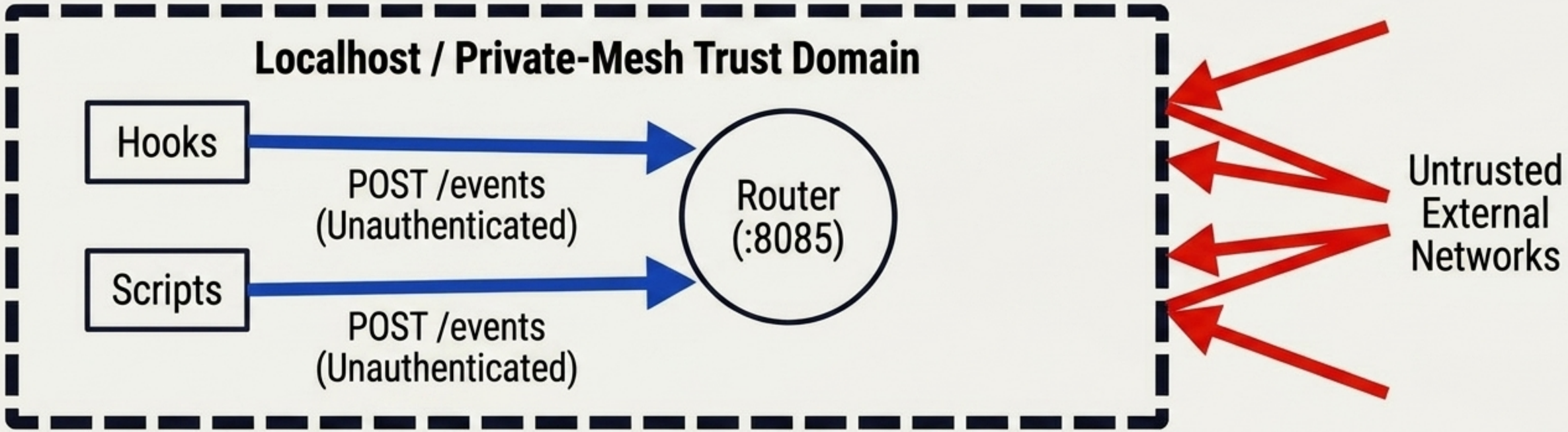
Observe
Log decisions & track workflow health

Emission Strategy Matrix: How producers push events

Fire-and-Forget (Async)	Synchronous
Returns 202 Accepted immediately (routing finishes in background).	Waits for full routing lifecycle to complete.
< 50ms (Captures event_id and releases caller).	Dependent on downstream webhook latency.
Primary Use Case: Claude Code hooks, shell scripts, critical path blockers.	Primary Use Case: Testing, or when the producer explicitly requires the routing decision.
Endpoint: POST /events	Endpoint: POST /events/sync

Note: Both methods support shell-based triggers via emit-event.sh and Python triggers via hook-dispatch.py.

The Receive Phase relies on a Localhost Trust Model



Zero-Friction Ingress
POST /events is explicitly unauthenticated by design.

Network Boundary
The router must only run where every producer is already inherently trusted.

Security Consequence
Exposing this port externally allows arbitrary event emission and trigger manipulation.

The 11-category event taxonomy establishes ground rules

`taxonomy.yaml` defines accepted event categories and payload schemas. If no taxonomy is loaded, the router fails open.

`session`
start, end

`memory`
store, recall

`agent`
dispatch, complete

`governance`
violation, audit

`security`
threat_detected

`infra`
container_restart

`schedule`
daily, weekly

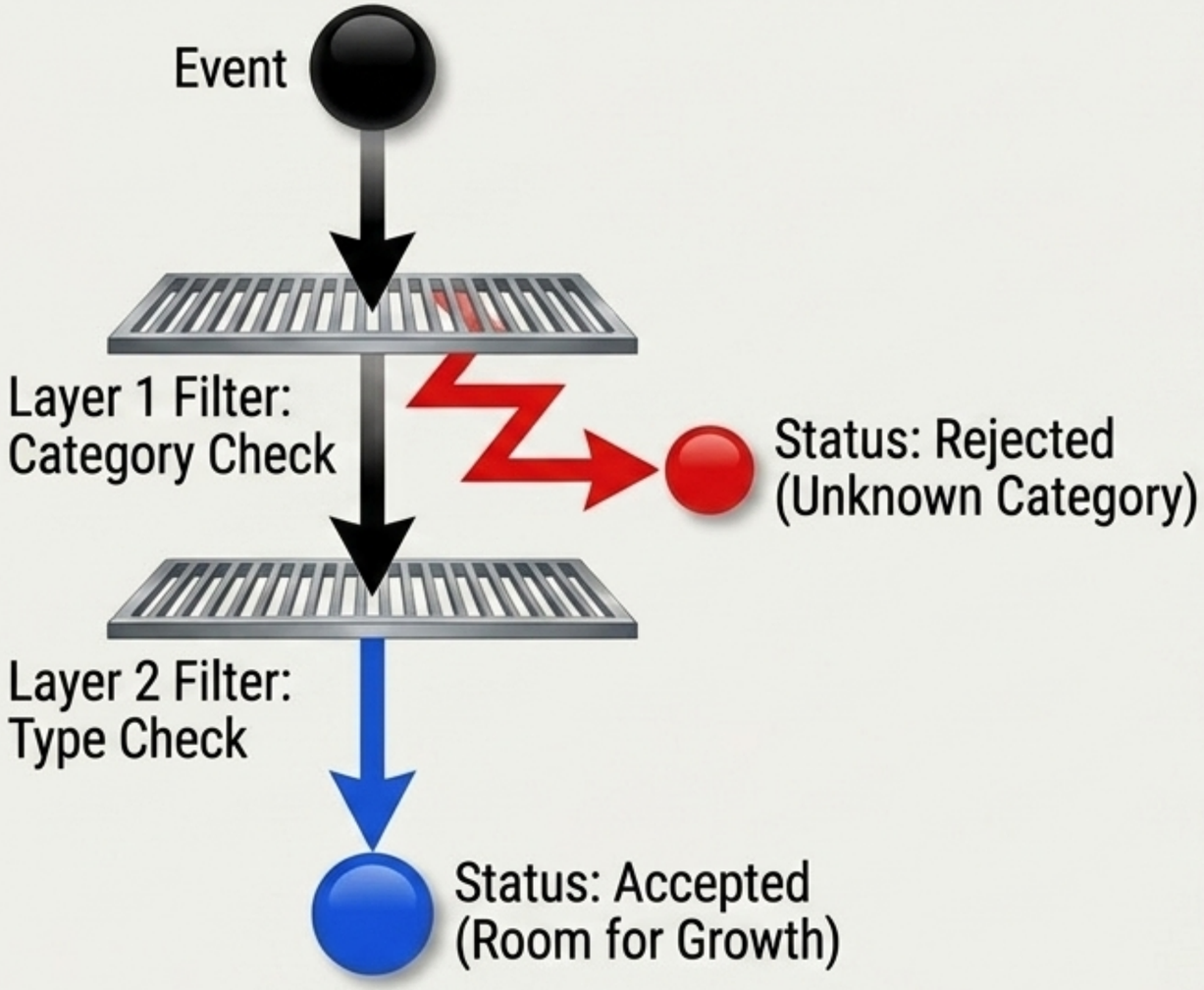
`git`
commit, push

`external`
webhook_received

`recovery`
attempt, success

`cost`
recorded

Strict on categories, flexible on types

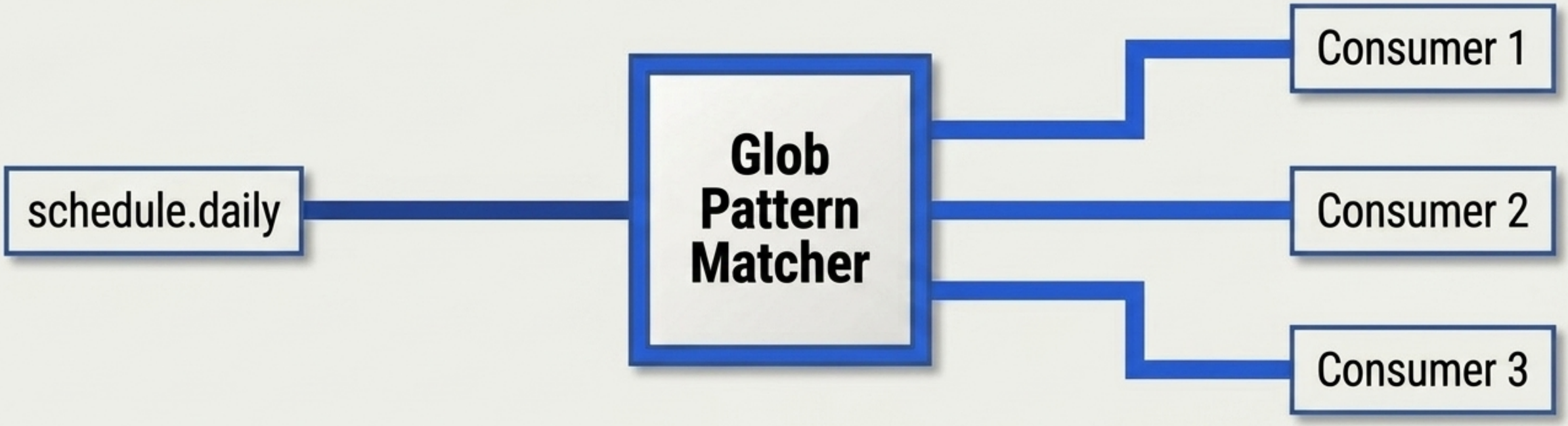


Unknown categories are strictly rejected to maintain system hygiene.

Unknown event types within a known category are intentionally accepted.

The taxonomy leaves room for growth without requiring immediate configuration updates for new event subtypes.

Glob-pattern routing enables effortless fan-out



Pattern Matching

routing-rules.yaml maps events using fnmatch (e.g., schedule.* catches all schedule events).

Multi-Match Fan-Out

A single event can trigger multiple separate rules and fan out to multiple consumers simultaneously.

Zero-Downtime Updates

The background watcher polls configuration files every 30 seconds. Changes apply instantly.

Three distinct consumer channels for event dispatch

Webhook

Mechanism: HTTP POSTs payload.

Target: Resolves via direct URL, env var, or n8n workflow registry.

Behavior: The standard bridge to external services and orchestration loops.

Conductor State

Mechanism: Writes event to conductor-state.json.

Target: Specific target_directory or global fallback.

Behavior: Silently updates local multi-agent orchestrator files via dot-path injection.

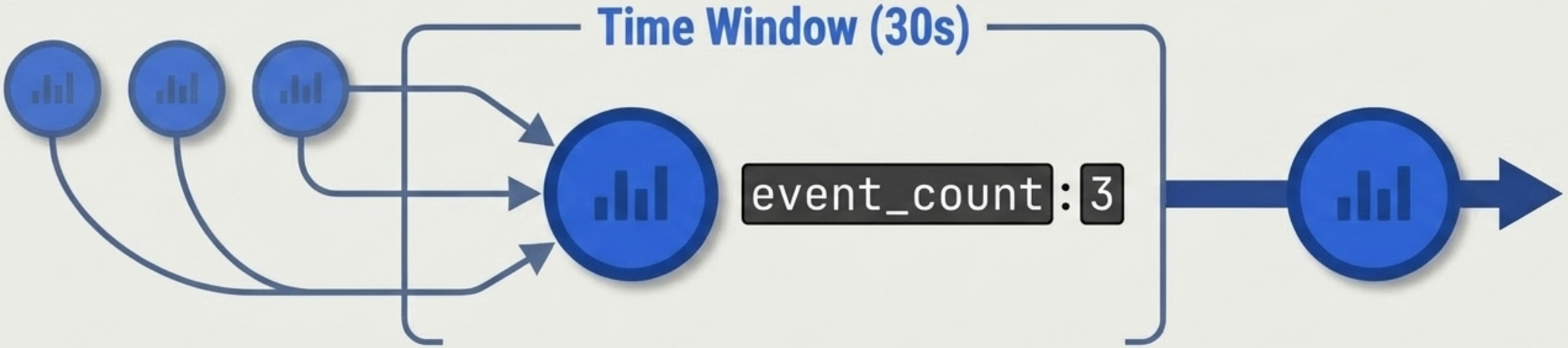
Direct Action

Mechanism: In-process execution.

Target: Console logs or WebSocket broadcasts.

Behavior: Zero-latency
Behavior: Zero-latency visibility and debugging.

Delayed dispatch collapses noisy event bursts



1. The Window

The first event opens a fixed time window (delay_seconds).

2. The Collapse

Subsequent events sharing the same dedup_key collapse into a single pending row.

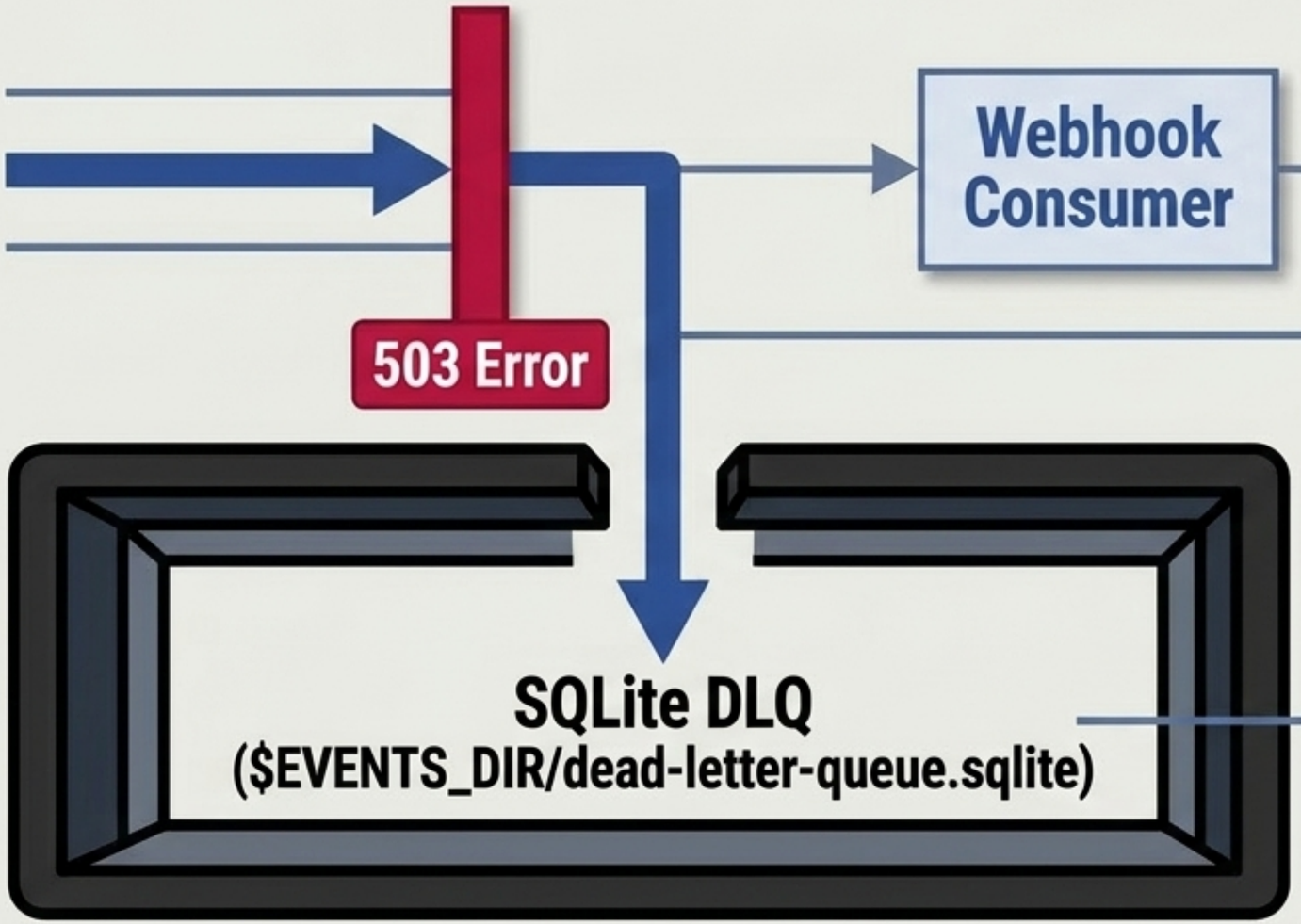
3. The Payload

The latest payload wins, merging data while incrementing the event_count.

4. The Release

After the fixed delay, one consolidated payload is dispatched.

The Dead-Letter Queue isolates failures from producers



Durable Storage

Failed webhook dispatches or delayed deliveries are securely persisted to a single SQLite file with WAL mode enabled.

Silent Protection

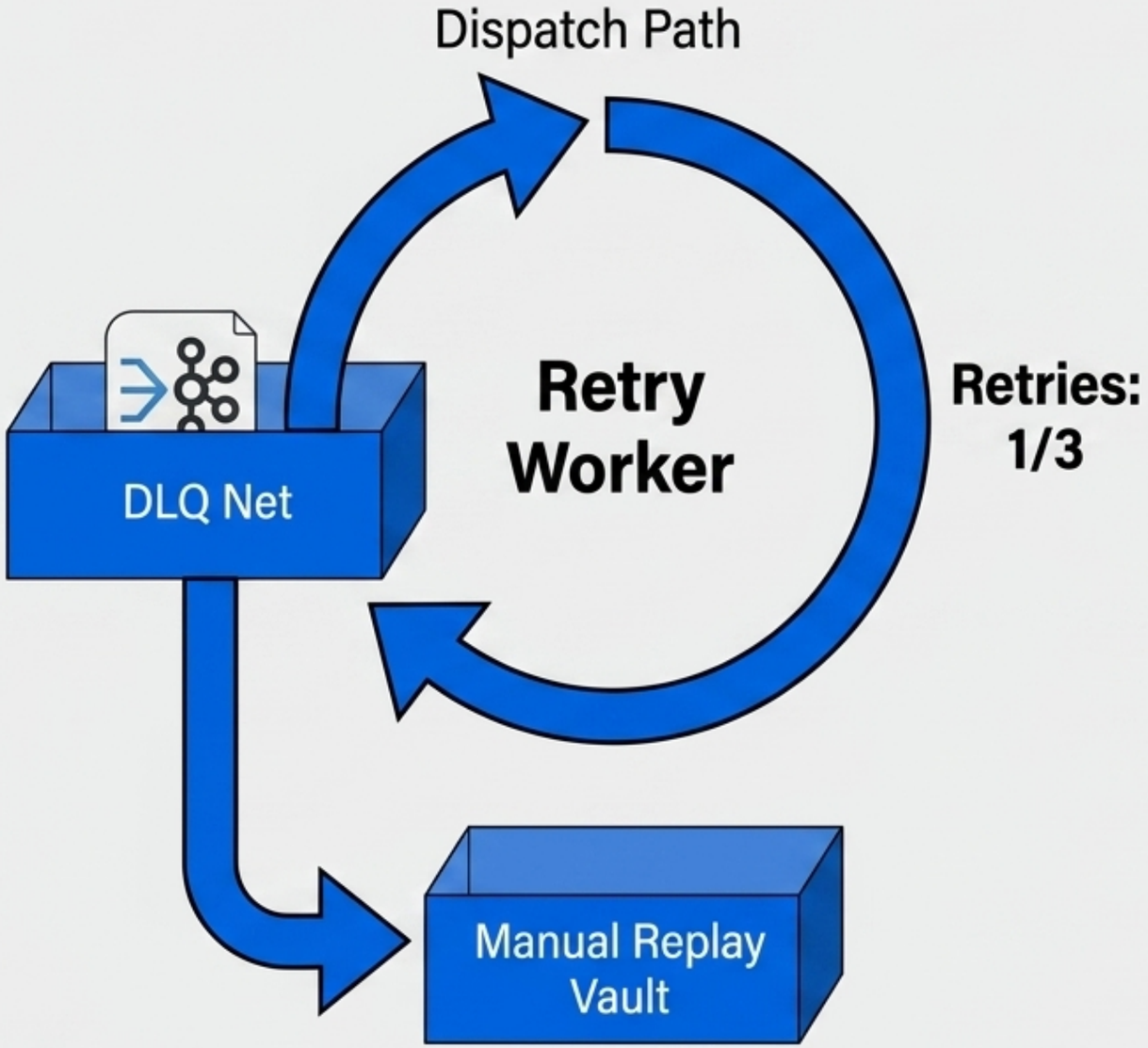
Because the router sits between producers and consumers, a downstream outage results in a DLQ entryentry, not a crashed producer script.

Data Sensitivity

Payloads are preserved entirely in the DLQ.

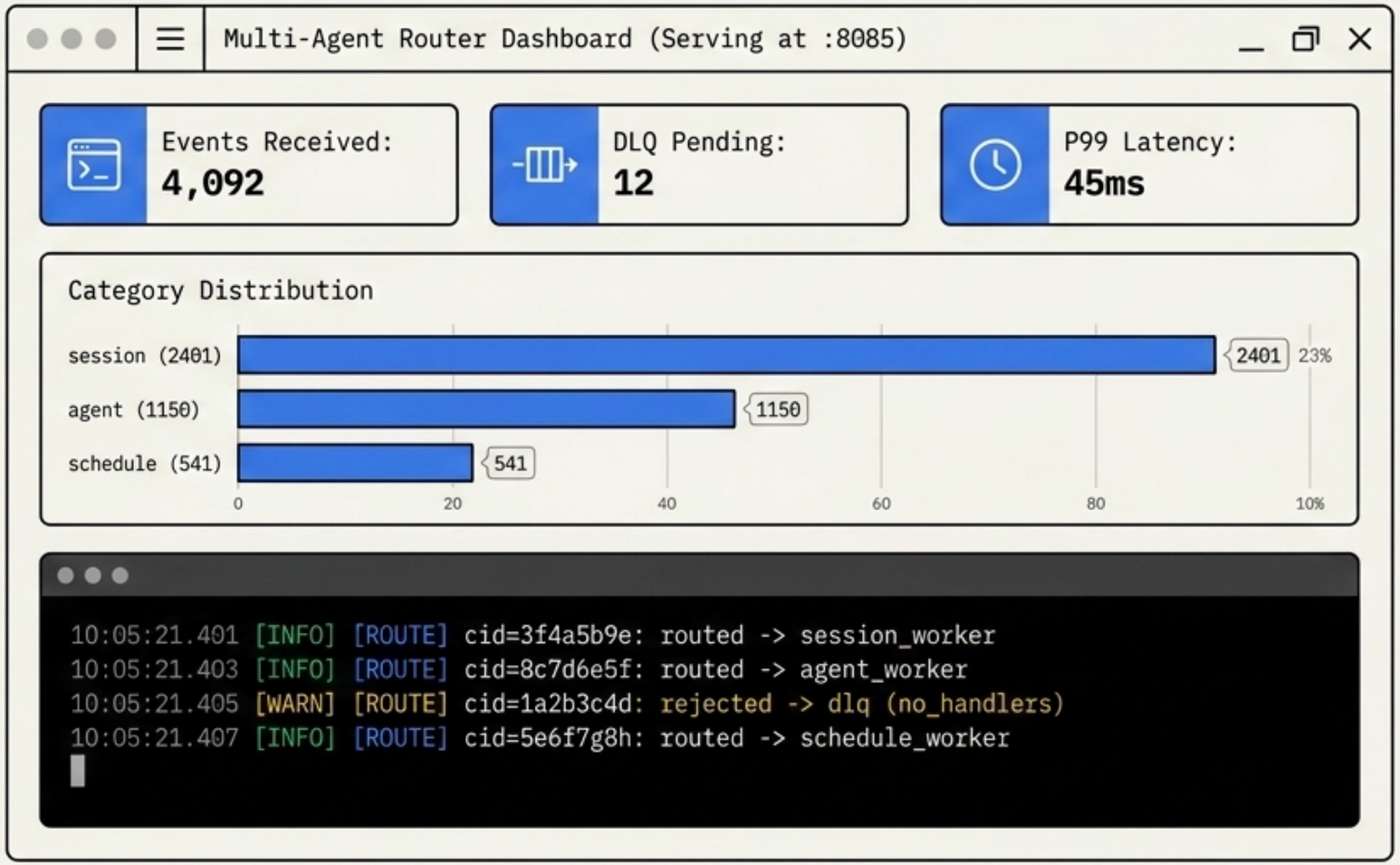
CISO note: Deploy careful routing for sensitive categories.

Automatic backoff and manual replay capabilities



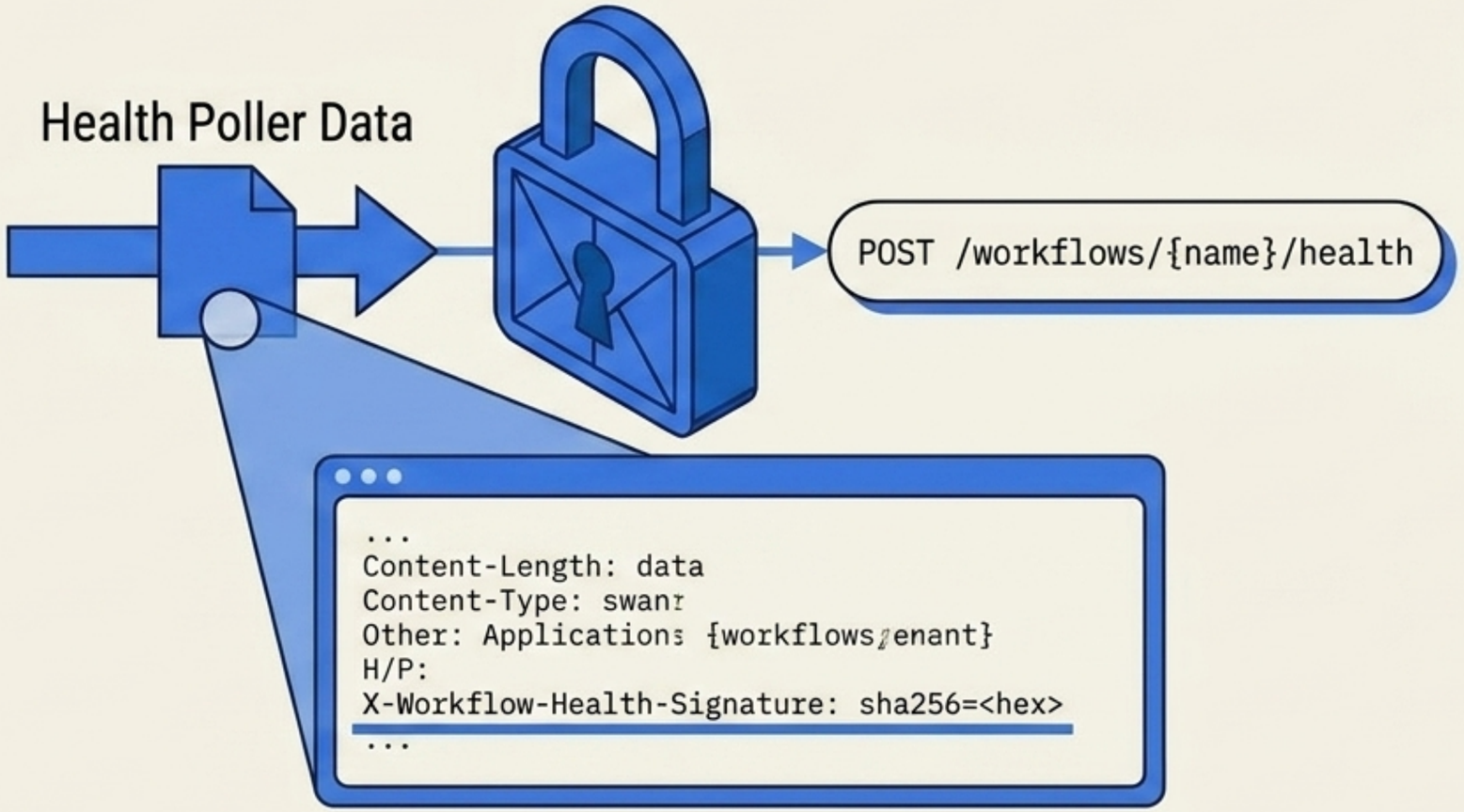
1. **Catch & Queue:** Delivery fails; event enters DLQ.
2. **Worker Cadence:** Retry worker wakes every DLQ_RETRY_INTERVAL_SECONDS (60s).
3. **Exponential Backoff:** Retries escalate using a 2000ms base backoff.
4. **Exhaustion:** Max retries hit -> event transitions to permanent failure state.
5. **Manual Intervention:** POST **/dlq/replay** enables bulk manual replay once downstream health is restored.

Real-time visibility into the multi-agent backbone



- **The Routing Log:** Every decision (routed, rejected, no_handlers) is written to SQLite.
- **End-to-End Tracing:** Correlation IDs are automatically injected and propagated for session tracing.
- **Live Dashboard:** 5-second polling intervals paired with a /ws WebSocket for push updates.

Tracking n8n SLAs with HMAC-secured workflow health



SLA Tracking
Aggregates 24-hour health against SLAs defined in workflow-registry.yaml.

The Sole Authenticated Route
Unlike the unauthenticated ingress, workflow health updates require a cryptographic signature.

Constant-Time Verification
Uses hmac.compare_digest with WORKFLOW_HEALTH_HMAC_SECRET to prevent timing attacks.

The Runtime Layer makes the ecosystem turnkey

Producers (Input)

- **hook-dispatch.py**
  Translates Claude Code hook payloads into proper taxonomy events (e.g., SessionStart -> session.start).
- **emit-event.sh**
  Fast, blocking-free CLI wrapper for shell scripts.

The Bridge Daemon (Output Mirrors)

Tails the SQLite routing log and mirrors events to external downstream targets (Postgres data-planes, metrics databases).

- **Idempotent & Fails Soft:** Uses a cursor state, skips dead downstreams without crashing.
- **Opt-In Only:** Requires specific DSN env variables; out of the box, it takes no harmful action.

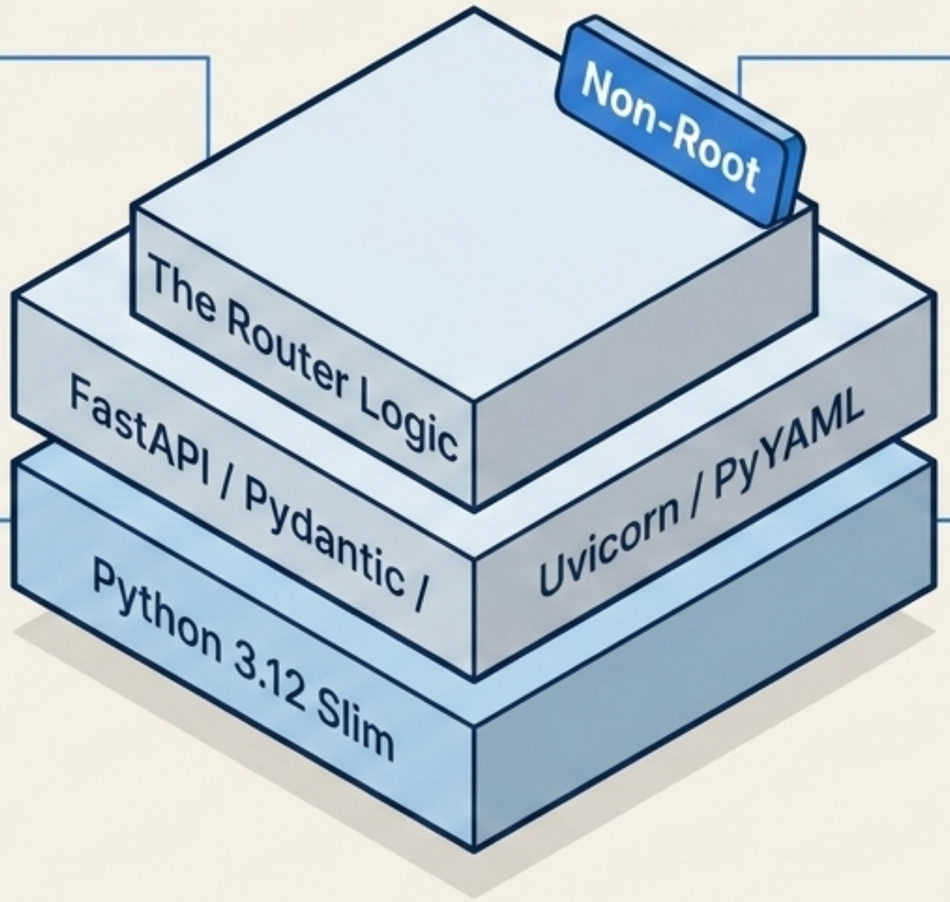
Lightweight stack, hardened by default

Minimal Footprint

Only 6 pinned direct dependencies. No heavy frameworks.

No Copyleft Risk

100% OSI-approved runtime dependencies (MIT, BSD-3-Clause).



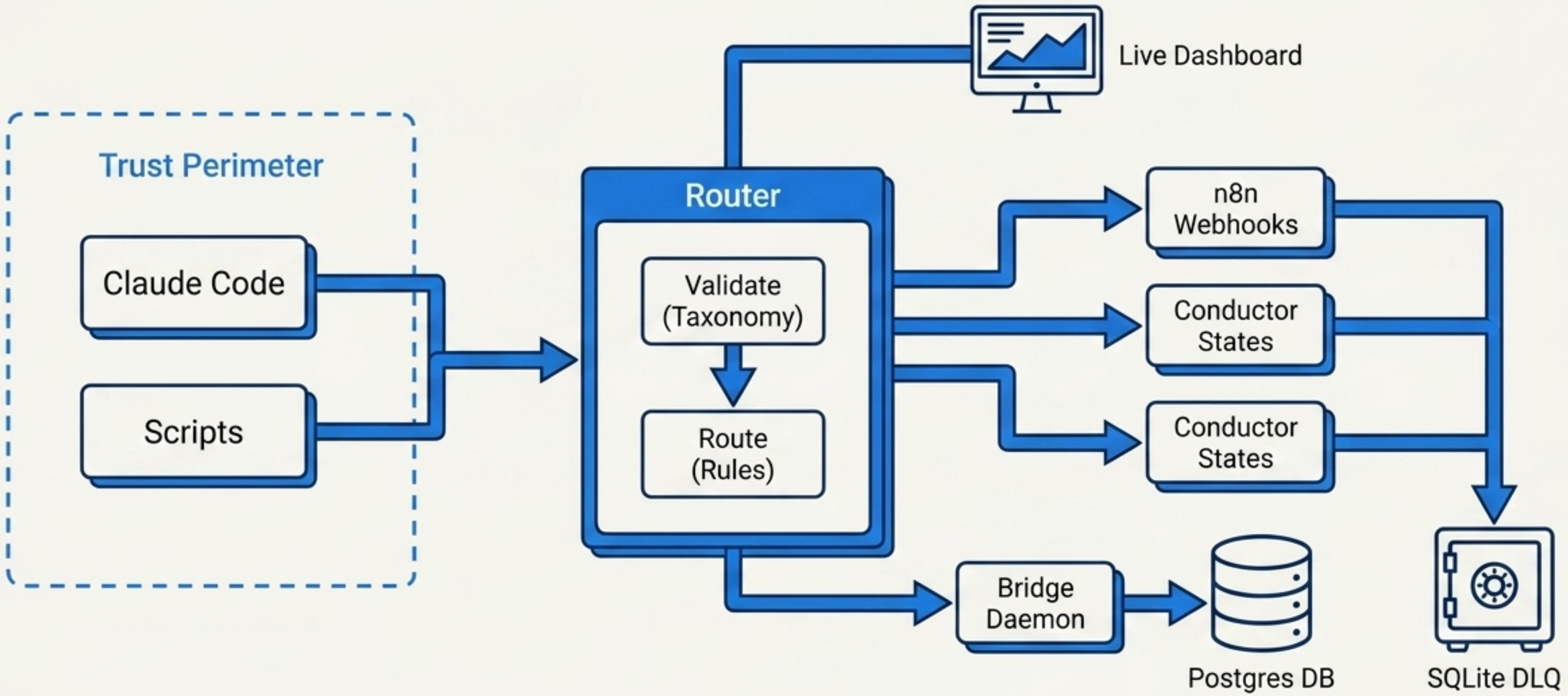
Container Hardened

Runs as non-root appuser (uid 10001). Pre-created paths, built-in health checks.

In-Process Scaling

Designed to run with `--workers 1` to consolidate in-memory WebSocket states.

The definitive nervous system for multi-agent architectures



Point-to-point chaos replaced by declarative, typed, resilient event routing.
Emit once, dispatch everywhere.