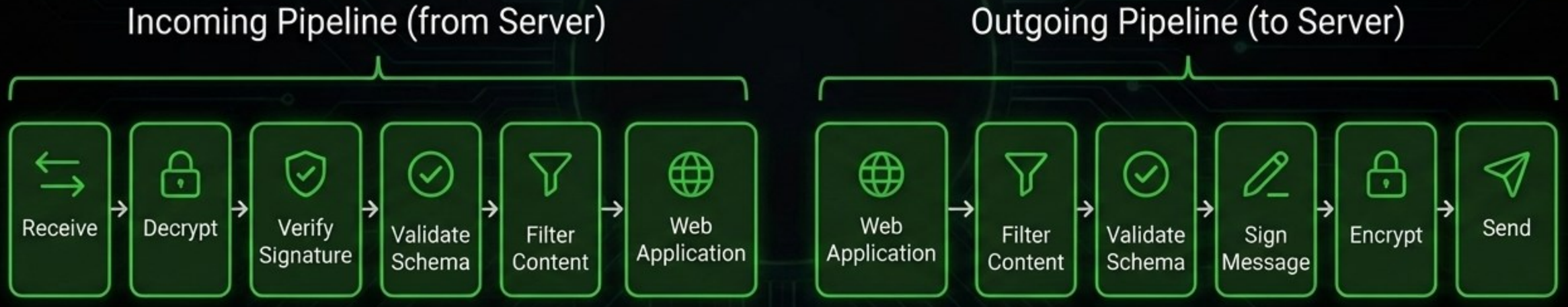
BizFirstAi

EdgeInteract: Browser-Side Message Pipeline

Secure, Validated, and Filtered Real-Time Communication for the Web.



10,000+

Msg/sec verified throughput

<100ms

P95 latency across the pipeline

100%

TypeScript strict mode for total type safety

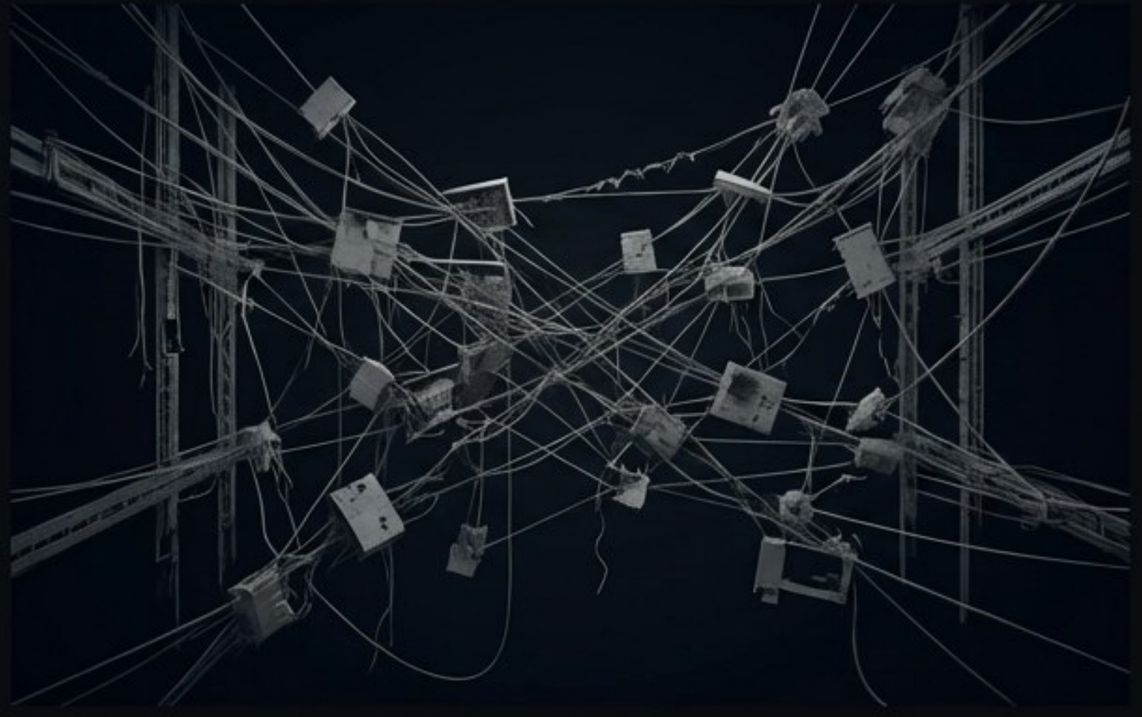
0

Core external dependencies in the base architecture

Enterprise-grade observability and zero data loss guaranteed.

The Enterprise Messaging Dilemma

The Tangled Web



Custom Sockets: Fast to start, impossible to scale. High maintenance and completely lacks built-in observability or retry logic.

The Bloated Overkill

Legacy Brokers (Kafka/RabbitMQ): Enterprise-grade, but heavy. Requires dedicated clusters, complex Zookeeper setups, and introduces high latency for real-time web UI synchronization.

The EdgeStream Architectural Philosophy

Universal Standard

Unifies the ecosystem using a single IEnvelope format with strict CNCF CloudEvents compliance.

Zero-Infrastructure

Embedded and serverless capable. Eliminates the need to manage separate stateful services or complex clusters.

Total Observability

Every message is tracked. Built-in hook execution logs, subscriber stats, and compliance audit trails by default.

The 4-Layer System Architecture

Dev/Inspector Tools
Standalone visual debugging and real-time message exploration.

Framework Bindings
React hooks, ChatWindow components, and SignalR Hub Adapters.

Reserved / Extension
Infrastructure for future protocol adapters and WebAssembly plugins.

Core Transport & PubSub
The lightweight 15KB execution engine, hook pipeline, and SubscriptionManager.

Anatomy of a Message: The IEnvelope

Metadata (The Envelope)

Source ID, Event Type (e.g., chat, flow-engine), Correlation ID, and Timestamp. Guarantees CloudEvents backward compatibility.

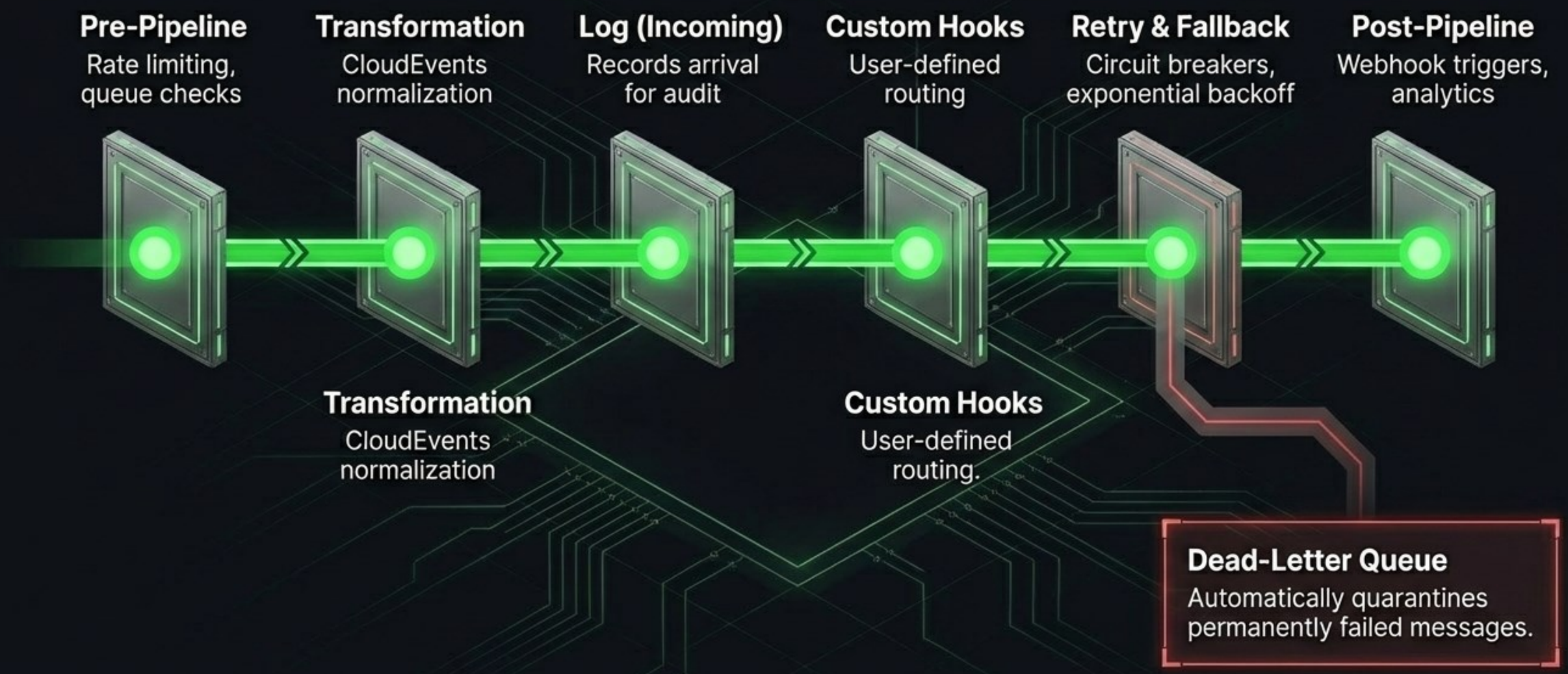
Context (The Security)

User Context, Session ID, and Multi-Tenant Isolation boundaries automatically preserved and carried through the entire pipeline.

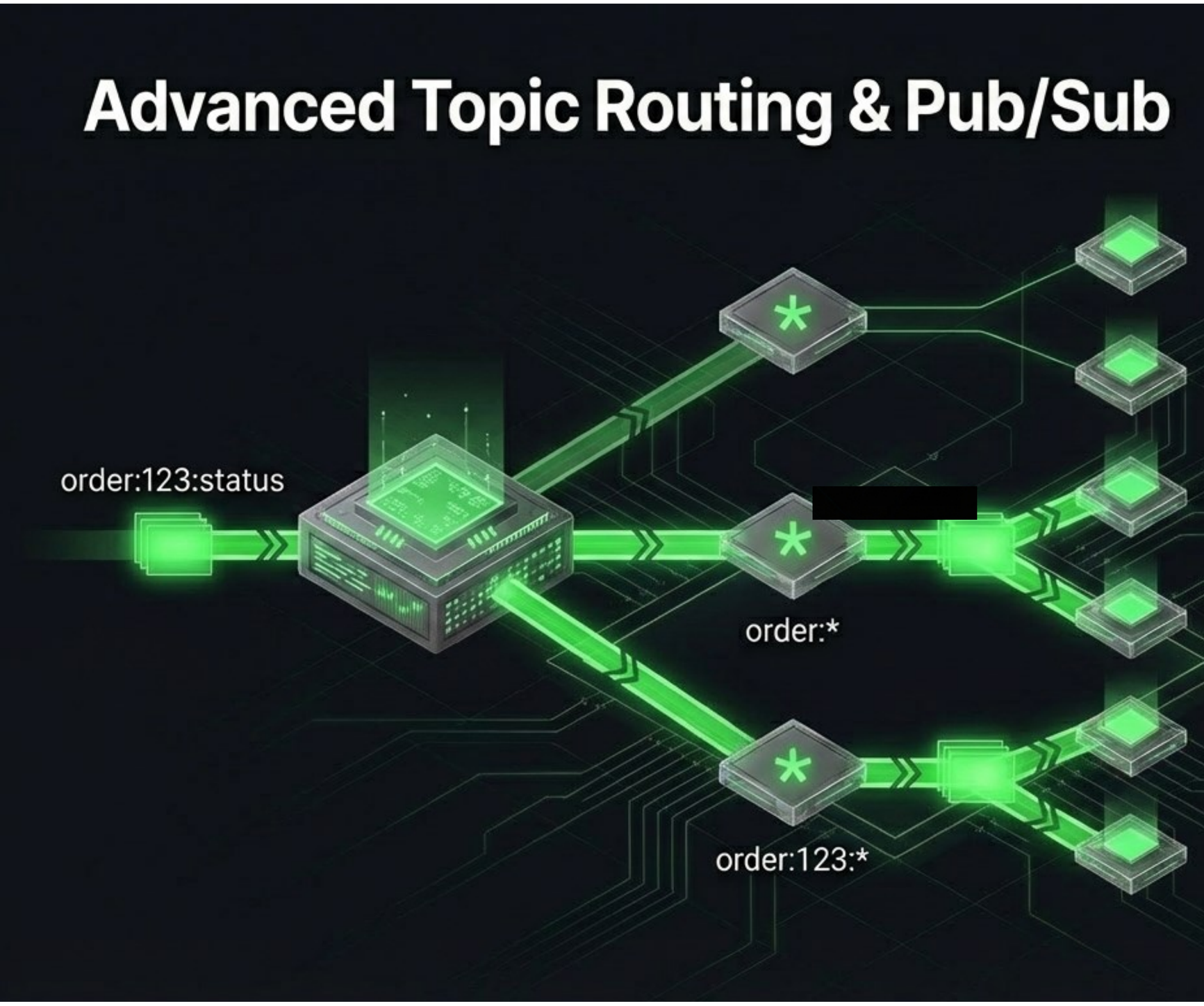
Payload (The Cargo)

The highly typed, strictly inferred data body driving the application state.

The Hook Pipeline Engine



Advanced Topic Routing & Pub/Sub



Hierarchical Naming

Topics are strictly structured for granularity (e.g., order:123:status).

Wildcard Matching

Regex and wildcard subscriptions dynamically map and filter streams to appropriate users.

Context-Aware Routing

Filters actively drop or forward messages based on the User/Tenant context embedded in the IEnvelope.

4 Standardized Communication Patterns

Fire-and-Forget (202)

Asynchronous notifications and background logging. No waiting for a response.

Request-Reply (200)

Synchronous operations. Wait for completion or timeout.

Streaming (SSE)

Chunked response delivery. Sub-100ms real-time token and data streams.

Task-Start

Async polling. Obtain a Task ID immediately, then poll or listen for the final result.

Built-In Enterprise Observability

Debug impossible scenarios in minutes, with zero external APM overhead.

Complete Audit Trail

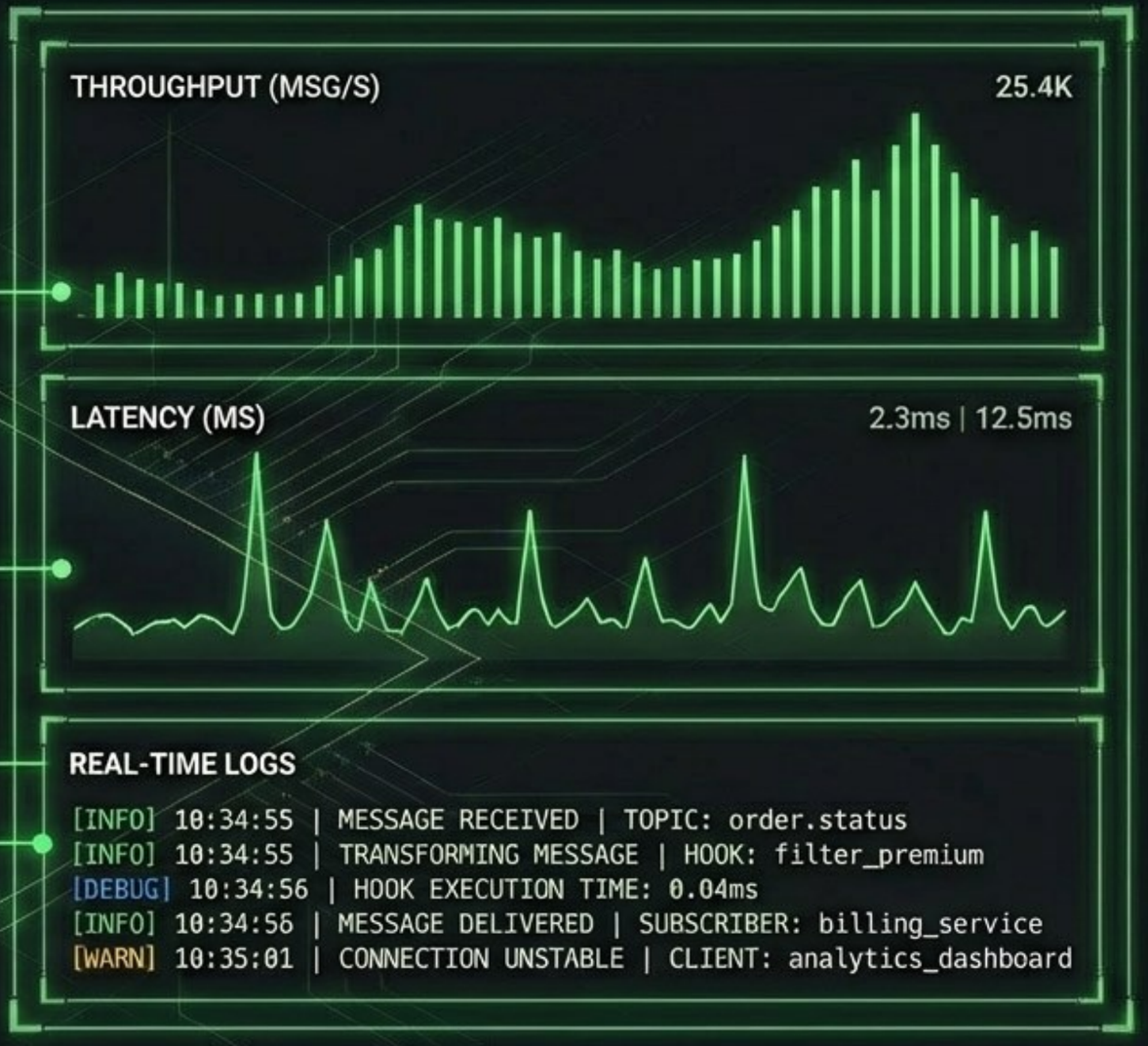
Every message arrival, transformation, and delivery is permanently logged.

Hook Telemetry

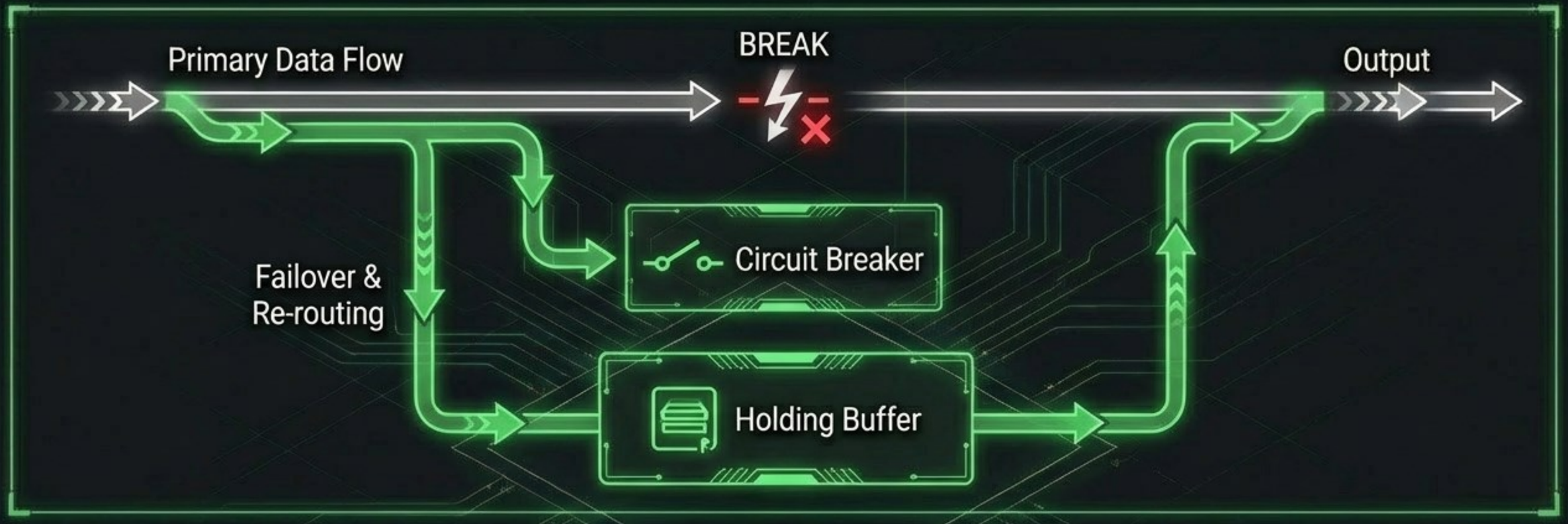
Track precise execution times and failure rates of individual pipeline hooks.

Subscriber Stats

Monitor dynamic subscription counts, topic density, and connection health in real time.



Resilience & The Zero Data Loss Guarantee



Automatic Persistence
Distributed state tracking across nodes prevents in-memory loss during crashes.

Exponential Backoff
Intelligent, automated retry mechanisms handle transient network failures.

Dead-Letter Queues
Automatic quarantine and continuous monitoring of permanently failed messages.

Graceful Degradation
Circuit breaker patterns ensure system stability during partial infrastructure outages.

Ecosystem Integration: The Central Nervous System

Competitive Diagnostic Matrix

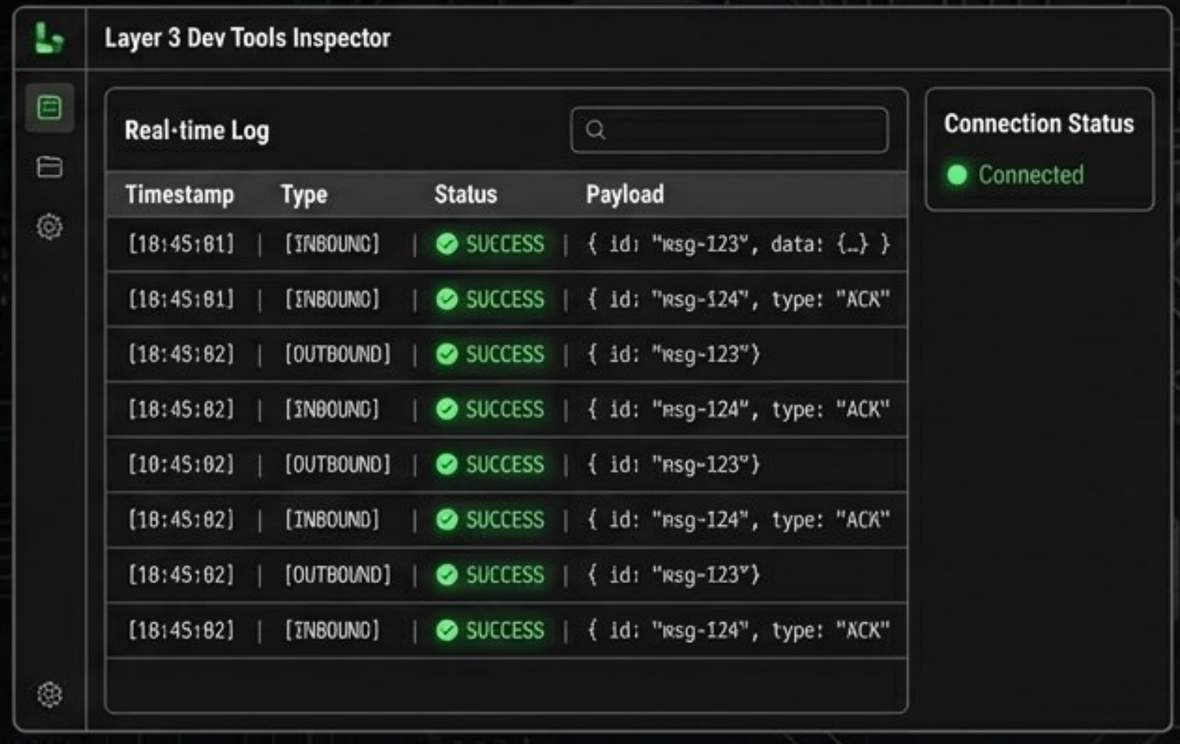
Feature	EdgeStream	Kafka	RabbitMQ / Custom Sockets
Setup & Infrastructure	✓ Zero / Embedded	✗ Complex Zookeeper/Cluster	✗ Separate Stateful Service
Latency / Web-First	✓ <100ms / Native SignalR	✗ 100-500ms / Backend only	✗ Requires web adapters
Observability	✓ Built-in Activity Logs	✗ Requires ELK/Splunk	✗ Manual implementation
Routing Complexity	✓ Simple Topic Wildcards	✗ Partitions/Offsets	✗ Exchanges/Bindings

The Developer Experience

```typescript
import { StrictMessage } from 'edge-stream-types';

interface MyPayload {
  id: string;
  data: Record<string, any>;
}

const handler = (msg: StrictMessage<MyPayload>) => {
  // Compile-time type safety
  const { id, data } = msg.payload;
  console.log(id.toUpperCase());
};
```



100% Strict Type Safety

Catch routing and payload errors instantly at compile-time.

Native React Bindings

Connect entire UIs in minutes using dedicated useEdgeStream hooks.

Standalone Visual Inspector

Explore message history, view registered hooks, and monitor connection status visually.

BizFirstAi

Strategic ROI: The New Standard for Real-Time

Accelerated Time to Market

Go from weeks of custom socket implementation to days using a standardized, predictable pipeline.

Slashed Infrastructure Costs

Eliminate the need for dedicated messaging clusters or expensive external APM licenses.

Unified & Observable

A single, compliant standard that guarantees every real-time event across the ecosystem is tracked, secure, and delivered.