Metadata Up, Media Down: A Hybrid Edge-Cloud Architecture for Multi-Tenant Video Management

A Dhi Technologies whitepaper — Dev Sanghvi, Dhi Technologies · July 2026 (draft)

Executive summary

Video management systems are sold as a dichotomy: cloud VSaaS (convenient, fleet-wide, expensive, privacy-fraught) or on-premises NVRs (private, cheap to run, siloed, stuck with decade-old UX). The dichotomy is false. It comes from treating "the video system's data" as one thing, when it is two things with opposite economics:

- **Media** — heavy, continuous, privacy-sensitive, rarely watched.
- **Metadata** — light, queryable, operationally precious, watched constantly.

The Dhi platform splits the system along that line. Media stays on the edge device that produced it and flows *down* to an authorized viewer's browser on demand, point-to-point, never stored in the cloud. Metadata — events, alerts, identity sightings, device health — flows *up* continuously into a serverless, multi-tenant control plane that costs almost nothing at rest and scales by adding rows, not servers. The result behaves like cloud software (single dashboard, any number of sites, modern UX, guest demos) with the data posture of an on-premises system (footage never leaves the building).

This paper describes the architecture as deployed: a Jetson-class analytics device running 26 configurable analytics; a control plane on Cloudflare Workers with a D1 (SQLite-semantics) database of 34 tables; row-level multi-tenancy enforced beneath a zero-trust SSO layer; an in-process metadata forwarder with measured live throughput; tunnel-based media delivery including native H.265; and a pairing-code onboarding flow that claims a new site's device into a tenant in minutes.

---

1 The false dichotomy

Cloud VSaaS puts cameras' streams in the vendor's cloud. Fleet-wide visibility and modern software come at the cost of continuous upstream bandwidth, per-camera subscription pricing, and a compliance posture in which the vendor's cloud is a giant archive of your most sensitive data. **On-prem NVRs** invert every property: footage stays home, costs are capital rather than recurring — and each site is an island with a local-only interface, no cross-site view, and no path to modern analytics.

Operators keep asking for the obvious third thing: *cloud software, on-prem data*. That is an architecture problem, not a pricing problem, and it has a clean solution once media and metadata are treated separately.

2 Architecture overview

Figure 1. Two planes. The metadata plane is cloud-native and always-on; the media plane is point-to-point and on-demand.

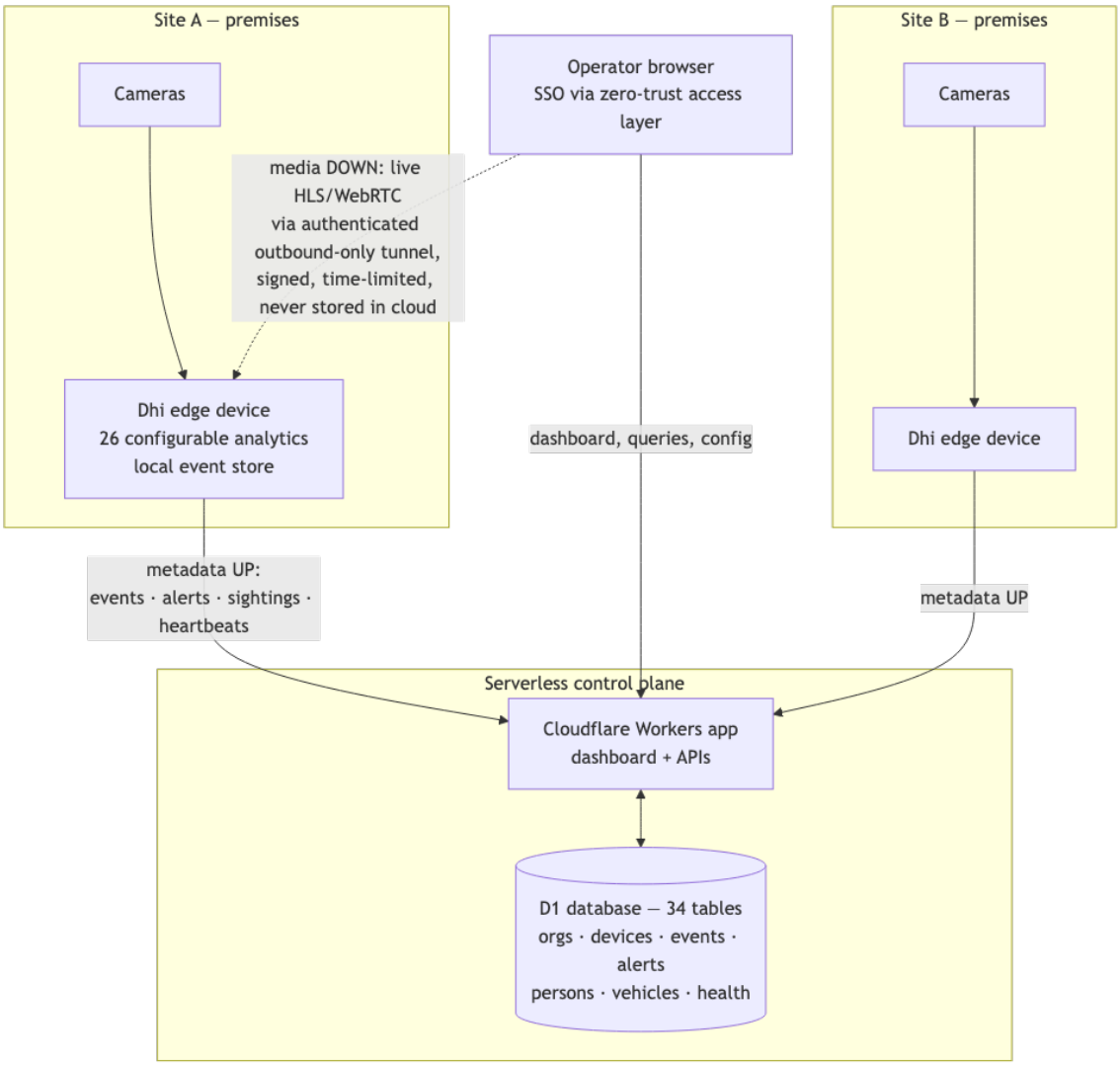

Figure 1: Figure 1. Two planes: metadata up, media down.

Three components:

1. **The edge device.** A Jetson-class unit runs the site's full analytics stack — a shared-inference architecture (documented in our systems paper) hosting any subset of 26 canonical use cases across 11 detector engine families, an authenticated local API, and a retention-bounded local event store.
2. **The control plane.** A single-page dashboard (≈40 k lines of TypeScript/React, 52 pages) served from Cloudflare Workers, backed by D1 — 34 tables covering organizations, members, sites, cameras, devices, events, alerts, persons, vehicles, and health. There are no servers to size: the control plane is functions plus a database, priced at entry-tier serverless rates at pilot scale.
3. **The identity layer.** Operators authenticate through a zero-trust access layer with Google SSO; every request carries an organization identity, and every table carries an organization foreign key. Tenancy is enforced at the row level in one place, beneath the application.

3 Metadata up

Every analytic event on the device passes through a single ingestion endpoint (a hard invariant of the platform — nothing writes event rows directly). Immediately after local persistence, an **in-process forwarder** — a small thread pool inside the API service, not a sidecar daemon — pushes the relevant rows to the control plane:

- **Alerts** (the operator-facing subset of events) upsert idempotently on a (organization, camera, type, timestamp) key, so retries never duplicate.
- **Identity sightings** unify people and vehicles under one abstraction: a person track carries an appearance token; a license-plate read mints a vehicle identity keyed on the normalized plate (`plate:<NORMALIZED>`), so "MH-12 AB 1234" and "mh12ab1234" collapse to a single cross-camera identity. The dashboard's person/vehicle timelines are built from these sightings.
- **Heartbeats** give the fleet view a lightweight liveness signal (throttled to ≈30 s).

Two properties matter operationally. *It is in-process*: forwarding happens at the moment of ingestion, on a bounded thread pool that never blocks the ingest path, replacing an earlier generation of polling daemons — there is no second process to deploy, monitor, or fall behind. *It is best-effort by design*: the device is the system of record; the cloud copy is a queryable projection. In live verification the forwarder sustained **35 identity sightings and 189 alerts per 60-second window** from a single six-camera device — far below its ceiling and comfortably above the workload.

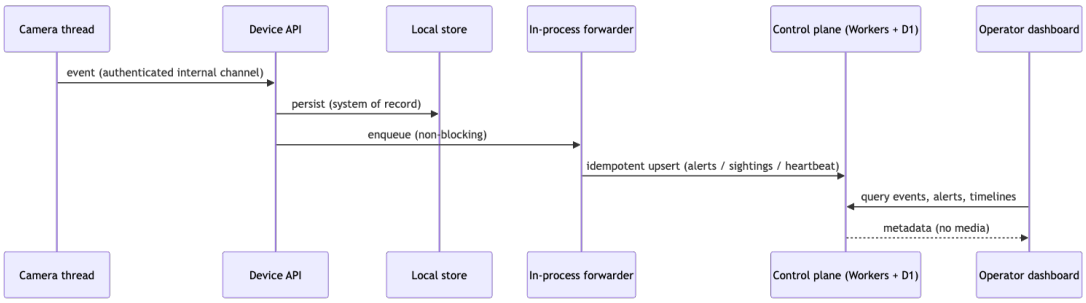

Figure 2: Figure 2. The event path: the cloud is a projection; the device is the record.

Figure 2. The event path. The cloud is a projection; the device is the record.

4 Media down

Live video never transits the control plane. The device publishes its streams through an **outbound-only tunnel** — no inbound firewall holes, NAT-friendly — and the dashboard player fetches them directly:

- **HLS in low-latency fragmented-MP4**, which serves H.265 camera streams *natively* (no transcode — significant on devices without hardware encoders) to browsers that decode H.265, alongside H.264.
- **WebRTC** for sub-2-second interactive viewing where HLS's 5–10 s is too slow.
- **Signed access**: the dashboard requests a stream authorization from the control plane, which checks organization membership and device scope and returns a time-window-signed token the media broker verifies before emitting a single frame.

If a site's device is offline, its live video is unavailable until it returns — the honest cost of a system in which no cloud copy exists. Event history, alerts, and timelines remain fully available throughout, because the metadata plane is independent.

5 Multi-tenancy without a tenancy layer

Because tenancy lives in two primitives — the SSO-derived organization claim and the organization foreign key on every table — the application code above it stays ordinary. Consequences:

- **Row-level isolation**: an operator sees exactly their organization's cameras, events, and devices; cross-tenant access is structurally absent rather than filtered in application logic scattered across pages.
- **Guest mode**: unauthenticated visitors are enrolled into a shared, seeded demo organization — the same code paths, real analytics output, zero exposure of any customer tenant. (Sales demos run on the production system, not a mock.)
- **Per-organization device registry**: each edge device belongs to one organization, with its access credential stored only as a hash.

6 Onboarding a site in minutes

Figure 3. Pairing-code claim: admin-gated, time-boxed, single-use; per-device credentials generated on the device and stored only as hashes.

The flow is deliberately boring: an admin mints a code in the dashboard, the installer runs one command on the device, and the device claims itself into the tenant — generating its own credential (never a shared fleet secret), bringing up its outbound tunnel, and appearing in the fleet view when its health probe passes. The schema and claim flow shipped in two phases; the third phase — reading events primarily from the cloud projection so NAT-only sites need the tunnel solely for live video — is the current roadmap step.

7 A typed contract across the boundary

Hybrid architectures usually rot at the seam: the cloud dashboard and the edge runtime drift until every release is an integration project. We pin the seam with a **shared typed contract**: the 26 canonical use cases, their labels, and their deterministic mapping onto the 11 detector engines live in one contract module consumed by both sides, and contract tests fail the build on any drift

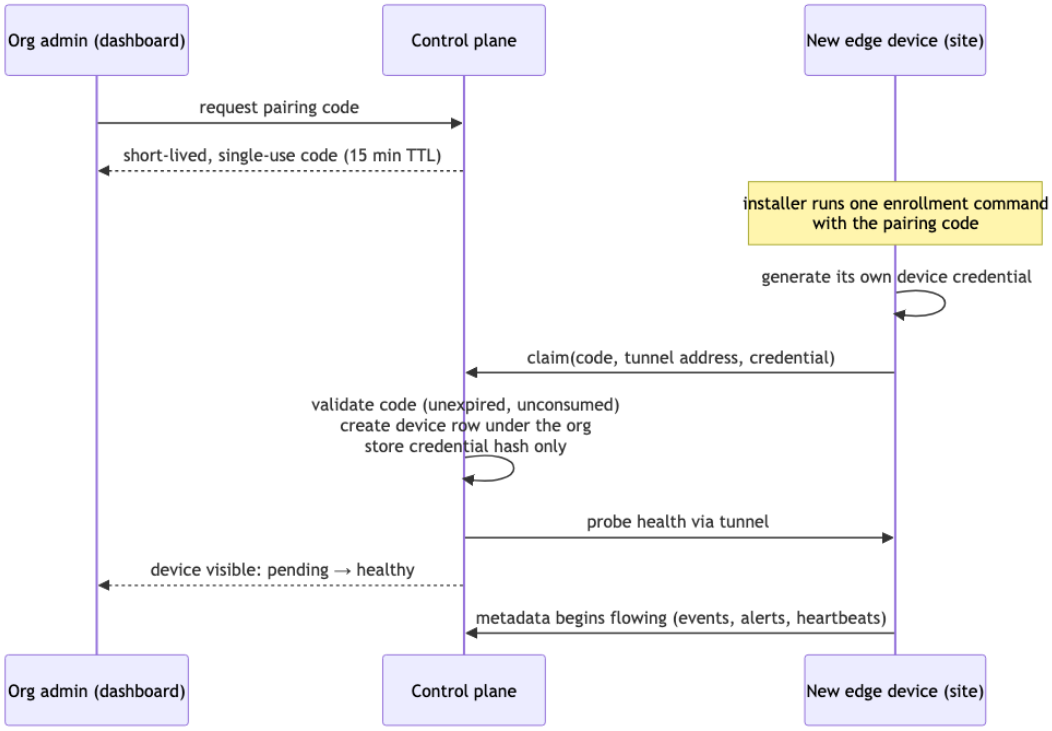

Figure 3: Figure 3. Pairing-code onboarding.

— completeness (all 26 present), uniqueness, and engine determinism are asserted mechanically. The device's use-case registry and the dashboard's rule UI are two views of one artifact. This is the unglamorous piece that makes the rest maintainable, and we would call it the paper's most transferable practice.

8 Economics and operations

- **Control plane at rest: ≈$0.** Serverless functions plus a small database at pilot-fleet scale sit within entry-tier pricing; there is no idle VM burning money while cameras see nothing.
- **Bandwidth: proportional to what humans watch,** not to what cameras see. Continuous upload is replaced by metadata (bytes per event) plus on-demand streams.
- **The device is the failure domain.** A site outage degrades that site's live video and pauses its metadata flow; the fleet, other tenants, and all historical metadata are unaffected. Devices restart into a known state (services are supervised; configuration is database-driven and reconciled).
- **Deploys are decoupled.** The dashboard ships continuously (build-gated, contract-tested); devices update on their own cadence; the typed contract is the compatibility keel between them.

9 When this architecture fits — and when it does not

It fits when footage is sensitive or regulated, uplink is constrained, sites are many and small, and the operational questions are metadata-shaped ("what happened, where, show me that one clip"). It fits poorly when the product requirement is *cloud video archival itself* — long-term offsite reten-

tion of full streams for evidentiary mandates — or when no on-site compute can be installed at all. In between, the split is tunable: nothing prevents selective clip escrow to cloud storage for flagged events, and the metadata plane already carries the references such a feature needs.

Appendix: scale facts (as measured, July 2026)

Fact	Value
Dashboard SPA	≈40,500 lines TS/React, 52 pages
Control-plane database	34 tables, 16 migrations
Edge runtime	≈85,500 lines Python, 361 modules
Analytics contract	26 use cases × 11 engine families, contract-tested on both sides
Live metadata sync (single 6-camera device)	35 sightings + 189 alerts / 60 s window (verified)
Whole-device memory under load	≈2.1 GB of 7.4 GB (see companion systems paper)

Contact: Dhi Technologies · dhi-tech.com