WHITE PAPER · V0.1

Keeping what a home robot sees and hears inside the home.

A privacy-first operating system for humanoid robots, where raw camera and microphone data *physically cannot* leave the device — and you still get cloud AI, on sanitized abstractions.

Cardea Project June 2026 Status: **core runnable today**, kernel jail in production posture

cardea.m9sh.com

ABSTRACT

Home humanoids are arriving — Tesla Optimus, Figure, 1X NEO, Apptronik, Unitree — and nearly every one runs a data-harvesting flywheel: cameras and microphones stream to a vendor's cloud (or to human teleoperators) to train models. It is the most invasive consumer device ever built, shipping with the weakest trust model: "*we promise*." Cardea is an operating-system layer that sits above any maker's hardware and enforces a single property: **raw camera and microphone data never crosses the home's network boundary**. This paper states the threat model, shows how the boundary is enforced by the operating system rather than a privacy policy, and — just as deliberately — marks exactly what is guaranteed today, what is production posture, and where the residual risk lives. A privacy product that overstates its guarantees has already failed.

1 The problem is now

A humanoid in your home is an always-on camera and microphone in every room — the kitchen, the bedroom, your children's rooms. The default

architecture streams all of it out.

This is not hypothetical. Two dated, verified failures already exist in this exact segment:

- **Unitree G1** was found exfiltrating audio and video roughly **every five minutes** (Sept–Oct 2025), and the behavior persisted across a firmware update.
- **1X NEO**, a ~$20K *home* robot, relies on **human teleoperators watching through your in-home cameras**, with half-baked privacy controls.

The default is being set right now. The data-harvesting architecture is calcifying into every shipping humanoid, and the capability layer (VLA models such as π0 and GR00T) has matured — so the one differentiator left for a privacy entrant is the **trust layer above the models**, which today is empty. The closest proven playbook is Apple's Private Cloud Compute: an attested boundary, a transparency log, and no raw egress. Nobody has built that synthesis for robots.

2 One property to enforce

Cardea exists for a single, falsifiable guarantee:

> *Raw camera and microphone data never crosses the home's network boundary. Only small, de-identified abstractions you have consented to may leave — and every crossing is logged.*

Everything else in the system is in service of that one line. Cardea is **not** a new kernel and **not** another humanoid or another model. It is a hardened userspace platform — "Android for robots," but the permission model is about *data egress*, not app features, and the default is *local*. It wraps incumbents rather than out-capabilities them: it runs on Jetson-class silicon, consumes open VLA models locally, and interoperates with ROS 2.

3 Threat model

The assets Cardea protects, in order of sensitivity:

1. **Raw exteroceptive streams** — camera frames, microphone audio, depth, the home map.
2. **Biometric identity** — face/voice vectors, "who lives here" (treated as *extreme PII*).
3. **The Hearth vault** — photos, documents, contacts, the home knowledge graph.
4. **Behavioral data** — routines, who is home when.

And the adversaries it is designed against:

Adversary	What they want	Why the default fails them
Cloud AI vendors / OEMs	the data flywheel: your footage to train models	their business *is* harvesting; "we promise" is the entire trust model
A compromised external model	to exfiltrate more than it was given	it only ever receives what already crossed the Threshold
Malicious / compromised skills	to phone home	on the robot they have no network interface at all
Network attackers	to intercept egress	only the Threshold has a route; egress is mTLS and logged

4 The shape of a humanoid

An Optimus-class humanoid is a multi-rate, hard-real-time control system with a perception and cognition pipeline stacked on top — a **frequency pyramid**. Crucially, the privacy boundary maps almost perfectly onto it:

Tier	Rate	Nature	Privacy class
Joint torque control	~1 kHz	hard real-time, µs jitter	proprioceptive
Whole-body balance / MPC	50–200 Hz	firm real-time	proprioceptive
Perception fusion	10–30 Hz	compute-heavy	**exteroceptive PII**
Task planning (VLA / LLM)	0.2–1 Hz	latency-tolerant	abstractions only

Raw, high-rate, privacy-sensitive data (camera, mic, biometrics, home map) lives in the lower and perception tiers and must never egress; only the top cognition tier — abstracted goals, sanitized scene text — is ever a candidate for external AI. Cardea's job is therefore three things: a **hardware abstraction layer** that tags each sensor with a privacy class; a **plane separation** so a stalled cloud call can never starve the 1 kHz control loop; and a **hard egress boundary** — the Threshold — between perception and any cloud model.

The stack

```
L4  Skills + External AI        WASM skills (capability-scoped) · cloud LLMs
    ──────────── THRESHOLD ────────────   ← the only thing that crosses the home boundary
L3  Cardea planes (Rust)
      ├─ RT control plane     isolated cores, SCHED_FIFO, EtherCAT → motors
      ├─ Sensor enclave       owns /dev/video,/dev/snd; local perception; NO network
      ├─ Threshold proxy      the ONLY networked process; policy + consent + audit
      ├─ Hearth vault         LUKS/TPM-sealed store + on-device semantic index
      └─ AI manager           model registry + router (local vs external by class)
L2  Cardea base               immutable image + systemd hardening + egress policy
L1  Linux kernel              PREEMPT_RT · netns · cgroups v2 · seccomp · Landlock
L0  Firmware / boot           Secure Boot → measured boot (TPM) → dm-verity
```

5 Two-layer enforcement, stated honestly

This is the single most over-claimed point in the whole category, so Cardea is precise about it. There are two layers, and only one of them stops a determined attacker.

Load-bearing: OS-level isolation

On the robot, every process except the Threshold proxy runs in a Linux **network namespace with no route to the internet**. The camera and microphone devices are reachable only by a **sealed sensor enclave** (device cgroups + udev). A compromised skill cannot leak what it has no wire to send. This layer does *not* trust Cardea's own application code.

Guarantee	Mechanism
Only the egress proxy reaches the internet	**network namespaces** — every other service has no route; a unix socket to Threshold is its only channel
Only the sensor enclave opens camera/mic	**device cgroups** + udev deny `/dev/video*`, `/dev/snd` to all else
Skills can't exfiltrate or escalate	**seccomp-bpf** + **Landlock** + a **WASM** capability sandbox (no ambient authority)
Tamper-evidence & attestation	**Secure/measured boot + TPM + dm-verity**; the Hearth key unseals only on an attested boot

Most of this is declarative `systemd` (`PrivateNetwork=`, `DeviceAllow=`, `RestrictAddressFamilies=`, `SystemCallFilter=`). The Threshold proxy is the *one* unit without `PrivateNetwork`. The entire egress boundary is a handful of auditable unit files plus one namespace — readable in an afternoon. That legibility is the trust asset.

Defense-in-depth: the Rust type system

Inside the application code, `Warded<T>` / `Sanitized<T>` and the sealed `Abstraction` marker turn accidental egress into a **compile error** and make every local-only read site greppable. They catch *Cardea's own* bugs early and document data flow. They are **not** what stops a determined attacker, and Cardea never claims they are.

> **THE HONEST FRAMING**
>
> Belt (kernel) and suspenders (types). Anyone who tells you a type system alone prevents data exfiltration is hand-waving. In this category, honesty is itself a feature.

6 The Threshold

The Threshold is the single egress chokepoint — the only component permitted to cross the home boundary. In the codebase, the boundary is also expressed in types so it is impossible to express raw data leaving:

```
// Raw sensor data — no API yields it for egress; only local sanitization, which records
provenance.
Warded<CameraImage>            // PrivacyClass::ExteroceptivePii

// Only an `Abstraction` may be carried across; raw pixel/audio types don't implement it,
// so `Sanitized<CameraImage>` is inexpressible.
Sanitized<SceneDescription>     // provenance: ExteroceptivePii (declassified)

// The single chokepoint. egress<T: Abstraction + Serialize>(...) — Warded won't compile.
Threshold::egress(topic, target, &sanitized) -> Verdict
```

`PrivacyClass` is a lattice — `Public < Proprioceptive < ExteroceptivePii < ExtremePii` — and provenance **folds** via a join, where the most restrictive class always wins: anything combined with biometric data becomes `ExtremePii` and is refused.

Declassification is perception's job

Doesn't everything touched by a face match become extreme PII? Yes — *unless perception explicitly drops the identity*, which is exactly its job. Faces are recognized **locally** against an enrolled gallery; that identity stays in a local-only scene. What becomes egress-eligible is a `SceneDescription` with counts and presence but **no names**. In the flagship demo, the cloud model sees "*1 person near the doorway, not a known resident*" — never "*Alice*."

The decision

```
raw sensor data?           → DENY: RawSensorData      (the core rule)
derived from extreme PII?  → DENY: ExtremePii         (biometrics never cross)
not consented?             → DENY: NoConsent          (default-deny by topic+target)
over the size budget?      → DENY: OverBudget         (covert-channel guard)
otherwise                  → ALLOW → transmit + audit
```

Consent is default-deny: the user grants `(topic, target)` pairs; everything else is refused. The **egress byte-budget** refuses a frame-sized payload smuggled out as a giant "abstraction" string — a bound on covert channels through the allowed path.

7 Verifiable, not trusted

Every decision — allow or deny — is appended to a **BLAKE3 hash-chained** log: each entry folds in the previous entry's hash, so truncation or modification is detectable by a `verify()`. Every field is human-readable. Cardea's core is open source: the entire boundary that decides what may leave your home is a handful of files anyone can audit.

> **HONEST SCOPE**
>
> A self-contained hash chain is tamper-*evident* against edits to a stored log, but a device that controls its whole log could recompute it. Real tamper-*resistance* requires externally anchoring the head — publishing it to the user's phone or a transparency log — so the device can't rewrite its own history. That anchoring, plus reproducible builds and remote attestation, is on the roadmap. Until it ships, "verify it yourself" is partially aspirational, and we say so.

8 What is real today

Cardea is a single pure-Rust workspace that builds clean on a developer laptop, no ROS or Gazebo required. The privacy spine and a deterministic simulator run today; the kernel jail is the production deployment posture and is *not* in the open repository because it cannot be exercised on a dev machine.

Component	What it does	Status
Privacy spine	`Warded` / `Sanitized` / `Abstraction`, the Threshold proxy, default-deny consent, byte-budget, hash-chained audit, the `raw_bytes_crossed() == 0` invariant	**Runnable**
Deterministic simulator	headless semantic-raycast camera + privacy-tagged HAL identical for sim and real; PD-stabilized bipedal balance with push recovery	**Runnable**
Full privacy loop	raw → de-identified abstraction → external model behind the Threshold → action; the `stranger-at-door` scenario doubles as a test asserting zero raw bytes crossed	**Runnable**
Egress scanner	"did my robot phone home?" — a drop-in OpenAI/Anthropic-wire proxy that recursively detects embedded base64 camera/mic data and blocks it (HTTP 403); plus a passive watch mode that flags phone-home beacons from connection metadata alone	**Runnable**
Kernel egress jail (L0–L2)	network namespaces, sealed sensor enclave, immutable image, measured boot, attestation — *where the privacy claim becomes real, not simulated*	**Production posture**
RT scheduler · WASM skill host · Hearth vault	multi-rate scheduler, capability-scoped WASM skills, encrypted local-first household vault	**Roadmap**

STATUS

The simulator demonstrates the **L3 plane separation** only — the Threshold is a real component, and nothing but the Threshold is ever handed a network handle. The robot adds the kernel jail underneath the very same code. No macOS timing number is evidence of real-time capability, and none is presented as such.

9 Known limitations

- **Sanitization sufficiency is not a type-system problem.** A type-valid sanitized caption can still name a person; an embedding can be invertible. The `Abstraction` marker stops the *trivial* one-line laundering — raw pixels are inexpressible as an egressable type — but it cannot judge *semantic* sensitivity. Mitigations: the size/rate budget, redaction, k-anonymity for aggregates, the audit log, and — load-bearing — the OS jail plus the fact that abstractions are produced by a reviewed local model, not arbitrary skill code.
- **Covert channels via allowed channels.** A skill with a legitimate egress grant could try to encode data into an allowed abstraction. This is bounded by budget, rate limits, and audit; the residual risk is acknowledged, not waved away.
- **Embedding egress** defaults to deny and is gated.
- **The load-bearing L0–L2 layer is not in the open repository** and cannot be demonstrated on macOS. One demonstrated egress bypass would damage the trust brand; that is the existential risk, mitigated by open-source legibility and by never claiming more than the architecture delivers.

10 Regulatory alignment

Data-minimization-by-architecture is a concrete **GDPR** asset (data minimization, purpose limitation) and an **EU AI Act** asset. Uncontrolled sensor egress from a home is a market-access and liability problem for OEMs *today* — independent of consumer volume. Cardea makes a humanoid defensible by architecture, not by a policy PDF, which is why the near-term buyer is an OEM selling into the EU, not yet the household.

11 Conclusion

The trust layer for home robotics is being defined right now, and it is empty. Cardea's bet is that the winning layer is vendor-neutral, open, and legible — a single audited Threshold over a kernel-enforced jail, with a public "did-it-phone-home" standard — precisely the position a data-harvester cannot credibly occupy. The guarantee is narrow and falsifiable on purpose: what your robot sees and hears stays home, and you can check that it did.

This document describes a system under active development. Components are marked runnable, production posture, or roadmap throughout; nothing here should be read as a claim that the kernel-enforced jail is exercised in the open repository. Corrections and security findings are welcome — Cardea treats honesty as a feature in a category where everyone overclaims.

Cardea

Guardian of the threshold.

No trackers. No cookies. No CDN. This page phoned home to no one.