🤫

HUSSH TECHNOLOGIES CORPORATION

Personal Supercomputing Infrastructure

A Systems Design

This paper describes the architecture of a fully on-device, consent-first personal data agent and the infrastructure that lets a person's own machines — and the AI agents acting for them — hold, organize, and act on personal information efficiently, safely, and without ceding custody. It states design goals and explicit non-goals, the data- and control-plane separation, the consent protocol and its receipt schema, a latency budget for the system's defining operation, the on-device/cloud-burst placement decision, the threat model, and an honest account of what is implemented versus targeted.

Revision 2026-06-12. Forward-looking design; sections marked TARGET are budgeted, not yet measured. Authored by hushh engineering with AI assistance (Claude).

Design tenets

1. Data is placed on hardware the owner controls; computation moves to the data, not the reverse.
2. Consent is a protocol with signed receipts and an append-only log — not a UI checkbox.
3. The owner is sovereign: capability with legible, single-party loyalty. Everything else is a non-goal.

1 · MOTIVATION

The distributed database of one.

Hushh Technologies Corporation was founded in August 2021, the same week a Bernedoodle puppy named Rio Grande joined the founder's household, and shortly after that household experienced identity theft. The two events are not unrelated in our telling: a dog is a system with high capability and a single, legible loyalty — it fetches, it guards, it is always on, and its incentives are observable rather than buried in a policy document. That is precisely the property absent from the software that manages personal data today.

Frame the individual as a system. Each person is, in effect, a sharded database with no primary: identity and records are partitioned across dozens of institutions — banks, brokerages, insurers, carriers, clinics, retailers, mail providers — each keying its shard on the same natural key, the phone number. The data describes the person; the person cannot read it whole, cannot run a query across it, and cannot revoke a downstream copy. The owner of record has the weakest read path of any party in the system.

The cloud-AI pattern of the early 2020s worsened this. It offered intelligence in exchange for custody: upload the corpus, trust the operator's retention policy, accept that the model provider now holds a copy. That trades one custodian for another. We wanted the opposite primitive — intelligence that runs where the data already lives, under the owner's key, with no new custodian created.

2 · REQUIREMENTS

Design goals and non-goals

Goals

- **G1 · Locality.** Personal data is processed on hardware the owner controls. Models run on-device for the common case; remote compute is the exception, not the default.
- **G2 · Legible consent.** Every access to personal data emits a signed, content-addressed consent receipt and an entry in an append-only transparency log the owner can read in full.
- **G3 · Revocability.** Revocation is a first-class, low-latency control-plane operation. A revoked grant stops future access and is itself a logged event.
- **G4 · Single-party loyalty.** The agent serves exactly one principal. No second reader is introduced — not the vendor, not an advertiser, not a model provider.
- **G5 · Agent-interoperability.** Third-party agents (the owner's or a trusted partner's) can act through the same consent-gated interface as the first-party agent, over open protocols.

Non-goals

- **N1.** We do not build a data lake. There is no central corpus of users' personal data, by construction — there is nothing to breach centrally.
- **N2.** We do not provide financial, legal, insurance, or medical advice. The agent organizes and surfaces; the human (or their licensed professional) decides.
- **N3.** We do not optimize for engagement, retention, or attention. The agent's success metric is task completion under consent, then getting out of the way.

- **N4.** We do not claim certifications we do not hold, or specifications we have not shipped. In-progress is stated as in-progress.

3 · ARCHITECTURE

System architecture

The system separates a data plane (where personal data is read, transformed, and stored — always on owned hardware) from a control plane (consent grants, receipts, the transparency log, device lifecycle — small, auditable, and the only thing that ever needs coordination). Personal data never traverses the control plane; the control plane carries capabilities and proofs, not content.

Layers

- **Hardware.** Owned device (Apple silicon today: M-series unified memory for local inference; A-series with the Secure Enclave as the identity/key root). Optional governed cloud tenant for burst, owned by the customer, with confidential computing and customer-managed keys.
- **Platform (hushh One).** Identity anchoring on the phone-number key; the PCHP consent engine (grants, receipts, revocation); the append-only transparency log; connector adapters to information sources; device/fleet lifecycle; the placement scheduler that decides on-device vs burst.
- **Agent (One Puppy).** The first-party agent. Fetches via consent-gated connectors, builds the normalized picture locally, runs local models for organization and insight, and exposes actions through platform intents (interoperating with Siri and other assistants).

Data plane / control plane

```
# control plane (small, auditable) — capabilities + proofs only
Grant{ principal, source, scope, expiry, sig }
Receipt{ grant_id, action, fields[], purpose, ts, content_hash, sig }
LogEntry{ prev_hash, receipt, ts }   # append-only, hash-chained

# data plane (on owned hardware) — content never leaves without a Grant
fetch(source, Grant) -> raw            # connector, scoped by grant
normalize(raw) -> picture.json         # local
infer(picture, local_model) -> insight # local; burst only if oversized
```

Because the control plane manipulates capabilities and hashes rather than content, the audit trail can be complete without the audit system ever seeing the data. The transparency log is hash-chained: any tampering with a past entry invalidates every subsequent hash, giving the owner a verifiable history of access.

4 · THE CONSENT LAYER

PCHP: consent as a protocol

PCHP (the permission & consent layer) treats every data access as a transaction that must present a valid, unexpired, sufficiently-scoped grant, and that must emit a receipt. Defaults are closed: a freshly provisioned device consents to nothing until the owner grants, item by item, during the in-person setup ceremony.

Receipt schema (illustrative)

```
{
  "grant_id":   "g_7af3…",            # which permission authorized this
  "action":     "fetch.statements",   # what was done
  "fields":     ["balance","renewal_date"],
  "purpose":    "assemble financial picture",
  "content_hash":"sha256:…",          # proof-of-what, not the what
  "ts":         1781…, "device":"the owner's",
  "sig":        "ed25519:…"           # device key in Secure Enclave
}
```

Receipts are content-addressed and signed by a device key held in the Secure Enclave, so a receipt proves that a specific action over specific fields occurred without disclosing the field values. Revocation writes a tombstone grant; the scheduler refuses any future action whose grant chain is tombstoned, and the refusal is itself logged. This makes "what does this agent know about me, and who said it could" a query the owner can actually run.

5 · THE DEFINING OPERATION (TARGET)

The 60-second fetch: a latency budget

The operation that defines the product is first-fetch: from a single connected account to a complete, normalized financial picture with a quantified insight, in about a minute, entirely on the owner's device. We budget it the way any latency-critical path is budgeted — explicitly, with headroom — and we label it TARGET because it is in active development, not yet measured at production scale.

Stage	Budget	Notes
OAuth consent + token	≤ 8 s	human-paced; one tap, scoped read grant
Source enumeration	≤ 6 s	identify financial senders/domains, last 12 mo
Fetch + parse	≤ 25 s	connector pulls; local statement/alert parse
Normalize → picture	≤ 8 s	accounts, balances, recurring, renewals
Local inference (insight)	≤ 10 s	on-device model; the dollar-denominated finding
Render + receipts	≤ 3 s	picture view + transparency-log writes
Total	≤ 60 s	tail-latency target; p50 materially lower

Two design consequences follow from the budget. First, fetch+parse dominates, so connectors are parallelized per source and parsing is incremental — the picture renders progressively rather than blocking on the slowest source. Second, inference is bounded: if a model that fits the device cannot produce the insight within budget, the scheduler may burst that single step to the governed cloud tier (below) rather than blow the tail. Personal data crossing that boundary is itself a consent event, logged.

6 · PLACEMENT

On-device vs governed-cloud burst

Placement is a scheduling decision, not a dogma. The default site of computation is the owned device, because that is where the data already is and moving data is the expensive, risky operation. Compute moves to the cloud only when a workload provably exceeds the device — a large fine-tune, a batch beyond local memory, a model larger than unified memory can hold.

When it does, it bursts into a tenant the customer owns: confidential-computing VMs, customer-managed encryption keys, and the same receipt/log discipline as on-device. We characterize the economics honestly — independent analysis places Google's Ironwood TPU at materially lower total cost than comparable GPU servers for model-scale work, and Arm-based instances (Axion-class) as the cheapest substrate for always-on orchestration — but the architectural point is governance, not price: the cloud is a rented extension the owner governs, never a custodian. Owned core, governed surge.

7 · SECURITY

Threat model

What we defend against

- **Central breach.** There is no central corpus (N1). Compromising the platform yields capabilities and hashes, not anyone's personal data.
- **Silent access.** Every read is gated by a grant and produces a hash-chained log entry; covert access is detectable by the owner.
- **Custody creep.** No second reader is introduced. A model provider or partner agent acts only through scoped, revocable grants — it never receives a durable copy.
- **Device loss.** Keys live in the Secure Enclave; data at rest is encrypted under those keys; a lost device is revoked from the control plane and its grants tombstoned.

What we do not yet claim

- **Formal certification.** FedRAMP High is being pursued, not held. Stated as in-progress, always.
- **Nation-state endpoint compromise.** An attacker with persistent code execution on the owner's unlocked device is outside the model; we reduce blast radius (Enclave keys, least-scope grants) but do not claim immunity.

8 · WHAT IS REAL, WHAT IS TARGET

Implementation status

Engineers judge a design by the honesty of its status section. Here is ours.

Component	Status
Consent receipts + append-only log (data model, backend)	Implemented (v0.1, tested)
Reservation/commerce backend + Order MCP server	Implemented (v0.1, tested)
Agent-native MCP / A2A / AP2-class mandate surface	Implemented (interface), expanding
60-second on-device first-fetch pipeline	In development — the critical path
On-device model selection + inference budget	Design + prototyping
Governed-cloud burst boundary	Designed; not yet wired
Agent runtimes: Android / Windows / Linux	Roadmap, demand-gated
FedRAMP High authorization	In progress

The single most important line is the fourth: the first-fetch pipeline is the product, and it is the work in front of us. Everything else in this paper exists to make that operation trustworthy when it runs.

9 · INVITATION

A computer that is finally yours

The personal computer promised the machine would work for the person. The cloud era quietly inverted that. Personal supercomputing infrastructure is the correction: capability that is local, loyal, legible, and owned — built to the standard a good dog sets, useful every day and never once betraying the household.

We are looking for systems engineers, on-device ML engineers, and security engineers who find the above under-specified in the right places and want to fix it. The hard, unsolved, interesting problems are the on-device inference budget, the connector reliability surface, and the placement scheduler.

Build with us

engineering@hushh.ai · partners@hushh.ai · hushh.ai/one