Who Controls Your AI Agent?

Separating the Operator from the Analyst to Close the Agentic AI Surveillance Gap

Kennt Kim
Calida Lab
kennt@calidalab.ai

Working Paper v1.0 — July 2026

Abstract

Agentic AI systems such as Open Interpreter, Claude Code, and Cursor delegate *operator privileges* — arbitrary code execution, filesystem access, and network calls — to cloud-hosted large language models. We argue that the industry has normalized a qualitatively new surveillance surface by conflating two fundamentally different trust models: *data sharing* (bounded, user-gated, discrete) and *remote system control* (unbounded, continuous, delegated). Every prompt, tool invocation, file content, and command output becomes observable by a single US-jurisdiction provider and legally compellable via instruments including subpoenas, search warrants, National Security Letters, and FISA §702. Recent US legal developments — chat logs introduced as criminal evidence, a litigation-hold order that overrode a provider's own deletion policy, and the first known search warrant seeking to unmask a user from their prompts — demonstrate that this threat model is no longer hypothetical.

We present SnowClaw, a reference architecture that closes this gap with three composable mechanisms. First, *operator/analyst decomposition*: the component that selects and invokes tool calls (the operator) is pinned to a local model with network egress blocked by a capability-gate framework, while content reasoning (the analyst) is exposed to the operator as an ordinary tool whose backend may be local or cloud. Second, *three-domain trust separation*: cloud analyst queries travel through an entry relay, a gateway, and a token issuer operated as distinct trust domains, so that no single party ever holds both a user's identity and their content — a topology equivalent to Oblivious HTTP (RFC 9458), extended with anonymous usage tokens following the Privacy Pass architecture (RFC 9576) to unlink billing from queries. Third, an on-device *reversible pseudonymization gate* that replaces sensitive entities with format-preserving placeholders before any query leaves the device, falling back to a local analyst whenever de-identification cannot be assured.

The architecture is grounded in a partial reference implementation built on a production-deployed end-to-end encrypted messenger and follows an explicit claims-to-invariants discipline: every privacy claim is mapped to a machine-checkable test, and unimplemented layers are disclosed rather than implied. We analyze what each adversary — including a legally compelled operator of each component — can observe or produce, compare threat-model coverage against stock cloud agents, local-only systems, and confidential compute, and state the residual limitations honestly.

Keywords: privacy-enhancing technologies; agentic AI; local-first inference; operator/analyst separation; Oblivious HTTP; Privacy Pass; reversible pseudonymization; threat modeling.

1 Introduction

Agentic AI systems achieve their productivity gains by granting language models what we call *operator privileges*: the ability to execute code, read and write files, and issue network requests on the user's machine, iteratively and with limited human review. In the dominant deployment pattern — Open Interpreter [18], Claude Code [1], Cursor [2], and their descendants — the model exercising these privileges is hosted by a cloud provider. The consequence is easy to state and easy to underestimate: **every prompt, every tool**

invocation, every file the agent reads, and every command output is transmitted to, observable by, and retained at the discretion of a single provider, typically under US jurisdiction.

We argue this design conflates two categorically different trust models. *Data sharing* is bounded, user-gated, and discrete: the user decides, per interaction, what to reveal. *Remote system control* is unbounded, continuous, and delegated: once an agent session begins, the union of everything the agent touches flows outward, and the user cannot meaningfully enumerate it in advance. A chat assistant leaks what you choose to tell it; an agent leaks a forensic image of your working life. A legal demand served on a chatbot provider yields conversations; the same demand served on an agent provider yields file contents, directory structures, credentials embedded in configuration, shell history, and a behavioral record of how the user works.

1.1 The threat model is no longer hypothetical

Three independent US legal developments between 2025 and 2026 converted this concern from speculation into precedent, at each of the three levels at which state power reaches data.

Criminal evidence. In the federal prosecution arising from the 2025 Pacific Palisades fire, prosecutors introduced the defendant's ChatGPT history — generated images, emotional queries, and questions about liability for accidental fires — as evidence of intent [27]. Contemporary reporting described chat histories as "a treasure trove" for investigators [8]. Whatever the eventual verdict, the evidentiary door is open: conversations that users experience as private, even confessional, are ordinary business records in litigation. OpenAI's chief executive has publicly acknowledged that no legal privilege attaches to them [26].

Civil discovery. In *The New York Times v. OpenAI*, a federal court in May 2025 ordered OpenAI to preserve *all* ChatGPT output logs — including conversations users had deleted and conversations covered by the provider's own 30-day deletion policy [19]. OpenAI called the order a "privacy nightmare" and contested it; the order was affirmed, and although later narrowed [12], the court subsequently ordered production of twenty million de-identified conversations to the plaintiffs [28]. The lesson generalizes beyond this case: **a provider's privacy policy, including its deletion promises, is subordinate to a court order**. Policy-based privacy did not survive first contact with litigation.

Law-enforcement process. In October 2025, the Department of Homeland Security obtained what is believed to be the first search warrant compelling OpenAI to identify a user from their prompt history [5, 23] — the generative-AI analogue of reverse keyword warrants long served on search engines. Beyond warrants, US law provides compulsion instruments that operate without user notice: National Security Letters [30] and FISA §702 directives [32] carry gag provisions, so the user may never learn that disclosure occurred; the Stored Communications Act [31] and the CLOUD Act [33] extend reach across data categories and borders. Under the third-party doctrine [24], records held by a provider receive systematically weaker constitutional protection than data on the user's own device, and the partial exception carved by *Carpenter* [25] has not been extended to chat or agent logs.

1.2 Only architectural non-possession survives compulsion

These developments share a single structural moral. Every mitigation that depends on provider behavior — retention limits, deletion policies, "we do not train on your data" commitments — is a promise, and promises yield to compulsion. The only property that survives a court order is *non-possession*: a party cannot be compelled to produce what it never held. This is the operating principle that lets Signal answer subpoenas with nothing but a registration timestamp and a last-connection date [22]. The design question for agentic AI is therefore not "which provider do we trust?" but "**how little must each party hold for trust to be unnecessary?**"

1.3 Contributions

SnowClaw is our answer for consumer agentic AI. This paper makes four contributions:

1. **A compelled-provider threat model for agentic AI** (§3). Existing agentic-security work, led by OWASP and industry red teams, focuses on attacks *against* agents — prompt injection, tool misuse, memory poisoning — and implicitly assumes a benign provider [20]. We model the structurally different adversary: the provider itself, or a state actor compelling it.

2. **Operator/analyst decomposition** (§4.1). We separate the agentic pipeline along an orthogonal axis — *execution* versus *interpretation*, not task difficulty. The operator (tool selection and invocation) is pinned to a local model with no network egress; interpretation is exposed to it as an ordinary tool whose backend it cannot distinguish. We explain why this decomposition is deliberately *calibration-free*: it removes the self-assessment decisions that small local models empirically cannot make.

3. **Three-domain trust separation from standard primitives** (§4.3). Cloud analyst queries traverse an entry relay, a gateway, and a token issuer in distinct trust domains, so no single party holds (identity, content) simultaneously. The topology is deliberately equivalent to Oblivious HTTP [29]; billing is unlinked from queries with Privacy Pass tokens [9, 10], the standardized descendant of Chaumian blind signatures [7]. On top, an on-device *reversible pseudonymization gate* (§4.4) minimizes what the frontier model can read at all, with a fail-closed fallback to local inference.

4. **A claims-to-invariants discipline** (§5). Every privacy claim in this paper is mapped to a machine-checkable invariant test; layers that are specified but not yet implemented are stated as such. The reference implementation builds on SnowChat, our independently developed, production-deployed end-to-end encrypted messenger, whose Signal-protocol library (X3DH, Double Ratchet, Sealed Sender) ships 115 library-level tests.

2 Background and Related Work

Agentic systems. Open Interpreter demonstrated the value of language-model-driven execution on consumer machines [18]; Claude Code and Cursor brought the pattern to mainstream software development [1, 2]. All retain a cloud model in the operator role: the cloud emits the tool calls, so the cloud must see the tool results. The exposure is not a bug in these systems; it is the direct consequence of where the operator runs.

Local-only inference. Ollama, LM Studio, and similar runtimes [17] eliminate the cloud entirely, at the cost of a capability gap: models that fit on consumer hardware trail frontier models on long-context reasoning and analysis. In practice the gap pushes users back to hosted services for exactly the tasks that involve their most sensitive material — the documents long enough, and the questions hard enough, to exceed local capability.

Confidential compute. Apple's Private Cloud Compute is the strongest deployed mitigation in the provider-trust family: hardware-attested nodes, ephemeral memory, no operator access to plaintext [3]. But PCC's guarantee is about *content confidentiality against the infrastructure operator*, not anonymity: the device authenticates to Apple, requests are attributable to an account, and the legal posture of the metadata is that of any provider-held record. Confidential compute narrows *what* can be compelled; it does not sever *who* from *what*.

Anonymity primitives. Chaum's blind signatures [7] began the line of work on unlinkable credentials that Privacy Pass standardized for the modern web [6, 9, 10]. Signal's Sealed Sender hides sender identity from the message router itself [15]; follow-up work quantified its limits under statistical disclosure attacks [16]. Oblivious HTTP [29] standardizes the two-party split — a relay that sees the client but not the content, a gateway that sees the content but not the client — with HPKE encapsulation [4]. Mix networks and onion routing [11] provide stronger network-layer anonymity at higher latency. These primitives are mature; what has been missing is their composition into a consumer agentic-AI architecture. To our knowledge, SnowClaw is the first publicly documented consumer-facing agent architecture that combines a local operator, an anonymized cloud analyst path, and unlinkable usage accounting.

	Adversary	Capability	What SnowClaw lets it hold
A1	Entry-relay operator†	Full view of own servers; compellable	Client IP, ciphertext, token validity
A2	Gateway operator†	Decrypts analyst queries; compellable	Pseudonymized content; no IP/identity
A3	Frontier provider	Observes/retains what reaches it	Pseudonymized text via pooled API keys
A4	Token issuer	Payment records; compellable	Purchase identity; no query linkage
A5	Network observer	Passive/active MITM	Ciphertext; timing (§7)
A6	Malicious content	Indirect prompt injection	Nothing (structural isolation, §4.2)
A7	Device seizure (at rest)	Disk forensics	Encrypted vault only

Table 1: Adversary classes. Each of A1–A4 is modeled as *honest-but-curious and legally compellable*: whatever the component holds, assume a court can obtain. †A1 and A2 are assumed non-colluding (§7).

3 Threat Model and Design Principles

We distinguish two axes of adversary that prior documents (including our own) tended to blur: adversaries that *penetrate the device* and adversaries that *compel a provider*. Application architecture cannot defeat a live implant on the user's device; no design in this space should claim otherwise. SnowClaw targets the second axis, plus at-rest device seizure.

Table 1 lists the adversary classes. The defining move is that A1–A4 are each modeled as *compellable*: we do not ask whether the operator of a component is trustworthy, but what a court order served on that component can yield. Out of scope are live device implants (nation-state endpoint compromise) and a global passive adversary correlating all traffic; the latter is mitigated only best-effort (§7).

Five design principles follow, and function as the constitution of the architecture:

1. **Architectural guarantees only.** Every privacy property must hold because a party *cannot* violate it, not because it promises not to. "We do not log" is a policy; "there is nothing to log" is an architecture.

2. **Fail-closed.** If a protective layer cannot verify that it is active, the path it protects is disabled — never silently degraded. (This principle is a direct lesson from our own v1 implementation, which printed "sandbox active" without verifying activation; §5.3.)

3. **Claims map to invariants.** Every claim in this paper corresponds to a machine-checkable test (Table 2). A claim without a fixture does not appear.

4. **Standard primitives first.** Where an IETF-standardized construction exists (OHTTP, Privacy Pass, HPKE), we adopt it or document exact equivalence, rather than designing bespoke cryptography.

5. **Residual limits are disclosed** (§7), not discovered by reviewers.

4 Architecture

4.1 Operator/analyst decomposition

SnowClaw splits the agentic pipeline along the axis of *execution versus interpretation*. The **operator** is the component that selects and invokes tool calls — reading files, running commands, querying device data. It runs exclusively on-device (Gemma 4 [14]: 4B Dense or 26B-A4B MoE via llama.cpp Metal [13] on macOS; the 2.3B E2B variant on phones), and its process has no network egress: the inference runtime is in-process, the AI modules are forbidden by construction from importing network layers, and outbound traffic is blocked at the OS level (§4.2). The **analyst** is the content-reasoning role — summarize this document, explain this error, draft this reply. Crucially, the analyst is exposed to the operator not as a phase of the pipeline but as an ordinary tool: `analyst.ask(question, context)`.

This decomposition is deliberately *calibration-free*. Our first architecture (v1) asked each model to assess its own capability: "handle this if you can, otherwise delegate to a larger model." It failed in a characteristic way — small models are poorly calibrated on *can-I-do-this* meta-questions, oscillating between overconfidence (which leaks tasks to the wrong tier) and overcaution (which makes the agent useless). The same models are,

however, well-trained on *tool selection*. v2/v3 therefore remove every self-assessment decision from the model: whether the analyst backend is the local mini-analyst or a frontier cloud model is decided by a fixed routing policy bound to an explicit user capability setting (`local_only` / `prefer_local` (default) / `prefer_cloud` / `cloud_only`), and the operator cannot observe which backend served a call. The routing decision is thereby bound to *user intent*, not model self-assessment, while the execution surface remains entirely local.

Two structural consequences matter for the threat model. First, the cloud analyst *cannot* issue tool calls: its responses re-enter the operator loop as structured tool results (§4.2), not as instructions. Second, the cloud path becomes optional at the architecture level — a user who selects `local_only` has a complete, if less capable, agent with zero cloud exposure.

4.2 Execution path: fail-closed local enforcement

The operator's privileges are mediated by a **capability gate**: a policy framework in which egress-bearing capabilities default to off, write-operations require confirmation, and permissions are granted just-in-time. Beneath it sit three enforcement layers:

1. **Model isolation by construction.** Inference runs in-process; the AI modules cannot import HTTP or socket layers (enforced by static audit). Runtime external HTTP from the agent process is zero by default.

2. **OS-level egress blocking, verified.** Outbound network access from the execution sandbox is blocked by firewall policy. The v3 rule — adopted after our v1 implementation shipped a fail-open bug (§5.3) — is that the "sandbox active" state is entered only after a runtime self-test: the sandbox attempts an outbound connection to a canary host and must observe it fail. If verification fails, high-risk tools (code execution) are disabled or require explicit per-use consent, with a visible warning. Displayed state and actual state cannot diverge silently.

3. **Structural isolation of analyst output.** Analyst responses return to the operator as structured tool-result records tagged `analyst_text`; the operator's parser syntactically ignores code fences and tool-call patterns inside them, and boundary sanitization strips invisible-Unicode injection vectors (tag blocks, bidirectional overrides, zero-width characters). The property "a document that tricks the analyst cannot make the operator execute code" is enforced by adversarial regression fixtures — including a PDF whose body embeds `rm -rf` inside a code fence — that must pass in CI (invariant I1, Table 2).

4.3 Three-domain trust separation on the cloud path

When the routing policy selects the cloud backend, the query traverses three parties operated as distinct trust domains (Figure 1):

Entry relay (domain R). The client seals the query to the gateway's public key (X25519 + XSalsa20-Poly1305 with per-request ephemeral sender keys — the Sealed Sender pattern [15] our messenger already runs in production; the construction plays the role HPKE [4] plays in OHTTP). The relay sees the client's IP address and an opaque ciphertext. It verifies the attached usage token *without learning identity*, applies rate limits, batches and jitters forwarding to blunt timing correlation, and holds no durable state: the relay codebase is structurally excluded from persistence layers (no database imports; audited, invariant I3).

Gateway (domain G). The gateway decrypts the pseudonymized query, forwards it to the frontier provider over pooled API keys — so the provider observes an aggregate stream, attributable to the gateway, not to users — and seals the response back. It never sees an IP (only the relay connects to it) and never a stable client identifier (envelopes carry per-request ephemeral keys, invariant I4).

Token issuer (domain T). Usage accounting must not become the identity leak. Clients purchase token batches from the issuer using the Privacy Pass issuance protocols [6]: the issuer signs blinded tokens, so tokens presented at the relay are unlinkable both to the purchase and to each other [7, 9]. The issuer knows

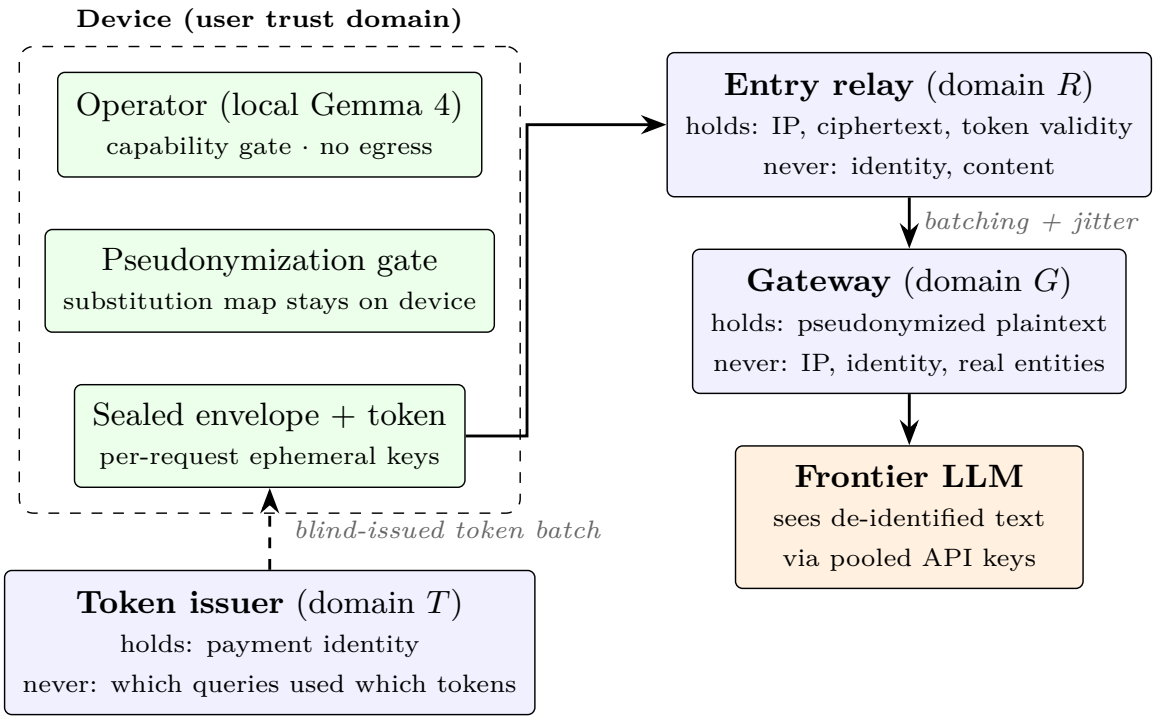

Figure 1: The cloud analyst path. No single domain holds (identity, content) simultaneously: R sees who is connecting but not what is asked; G sees what is asked — pseudonymized — but not by whom; T knows who paid but not what for. The topology is equivalent to Oblivious HTTP (RFC 9458) with a Privacy Pass token layer (RFC 9576) and an on-device content-minimization layer added.

who paid and how much; it cannot know which queries — or even which relay sessions — the tokens funded (invariant I5). Double-spend prevention lives at the gateway via a nonce cache.

The decomposition is intentionally isomorphic to Oblivious HTTP [29]: R is the Oblivious Relay Resource, G the Oblivious Gateway Resource. We claim no novelty for the split itself — that is the point. The contribution is the composition: OHTTP-equivalent transport, plus standardized unlinkable payment, plus the content-minimization layer below, assembled for the agentic setting where the local operator (not the cloud) holds execution authority.

4.4 The pseudonymization gate

Trust separation unlinks the query from the user's identity, but the frontier provider still reads the query *content* — and content frequently re-identifies its author (names, addresses, credentials, internal hostnames). SnowClaw therefore interposes a final on-device layer on the cloud path: a *reversible pseudonymization gate*.

Detection is layered. **L1** is deterministic and reproducible: pattern scanners for secrets (API keys, private keys, bearer tokens), email addresses, phone numbers, IP addresses and hostnames, payment card numbers, and wallet addresses — plus a *device-local dictionary* built from the user's own contacts, calendar, and account names. This dictionary is the on-device advantage: a cloud DLP service cannot know that "Mina" is the user's daughter, but the device does, at exact-match precision, without that knowledge ever leaving the device. **L2** is a local-model NER pass flagging person, organization, and location spans that L1's patterns cannot enumerate.

Detected entities are replaced with format-preserving, session-consistent placeholders (`PERSON_1`, `HOST_2`, key-shaped placeholders that preserve prefix and length), so the analyst's reasoning over structure survives. The substitution map exists only in device memory and the local encrypted vault; the gateway response is rehydrated on-device before the operator sees it.

Detection is necessarily best-effort, so the gate is governed by the fail-closed principle: **if a query cannot**

be confidently de-identified — L2 uncertainty, high-entropy blobs, dense unfamiliar identifiers — the router does not send it with a warning; it downgrades the request to the local mini-analyst. The cloud path receives only what the gate affirmatively cleared (invariant I6). We state plainly what this layer is: exposure-surface reduction, not content hiding. Stylometry and structural features survive pseudonymization (§7); this is why the default routing policy is `prefer_local` and the gate is the last, not the only, line of defense.

4.5 Request lifecycle

End to end, a cloud analyst call proceeds: (1) the operator invokes `analyst.ask`; (2) the routing policy selects the cloud backend (user setting permitting); (3) the pseudonymization gate de-identifies the query or fails closed to local; (4) a Privacy Pass token is attached; (5) the query is sealed to the gateway key with fresh ephemeral keys; (6) the entry relay verifies the token, rate-limits, batches, forwards; (7) the gateway checks double-spend, decrypts, queries the frontier model over pooled keys, seals the response; (8) the device rehydrates placeholders and delivers the text to the operator as an isolated `analyst_text` tool result. The user-visible latency cost is one relay hop plus gate processing; the privacy gain is that steps 6–8 each execute inside a domain that could be compelled tomorrow without yielding the pair (who, what).

5 Implementation Status and Claims Discipline

5.1 What is implemented

The reference implementation is anchored in SnowChat, our independently developed, production-deployed E2EE messenger, built on a Pure Dart implementation of the Signal protocol suite — X3DH, Double Ratchet [21], Sender Keys, and Sealed Sender — with 115 library-level tests (HKDF RFC vectors, all-zero DH rejection, replay resistance, forward-secrecy verification) and six rounds of internal code audit. The Sealed Sender construction used on the cloud path is not merely cited; it is the same code that routes production message traffic.

On top of this foundation, SnowClaw currently implements: the local operator loop with the capability-gate framework (macOS: Gemma 4 4B/26B-A4B via llama.cpp Metal; Android/iOS: Gemma 4 E2B via an on-device runtime, sharing the Dart protocol code); the structural isolation of analyst output with its adversarial fixture suite; the sealed-envelope client and a relay skeleton whose RAM-only property is enforced by import audits and unit tests; and an E2EE phone–desktop delegation channel.

5.2 What is specified but not yet implemented

Consistent with our claims discipline, the following v3 layers are *designed and scheduled, not shipped*: the Privacy Pass issuance and verification path (the current relay accepts well-formed placeholder tokens — wire-format frozen so the verifier can land without client changes); the pseudonymization gate (L1/L2 stack specified as in §4.4); the physical separation of relay and gateway into distinct operational domains (today both would be operated by Calida Lab; separation begins as distinct keys, hosts, and log regimes, with independent operation as the stated goal); and extension of the encrypted vault to the mobile message store. Until these land, the honest description of the deployed system is: *local-operator agentic execution with structurally isolated cloud analysis over a sealed, identity-minimizing relay — with anonymity and content-minimization layers specified but pending.*

5.3 Lessons encoded in the design

Two of v3's principles are generalizations of our own failures, which we report because they are instructive for anyone building in this space.

The calibration cliff is real. v1 delegated by model self-assessment and failed as described in §4.1. The durable fix was not a better prompt but removing the meta-decision from the model entirely.

Fail-open protection is worse than none. An early build constructed its network sandbox object but never activated it — while printing "sandbox active (outbound network blocked)" at startup. Nothing in the

#	Claim	Invariant / fixture	Status
I1	Analyst has no device agency	Adversarial-content fixtures (incl. hostile PDF)	Implemented
I2	Operator has zero egress	Static import audit + canary egress self-test	Partial[a]
I3	Relay holds no plaintext/state	No-persistence import audit + unit tests	Implemented
I4	Gateway never sees identity	No stable IDs in envelope; ephemeral-key tests	Implemented
I5	Token unlinkability	Privacy Pass test vectors (RFC 9578)	Specified
I6	Gate blocks sensitive entities	Red-team corpus → zero leaks; fail-closed route	Specified
I7	No plaintext at rest	Cold-storage scan finds no analyst plaintext	Partial[b]

Table 2: Claims-to-invariants matrix. [a]Import audit and gate checks implemented; the canary self-test replaces a v1 fail-open bug (§5.3). [b]Desktop vault implemented; mobile store extension scheduled.

	Cloud agent	Local-only	Conf. compute	SnowClaw (v3)
Frontier-level analysis	•	—	•	•
Provider sees content	full	—	attested-hidden	pseudonymized
Provider sees identity	•	—	•	—
Provider has device agency	•	—	—	—
Compellable (who, what) pair	•	—	•[c]	—[d]

Table 3: Threat-model coverage. [c]Confidential compute hides content from the operator but the account linkage and metadata remain compellable. [d]Requires the R–G non-collusion assumption (§7); compelling any single SnowClaw domain yields at most one of (who, what).

architecture made that divergence impossible; it was found by review, not by a test. The v3 rule — protective layers must verify themselves active, and their guarded paths must close otherwise — exists so that this class of bug becomes a build failure rather than a silent privacy hole. This is also why Table 2 exists: a privacy claim without a machine-checkable invariant eventually drifts from the code.

6 Security Analysis

We analyze by asking, for each party, *what does a court order served on it yield?*

Entry relay (R). Connection metadata: client IPs, timing, ciphertext volumes, token-validity events. No identities (tokens are unlinkable; envelopes are sealed), no content, and — structurally — no logs to preserve. A litigation hold served on R has nothing retrospective to attach to, which is precisely the property the NYT preservation order demonstrated providers lack [19].

Gateway (G). Pseudonymized query plaintext transiting RAM, plus the frontier API relationship. No IPs, no account linkage, no substitution maps (those never leave devices). Compelling G prospectively turns it into a wiretap of de-identified text — serious, and disclosed — but it cannot answer "what did user X ask?" because nothing at G resolves X.

Frontier provider. Receives de-identified text attributable only to the gateway's pooled keys. The provider's retention, training, and compliance posture apply to an aggregate stream; the reverse-warrant pattern of [5] — resolve a prompt to an account — terminates at the gateway's account, not a user's.

Token issuer (T). Payment identities and issuance volumes: the same information a bank already holds about a subscription. The blind issuance protocol makes redemption cryptographically unlinkable to purchase [6].

The device. All real secrets concentrate here by design: the substitution maps, session keys, history — inside an encrypted vault (I7), because the device is the one party the user actually controls, and at-rest seizure (A7) is in scope.

Table 3 compares coverage against the alternatives. The stock cloud agent concedes every row; local-only concedes capability; confidential compute concedes identity and compellability of the account linkage. SnowClaw's row is conditional on stated assumptions — non-collusion and the best-effort layers — which we now make explicit.

7 Limitations

1. **Non-collusion of R and G.** If the relay and gateway collude (or are compelled together and correlate), IP-to-content linkage is reconstructible. This is OHTTP's assumption, inherited knowingly. Initially both domains will be operated by Calida Lab with organizational separation only — distinct keys, hosts, and log regimes; that is disclosed as the architecture's weakest present link, and migration of one domain to an independent operator is the stated roadmap.

2. **Small anonymity sets.** Early deployment means few concurrent users; batching and jitter blunt but do not defeat statistical disclosure and timing correlation, as the sealed-sender literature shows [16]. Anonymity here grows with adoption; we do not claim strong network-layer anonymity at small scale and note Tor/Nym transport as a composable upgrade [11].

3. **Re-identification through style and structure.** Pseudonymization does not remove authorial style, code idiom, or rare structural fingerprints. The gate reduces the exposure surface; it does not anonymize prose. The mitigations are the `prefer_local` default and fail-closed routing, not a stronger claim.

4. **Semantic inference at the frontier.** The frontier model reads coherent (de-identified) queries and can infer sensitive facts from what remains; content minimization is not content hiding.

5. **Device compromise is out of scope.** A live implant sees everything before any gate. SnowClaw addresses provider-side compulsion and at-rest seizure; endpoint integrity is delegated to the platform.

Future work follows directly: land the Privacy Pass verifier and the pseudonymization gate (with a red-team corpus as the acceptance bar); separate the gateway operationally; evaluate the capability cost of the local operator and the latency cost of the relay path against stock cloud agents; and quantify gate recall/precision on realistic corpora.

8 Conclusion

The agentic-AI industry has, in three years, normalized an arrangement that no one would have accepted if proposed directly: a single remote party that executes code on your machine, reads what it touches, retains what it reads, and answers to whatever legal process arrives. The 2025–2026 record — chat logs as criminal evidence, deletion policies overridden by litigation hold, users unmasked from prompts — establishes that this arrangement's risks are not theoretical, and that provider policy is not a defense.

SnowClaw's answer is separation: of execution from interpretation, so that no remote party holds agency over the device; of identity from content, so that no single party can be compelled into producing both; and of payment from usage, so that accounting does not quietly rebuild the identity link. None of the individual mechanisms is novel — that is deliberate. Sealed-sender routing runs in our production messenger; the relay topology is standardized as Oblivious HTTP; the tokens are standardized as Privacy Pass. The contribution is the composition, an agentic trust model in which the question "what can be compelled from you?" has, for every party except the user's own device, a boring answer.

The architecture is partially implemented, and this paper has been explicit about which layers are running and which are specified. We offer the design, its claims-to-invariants discipline, and its disclosed limitations to the community as a concrete, criticizable position: consumer agents can keep frontier capability without conceding a surveillance surface, and the way to do it is to make sure nobody — including us — is in a position to betray the user.

Availability

The working paper is maintained at https://snowclaw.calidalab.ai/. The Signal-protocol library, messenger, and agent reference implementation are being prepared for artifact release alongside the v3 implementation phases.

References

[1] Anthropic. Claude Code. https://claude.com/claude-code. Accessed 2026-07-06.

[2] Anysphere. Cursor: The AI code editor. https://cursor.com. Accessed 2026-07-06.

[3] Apple Security Engineering and Architecture. Private Cloud Compute: A new frontier for AI privacy in the cloud. https://security.apple.com/blog/private-cloud-compute/, 2024. Accessed 2026-07-06.

[4] Richard Barnes, Karthikeyan Bhargavan, Benjamin Lipp, and Christopher A. Wood. Hybrid Public Key Encryption. RFC 9180, IRTF, 2022.

[5] Thomas Brewster. DHS ordered OpenAI to share user data in first known warrant for ChatGPT prompts. Forbes, https://www.forbes.com/sites/thomasbrewster/2025/10/20/openai-ordered-to-unmask-writer-of-prompts/, October 2025. Accessed 2026-07-06.

[6] Sofía Celi, Alex Davidson, Steven Valdez, and Christopher A. Wood. Privacy Pass Issuance Protocols. RFC 9578, IETF, 2024.

[7] David Chaum. Blind Signatures for Untraceable Payments. In *Advances in Cryptology: Proceedings of CRYPTO '82*, pages 199–203. Plenum Press, 1983.

[8] CNN. How ChatGPT conversations became 'a treasure trove' of evidence in criminal investigations. https://www.cnn.com/2026/05/02/us/chatgpt-ai-privacy-crime, May 2026. Accessed 2026-07-06.

[9] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy Pass: Bypassing Internet Challenges Anonymously. *Proceedings on Privacy Enhancing Technologies*, 2018(3):164–180, 2018.

[10] Alex Davidson, Jana Iyengar, and Christopher A. Wood. The Privacy Pass Architecture. RFC 9576, IETF, 2024.

[11] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-Generation Onion Router. In *13th USENIX Security Symposium*, 2004.

[12] Engadget. OpenAI no longer has to preserve all of its ChatGPT data, with some exceptions. https://www.engadget.com/ai/openai-no-longer-has-to-preserve-all-of-its-chatgpt-data-with-some-exceptions-192422093.html, October 2025. Accessed 2026-07-06.

[13] Georgi Gerganov and llama.cpp contributors. llama.cpp: LLM inference in C/C++. https://github.com/ggml-org/llama.cpp. Accessed 2026-07-06.

[14] Google DeepMind. Gemma 4 model family. https://ai.google.dev/gemma, 2026. Accessed 2026-07-06.

[15] Joshua Lund. Technology preview: Sealed sender for Signal. Signal Blog, https://signal.org/blog/sealed-sender/, 2018. Accessed 2026-07-06.

[16] Ian Martiny, Gabriel Kaptchuk, Adam Aviv, Dan Roche, and Eric Wustrow. Improving Signal's Sealed Sender. In *Network and Distributed System Security Symposium (NDSS)*, 2021.

[17] Ollama contributors. Ollama: Run large language models locally. https://ollama.com. Accessed 2026-07-06.

[18] Open Interpreter contributors. Open Interpreter: A natural language interface for computers. https://github.com/openinterpreter/open-interpreter. Accessed 2026-07-06.

[19] OpenAI. How we're responding to The New York Times' data demands in order to protect user privacy. https://openai.com/index/response-to-nyt-data-demands/, June 2025. Accessed 2026-07-06.

[20] OWASP GenAI Security Project. Agentic AI — threats and mitigations. https://genai.owasp.org, 2025. Accessed 2026-07-06.

[21] Trevor Perrin and Moxie Marlinspike. The Double Ratchet Algorithm. Signal Specifications, https://signal.org/docs/specifications/doubleratchet/, 2016. Accessed 2026-07-06.

[22] Signal Foundation. Grand jury subpoena for Signal user data, Eastern District of Virginia. https://signal.org/bigbrother/eastern-virginia-grand-jury/, 2021. Accessed 2026-07-06.

[23] Stanford Center for Internet and Society. Eight (or so) questions to ask about the ChatGPT warrant. https://cyberlaw.stanford.edu/blog/2025/10/eight-or-so-questions-to-ask-about-the-chatgpt-warrant/, October 2025. Accessed 2026-07-06.

[24] Supreme Court of the United States. Smith v. Maryland, 442 U.S. 735, 1979.

[25] Supreme Court of the United States. Carpenter v. United States, 585 U.S. 296, 2018.

[26] TechCrunch. Sam Altman warns there's no legal confidentiality when using ChatGPT as a therapist. https://techcrunch.com/2025/07/25/sam-altman-warns-theres-no-legal-confidentiality-when-using-chatgpt-as-a-therapist/, July 2025. Accessed 2026-07-06.

[27] TechSpot. Prosecutors used a man's ChatGPT logs to try to link him to the Pacific Palisades fire. https://www.techspot.com/news/112929-chatgpt-logs-used-evidence-palisades-fire-case-ended.html, 2026. Accessed 2026-07-06.

[28] Terms.Law. OpenAI v. New York Times: 20 million ChatGPT chat logs ordered produced. https://www.terms.law/2025/11/12/openai-v-new-york-times-stopped-being-just-a-copyright-case-the-moment-the-court-turned-to-your-chatgpt-logs/, November 2025. Accessed 2026-07-06.

[29] Martin Thomson and Christopher A. Wood. Oblivious HTTP. RFC 9458, IETF, 2024.

[30] United States Code. 18 U.S.C. § 2709 — counterintelligence access to telephone toll and transactional records (National Security Letters), 1986. As amended.

[31] United States Code. 18 U.S.C. §§ 2701–2712 — Stored Communications Act, 1986. As amended.

[32] United States Code. 50 U.S.C. § 1881a — procedures for targeting certain persons outside the United States (FISA § 702), 2008. As amended.

[33] United States Congress. Clarifying Lawful Overseas Use of Data (CLOUD) Act, 2018. Enacted as part of the Consolidated Appropriations Act, 2018.