Omyra

The Private Autonomous AI Protocol

Private inference · permissionless compute · provable receipts · sovereign memory — one stack on Solana

Omyra Labs · omyra.xyz · June 2026 · v1.0 (Pre-launch Draft)

Abstract. Using AI today means trusting a custodian. To run a model you surrender your prompts, your data, and any proof of what actually happened. Omyra removes that bargain: inference runs inside hardware enclaves (Intel TDX, AMD SEV-SNP) that GPU operators cannot see into, every inference emits a verifiable cryptographic receipt, and an agent's memory is encrypted on-chain under keys only the user holds. This paper is utility-first — it details what you can *build and do*, the dApps the protocol enables, the features that make them possible, and the goals that order our work. Architectural specifics herein are engineering *targets* informed by production-proven zero-knowledge privacy-layer designs on Solana; they describe the standard Omyra builds toward, not features claimed as shipped.

1 The Problem

When you call a hosted model, the operator must see your input in plaintext to run it. "We don't store your data" is policy, not proof — reversible and unverifiable. For individuals this is surveillance; for enterprises a compliance wall; for autonomous agents (software that transacts for you) it is disqualifying, because an agent that leaks its memory or instructions can be hijacked.

Omyra's thesis: privacy in AI must be *enforced by hardware and proven by cryptography*, not promised in a terms page. Three properties follow — confidential execution (the chip hides the prompt, not the operator's goodwill), verifiable computation (a receipt proves what ran), and sovereign memory (the user holds the keys).

2 Protocol at a Glance

Omyra is a protocol, not a product: no central API, no API key, no account. A wallet is identity. Four roles interact —

- **Users** pay for private inference, memory, and workflows.
- **Providers** stake $OMYRA, run TEE GPUs, earn per verified job.
- **Validators** verify proofs, settle multi-party jobs, earn fees.
- **Builders** compose agents/workflows, earn execution fees.

Trust-critical state settles on Solana; heavy compute and proving run off-chain, anchored on-chain so correctness never depends on an off-chain party staying honest.

3 The dApps Omyra Enables

Omyra is infrastructure; its value shows up in the applications built on it. These are the flagship dApps the protocol is designed to support — some built by Omyra Labs as references, all open for anyone to build.

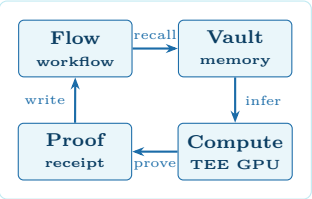

Figure 1: The four products form one pipeline: a workflow recalls memory, runs private inference, emits a chained receipt, and writes back — settling on Solana.

3.1 Sentinel — private personal AI

A personal agent that holds your financial context, correspondence, and notes, runs inference no operator can read, and remembers across sessions under your keys. Sentinel is the consumer face of Omyra: install once, own your memory forever, never leak a prompt.

3.2 Vault Console — sovereign memory manager

A dashboard to browse, search, export, and *provably delete* an agent's memory. Each entry shows its on-chain commitment; deletion produces a nullifier receipt you can verify. The GDPR answer made into a product: deletion you can prove, not deletion you're told happened.

3.3 Forge — no-code agent & workflow builder

A visual builder where anyone composes autonomous workflows (declarative YAML under the hood), wires Vault recall to Compute inference, and publishes to a marketplace. Builders earn execution fees whenever others run their workflow. Forge turns "write an agent" into "draw an agent."

3.4 Proof Explorer — verifiable-AI explorer

A public explorer for inference receipts: look up any receipt by hash and verify the model, input, and output bindings plus the TEE attestation. For audited AI in finance, legal, or healthcare, Proof Explorer is the receipt-of-record.

3.5 Compute Hub — the provider dashboard

Where GPU providers register, stake $OMYRA, watch reputation and earnings, and claim rewards. Live metrics (verify latency, jobs served, slashing record) keep the supply side transparent and competitive.

4 Use Cases in Depth

The protocol earns its keep by making concrete jobs possible that previously required trusting a custodian.

4.1 Private personal agents

A single agent retains your whole life-context and acts on it — drafting, budgeting, scheduling, triaging — while every prompt stays sealed in a TEE and every memory entry stays encrypted under your keys. The receipt trail lets you later prove what the agent did. The custodian risk simply does not exist, because there is no custodian who can read.

4.2 Confidential enterprise inference

A bank, hospital, or law firm runs regulated data through a model without exposing it to the hosting provider. The TEE supplies the confidentiality regulators demand; the receipt supplies the audit trail compliance demands; provable deletion supplies a defensible answer to retention obligations. "Where did this data go, and can you prove it's gone?" becomes answerable with math.

4.3 Autonomous machine-to-machine commerce

Agents that buy work from each other — an inference, a forecast, a computation — need privacy and replay protection. Omyra binds every action to a time-bounded validity window: a leaked or intercepted instruction is useless after its slot passes. This is what makes *unattended* agent-to-agent settlement safe rather than reckless.

4.4 Sealed cross-agent computation

Multiple agents collaborate on a computation (sandboxed WASM with encrypted I/O) while intermediate results stay sealed until settlement. Outputs are cryptographically bound to the requester's wallet, so no third party can claim another agent's result. This is the substrate for privacy-preserving data pipelines and strategies that must remain hidden until executed.

4.5 Verifiable AI for high-stakes decisions

When AI output drives a financial, legal, or medical decision, "the model said so" is not enough. The receipt proves *which* model ran on *which* input and produced *which* output, inside attested hardware — auditable after the fact without re-running it and without exposing the underlying data.

4.6 Compliant AI marketplaces

Workflow builders monetize agents through Forge while every execution is provable and every payment settles to the provider that did the work — a marketplace where buyers verify quality cryptographically and sellers are paid without a platform skimming a cut.

5 Core Features

The features below make the use cases real. Each is a design target; the disclaimer in §8 applies.

Dual-TEE confidential inference
Inference runs inside Intel TDX *or* AMD SEV-SNP. Dual support is deliberate redundancy — a flaw in one vendor's enclave does not collapse privacy, because the other holds.

Verifiable receipts (chained)
Each inference emits: model hash + input hash + output hash + TEE attestation + signature. Step N embeds step $N-1$'s proof hash $\rightarrow$ a tamper-evident audit trail for a whole workflow.

Sovereign, provably-deletable memory
Memory is encrypted on-chain under user keys. Deletion spends a one-time nullifier into an on-chain account — a one-way door. The entry then becomes cryptographically unrecoverable, and a ZK proof attests deletion without revealing what was deleted.

Replay-protected agent ops
Every action carries an expiration window in the proof's public inputs. Outside the window the proof is invalid — closing the replay vector that makes unattended agents dangerous.

Reputation-gated compute + slashing
Providers stake $OMYRA. Reputation is earned via uptime and proof correctness, never bought, and weights job allocation. Equivocation or persistent downtime burns stake (partial to full).

Private WASM compute
Lightweight jobs in isolated sandboxes with AES-256-GCM encrypted I/O; output notes bound to the requester's wallet so nobody else can claim a result.

On-chain-anchored BFT settlement
Multi-provider jobs settle through a reputation-weighted BFT cohort, then the Solana program *re-verifies* proofs against stored state — so a compromised off-chain cohort cannot override on-chain truth.

Sub-second verification
Target: constant-size proofs verifiable in $\sim$10 ms on a single CPU core, no GPU. Any laptop or phone can check an Omyra receipt; on-chain verification stays < 1 s.

6 Token Utility

$OMYRA is required for the protocol to function — not a governance ornament on a free service.

- **Staking.** Providers and validators bond $OMYRA; the bond makes honesty rational and is slashed on misbehavior.
- **Reputation weight.** Stake is necessary but not sufficient — reputation (earned, not bought) gates job allocation and consensus weight.
- **Fee settlement.** Inference, memory, and workflow fees settle in protocol terms to the providers who did the work.
- **Slashing collateral.** Equivocation or downtime burns a portion of stake, scaled to severity.

6.1 No founder cut

Earned fees accrue to a provider's on-chain rewards balance and are claimed by the provider. No admin key drains fees; no foundation tithe is skimmed. A platform takes a percentage — Omyra has no middleman percentage to take.

Parameter	Value
Symbol	$OMYRA
Settlement chain	Solana
Launch	Fair launch (Pump.fun), no presale
Emission	Deflationary: share of fees burned
Provider stake	Required to register; slashable

6.2 The flywheel

More users → more jobs → more $OMYRA burned & more provider revenue → more compute supply → lower latency & cost → more users. The loop is driven by usage, not by emissions that evaporate.

7 Technical Design Direction

Engineering targets informed by production-proven ZK privacy-layer designs on Solana. Design goals, not shipped claims.

Proof system. Groth16 over BLS12-381 — a battle-tested pairing — targeting constant-size proofs (∼200 bytes) verifiable in ∼10 ms on one CPU core.

SNARK-friendly memory. Poseidon hashing (ZK-optimized), Pedersen commitments (bind without revealing), sparse Merkle trees (efficient inclusion proofs).

Trusted setup. Multi-party ceremony with a publicly auditable transcript; one honest participant suffices for soundness. Circuits don't activate on mainnet until the ceremony is complete and verified.

Networking & ops. P2P mesh (DHT discovery, gossip propagation, request/response queries) with peers from the on-chain registry, not hardcoded bootstrap nodes. Every node exposes health/readiness/metrics endpoints. A client version handshake against the on-chain program version blocks stale clients.

8 Goals

We order our work by capability, not calendar. Each goal gates on the previous being real.

1. **Make privacy provable.** Ship Vault commitments, the Proof receipt format, and a public verification registry, with SDKs (TS/PY/RS/Go). *Goal: a receipt anyone can verify on any device.*
2. **Make compute permissionless.** Launch the TEE GPU marketplace with staking, slashing, and an on-chain pricing order book. *Goal: anyone can supply private compute and get paid trustlessly.*
3. **Make agents autonomous.** Ship Flow, the workflow marketplace, and on-chain-anchored settlement. *Goal: agents that act unattended and safely.*
4. **Make it durable.** Complete the trusted-setup ceremony, external audit, and the transition of protocol parameters to the community. *Goal: a protocol that outlives its founders.*

North star. Make private, verifiable, autonomous AI a *public utility* — as ordinary to use as calling an API, but without surrendering your data, your proof, or your memory to anyone.

9 Why No One Else Is Here

Confidential-compute platforms have TEEs but no agents, memory, or workflows. Agent economies have autonomy but run inference in the clear. GPU markets have supply but no confidentiality. Omyra's position is the intersection: hardware privacy *and* an agent economy *and* cryptographic proofs *and* sovereign memory, in one permissionless stack on Solana. The contribution is the integration, not any single primitive.

10 Conclusion

Omyra removes the custodial bargain at the heart of AI: privacy becomes a property of silicon, correctness a property of cryptography, memory the property of the user. $OMYRA is the bond that secures providers, the weight that gates reputation, and the unit in which work is paid. We state plainly what is a target and what is shipped — a protocol that exaggerates its present forfeits the trust it asks others to verify. Everything here is meant to be checked, not believed.

Links. omyra.xyz · docs.omyra.xyz · dapp.omyra.xyz · github.com/omyraxyz · t.me/omyraxyz · @omyraxyz

Pre-launch technical draft, informational only. Not financial advice; not an offer of any security. $OMYRA is a utility token

required for protocol operation; nothing here guarantees future functionality, value, or returns. Forward-looking statements are design targets and may change.