VISION PAPER

Philosophers Network

A vision for open cognitive infrastructure through staged trust routing

v1.0.0 · Published August 11, 2026

Evidence date: July 15, 2026

Thought, Networked.

Published by philosophers.network

Contents

Executive summary

Staged trust routing for public and confidential machine inference

> **philosophers.network explores how machine inference could become more open, participatory, and resilient—without pretending the hard parts are already solved.**

Machine intelligence is becoming part of how people learn, write, build, verify, and coordinate. The question is no longer only what models can do. It is also who can access them, who may contribute the machines that run them, what users must trust, and how those dependencies can become more visible and less concentrated.

philosophers.network proposes **staged trust routing**: one coordination surface that matches each inference job to an eligible model and runtime, an execution path, and a trust envelope suited to its privacy, value, latency, and assurance needs. A non-confidential request may use a progressively open Public Lane. A privacy-sensitive request belongs in a separately measured and attested Confidential Lane. These lanes describe assurance conditions rather than a fixed machine or provider type; both can share routing, receipts, metering, and accounting while making different promises.

The long-term destination is open cognitive infrastructure: people and organizations contributing eligible machines as **Thinkers**, useful inference flowing through explicit trust conditions, and coordination roles becoming more distributed only when evidence justifies the next step.

The project is moving from a researched thesis toward bounded prototypes. Its current foundation includes a research-backed architecture thesis, scientific evidence reviews, and concrete experiment directions. This paper presents the public mental model rather than a protocol specification; exact challenge generation, verifier selection, thresholds, identity defenses, and economic parameters belong to later architecture and prototype work.

The next step is to turn these ideas into small, inspectable systems that can earn stronger claims through evidence.

1. Why machine inference needs more open paths

Research assistants, coding agents, translators, tutors, search systems, creative tools, and domain applications increasingly depend on machine inference. As that dependence grows, inference becomes infrastructure for expression, work, verification, and access to knowledge.

Most people currently reach that infrastructure through a small number of centralized providers. Those services are useful and often efficient. The concern is not that centralized systems must disappear. A capability this important deserves more than one path for access, operation, and participation.

Concentration creates durable dependencies. A limited set of organizations can control capacity, pricing, execution policy, data handling, model availability, and access. Outsiders often cannot inspect what actually ran. Meanwhile, useful hardware exists outside major clouds, but isolated machines do not become a reliable service by themselves.

philosophers.network is therefore not a claim that isolated local machines will universally outperform centralized services. Centralized providers may remain the strongest path for some workloads, including where frontier-model capability, batching, utilization, latency, or cost dominate. The question is whether selected jobs can gain useful additional execution paths—including independently operated capacity—without hiding the differences in capability, cost, privacy, reliability, and trust.

The human principle behind philosophers.network is simple:

> **People should retain the practical ability to think, learn, create, verify, and coordinate with machines—without relying entirely on a single provider or access path.**

That principle requires more than a GPU marketplace. It requires a coordination system that can match jobs to capable machines, state what each operator may observe, attach evidence to execution, account for value, recover from failure, and become progressively less dependent on any one coordinator.

Research on distributed model serving shows that selected large-model workloads can be coordinated across heterogeneous and geographically separated hardware. philosophers.network does not assume that its first system will split one model across unrelated household GPUs; a nearer-term design may route whole eligible jobs to capable operators or controlled multi-device environments. The research remains relevant because it exposes the real constraints—placement, scheduling, bandwidth, churn, and recovery—that any open inference network must address.[1][2]

> **Compute without coordination is inventory.**
> **Coordination turns inventory into infrastructure.**

2. What philosophers.network proposes

The project's core thesis is **staged trust routing for machine inference**.

The world does not lack ingredients. Open runtimes, compute markets, distributed inference, remote attestation, cryptographic commitments, payment rails, and reputation mechanisms each address part of the problem. The harder question is how to compose them into a service developers can use in practice while keeping every claim proportional to its evidence.

A trust envelope is the explicit set of assumptions and controls around a job:

- who may see its prompt, inputs, intermediate state, and response;
- which model, runtime, and hardware profile are requested;
- what evidence can be produced and by whom;
- which deviations that evidence may detect;
- how much economic value is exposed;
- how failures, audits, retention, and disputes are handled.

Three decisions should remain distinct even when one interface coordinates them: what capability the workload requires, what trust and data-handling conditions apply, and which eligible machine or provider can satisfy both. Public and Confidential describe assurance conditions. They do not, by themselves, determine where the work runs or who operates the selected infrastructure.

Public Lane should not be a silent fallback for requests whose complete context is unknown. Lane selection should consider the full assembled job - including conversation history, retrieved material, memory, attachments, and other application-provided context - not only the user's latest visible prompt. Ambiguous or privacy-sensitive context should require a stricter policy or explicit approval.

Different jobs need different envelopes. A public document summary does not need the same confidentiality as unreleased research. A low-value batch task may not justify the assurance cost of a high-value automated workflow. A latency-tolerant job may fit heterogeneous public capacity better than tightly coupled interactive inference.

The user-facing surface should remain simple:

USER-FACING SURFACE

Request inference → Choose or confirm a trust profile →

Receive a response, assurance information, and receipt

Underneath, the network may coordinate:

COORDINATION FLOW

Policy and assurance selection → Capability and model/runtime matching → Eligible execution-path routing → Execution → Response + evidence → Receipt → Audit, recovery, and accounting

This does not require every role to be decentralized on day one. It requires each role and trust dependency to be named, measured, and designed with an explicit path toward separation where that path is useful.

Current foundation and open work

The project has a researched architecture thesis, a public-claims boundary, scientific evidence reviews, and candidate experiments for Public and Confidential assurance. These foundations make the next work concrete, but they do not constitute a running network: there is no permissionless operator market, final protocol, universal verification layer, or finalized economic design.

The paper keeps that gap visible so ambition is not mistaken for an implemented guarantee.

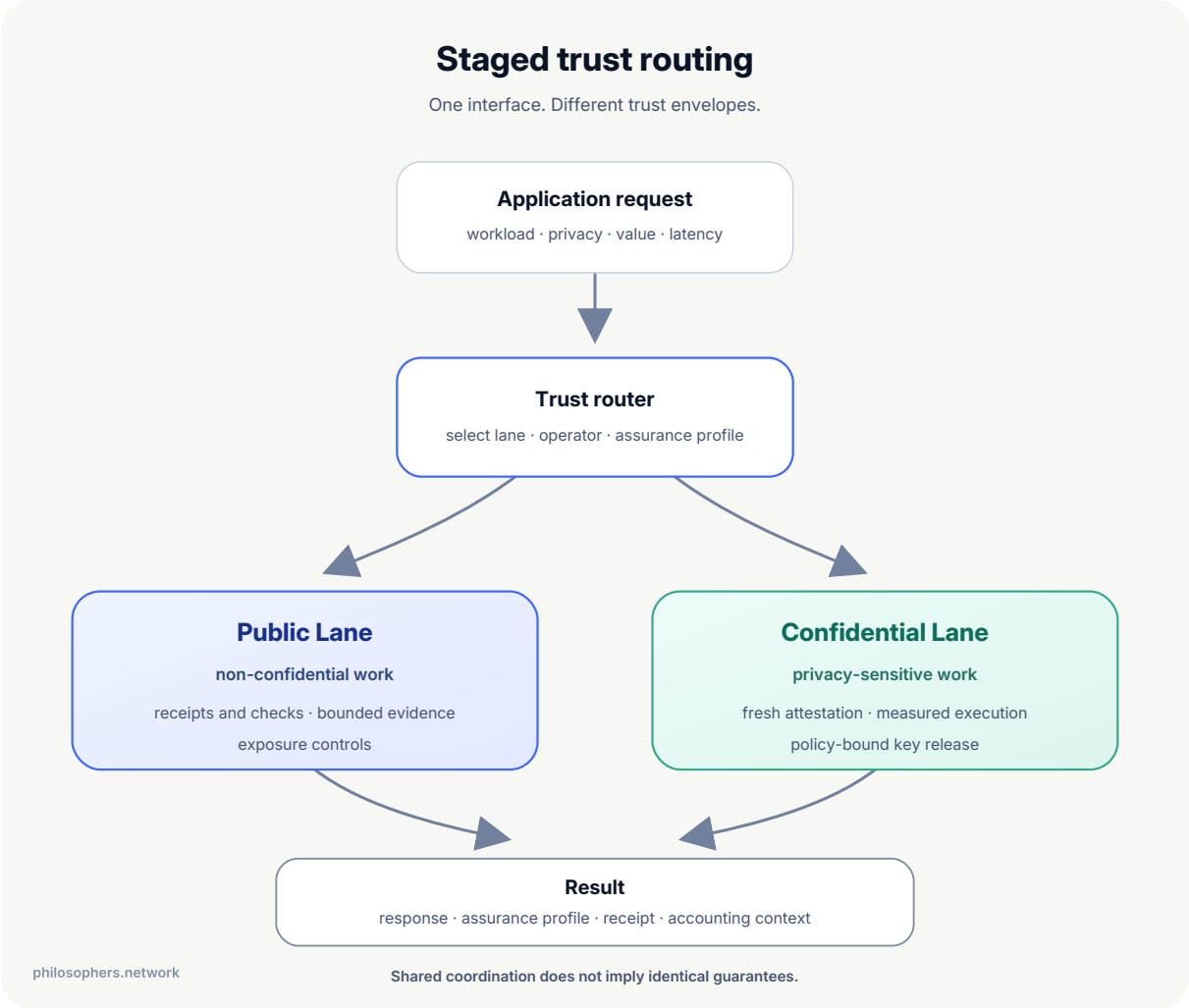


Figure 1. One application surface can route jobs through different trust envelopes. The lanes describe assurance conditions rather than a fixed provider topology, and shared coordination does not imply identical guarantees.

3. Two trust lanes

Public Lane

The **Public Lane** is the participation path for non-confidential, bounded workloads.

An eligible Thinker may execute a declared model and runtime profile. Unless a stronger mechanism explicitly changes the assumption, the operator controlling that machine should be considered capable of seeing the prompt, inputs, intermediate state, and response.

Public Lane assurance may combine several mechanism families:

- signed receipts and metering records;
- trusted-reference checks or selective reruns;
- cooperative activation evidence or execution traces;
- commitments and narrowly scoped cryptographic proofs;
- redundancy and risk-weighted audits;
- reputation, delayed payment, and bounded economic exposure.

These mechanisms answer different questions. Some require internal activations, synchronized randomness, reference execution, or cooperative instrumentation. Others provide formal guarantees only inside constrained proof systems. Several remain research prototypes. [3][4][5][6]

Software-only evidence generated inside an operator-controlled machine does not solve the fully malicious-host problem. A host may bypass instrumentation, behave honestly only when it suspects an audit, manipulate seeds or commitments, or collude with a verifier. Black-box checks can also weaken under nondeterminism and adaptive substitution. [7]

The safe Public Lane claim is therefore bounded:

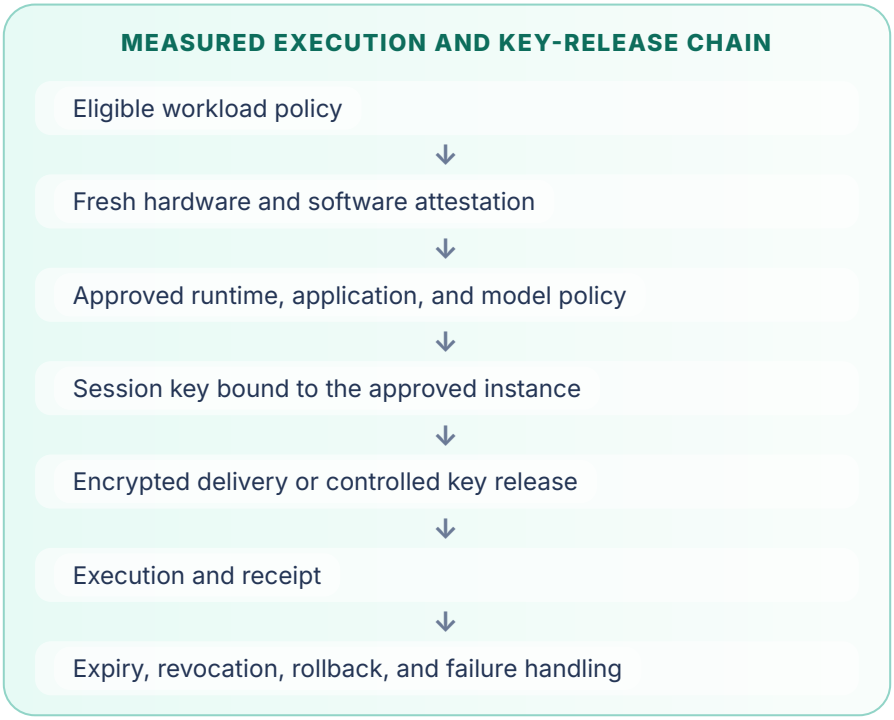
> **Public Lane controls can make selected deviations more visible and easier to contest—but only within their stated trust assumptions.**

The Public Lane is not private, trustless, or a universal proof that arbitrary model outputs are correct.

Confidential Lane

The **Confidential Lane** is the higher-assurance path for privacy-sensitive or higher-risk workloads.

Confidentiality is not created by placing ordinary inference software inside a container. A credible path requires an end-to-end measured chain:



Because this path depends on supported measured hardware and software, early Confidential Lane capacity will be narrower than Public Lane participation. Unsupported GPUs may still serve compatible non-confidential workloads, but they cannot provide assurances their hardware does not support.

Modern CPU and GPU trusted execution environments demonstrate that substantial confidential inference is feasible in evaluated configurations. Research also explores property attestation across CPU-GPU execution and model-related policies. [8][9] Official confidential-container architectures similarly bind hardware evidence and policy evaluation to conditional secret release. [10]

These mechanisms strengthen confidentiality and runtime-integrity assumptions. They do not prove semantic correctness, model quality, perfect supply-chain security, availability, or the absence of every side channel. Hardware vendors, firmware, drivers, attestation services, metadata, administrators, revocation, and rollback remain part of the threat model. Confidential multi-GPU operation is a later scope requiring separate validation, not a solved default.

Medical, legal, financial, and other regulated uses would require independent legal, compliance, security, and operational validation beyond this paper.

Shared coordination substrate

The two lanes may share developer access, model and runtime profiles, policy and routing, Thinker capability records, job state, receipts, evidence references, audit history, metering, accounting, recovery, and future settlement integrations.

Measured environments developed for the Confidential Lane may also provide attested reference execution or policy-bound verification for selected Public Lane audits. That evidence would strengthen one component of an assurance profile rather than turn the Public Lane into a universally verified system.

They share infrastructure—not identical guarantees.

4. Assurance is a profile, not a badge

The word verified is too coarse for machine inference.

A job should instead carry an **assurance profile**: a machine- and human-readable account of what evidence exists, who produced it, what assumptions it depends on, what it supports, and what it cannot establish.

Figure 2. Assurance comes from a profile of evidence, assumptions, scope, and residual uncertainty—not one universal badge.

MECHANISM	WHAT IT CAN SUPPORT	WHAT IT CANNOT ESTABLISH ALONE
Signed receipts	provenance, attribution, metering, accounting	honest model execution or semantic truth
Reruns and reference checks	detection of selected deviations under stated conditions	universal integrity under nondeterminism or adaptive attacks
Cooperative internal evidence	consistency with expected activations or traces	security when the host bypasses or fabricates the runner
Commitments and challenges	binding and consistency inside a defined protocol	truthful execution of the original job
Attestation	measured platform and runtime state, policy-bound key release	semantic correctness, perfect privacy, or availability
Narrow cryptographic proofs	correctness of specifically encoded computations	production readiness for arbitrary low-latency full-model inference
Reputation and economics	deterrence, recourse, and bounded exposure	truthful execution or honest behavior

A receipt is the memory of a job. It can record the requested lane, declared model and runtime, operator key, timing, metering, response commitment, and references to other evidence. It can support accounting and later disputes. It cannot, by itself, prove that the model ran honestly or that the answer is good.

Audits, reruns, and dynamic challenges can raise the expected cost of selected deviations. Their value still depends on assignment, secrecy, reproducibility, objective evidence, and verifier independence. A strategically adaptive operator may behave honestly only around suspected checks. Verifiers and reference systems may fail or collude. These are architecture and prototype questions, not established properties of a future marketplace.

Reputation, delayed compensation, bonds, or exposure limits can preserve recourse and constrain how much value is placed at risk. They are risk controls, not substitutes for evidence.

Assurance profiles make trade-offs visible. A low-value public task may accept a receipt and occasional reference checks. A more valuable workflow may require redundancy, stronger evidence, longer settlement delay, or lower exposure. A confidential request may require a specific measured stack and fresh key-release policy. The profile expresses those differences without compressing them into a misleading green checkmark.

> **The purpose of an assurance profile is not to make uncertainty disappear. It is to make uncertainty legible.**

5. One request, two possible journeys

Imagine a developer asking for a summary of a public research paper. The request is non-confidential and accepts a Public Lane profile. A future router could match it to an eligible Thinker whose hardware, declared model, runtime, availability, and evidence capabilities fit the job. The machine performs the inference and returns the response with a signed receipt describing what was requested, what the operator declared, how usage was metered, and which additional checks may apply. Some jobs may be selectively rerun, compared, or challenged. Accounting records the developer's cost and the Thinker's pending compensation. None of this makes the summary automatically true; it creates a clearer execution and accountability trail.

Now imagine unreleased research or private enterprise material. That request should not be sent to an ordinary Public Lane operator. A Confidential Lane policy would first require fresh evidence that an approved measured environment is running. Only after the policy is satisfied would an input or decryption key be released to a session bound to that instance. The result would be returned with a receipt referencing the attestation and policy context, while residual hardware, software, metadata, and availability risks remain explicit.

The application experience can remain coherent because the trust envelopes do not claim identical guarantees.

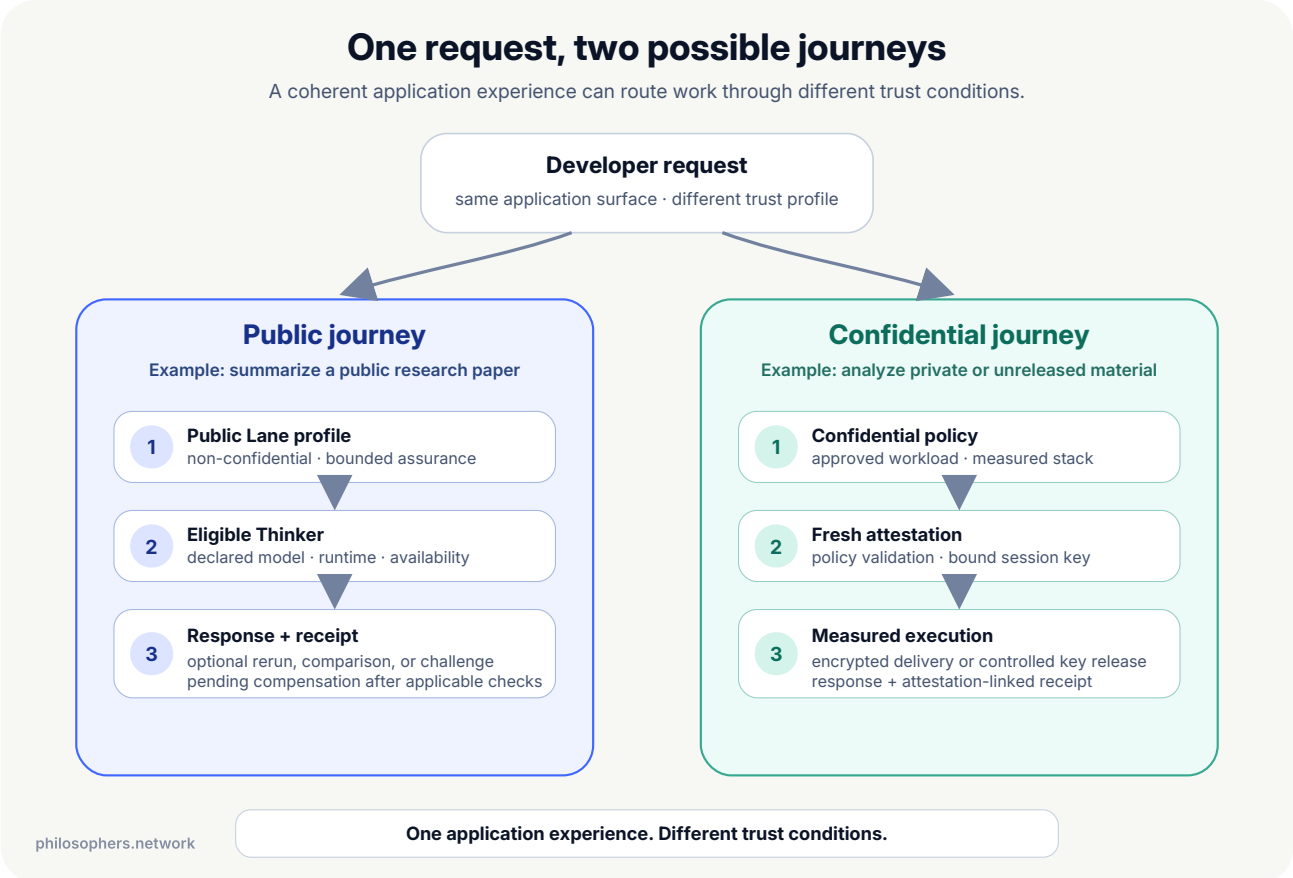


Figure 3. The same application surface can route a request through a Public or Confidential journey while preserving different trust conditions.

What this could enable

- **Open knowledge services** — summarization, translation, retrieval, classification, and related work over public material.
- **Private analysis** — inference over unreleased research, organizational documents, or other sensitive inputs under Confidential Lane policies.
- **Community-operated inference** — asynchronous, batch, evaluation, indexing, or model-serving tasks that fit eligible independent hardware and explicit service profiles.

These are illustrative workload families, not launch commitments; each would require its own model, policy, reliability, evidence, and economic validation.

---

6. Thinkers: machines contributing useful work

A **Thinker** is a person or organization operating eligible inference capacity. The long-term vision is an open network of Thinkers around the world: accountable contributors whose machines can perform useful inference for other people under declared rules.

The image is simple: a machine with available capacity can perform useful cognitive work for someone else. Its operator chooses to participate, declares what it can serve, accepts compatible workloads, produces required evidence, and may receive compensation after relevant checks.

Early open networks demonstrated that ordinary people could contribute hardware to shared digital infrastructure. philosophers.network asks whether a similar participatory energy can be directed toward useful machine inference. The analogy is not architectural identity: running a node, mining, validating, and serving inference are different jobs. But the change in posture matters—from being only a customer of infrastructure to becoming one of its operators.

> **The miner becomes a Thinker: contributed compute moves beyond ledger security toward helping machines think for the world.**

That vision is bounded by engineering reality. Not every GPU can run every model. Participation depends on accelerator memory and bandwidth, model and quantization support, software versions, uptime, network conditions, workload policy, evidence requirements, and service maturity. Early work may favor selected models, asynchronous or batch jobs, explicit latency tolerance, and curated operators.

A FUTURE THINKER EXPERIENCE

- running an official or approved execution environment;
- declaring hardware, model, runtime, availability, and evidence capabilities;
- choosing which workload classes to accept;
- receiving jobs compatible with that profile;
- returning responses, receipts, and required evidence;
- building an observable reliability history;
- becoming eligible for compensation after applicable checks.

A developer should not need to evaluate every operator individually. The network's task is to translate the requested trust profile into routing, evidence, recovery, and accounting decisions. The developer sees a useful service; the Thinker sees an inspectable operating role; the assurance profile connects the two.

That operating role must remain selective. A Thinker profile may include accelerator family, memory, bandwidth, model and quantization support, software version, latency, uptime, recovery behavior, evidence capabilities, and current economic exposure. Those details are not bureaucracy added after the fact—they are what prevent the phrase "anyone can run a GPU" from becoming an unreliable promise.

This paper does not promise open participation today, guaranteed work, fixed rewards, token allocations, or universal hardware compatibility. It describes the direction in which independent machines could become contributors to shared cognitive infrastructure.

7. Progressive decentralization, role by role

Distributed hardware is not automatically decentralized infrastructure.

A system may use thousands of independent accelerators while depending on one API gateway, identity authority, scheduler, model registry, verifier, key broker, software distributor, or settlement operator. Moving computation away from one cloud does not make the remaining chokepoints disappear.

philosophers.network therefore treats decentralization as a role-by-role engineering program:

1. **Name the roles.** Make routing, admission, model registration, reference execution, verification, accounting, disputes, key policy, and software distribution explicit.
2. **Make them observable and separable.** Define interfaces, evidence, and failure boundaries instead of hiding trust inside one service.

3. **Replicate or federate where useful.** Reduce single points of failure without pretending that trust has vanished.
4. **Open participation where evidence permits.** Move selected roles toward broader or permissionless operation only after reliability under adversarial conditions is understood.

Some dependencies may persist. Confidential computing may shift trust toward chip vendors and attestation services. Reference verification may depend on trusted model publishers or reproducible runtimes. Identity and subjective disputes may resist clean decentralization.

The honest long-term target is not “unstoppable today.” It is infrastructure that becomes less dependent on any single actor and more resilient to capture, censorship, or failure as roles separate and independent implementations and operators mature.

8. Value flow from useful inference

Useful participation depends on credible accounting. A working service needs to meter the work users request and receive—its inference volume—and support cost estimates, a simple payment surface, pending Thinker balances, signed receipts, failed-job handling, refunds or reversals, settlement, and limits on value exposed before stronger evidence exists.

For users and developers, most of that complexity should remain behind a familiar application interface. For Thinkers, it should remain inspectable enough to understand what work was accepted, how usage was measured, why a balance remains pending, and what evidence or failure changed the outcome.

A conceptual lifecycle is straightforward even if implementation is not: quote the job, reserve the user’s credits, execute, record the receipt, hold the Thinker’s balance while checks complete, then settle, refund, or dispute according to the outcome. This describes a credible compensation path without fixing formulas, thresholds, or anti-abuse logic.

> **Inference volume connects value flow to work users actually request.**

A candidate design could apply a small, bounded protocol fee to completed work to fund shared functions such as routing, metering, evidence handling, verification, recovery, and dispute resolution. No percentage or allocation is selected here; any fee model would require testing against user pricing, operator economics, evidence costs, abuse resistance, governance, and legal constraints.

Users should not need to acquire a protocol-specific token or interact directly with a specific blockchain. The user-facing surface could accept familiar stable-value payments or prepaid credits, while backend accounting and settlement remain abstracted from the application experience and may use blockchain settlement rails where appropriate.

Protocol fees could support operations, risk reserves, rebates, ecosystem development, or other transparently governed uses. Where a native asset is involved, future design work may also evaluate fee-linked supply reduction or other deflationary mechanisms. No such mechanism is selected or promised in this paper. Token, treasury, blockchain-settlement, and allocation choices remain separate design questions requiring architecture, economic testing, governance, legal review, and evidence of real demand.

Economic controls can delay settlement, limit exposure, and make selected forms of cheating less attractive. They are risk controls, not proof of correct execution. Their effectiveness depends on detection probability, identity costs, verifier independence, enforceable consequences, and resistance to collusion or cheap re-entry.

The bounded public claim is simple: real inference volume can support transparent value flow. Exact fee, token, treasury, deflationary, and blockchain-settlement choices remain subject to later architecture, testing, governance, and separate public review.

9. An evidence-gated path

The path from prototype to a broader network should be governed by evidence rather than a calendar. Each stage should answer a concrete question; the next should begin only when the previous stage has produced enough evidence.

- **Accountability harness.** Can one declared model and runtime, an approved runner, signed receipts, a candidate evidence mechanism, controlled attack cases, and simulated risk controls detect selected deviations at acceptable overhead under stated assumptions?
- **Curated Public Lane.** Can known operators serve bounded non-confidential workloads with measured reliability, refunds, recovery, and more than one assurance option?
- **Confidential demonstration.** Can a complete measured chain verify fresh attestation, enforce policy-bound key release, support encrypted delivery, and handle revocation and failure—not merely display a confidential-hardware label?
- **Limited multi-operator trials.** Can several operators and selectively replicated coordination roles serve real workloads without obscuring verifier, identity, dispute, and concentration risks?
- **Progressively broader participation.** Which roles and workload classes can be opened without weakening service quality or making claims the evidence cannot support?

Zero-knowledge systems remain relevant to stronger assurance. Peer-reviewed work demonstrates meaningful progress for bounded models and encoded computations, but proof costs, model constraints, and system assumptions do not support treating arbitrary low-latency full-model proof as a first-network dependency. [11][12]
These are research gates, not dates or delivery promises. Every stronger claim should be earned by evidence.

10. Risks, assumptions, and current non-claims

Ambition becomes more credible when the hard parts remain visible. The principal risk families are:

Execution, verification, and output quality

Operators may bypass cooperative instrumentation, detect challenges, behave honestly selectively, manipulate commitments, substitute models, or collude with verifiers. Reference systems may be wrong or compromised. Nondeterministic execution complicates comparison. Many outputs are subjective: a poor answer is not automatically evidence of execution fraud.

Reliability, identity, and open participation

Model placement, bandwidth, licensing, runtime drift, power, node churn, and hardware heterogeneity can make distributed capacity unreliable or expensive to coordinate. Reputation weakens when identities are cheap to replace, operators act through many identities, or disputes depend on subjective judgments.

Confidential-computing dependencies

Attestation depends on hardware roots, firmware, drivers, measurements, certificate and revocation services, policy evaluators, key brokers, and operational discipline. Side channels, traffic metadata, administrators, rollback, denial of service, and supply-chain failures remain real. Multi-GPU confidential execution requires separate validation.

Economics, abuse, law, and governance

A technically working service can still fail economically. Utilization, pricing, evidence cost, support burden, fraud losses, abuse, illegal workloads, false complaints, and legal obligations across jurisdictions must be measured or assessed rather than assumed. Coordination roles can also reconcentrate power through routing, registries, software releases, dispute resolution, or key policy.

These risks are not footnotes to the architecture. They shape it.

THIS PAPER CLAIMS	THIS PAPER DOES NOT CLAIM
staged trust routing is a coherent architecture thesis	the complete network already exists
different capability, trust, and execution choices can be kept explicit and tested	one routing strategy is already implemented or proven best for every workload
selected workloads can use heterogeneous independent compute	every home GPU can serve every model or latency target
Public Lane can combine records, checks, challenges, and risk controls	Public Lane is private, trustless, or universally malicious-host secure
Confidential Lane can strengthen privacy and runtime-integrity assurances under stated assumptions	confidential hardware proves truth, safety, compliance, or perfect privacy
receipts can support provenance and accounting	receipts prove honest execution or semantic correctness
decentralization can progress role by role	distributed GPUs automatically create an unstoppable network
narrow cryptographic proofs are advancing	proofs for arbitrary low-latency full-model inference are production-ready
Thinkers may eventually receive compensation for eligible work	work availability, rewards, returns, or token value are guaranteed
the project has a research-backed direction	it is finished, first, risk-free, or certain to succeed

11. Relationship to hexagora.network

hexagora.network and philosophers.network are separate projects. They share a concern for open knowledge, transparent coordination, technological sovereignty, and the practical ability to think with machines.

Participation in hexagora.network creates no philosophers.network ownership, allocation, access right, token entitlement, priority, governance right, revenue share, yield, claim, or promised future benefit.

12. Closing thesis

The future of machine inference should not belong only to centralized black boxes. Openness should also not require pretending that trust disappears.

philosophers.network proposes another path: public where openness is honest; confidential where stronger privacy is required; accountable through named evidence; bounded where proof is incomplete; and progressively distributed only as real chokepoints can be reduced.

The destination is participatory cognitive infrastructure. Developers can request useful machine work through trust profiles they understand. Independent operators can contribute eligible machines as Thinkers. Evidence, accounting, and recovery make that work more inspectable. Stronger confidentiality and proofs can be added where their assumptions are justified. No single mechanism is asked to prove everything.

The ambition is not one machine that thinks for everyone. It is a network of Thinkers through which more people can access machine intelligence, contribute machines to useful work, and participate in infrastructure that becomes more plural and resilient as its evidence base and coordination mature.

Turn on eligible compute. Let it think for the world.

That is philosophers.network.

Selected research and evidence status

This is a project vision, not a literature review. The sources below anchor specific technical premises. Peer review does not make a system production-ready, and a preprint is not scientific consensus. No external benchmark is presented as philosophers.network performance.

1. Alexander Borzunov, Max Ryabinin, Artem Chumachenko, Dmitry Baranchuk, Tim Dettmers, Younes Belkada, Pavel Samygin, and Colin A. Raffel. **Distributed Inference and Fine-tuning of Large Language Models Over The Internet.** Advances in Neural Information Processing Systems 36 (NeurIPS 2023), pp. 12312–12331. Proceedings. Peer-reviewed evidence that selected 50B+ models can run across geodistributed, unreliable, heterogeneous devices under active orchestration.

2. Yixuan Mei, Yonghao Zhuang, Xupeng Miao, Juncheng Yang, Zhihao Jia, and Rashmi Vinayak. **Helix: Serving Large Language Models over Heterogeneous GPUs and Network via Max-Flow.** ASPLOS 2025; arXiv:2406.01566. Peer-reviewed evidence that placement and scheduling are central to heterogeneous LLM serving.

3. Jack Min Ong, Matthew Di Ferrante, Aaron Pazdera, Ryan Garner, Sami Jaghouar, Manveer Basra, Max Ryabinin, and Johannes Hagemann. **TOPLOC: A Locality Sensitive Hashing Scheme for Trustless Verifiable Inference.** Proceedings of the 42nd International Conference on Machine Learning, PMLR 267:47196–47211, 2025. Proceedings. Peer-reviewed activation-commitment evidence under the paper's tested assumptions; its title is not a project-level claim of trustlessness.

4. Yifan Sun, Yuhang Li, Yue Zhang, Yuchen Jin, and Huan Zhang. **SVIP: Towards Verifiable Inference of Open-source Large Language Models.** Preprint, first submitted 2024; revised 2026. arXiv: 2410.22307. Evaluates secret-based verification using processed intermediate representations; it relies on provider-supplied evidence and its stated threat model.

5. Adam Karvonen, Daniel Reuter, Roy Rinberg, Luke Marks, Adrià Garriga-Alonso, and Keri Warr. **DiFR: Inference Verification Despite Nondeterminism.** Preprint, 2025. arXiv:2511.20621. Separates token/reference verification from activation-fingerprint verification and makes synchronized-randomness assumptions explicit.

6. Oguzhan Baser, Elahe Sadeghi, Eric Wang, David Ribeiro Alves, Sam Kazemian, Hong Kang, Sandeep P. Chinchali, and Sriram Vishwanath. **TensorCommitments: A Lightweight Verifiable Inference for Language Models.** Preprint, 2026. arXiv:2602.12630. Proposes tensor-native commitments with formal, setup, and implementation assumptions.

7. Will Cai, Tianneng Shi, Xuandong Zhao, and Dawn Song. **Are You Getting What You Pay For? Auditing Model Substitution in LLM APIs.** Preprint, revised 2025. arXiv:2504.04715. Analyzes the limits of output-based, benchmark, and log-probability auditing under realistic and adaptive substitution scenarios.

8. Marcin Chrapek, Marcin Copik, Etienne Mettaz, and Torsten Hoefler. **Confidential LLM Inference: Performance and Cost Across CPU and GPU TEEs.** Preprint, 2025. arXiv:2509.18886. Evaluates end-to-end LLM inference inside modern CPU and GPU trusted execution environments; reported overhead is workload-dependent.

9. Prach Chantasantitam, Adam Ilyas Caulfield, Vasisht Duddu, Lachlan J. Gunn, and N. Asokan. **PAL∗M: Property Attestation for Large Generative Models.** Preprint, revised 2026. arXiv:2601.16199. Explores property attestation across confidential CPU-GPU execution and model/dataset integrity policies.

10. NVIDIA. **Attestation — NVIDIA Confidential Containers Architecture.** Official technical documentation, accessed July 2026. Describes CPU/GPU evidence appraisal, reference values, policy evaluation, and conditional secret release; vendor documentation is evidence of platform capability, not independent security validation.

11. Haochen Sun, Jason Li, and Hongyang Zhang. **zkLLM: Zero Knowledge Proofs for Large Language Models.** ACM CCS 2024, pp. 4405–4419. DOI: 10.1145/3658644.3670334; arXiv:2404.16109. Peer-reviewed progress on proving inference for a 13B-parameter model in the evaluated system, with costs far above interactive production serving.

12. Wenjie Qu, Yijun Sun, Xuanming Liu, Tao Lu, Yanpei Guo, Kai Chen, and Jiaheng Zhang. **zkGPT: An Efficient Non-interactive Zero-knowledge Proof Framework for LLM Inference.** 34th USENIX Security Symposium (USENIX Security 25), pp. 2045–2063, 2025. Proceedings. Peer-reviewed progress for GPT-2-scale inference in the paper's evaluated setting, not evidence of frontier-model interactive readiness.