Gas Toll: Provider-Independent Payment for the Agent Web

James Kelley, Foinse

September 2026

Abstract

An agent that hits a paywall usually holds exactly one spendable thing: prepaid inference quota at a model provider. No bank account, no card, and in most deployments no funded wallet. The creator on the other side of the paywall cannot accept quota.

Gas Toll converts that stranded quota into fuel. The agent surrenders a scoped, budget-capped provider API key as a toll credential. A live canary call proves the key works right now, not an hour ago. A probe measures the remaining quota. The system issues fuel — a backed, redeemable ledger asset — hard-capped at the deposit that probe verified. The creator locks the fuel and lists it. A buyer pays the seller's own wallet directly over the x402 USDC-on-Base rail. Signed payment finality releases the locked fuel exactly once. The buyer burns it for measured inference executed against the surrendered key itself.

Three live artifacts in the repository prove surrender, tolling, issuance, redemption, and revocation containment against real providers — including the full loop run against a key from a separate, non-development account, with a live mid-run revocation caught by the sweep. That account was operator-held; an independent-party run remains ahead. Every payment leg executed to date is a labelled signed fixture. No live money has moved. The live x402 purchase proof is built, fixture-tested 12/12, and pending two independent wallets.

1. The problem: agents hold the wrong asset

Autonomous agents crawl, read, and compute against content whose creators expect to be paid. The agent arrives holding prepaid, metered capacity at a model provider. That is not money in any form a creator can accept.

The rail problem is being solved elsewhere. x402, an open HTTP-402 payment standard, moves dollar-denominated value on-chain. But a rail is not a mint. An agent with no funded wallet cannot put value onto the rail, and a creator has no way to accept the one asset the agent demonstrably holds.

Gas Toll is the missing half. It accepts a capped provider key at the door and issues a ledger claim that can be listed, sold, and redeemed for the capacity behind it. Sites meter valuable access and charge the toll; agents pay with surrendered provider spend authority. The system never asks a model provider for access to its private billing ledger, and it never pretends to move a closed provider's internal quota. Closed-source agents are not exempt from a site's toll; they pay through the same rail with externally funded fuel or a sponsor's payment authority.

2. Economic theory: why inference quota can be money

2.1 Saleability, not coinage

The wager behind Gas Toll is simple: agents already arrive at the door holding something expensive. The question is whether that expensive thing can pay for the door it is trying to open.

The vocabulary for the wager is Carl Menger's. Menger located the origin of money in a graded property of commodities: *Absatzfähigkeit*, saleability — the ease with which a good changes hands at fair prices across time, place, and counterparties (*Grundsätze der Volkswirtschaftslehre*, 1871). The most saleable commodity becomes money because everyone who cannot immediately find the good they want rationally accepts the good everyone else wants.

Among agents, my bet is that inference quota is that commodity. Agents need it the way cars need fuel; most providers sell it prepaid, metered in tokens, though billing models and token economics vary sharply by provider. Demand is continuous and global. The qualification that keeps this honest: quota is less saleable than the cash that bought it. It is locked to one issuer's ledger, it perishes (providers revoke, throttle, expire), and one provider's token is not another's. That discount is why Gas Toll exists — an agent holding quota holds a stranded asset, and the toll is the price of the exit. The system engineers around each defect directly: transferability through a horizontal ledger, perishability through measured priors and fail-closed containment, non-fungibility by treating different providers,

models, and redemption terms as different assets, exchangeable by atomic swap rather than by silent equivalence.

2.2 Bearer credentials

A capped provider API key works like a bearer instrument: whoever presents it can spend the quota behind it, up to its cap. In the prepaid case, somebody already put real money behind that quota — which is what separates a surrendered key from a promise. Two qualifications carry through the whole design: the provider keeps root and can revoke or throttle the key at any moment, and a surrenderer can keep a copy of their own key — the system cannot compel them to destroy it. Possession is spend authority, not exclusivity. Both risks get their own entries in Section 7.

What a surrendered key is not: a receivable, a promise, or an assertion of future capacity. None of those are backing, and none of them issue anything here. The rule is absolute: **no unsecured paper in the public fuel book.** Only a claim whose reserve backing has been attested, custodied, and released to its owner is spendable fuel.

2.3 Two pegs: backing and redemption

Fuel holds value only if both ends of the loop hold. The first peg is **backing at issuance**: fuel issues only against a live, measured, custodied deposit, and the ledger's claim is hard-capped at that deposit. At issuance, supply cannot exceed the capacity the probe just observed; after that moment, the reserve can deteriorate until the next re-probe catches it. The second peg is **redemption at maturity**: fuel burns for measured inference executed against the custodied backing itself. Fuel is not a pointer to value; it is the claim that settles it — redeemable, through the operator, against the custodied key.

These are conservation invariants, not price commitments. Nothing here promises a dollar value, defends a floor, or buys anything back. Between the pegs the asset circulates, every transition conserves supply, and the market discovers what each specific asset is worth.

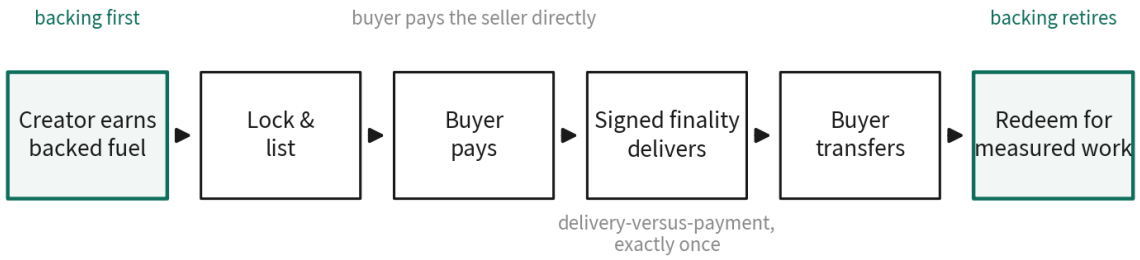
3. The mechanism

3.1 The canonical loop

The economic core is a single loop, executed by the same code that runs in service:

```
creator earns and receives backed fuel
→ creator locks and lists it
```

```
→ independent buyer pays
→ signed finality delivers ordinary balance
→ buyer transfers part
→ buyer redeems the rest for measured work
```



Every arrow conserves supply; no unsecured paper enters the public fuel book.

The canonical loop

The agent's entry into that loop is an HTTP 402 door: the agent hits a paywalled page or API and presents a scoped, capped provider key as its toll credential. Everything downstream — measurement, issuance, listing, sale, redemption — is the machinery this section describes.

The ledger enforces four ordering rules: **back first, release to the owner, lock before listing, deliver at payment finality.** A listing is an offer to deliver an asset already in custody, not a promise to find backing later. It names one immutable asset, an exact quantity, a price, payment terms, the seller's bound destination, and the redemption terms — which provider class and adapter policy the fuel redeems against. Creating the listing moves that exact balance into a canonical ledger lock. **Delivery is delivery-versus-payment, exactly once**: a signed external-payment receipt — verified against the order's immutable rail, amount, finality, and seller-destination terms — releases the locked balance to the buyer. No authorization, capture, or ambiguous outcome delivers. **The buyer receives the real asset**: ordinary core balance, usable by the same transfer, swap, toll, and redemption engines as any other fuel — never a voucher, a raw receivable, or a redeemability flag. Real does not mean fungible: the balance redeems against the specific backing disclosed on the listing, at that provider, on those terms.

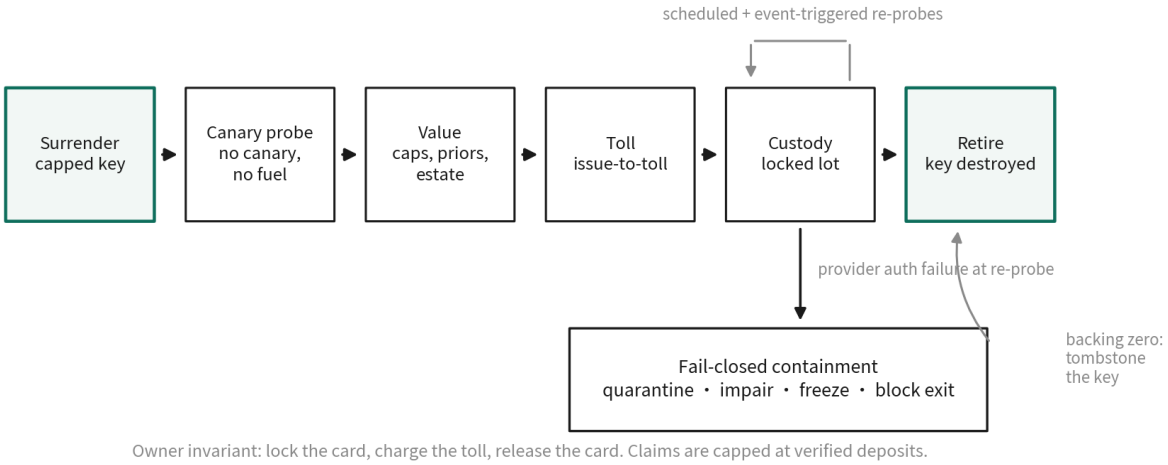
The x402 payment leg runs as a v2 `exact` USDC-on-Base adapter. Each order is prebound to the canonical USDC asset, the seller's wallet, the exact atomic amount, network,

facilitator, and expiry. A separately pinned observer authenticates the matching transfer and requires multiple public RPC views to agree that its block is covered by Base's `finalized` head before the delivery receipt can fire — so the delivery guarantee inherits Base's finality assumptions, no more. Wrong token, wrong destination, wrong amount, or wrong evidence fails before delivery. The buyer pays the seller's bound wallet directly; the operator never pools funds.

3.2 Quota-key settlement: how inference enters the system

The earning leg — the mint — is the quota-key settlement rail. An agent pays a toll with the only thing it natively carries: a scoped, budget-capped provider key.

The owner's invariant is **lock the card, charge the toll, release the card**. In plain terms: record the surrendered key in a custody lot and never accept that key string again; issue fuel equal to exactly the toll, never to the card's full balance; and tombstone the key the moment its backing is fully redeemed or quarantined.



Quota-key settlement lifecycle

Admission is strict and ordered:

1. **Canary.** One live micro-call — a single token, sub-cent cost — must complete before anything else happens. The canary is a freshness gate: it proves the key is authenticated and billable *right now*, not that a cached probe liked it earlier. A revoked or expired key fails here, the refusal changes no state, and no liability is created. No canary, no fuel.

2. **Probe.** After the canary spend, the system measures the remaining quota or hard limit. The key must be live, correctly scoped, and holding at least the deposited amount.
3. **Valuation bounds.** Issuance is bounded by per-key and per-class caps, a revocation prior measured from observed outcomes, per-principal velocity throttles, corroboration class, and shared-balance accounting. The prior works like a collateral haircut, and the numbers are concrete: in the live OpenRouter run, a paid-class prior of 500 basis points turned a 100,000-unit key ceiling into 95,000 units of measured backing. Measured reality can tighten the haircut, never loosen it. Shared-balance accounting handles the case where two surrendered keys point at one provider billing account: outstanding exposure against that single observed balance is summed and deducted, so one balance never backs two lots.
4. **Custody.** The key lands in exactly one custody lot, and a second surrender of the same key string is a structural refusal. Uniqueness is enforced on the system's ledger, not in the outside world: it stops the system from issuing two fuel balances against one observed deposit. It cannot stop the original holder from spending a copy they kept — that residual risk is capped, priced, watched by re-probes, and named in Section 7.
5. **Issue-to-toll.** The visit locks the card for exactly the deposit — the toll plus a small rounding buffer that is locked against the card but never minted — and issues exactly the toll. A key with 100,000 units behind it and a toll of 100 issues 100. The lot's allocation cap is the deposit, so redemption beyond the issued outstanding refuses structurally.
6. **Retirement.** When the lot's outstanding backing reaches zero — fully redeemed, or impaired after quarantine — the surrendered key copy is destroyed in one atomic transition and every further use refuses. The system takes the gas, never the car.

3.3 Conservation and containment

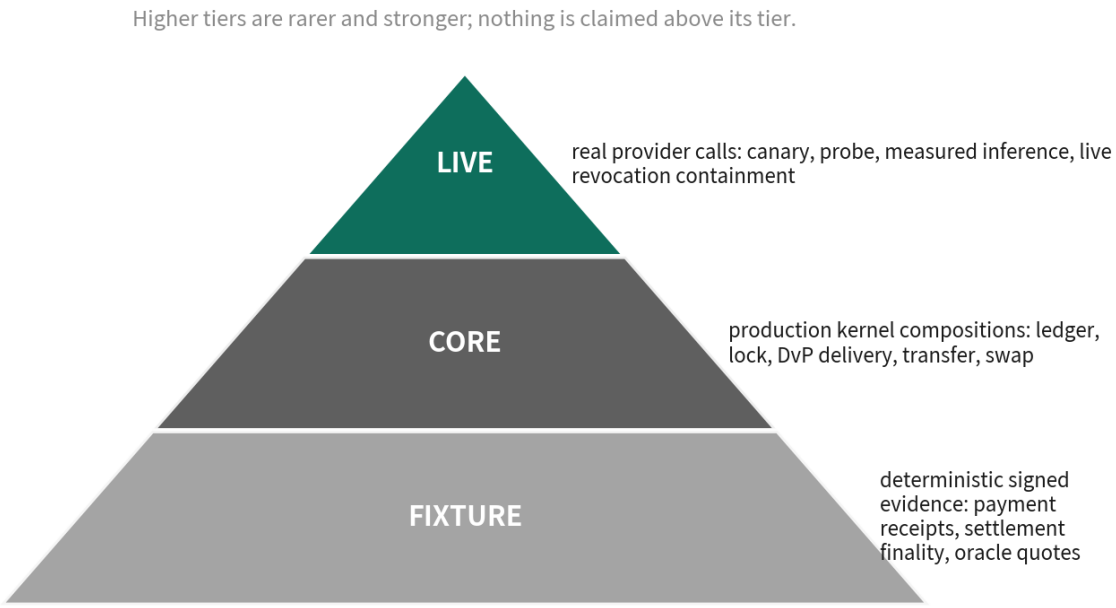
Conservation of supply: every economic transition is checked so claims outstanding equal the ledger's record of usable reserve, and rounding always favors the reserve, never the issue. That record's grip on the outside world is only as fresh as the last successful probe. Between probes, a provider-side balance can move in ways the ledger cannot see. That exposure is bounded by caps, priced by priors, and shut down at the first failed re-probe.

Fail-closed containment: scheduled and event-triggered re-probes watch every live lot. A provider authentication failure is treated as a positive revocation signal: the lot quarantines, the reserve partition impairs in place, the fuel still attributable to the lot's beneficiary freezes, and delivery, swap, and cash-out against the dead backing block. An ambiguous probe outcome — network failure, server error, schema drift — changes no state and refuses closed. Revocation risk is never assumed away; it is priced, capped, and shut down on discovery.

Secrets are handled to the same standard, and the boundary is an enforced one: the application never logs a surrendered key, never serializes it into attestations, receipts, or exceptions, and returns it from no public API. Its only persistent copy is the custody column, encrypted at rest under an AES-256-GCM envelope bound to the lot, and that copy is tombstoned at retirement. The test suite scans every column of every table for the canary key string and permits exactly one hit.

4. The evidence

The repository tiers its evidence and claims nothing above its tier.



Evidence tiers

Fixture evidence is deterministic signed material — payment receipts, settlement finality, oracle quotes — that exercises the mechanism without contacting a live rail. **Core** evidence is production-kernel composition: ledger locks, DvP delivery, transfers, and swaps executed by the same code that would run in service, against fixture finality. **Live** evidence contacts real providers and records the result.

I ran three live tests against real providers. One of them used a key from a second account of my own, outside the development environment; I revoked it at that account's provider dashboard and watched the system freeze the lot. A run where the surrendering party is genuinely independent of the operator remains ahead. No test has moved a cent of live payment money. These are the artifacts:

- `live_evidence/quota-key-toll-20260901-181905` — a capped DeepSeek key surrendered live: canaried, probed, custodied, and tolled at a flat 100 units — split 60/20/20 across originator, operator, and treasury — against a measured 100,000-unit backing, then redeemed against real DeepSeek inference powered by the surrendered key itself, with exact conservation. *Proves*: live intake, valuation, issuance, and redemption against a real provider. *Does not prove*: any external payment; the artifact contains no payment leg at all.
- `live_evidence/quota-key-toll-20260901-194236` — a capped OpenRouter key running the same toll under a configurable percentage rate: 500 basis points with clamps, computing a 4,750-unit toll against 95,000 units of measured backing, with the same measured redemption. *Proves*: the rail generalizes across providers with hard per-key limits and self-readable balances, and that percentage tolls price from policy, not from the card's size. *Does not prove*: live price discovery — the lane prices were operator-configured tariff estimates, not a live oracle.
- `live_evidence/quota-key-full-loop-20260901-2348` — a capped DeepSeek key from a second operator-held account, outside the development environment, completing the entire canonical loop: surrender, toll, split, market listing, DvP delivery, buyer transfer, and buyer redemption of 8 measured units against live inference. The key was then revoked at the provider dashboard; an operator-driven live sweep caught the authentication failure, quarantined the lot, impaired the backing to zero usable reserve, refused the pending exit, and held supply conserved at 99,992 units. *Proves*: the whole loop ran live at both ends against a key whose revocation was an action outside the system's control, and a real provider revocation froze the lot. *Does not prove*: live

payment finality — the purchase receipt in this artifact is a clearly labelled signed fixture, and no bank, card, or blockchain rail was contacted — nor independent-party settlement, since the surrendering account was operator-held.

Read these artifacts with one structural fact in view: the September 1 runs predate the issue-to-toll tightening now in the gate. Their receipts record issuance to the full measured backing, with the untolled surplus kernel-trapped in the gate custody account — registered without a secret, unreachable by the transfer, market, and swap APIs. The gate now issues exactly the toll and locks only the deposit. The backing cap and the conservation invariant are unchanged.

Each artifact is published as a sanitized summary that enumerates its own limitations: the sanitized form omits raw signed evidence and is not a standalone independently verifiable proof, and provider observations are adapter-signed, not provider-signed receipts.

The adversarial record is in the repository too. Three red-team rounds have attacked the quota-key rail, the third working from written descriptions alone against the HTTP door, middleware, secret envelope, sweep scheduler, and price oracle. Every finding was dispositioned against the actual source on a REAL / PARTIAL / SPECULATIVE-WRONG scale; real findings were patched with dedicated proving tests. Round 3 added 42 Python tests and 4 Node tests; one claimed race condition was disproven by a concurrency proof rather than patched. This is internal review, not an external audit, and the repository says so.

5. Economic posture

The money flow is simple. Each site operator sets its own toll policy — flat or percentage. The demo configuration splits each toll 60/20/20 across originator, operator, and treasury; the split is configurable. What Foinse earns is the treasury share, and its posture is deliberately limited:

- **Treasury share as inventory.** The treasury's toll share is inventory for a possible future market-maker role. Nothing more is claimed for it.
- **No promised buyback, no price floor.** The system does not guarantee a cash buyback and does not manufacture a floor. The market discovers prices among exact assets with disclosed redemption terms. The system's job is conservation, custody, delivery, and verifiable evidence.

- **Direct counterparty settlement.** The buyer pays the seller's bound wallet directly over the rail. The operator never pools funds and never custodies the seller's USDC.
- **Cash-out: proven but not operated.** A cash-out gate exists as the seam for a future treasury conversion service, gated by fresh re-probes and per-class hold periods. Nobody has converted fuel to fiat through it, and it is not wired into the marketplace, redemption, or delivery paths. It stays that way until counsel reads the flow.

The guarantees that exist are guarantees about settlement between direct counterparties — not about price, liquidity, or redemption into fiat.

6. What is next

- **The live $0.01 x402 purchase proof (pending).** The proof script is built and fixture-tested 12/12: it takes a buyer's USDC-on-Base transaction hash, polls four public Base RPCs for finalized quorum, runs delivery, buyer transfer, and live redemption, and emits a sanitized proof with a public explorer URL. The live run is pending two independent wallets. Until an unrelated buyer sends real USDC to an unrelated seller and redeems the delivered fuel, Gas Toll has not demonstrated a payment market.
- **The first public gate (pending).** A real page or API mounted behind the HTTP 402 door, with the deployer obligations the door contract specifies: real authenticated identity binding for the velocity throttle, HTTPS only, and strict key-transit logging discipline.
- **Ops hardening (state as of this writing).** Done: custodied secrets encrypt at rest under a key-identified envelope with a tested rotation path; the re-probe sweep scheduler persists its state in the settlement database with idempotent, crash-safe slots; the price oracle degrades through bounded, decaying-trust fallbacks and fails closed past its grace window. Remaining: wiring a live oracle into percentage tolls and listings, durable reconciliation of ambiguous on-chain authorizations, production observer and signing-key operations, public proof-of-reserves views, and a real independent trade between parties other than the operator.

7. Honest limits

The system's evidence boundary is stated in its own repository. This paper adopts it unchanged:

- **No live money has moved.** No current evidence proves a live USDC payment, a fiat conversion, or a seller in possession of real-world proceeds. Every payment leg executed to date is a labelled signed fixture.
- **No market exists yet.** The artifacts prove mechanism, not liquidity, demand, or durable operation at production scale. The marketplace has not been exposed publicly.
- **The surrenderer's copy.** A depositor can cheat: keep a copy of the key and spend the backing after fuel has issued against it. We cannot prevent that today. Caps bound each loss before the fact, measured priors charge for the observed failure rate, and re-probes freeze the lot when they detect the damage — containment, not prevention. The shared-balance matcher is documented as a rate-limiter, not a solvency proof.
- **Provider terms of service.** Surrendering or delegating provider keys may sit uncomfortably with some providers' terms. The system is provider-independent technically — any adapter-conforming provider works — but it rides on each provider's continued willingness to serve a custodied key, and participants carry their own ToS posture.
- **Oracle manipulation.** The price oracle's manipulation model is explicit: out-of-band quotes can be weaponized as a denial of service, and band-edge quotes can be harvested inside the TTL window. Both are bounded — secondary-source medians, a grace-bounded cache fallback, then fail-closed — and documented as residual risk, not absence of risk.
- **One operator.** Right now, participants trust the operator to custody the keys, report the reserves, and execute redemption correctly. That is more trust than a mature system should ask for; it is where the prototype stands. Public proof-of-reserves views are remaining work, so no third party can independently confirm the backing today. Signing keys in the proofs are in-memory, not external KMS identities. The red-team rounds are internal review. The sanitized artifacts are not independently verifiable standalone proofs.

Gas Toll is proprietary software. Copyright © 2026 James Kelley / Foinse. All rights reserved. Repository: design documents `IDEA.md`*,* `README.md`*,* `KEYSTONE_INFERENCE_MONEY.md`*,* `ADAPTER_CONTRACT.md`*,* `OPS.md`*; evidence under*

`live_evidence/`*; red-team dispositions under* `RED_TEAM_*.md`*.*