Nodexo: A Peer-to-Peer Market for Verified Compute

Anonymous
www.nodexo.ai

Abstract. Access to computation has come to rely on centralized providers serving as trusted third parties. A renter cannot verify that the hardware behind a dashboard is real, that the machine paid for is the machine received, or that access will not be repriced or revoked at the provider's discretion. We propose a peer-to-peer compute market in which the machine itself is the object of proof. Operators demonstrate physical possession of their silicon through timed hardware challenges bound to a persistent device fingerprint and to an on-chain identity. Settlement requires a signature rather than an account. The right to consume capacity can be earned rather than purchased: participants who perpetually lock stake to the subnet receive a daily allowance of compute in proportion to their conviction, while the stake itself is never spent. A second phase tokenizes the sub-subnet: each incentive mechanism inside the subnet becomes a market with its own token capitalized in the subnet's alpha, miner emissions divide between the base fleet and this market of markets, and admission requires the same proof of hardware. The network's value grows with its participants under Metcalfe's and Reed's laws, while Jevons' paradox underwrites demand for the capacity it serves.

1. Introduction

Computation has become the binding constraint on machine intelligence, and access to it runs almost entirely through trusted intermediaries. The arrangement works well enough for routine workloads, but it inherits the weaknesses of the trust-based model. A renter cannot inspect the data center. Hardware can be misrepresented, oversubscribed, or substituted. Prices and terms can change without recourse. Access itself can be narrowed or revoked when it becomes inconvenient to the provider, and as the means of frontier development concentrate into a small number of firms, the conditions attached to access multiply. Every guarantee in the present system is a promise, and every promise has an owner.

What is needed is a compute market based on cryptographic proof instead of trust, allowing any two willing parties to transact directly without a trusted intermediary [1]. Computational work is costly to prove in general, but two narrower statements admit practical proof: that a specific physical machine exists and is controlled by a specific identity, and that payment for its use was made. In this paper we propose a market built on exactly those proofs. Hardware is verified by challenge and bound on chain to the operator who controls it; settlement is authorized by signature alone; and sustained commitment to the network, measured on chain as conviction, entitles a participant to a recurring share of the network's capacity. We further describe a recursive extension in which new compute markets form inside the subnet itself, each carrying its own token, each admitted to the incentive mechanism only by the same proof of hardware.

2. Proof of Hardware

We define a miner as a machine offered to the network. Each miner is bound to a hardware fingerprint derived from immutable identifiers of the machine's physical components: the hash of its GPU device identifiers concatenated with its system identifier. The fingerprint is recorded in an on-chain registry at registration and checked thereafter; a change of hardware changes the fingerprint and is detected:

$$F = SHA\text{-}256(u_1 \,||\, u_2 \,||\, ... \,||\, u_n \,||\, u_{sys})$$

where u_1 through u_n are the device identifiers of the GPUs and u_{sys} is the system identifier.

Possession is established by timed challenges sized to the claimed silicon. Let r_c denote the effective throughput of device class c on the challenge workload. A validator issues a nonce-seeded workload of size W with a response deadline Dt chosen such that

$$W / r_c <= Dt < W / r_{c-1}$$

so that hardware of at least class c completes within the deadline and the next class down cannot. The response commits to the work, not merely the answer: outputs are partitioned and committed under a Merkle root, and validators audit randomly selected leaves, so fabricating a response requires doing the work the response claims. Because the deadline budgets compute time with little slack, relaying challenges to remote hardware spends the budget on round trips and misses the window. Challenges are issued continuously with fresh nonces, so proof of hardware is a property the fleet maintains, not a ceremony it performs once.

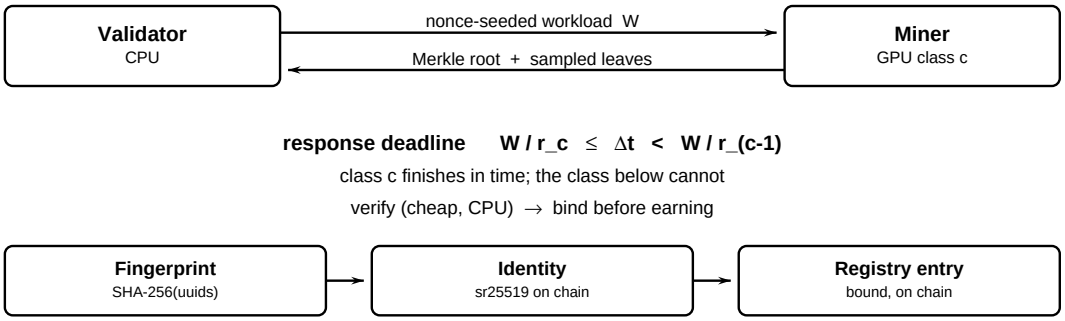

Figure 1: Proof of hardware. A timed, nonce-seeded workload yields a Merkle commitment the validator audits by sampled leaves; the deadline separates device classes, and fingerprint plus on-chain identity bind the machine before it earns.

The asymmetry is deliberate. Producing a valid response requires holding the claimed silicon; checking one requires recomputing a handful of sampled leaves against a Merkle root and reading a clock, work a commodity CPU performs in moments. Verification is cheap by construction, and section 4 develops the architectural consequence: validators need no GPUs at all.

Identity is established on chain. The operator signs with the key of its network identity, and the signature is verified by a native precompile, binding miner, fingerprint, and identity in a single registry entry before the machine may earn.

We are explicit about scope. Proof of hardware attests that the machine is real, of the stated class, and controlled by the stated identity. It does not attest to the semantic correctness of arbitrary workloads, a

problem we regard as important and adjacent. A market that is honest about what it proves is the predicate for any market that proves more.

3. Network

Nodexo operates as subnet 106 of Bittensor, a peer-to-peer intelligence market in which subnets define their own incentive mechanisms and are rewarded in the network's native asset [2]. The subnet's contracts run on the Bittensor EVM and reach consensus state through native precompiles: subnet membership and the metagraph, the staking module for stake and emissions, and signature verification for identity. Reads are taken at finalized heads, so the registry and the conviction ledger track state that cannot be reorganized.

Settlement is denominated in TAO. Deposits flow through a gateway contract that performs a transparent split: an owner share, capped at twenty percent in the contract, is routed to the subnet owner, and the remainder is staked to the subnet's validator. The split has a consequence worth stating plainly: paying for compute increases the stake securing the subnet that serves it. Usage and security are not in tension; one purchases the other.

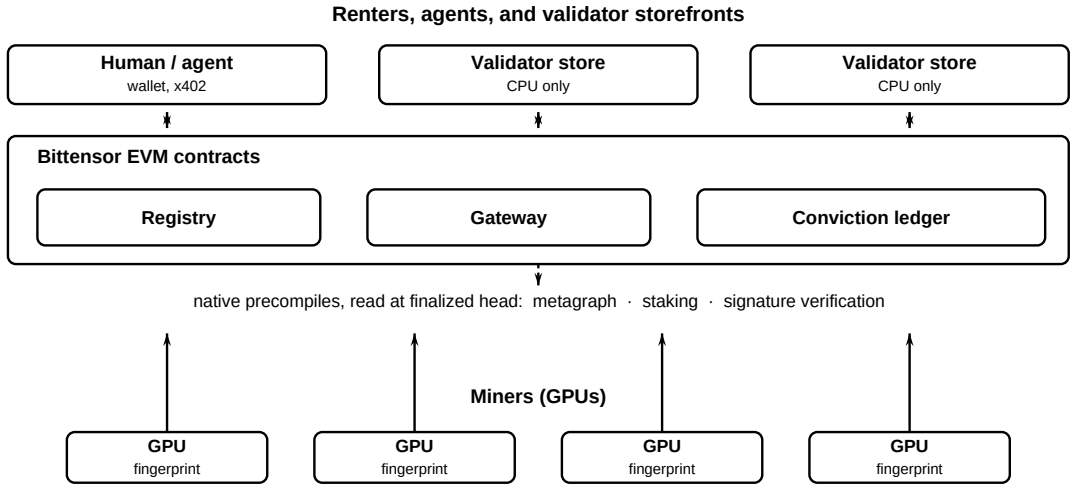

Figure 2: System architecture. EVM contracts read consensus state through native precompiles at finalized heads. Renters, agents, and independent validator storefronts transact against the same registered miner fleet; settlement and deposits flow in TAO.

4. Market

Operators list machines at prices of their choosing and earn through two channels for the same honest capacity: rental revenue from renters, and the subnet's emissions for maintaining provable hardware. The two channels discipline each other. Emissions reward capacity even when demand is slack, and rental revenue rewards capacity the market actually wants.

Renters have three paths to a machine. First, per-rental settlement through x402, an open standard for internet-native payments in which a wallet signature authorizes the payment itself [10]; no account, key, or stored balance exists to be created or revoked. The market is in this sense fully agentic by construction: an autonomous agent holding a wallet rents a machine exactly as a person does, there

being no step of onboarding a non-human cannot perform, which positions the network for demand that arrives as software rather than as users. Second, a prepaid credit balance funded in TAO or USDC, against which metered rentals run with no fixed end time, billed while the machine runs and terminable at any moment. Third, conviction allowance, described next. Credits are an accounting balance anchored to on-chain deposits; they are spendable on compute and are not withdrawable as cash, which confines the mechanism to its purpose.

Validation is open by the same asymmetry. Because checking proofs is CPU work, running a validator on subnet 106 requires no GPU, and the role is open to anyone rather than to whoever owns a datacenter. Validators are also distribution. Any validator may operate its own storefront and sell the network's capacity through it, competing on interface, service, and reach rather than on captive supply. Compute subnets to date have tended to route through a single platform's front door, one company's frontend standing between the network and its renters. Subnet 106 is arranged as a protocol with many doors: the capacity belongs to the network, and every door reaches all of it.

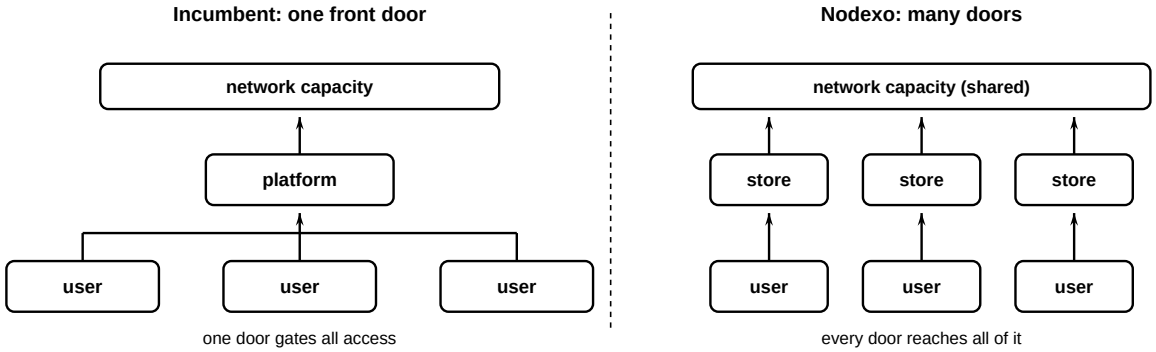

Figure 3: Distribution. Incumbent markets route through one platform frontend; on subnet 106 any CPU-only validator runs a storefront over shared capacity, so no single door is a chokepoint.

5. Conviction

Bittensor's conviction system, introduced as BIT-0011, allows stake to be locked and measures commitment as a function of locked mass and time [5]. A decaying lock loses locked mass on a half-life of roughly sixty days and may be unstaked only as it decays. A perpetual lock freezes its mass, accrues no decay, and cannot be unstaked until switched back to decaying mode and bled out along the same curve. A lock directed to the subnet owner's hotkey is credited at its full locked mass immediately. Nodexo recognizes perpetual locks only: the allowance pays for standing commitment, and a position scheduled to leave is not one. Switching a lock to decaying mode ends eligibility at the next daily calculation, while the stake itself releases only along the half-life curve. Nodexo reads lock state directly from the chain at finalized heads through the native lock query, recognizes one lock per coldkey, and rejects locks whose target is not the subnet owner's hotkey.

Each day the network computes a capacity value C, the revenue the listed fleet would produce in twenty-four hours at listed prices, and sizes an allowance pool as a tunable fraction m of it:

$$C = 24 \cdot (p_1 + p_2 + ... + p_n)$$

with p_j the hourly price of listed machine j. Let v_i be the conviction of participant i and V the sum of conviction across all participants. The participant's daily allowance is

$$a_i = min(m \cdot C \cdot v_i / V ,\ a_{cap})$$

where a_{cap} is a per-account ceiling. Allowance is credited once per day, is spent before paid credits, does not roll over, and resets at the next calculation. It is a flow, not a stock. The stake that earns it is never consumed and retains every property stake otherwise has on the network, so the participant's position continues to earn its native yield while paying a second yield denominated in compute.

The design has three properties worth noting. Because allowance is proportional to conviction, splitting a position across identities yields no advantage; the mechanism is sybil-neutral by construction. Because conviction weights time as well as mass, transient capital cannot harvest the pool by arriving before the daily calculation and leaving after it. And because the pool is a fraction of capacity rather than a fixed issuance, the network never promises compute it does not have; in the limiting case where no machines are listed, the capacity value is zero, the pool is empty, and the allowance is simply zero for that day rather than a claim on capacity that does not exist. Finally, eligibility is binary in the lock's mode: a perpetual lock holds its full share and a decaying lock holds none, so the share schedule is flat for exactly as long as the commitment is, and no passive participant's allowance erodes silently underneath them.

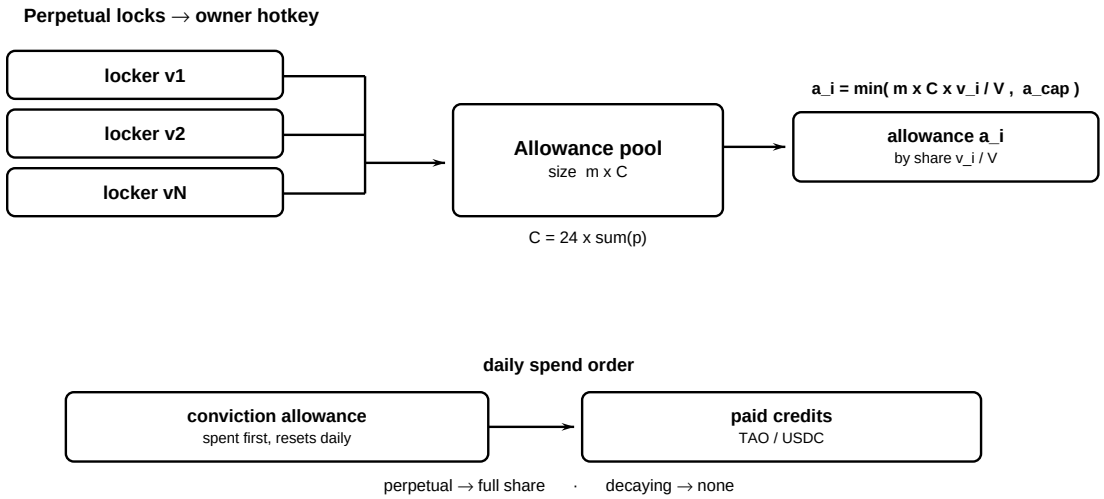

Figure 4: Conviction allowance. Perpetual locks to the owner hotkey are summed; the daily pool m·C divides by conviction share, capped per account, and is spent before paid credits.

6. Incentive

The incentive structure is arranged so that honesty is the profitable strategy for every class of participant. An operator who misrepresents hardware fails timed challenges and forfeits emissions; the fingerprint binding makes substitution detectable; and the same registry entry that lets a machine earn is the one that identifies it, so reputation accrues to real iron. A participant who locks conviction is paid in capacity for alignment, and the payment stops the moment alignment does. The subnet owner's share of deposits is capped and visible on chain, and the remainder of every deposit strengthens the validator that secures the system. The incentive may help encourage participants to stay honest, because it is more profitable to play by rules that pay twice than to defraud a system that detects and disconnects [1]. The same logic governs whole markets in the sub-subnet layer: a market's standing depends on the verified capacity its miners keep proving, so neither a miner nor a market earns by reputation once the proofs stop.

7. Tokenized Sub-Subnets

Bittensor recently introduced sub-subnets, since formalized as multiple incentive mechanisms within a single subnet [4]. A subnet owner may partition miners across mechanisms, each running Yuma consensus independently with its own weight matrix and bond pool, and apportion the subnet's emissions among them. The primitive is useful and deliberately limited: the mechanisms share the subnet's single alpha, their number is capped and rising, and the emission split between them is set by the owner at discretion. The arrangement echoes Bittensor before dynamic TAO, when emissions across subnets were apportioned by the votes of root validators rather than by a market; dynamic TAO replaced that vote-weighting with market pricing, letting the value of each subnet's alpha determine its share of emissions [3]. Inside a subnet, the analogous step has not yet been taken: the split across mechanisms remains an administered setting.

Phase two of Nodexo performs the same replacement one level down. Each sub-subnet of subnet 106 issues its own token on the Bittensor EVM and is capitalized through a pool denominated in the subnet's alpha: staking alpha into a sub-subnet prices its token, and the sub-subnet's share of emissions follows the net capital flowing into it, precisely as dynamic TAO now allocates among subnets by net flow, with 106 alpha standing in the role of TAO. The administered split becomes a market: allocation among the sub-subnets follows the price of their tokens rather than an owner's setting. The boundary between the base fleet and the layer is the one dial kept by hand, a policy parameter rather than a price. A market for inference on one model family, a market for one class of GPU, a market for one region's economics: each becomes a priced vertical with its own token, its own community, and its own claim on emissions.

Creation mirrors Bittensor's own subnet registration, one level down. Where registering a subnet on Bittensor costs TAO at a price that rises with each registration and decays back toward a floor, registering a sub-subnet on Nodexo costs alpha on the same dynamic schedule: a burst of registrations makes the next more expensive, a quiet interval makes it cheaper, and the fee is burned, so the cost of entry is paid to every holder as a reduction in supply rather than to any treasury. Bittensor stops there: a subnet is registered once the cost is paid. Nodexo adds a second stage of its own. Paying the fee buys the right to open a continuous clearing auction, a constant-product price-discovery process modeled on a liquidity bootstrap rather than on Bittensor, which has no such mechanism. Each sub-subnet has a fixed maximum supply of twenty-one million of its own token. Bidders contribute alpha, which is not spent but held to seed the market. At graduation the protocol mints four hundred twenty thousand of the

token, two percent of the cap, in two equal tranches: two hundred ten thousand distributed to bidders pro rata, and two hundred ten thousand paired against the bidders' alpha to open the constant-product pool. The auction thus discovers the token's price, distributes its opening supply, and seeds a two-sided pool in a single event, and the remaining supply, the balance of the twenty-one million, is emitted over the market's life as alpha itself is. Both tranches count against the cap. Only the swap fee on each bid is burned, as every later trade in the pool burns a fee; the bidders' alpha becomes liquidity, not ash.

The layer is capacity-bounded: at launch it holds four tokenized sub-subnets, a ceiling raised as sustained demand warrants. Graduation is resolved against that ceiling, by the same deregistration mechanism Bittensor applies to subnets at the root. If a seat is open, the graduate goes active and begins receiving emissions. If all four are full, the graduate displaces the weakest incumbent: the sub-subnet with the lowest exponential moving average of its token price, among those past their immunity period, is deregistered and the graduate takes its slot. A newly active market is immune from deregistration for one month, so a young market is given time to establish itself before its standing can evict it. Note that eviction reads price while emission follows flow: the two signals are deliberately distinct, mirroring Bittensor, where allocation tracks net inflow and deregistration tracks price. When a market is deregistered, its pool is unwound and the alpha it held is distributed pro rata to the holders of its sub-token, so a closing market returns its reserve to the participants who funded it rather than stranding them. The burned registration fee, the auction, and the cap with displacement together ensure the four live markets are always the four that have most recently earned their places.

Active status is earned continuously, not granted once. A sub-subnet's standing rests on two signals: the net capital flowing into its pool, and the verified capacity its miners prove. A market that ceases to attract net inflows, or whose miners stop passing challenges, loses its standing and decays out of emissions; neither capital alone nor a token price can sustain a market that has stopped producing provable compute. Both admission and survival reduce to the same test, and a market that falls to the bottom of the four is the one a new entrant displaces.

The layer is funded by division, not dilution. Miner emissions E_m, the forty-one percent share of subnet emissions that Bittensor directs to miners, divide into a fraction f directed to the base market of section 4, scored by proof of hardware, and the remainder injected into the liquidity pools of the sub-subnet layer:

$$E_{fleet} = f \cdot E_m , \quad E_k = (1 - f) \cdot E_m \cdot w_k$$

where w_k is the weight of pool k among those admitted. An injection is not a payout: alpha entering a pool deepens its alpha-side reserve, so emissions arrive as standing liquidity rather than as sell pressure, mirroring the way Bittensor injects emission into subnet reserves [3]. The weight follows the model Bittensor itself adopted. Bittensor moved its subnet allocation away from token price to net staking flow in late 2025, smoothing each subnet's net inflow with an exponential moving average and flooring negative-flow subnets to zero emission [3]. Nodexo's layer uses the same flow signal, with the subnet's alpha in TAO's place, and gates it by proof of hardware. Let s_k be the verified-capacity-weighted exponential moving average of net alpha flow into pool k, floored at zero. Then

$$w_k = s_k / (s_1 + s_2 + ... + s_n)$$

A market earns its share by attracting real capital and proving real hardware at once; smoothing over a long window makes transient flow unrewarding, and the zero floor denies emission to a market in net outflow. This is a deliberate improvement on the system it mirrors: Bittensor's flow model has no work

anchor, whereas here flow that is not backed by proven capacity earns nothing, so capital cannot purchase emission without compute beneath it. Both shares inject automatically every epoch; what is not automatic is the ratio between them. The division f, unlike the weights, is not a market variable: it is fixed in protocol and retuned only by hand, when the owner shifts the split between fleet and layer as the layer matures, and the native mechanism split that carries the change applies at most once per day on mainnet [4]. Owner and validator channels are unchanged.

Admission is not discretionary and cannot be bought. A sub-subnet is recognized only if its capacity passes the same proof of hardware that governs the base market, and its standing tracks the verified capacity its miners continue to prove. The filter is constitutional rather than curatorial: a token can be launched, but only proven silicon earns emissions, and only net inflows backed by that silicon move its share. This is the property that keeps a market of markets from degenerating into a casino of casinos. The deliberate ceiling of four reinforces it: scarcity of slots concentrates capital and verified compute into a small number of real markets rather than dispersing emission across a long tail of thin tokens, and the ceiling rises only as genuine demand fills the seats below it.

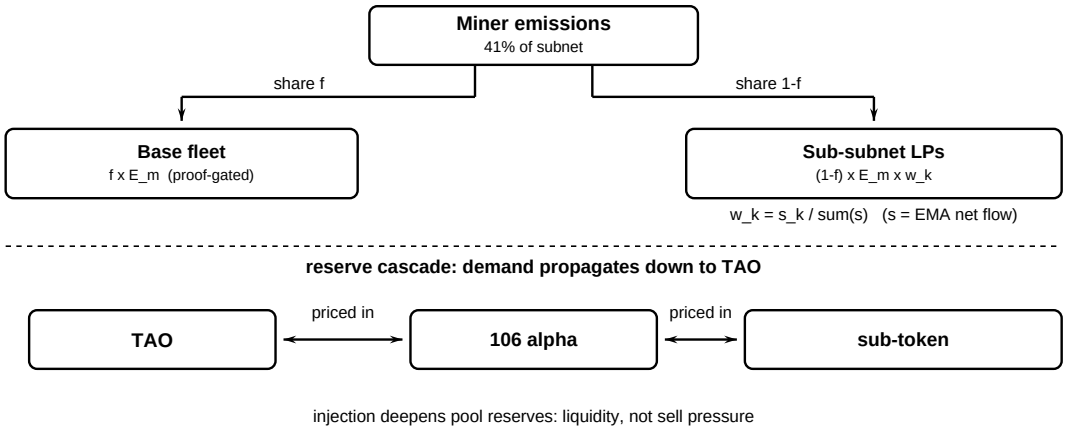

Figure 5: Emission split and reserve cascade. The 41% miner share divides by a manually fixed f into the proof-gated fleet and flow-weighted injections into the (capped) sub-subnet pools; entry cascades TAO → alpha → sub-token.

8. Value Flow to Bittensor

The design is arranged to be strictly additive to its host network. Six flows make this precise.

First, the reserve cascade. Entry into any sub-subnet runs through the chain of pools: TAO is staked into subnet 106 for alpha, and alpha is staked into a sub-subnet for its token. Sub-tokens are priced in alpha and alpha is priced in TAO, so every economy that forms at the leaves is collateralized, ultimately, in TAO. Demand at any layer propagates to the root, and under dynamic TAO sustained net inflow draws a larger share of emissions to the subnet, deepening its reserve of TAO further [3]. The layer's share of miner emissions is injected into these same pools as alpha-side reserve, so issuance deepens the books the economy trades on rather than landing as sell pressure.

Second, the lock cascade. TAO rests in the subnet's reserve; alpha is perpetually locked by conviction under section 5 and sequestered in sub-subnet pools; sub-tokens are bound in their own markets. Each layer removes the asset of the layer beneath it from circulation, converting speculative float into

structural commitment at every level, with TAO at the bottom of all of it.

Third, denomination. Gas on the Bittensor EVM is paid in TAO, rentals settle in TAO, and every deposit's remainder is staked to the subnet's validator under section 3. Activity in the compute economy raises TAO's transactional demand and the stake securing consensus at once.

Fourth, emission quality. A standing critique of emission systems is that their allocation signal, whether price or capital flow, can be manufactured while real substance cannot. Within subnet 106, no emission reaches a market whose hardware has not been proven, and net inflow moves a market's share only when proven capacity backs it, so the issuance routed through this subnet purchases verifiable capacity rather than narrative. To the extent the gate holds, each unit of TAO emitted here is backed by silicon that demonstrably exists, a stronger claim than the asset's critics usually contend with.

Fifth, scale without subsidy. Subnet slots at the root are scarce and costly, and the host network's emission budget is fixed. Sub-subnets multiply the count of functioning markets inside a single netuid, funded entirely from the subnet's own miner emissions and consuming no additional root resources. The ecosystem gains the group-forming growth of section 9 at zero marginal cost to the chain, and sub-subnets that outgrow the layer graduate to root registration, returning demand to TAO once more.

Sixth, the burn. Sub-subnet activity removes alpha from circulation through fees, while the capital itself is retained as liquidity. The registration fee that opens an auction is burned rather than collected. The alpha bid into an auction becomes pool reserve at graduation, but the swap fee on each bid is burned, as is the swap fee on every later trade in the pool. Launch, price discovery, and ongoing operation each return value to holders as a reduction in supply, while bid principal and emissions accumulate as standing reserve. As the asset beneath the sub-subnets, alpha is itself collateralized in TAO through the reserve cascade, so tightening alpha supply concentrates value that ultimately rests on TAO. The layer grows its host by issuing into reserves and shrinks its own float by burning fees, in the same motion.

9. Economic Foundations

Three regularities from network economics locate the design.

Metcalfe's law holds that a network's value scales with the square of its connected participants [7]. A compute marketplace is such a network: each additional operator widens every renter's choice, and each additional renter deepens every operator's demand. Liquidity begets liquidity.

Reed's law observes that networks which allow groups to form scale faster still, since the number of possible groups grows exponentially in membership [8]. Phase two is the deliberate exploitation of this regularity. Tokenized sub-subnets are group formation: every specialized market that forms inside the subnet creates value that no enumeration of pairwise rentals captures, and the reserve asset of all of them is the subnet's alpha.

Jevons observed that improvements in the efficiency of a resource's use increase, rather than decrease, total consumption of the resource [6]. Computation is the canonical modern instance: every fall in the cost of intelligence enlarges the set of problems worth applying it to, and aggregate demand rises. A market whose capacity is verified and whose access cannot be revoked is positioned on the right side of this curve. It does not need to win demand from incumbents; it needs only to exist when induced demand arrives.

Finally, the design can be read through Coase: firms exist where the cost of transacting across a market exceeds the cost of organizing within a hierarchy [9]. The cloud is a hierarchy that exists because

trusting strangers' hardware was expensive. Proof of hardware collapses that transaction cost, and the boundary of the firm recedes accordingly. What remains is a market.

10. Adversarial Considerations

We consider the principal attacks.

Spoofed hardware. An attacker claims silicon it does not hold. Timed challenges make the claim unprofitable: responses computed on lesser hardware miss the window, responses proxied to distant hardware pay latency, and a machine that fails challenges stops earning. The fingerprint binding prevents rotating identities across the same hardware, or the same identity across changing hardware, without detection. Renting real hardware to answer challenges does not pay either: challenges are sampled so that servicing them approaches the cost of honestly operating the claimed capacity, leaving expected profit nonpositive.

Sybil locking. An attacker splits stake across many identities to capture allowance. Share is linear in conviction, so the sum of the parts equals the whole; nothing is gained. Per-account ceilings bind the large, not the many.

Flash locking. An attacker locks immediately before the daily calculation and exits after it. Eligibility requires perpetual mode, so there is no exit to flash: leaving means switching the lock to decaying, which ends allowance at the next calculation and releases stake only along the half-life curve thereafter. Reads occur at finalized heads and one lock is recognized per coldkey. The fastest possible harvest is a day of allowance purchased with months of illiquidity, and it does not pay.

Whale capture. A single holder attempts to absorb the pool. The per-account ceiling a_{cap} bounds any account's daily draw, and the pool's definition as a fraction of capacity bounds the total giveaway regardless of the distribution of conviction.

Extraction. An attacker attempts to convert allowance into cash. Allowance credits are spendable on compute and are not withdrawable; the worst case is consumption of capacity the pool had already set aside.

11. Privacy and Access

The base market requires no account. A wallet signature is sufficient to rent a machine, and the network serves keys, not customers. Participants are pseudonymous to the protocol in the manner of public blockchains: identities are public keys, and what is public is the flow of settlement, not the identity of the party [1]. Because no intermediary stands between a key and a machine, there is no party positioned to narrow, reprice, or revoke access selectively, and distribution itself is plural: with validators free to run independent storefronts under section 4, no single frontend is a chokepoint through which access can be governed. Removal from the network is a consequence of failing its proofs, not of displeasing its operator.

12. Implementation Status and Roadmap

Phase one is implemented: the registry and gateway contracts on the Bittensor EVM, the challenge system and fingerprint binding, the marketplace with x402 and credit settlement, and the conviction ledger reading native lock state. It launches on subnet 106 in June 2026. The conviction allowance opens as Bittensor's native conviction primitives complete their rollout to mainnet, and capacity accounting and settlement move progressively deeper on chain as those primitives mature. Conviction v1 is live on mainnet; the instant owner-credit and native lock-read paths assumed in section 5 ship with v2, presently in test, and the allowance opens with them. Phase two, the tokenized sub-subnet layer, follows: token issuance and alpha-denominated pools on the EVM, the miner-emission split between the base fleet and the layer's liquidity pools, fixed in protocol and adjusted by hand, and the verification gate of section 7. We prefer to claim what is running over what is intended, and to label the difference.

13. Conclusion

We have proposed a market for compute without trust. We began with the observation that access to computation runs through promises, and that every promise has an owner. We replaced the promises that admit proof: possession of hardware, control of identity, and payment, each verified rather than assumed. We then inverted the basis of access, allowing capacity to be earned by measurable conviction instead of granted by an intermediary, with a stake that is never spent paying a yield denominated in compute. Finally we described how the market generalizes: specialized compute markets forming inside the subnet, each with its own token capitalized in the subnet's alpha, admitted to incentives only by proof of hardware, funded from the subnet's own emissions, and additive by construction to the network that hosts it. The network needs no trusted intermediary because it was arranged so that none is required: machines prove themselves, payments authorize themselves, and commitment measures itself. What the participants own, no one else can switch off.

References

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.

[2] Y. Rao, "Bittensor: A Peer-to-Peer Intelligence Market," 2021.

[3] Opentensor Foundation, "Dynamic TAO," 2025.

[4] Opentensor Foundation, "Multiple Incentive Mechanisms Within Subnets," Bittensor Documentation, 2025.

[5] Opentensor Foundation, "BIT-0011: Locked Stake and Conviction," 2026.

[6] W. S. Jevons, "The Coal Question," Macmillan and Co., 1865.

[7] G. Gilder, "Metcalfe's Law and Legacy," Forbes ASAP, 1993.

[8] D. P. Reed, "That Sneaky Exponential: Beyond Metcalfe's Law to the Power of Community Building," Context Magazine, 1999.

[9] R. H. Coase, "The Nature of the Firm," Economica, vol. 4, no. 16, 1937.

[10] Coinbase, "x402: An Open Standard for Internet-Native Payments," 2025.