Hearth Tokenomics
HRTH Emission, Creds, and the Retirement Split

July 10, 2026

DRAFT v2 · illustrative parameters, load-bearing formulas

Abstract

Hearth runs a two-asset economy that separates the token securing the network from the credits pricing its output. HRTH is hard-capped and scarcity-shaped: a 5% genesis premine plus a continuously disinflationary, back-weighted block reward that keeps early float scarce while emission asymptotes to a fixed cap over decades. Creds are per-model, USD-priced inference credits issued algorithmically to HRTH stakers, each model's issuance following measured demand and clamped every epoch to its verified servable capacity. When a Cred is spent it is retired and its value split three ways, 60% burned, 30% to the serving node, 10% to the verifier pool, and only the burned fraction counts toward consensus, which decouples node cash flow from the forging signal and keeps self-dealing loss-making. Because the burn is the sole deterrent, the node share is not a free constant but a governed function of network maturity, launched low and raised toward a deterrence-bounded ceiling as measured usage deepens. The formulas are the specification; all constants are illustrative and left for governance and genesis to set.

This document specifies the two-asset economy of Hearth: the HRTH emission curve, Cred issuance, the retirement split, the forging-boost mechanism, and the node economics that tie them together. Constants marked (genesis) or (governance) are tunable; the formulas are the specification. All worked figures are illustrative and self-consistent with the parameters stated in §8.

1 The two assets

HRTH is the network token: a hard-capped, scarcity-shaped asset that is staked to forge, spent as gas, used in governance, and issued as the reward

for verified work. It secures the chain and funds the bootstrap subsidy. On proven misbehavior it is suspended, never confiscated.

Creds are per-model inference credits (per-model from Phase 2 onward, a single Cred in Phase 1): USD-priced, capacity-backed, retired on use. One Cred-m redeems for one standardized unit of verified output from pinned model m. Creds are emitted to HRTH staked for Creds as a capacity-weighted basket, the yield that makes that position worthwhile, and they carry no peg; each Cred's market price is the discovered price of that model's verified inference.

The design separates the asset that secures the network (HRTH, scarce and held) from the assets that price its output (Creds, elastic and spent).

The separation is not a stylistic choice but a provable requirement. A single-token proof-of-stake economy cannot hold service prices stable without letting validator rewards grow without bound; Kiayias, Lazos, and Penna prove this and show that a two-token design, one asset staked by validators and another spent by users, attains viability, decentralization, price stability, and on-chain feasibility at once [1]. Hearth's HRTH/Cred split instantiates exactly that separation, HRTH the staked and secured asset, Creds the spent and priced one.

Burn-on-use service credits are otherwise the established DePIN pattern: Helium's USD-pegged Data Credits [2], Render's render credits [3], and Internet Computer cycles [4] are fiat-denominated, retired on use, and insulate consumers from network-token volatility, at the cost of an administered peg. Creds keep the retire-on-use discipline but depart in two ways. They carry no peg, so each model's Cred price is discovered on the market rather than administered, and they are issued to stakers as the yield on HRTH under a verified-capacity clamp rather than minted by burning the network token. The burn therefore sits at retirement, where it prices throughput and deters self-dealing (§4, §7), not at issuance [5].

Everything below makes that split precise.

2 HRTH emission, scarcity-shaped, hard-capped

HRTH has a hard cap, a tiny genesis premine, and a continuously disinflationary block reward. Unlike Bitcoin's front-loaded schedule, the curve is deliberately back-weighted in its early years: with only 5% minted at genesis, a front-loaded emission would flood the market and destroy the scarcity the asset needs while it bootstraps. The curve instead releases slowly, so

HRTH stays scarce through the first years and disinflates toward the cap over decades.

2.1 Supply structure

$$C_{\max} = P_{\text{gen}} + C_{\text{emit}}, \qquad P_{\text{gen}} = 0.05\, C_{\max}$$

- P_{gen}, genesis premine, 5% of supply. The only HRTH in existence at launch, split between the fallen-chains burn claim (the public genesis distribution), a DAO treasury, and a vested team allocation. No VC, no presale.

- C_{emit}, 95% of supply, emitted as the forging reward, block by block, over decades. This is the fair-launch core: almost all HRTH is earned by securing the network, not distributed at genesis.

Illustrative split of the 5% premine: fallen-chains burn claim 3%, DAO treasury 1%, team (vested) 1%.

The burn claim is earned, not sold. During a genesis window, holders of ten faded L1 chains provably burn their tokens on the home chains, a plain transfer to a published, provably unspendable burn address, and bind the source address to a Hearth address with a separately signed message (a transfer memo on chains whose wallets cannot sign bare messages). Each burn's credit is computed deterministically from public chain data: the burned amount is split into min-balance holding layers reconstructed from transfer history, each layer priced at the maximum weekly-average USD price of its token since the layer formed, and the layer credits summed, then published in a reproducible snapshot carrying a Merkle root and an independent recompute script. The weekly-average price is the credit formula's input, not a promised value: loyalty to the network converts to a share of it, credits convert one-to-one into HRTH at launch, and no floor or dollar value on the result is implied. The full mechanics, the chain list, and the dispute process are specified in the genesis burn specification and implemented by the genesis application.

Because the claim converts one-to-one and turnout is unknown in advance, the genesis supply floats with participation, so the absolute cap is derived at snapshot freeze rather than preset. Writing B for the total credits at freeze, the burn claim is pinned at 3% of supply, so $C_{\max} = B/0.03$, and the DAO (1%) and team (1%) allocations follow the same fixed proportions. Every proportion, every formula, and the emission curve's shape is

a turnout-independent genesis constant; the worked figures throughout use the illustrative $C_{\max} = 100\text{M}$, which corresponds to $B = 3\text{M}$ credits.

2.2 The emission curve

The block reward decays continuously, every block pays slightly less than the last, halving on a fixed cadence $T_{1/2}$:

$$R(h) = R_0 \cdot 2^{-h/H_{1/2}}, \qquad R_0 = \frac{C_{\text{emit}} \ln 2}{H_{1/2}}$$

where $H_{1/2} = T_{1/2} \cdot N_{\text{yr}}$ is the reward half-life in blocks, with N_{yr} the blocks per year (525,960 at 60 s blocks). The normalization makes total emission $\int R(h)\, dh = R_0 \cdot H_{1/2} / \ln 2 = C_{\text{emit}}$, so supply asymptotes to the cap:

$$S(t) = P_{\text{gen}} + C_{\text{emit}} \left(1 - 2^{-t/T_{1/2}}\right) \longrightarrow C_{\max}$$

Each block's HRTH reward is paid to that block's forger; the continuous Cred yield to stakers is separate (§3.3).

2.3 Why not a Bitcoin-style front-loaded halving

Bitcoin front-loads: its first four-year era mints half of all coins. Applied here (95M over a 4-year first era) that is ≈11.9M HRTH in year one against a 5M genesis float, a 238% first-year expansion, with circulating supply already ≈17% of cap by year one (a flat first-era mint; the continuous-half-life 4-year row in the table below gives 20%, the two being idealizations of the same schedule). That is the opposite of scarce. The half-life is therefore lengthened until early float is genuinely scarce:

reward half-life $T_{1/2}$	circ. y1	y2	y4	y8	y16
4 yr (Bitcoin-like)	20%	33%	52%	76%	94%
8 yr	13%	20%	33%	52%	76%
10 yr (chosen)	11%	17%	28%	45%	69%
16 yr	9%	13%	20%	33%	52%

Table 1: Circulating supply as a fraction of cap under alternative reward half-lives.

2.4 Instantiated schedule

With $C_{\max} = 100\text{M}$, $P_{\text{gen}} = 5\text{M}$ (5%), $C_{\text{emit}} = 95\text{M}$, $\tau_b = 60$ s, and $T_{1/2} = 10$ years (genesis), the initial reward is $R_0 \approx 12.52$ HRTH/block:

Year	Reward/block	Emitted that year	Circulating supply	% of cap
0 (genesis)	–	–	5.00M	5.0%
1	11.68	6.36M	11.36M	11.4%
2	10.90	5.94M	17.30M	17.3%
4	9.49	5.17M	28.00M	28.0%
8	7.19	3.92M	45.44M	45.4%
12	5.45	2.97M	58.65M	58.6%
20	3.13	1.70M	76.25M	76.2%
40	0.78	0.43M	94.06M	94.1%
∞	0	0	100M	100%

Table 2: HRTH emission, scarcity-shaped, 10-year half-life, 5% premine (illustrative).

Only ≈11% of supply circulates after year one and ≈28% after year four, the intended scarcity, while the year-one forging pool (≈6.4M HRTH, ≈ $1.27M at $0.20) is ample for security (§7). Because emitted HRTH is earned by stakers, who lock it to keep earning, the liquid float is scarcer still. $T_{1/2}$ is a genesis parameter; a shorter half-life releases faster, a longer one is scarcer for longer.

3 Cred issuance, algorithmic, demand-following, per-block

Phase 1 runs a single Cred, one USD-denominated, capacity-backed, burn-on-use credit, because relaying to a hosted aggregator means no per-model self-hosted capacity backs distinct claims and no network-discovered per-model price exists. The per-model index m below is therefore Phase-2 machinery: in Phase 1 it ranges over a single element, and every formula applies unchanged to the one Cred. Per-model Creds phase in model by model in Phase 2 as self-hosted verified capacity comes online, with the single Cred persisting as the settlement numeraire.

Creds are not on a fixed curve and are not voted on each epoch. Each model's Cred issuance is set by an algorithm from the previous epoch's consumption; the DAO's only role is to seed the initial supply and set the controller's constants. Issuance is then streamed to stakers every block.

3.1 The epoch and the control signal

An epoch is two weeks, $N_{\text{ep}} = 20{,}160$ blocks at 60-second block time. For model m in epoch E, the demand signal $C_m(E)$ is the burned, payer-gated value of that model's retirements, normalized back to Cred units:

$$C_m(E) = \frac{\text{burned payer-gated USD value of model-}m\text{ retirements in } E}{\phi_b \, \bar{p}_m(E)}$$

with $\bar{p}_m(E)$ the epoch's time-weighted average Cred-m price. Since each gated retirement burns exactly ϕ_b of its value, the normalization makes $C_m(E)$ the full count of payer-gated Creds retired in the epoch, so issuance still replaces consumption one-for-one; what changes is what cannot move the signal: retirements that never passed the payer gate, and the recycled node and verifier shares, contribute nothing. This closes the loop with §4.1: consensus weight and Cred emission read the same observable, the burned gated fraction, so a self-dealer buys neither with the share it recovers.

3.2 The algorithmic controller

Issuance for the next epoch targets consumption, so Cred supply neither starves nor accumulates. The controller smooths the signal and clamps it to verified capacity κ_m^{cap}:

$$\widehat{C}_m(E) = (1-\lambda)\,\widehat{C}_m(E-1) + \lambda\, C_m(E)$$

$$Q_m(E+1) = \min\Big((1+\gamma)\,\widehat{C}_m(E),\ \kappa_m^{\text{cap}}(E+1)\Big)$$

- $\lambda \in (0, 1]$, EMA rate (controller constant, DAO-set once) damping epoch-to-epoch swings.
- $\gamma \geq 0$, a small growth allowance letting supply lead rising demand.
- $\kappa_m^{\text{cap}}(E+1)$, model m's verified servable capacity, the protocol-enforced ceiling.

This is a closed loop with no per-epoch governance action: given the constants, $Q_m(E+1)$ is a deterministic function of measured on-chain consumption. $Q_m(0)$, the initial supply at a model's listing, is the one DAO-set quantity, the seed the controller then adjusts automatically.

3.3 The capacity ceiling and per-block distribution

The $\min(\cdot, \kappa_m^{\text{cap}})$ is protocol-enforced: no controller path can mint a Cred faster than its model's verified capacity, so "capacity-backed" holds mechanically at every epoch. Thin-demand models converge to low issuance automatically.

The epoch's issuance is streamed to stakers every block, not dropped at epoch end. Each block emits

$$q_m = \frac{Q_m(E+1)}{N_{\text{ep}}}$$

Creds of model m, distributed pro-rata to HRTH locked in the Cred-staking contract (position 2, §6.1). Cred-stakers thus receive a continuous, capacity-weighted Cred basket, the smooth USD-denominated yield on HRTH staked for Creds (§6), rather than a lumpy periodic payout.

4 Cred retirement, the split

When a Cred of USD value p is spent, it is retired and its value divided three ways:

$$p = \underbrace{\phi_b\, p}_{\text{burned}} + \underbrace{\phi_n\, p}_{\text{serving node}} + \underbrace{\phi_v\, p}_{\text{verifier pool}}, \qquad \phi_b + \phi_n + \phi_v = 1$$

Launch values (governance): $\phi_b = 0.60$, $\phi_n = 0.30$, $\phi_v = 0.10$.

- Burn ϕ_b, permanently removed. Creates scarcity, ties Cred float to throughput, and is the self-dealing deterrent (§7).
- Node ϕ_n, paid to the serving node as immediate USD-denominated cash flow, the operator's answer to fiat-denominated running costs (§5).
- Verifier ϕ_v, funds the audit committee that re-checks sampled jobs.

4.1 The critical rule: only the burned fraction counts toward consensus

Both workBoost and capacity-weighted Cred emission count only the burned, payer-gated fraction $\phi_b \cdot p$, the demand signal defined in §3.1, never the node or verifier share. This quarantines consensus from the node payout: the 30% handed to the node is round-tripped back to a self-dealer and so contributes

nothing to deterrence, but because it also contributes nothing to workBoost, the node share can be paid without weakening the consensus signal at all. Node cash flow and consensus integrity are thereby decoupled.

5 Node economics and cost coverage

A node's cost is denominated in fiat (power, amortized hardware); its Cred-share revenue is USD-denominated but usage-dependent. Coverage is the market's job at the margin and the subsidy's job in bootstrap, never a reason to enlarge ϕ_n (which would break §7).

5.1 Per-job accounting

For a job of price p (USD) on a node with cost c_{job}:

$$\text{revenue}_{\text{node}} = \phi_n \, p + \rho = 0.30 \, p + \rho$$

where ρ is the HRTH issuance reward for the verified job (§2 emission, distributed per work). Serving is elective: a node serves only when $\text{revenue}_{\text{node}} \geq c_{\text{job}}$, so entry/exit discovers the floor and the protocol never measures unobservable cost.

5.2 Break-even Cred price

The price at which the node share alone covers cost ($\rho = 0$):

$$p^* = \frac{c_{\text{job}}}{\phi_n}$$

Phase-2 worked figure. A self-hosted FP8 node at ≈\$2.5/GPU-hr sustaining ≈4,000 tok/s produces $4000 \times 3600 = 1.44 \times 10^7$ output tokens per GPU-hour, so its cost is $\$2.5/(1.44 \times 10^7 \text{ tok}) \approx \$0.174$ per 1M output tokens (≈\$0.17), giving

$$p^* = \frac{\$0.174}{0.30} \approx \$0.58 \,/\, \text{1M tokens}$$

The mid-2026 market for the same open-weights 70B spans \$0.12–0.90 / 1M, so take one Cred's standardized unit to be 1M output tokens, matching these \$/1M figures. Above ≈\$0.58 the node profits on the Cred share alone and HRTH is pure upside; below it, HRTH issuance bridges the per-unit shortfall, worked in §5.3. Positioning: verified inference at commodity

price, not cheaper than the cheapest unverified host, but proven and undeprecatable.

5.3 Subsidy sizing

The bridge is HRTH, not a bigger Cred slice. Size per-epoch emission so that, at target utilization, the median honest node is whole per standardized unit (one Cred, 1M output tokens):

$$\mathbb{E}[\rho] \geq c_u - \phi_n \, p_{\text{market}}, \qquad c_u \approx \$0.174 \text{ per unit}$$

At a mid-band Cred price $p_{\text{market}} = \$0.30$, the per-unit gap is $0.174 - 0.30 \times 0.30 = 0.174 - 0.09 \approx \$0.084$ per 1M tokens, so a 1k-token job, one-thousandth of a unit, leaves an $\approx \$10^{-4}$ shortfall for HRTH issuance to cover. The gap shrinks as the Cred price rises toward p^* and vanishes above it; the organic-revenue-to-subsidy ratio (§6) signals when to taper. Because the year-one pool (≈6.4M HRTH) dwarfs the aggregate gap at any realistic early volume, demand, not the subsidy, is the binding constraint, the intended bootstrap condition.

5.4 Phase-1 contrast

In Phase 1 (relay), c_{job} is fiat paid to an aggregator at provider list price (≈\$5/1M, non-discountable), roughly 29× the Phase-2 cost. No split or subsidy makes relay sustainably profitable; Phase 1 is a deliberate loss-leader whose purpose is to stand up the chain, token, and demand before self-hosted capacity (Phase 2) drops the cost floor into the coverable range.

6 The economic loop

6.1 Three positions for an HRTH holder

An HRTH holder chooses one of three mutually exclusive positions per coin.

1. Hold. HRTH on a plain balance earns nothing, no Cred yield and no emission; holding is exposure to the asset, not a claim on network cash flow.

2. Stake for Creds. HRTH locked in the Cred-staking contract earns the continuous Cred stream of §3.3 (in Phase 1 the single Cred, later the

capacity-weighted per-model basket), pro-rata to locked HRTH, and earns no HRTH emission.

3. Validate, or lease to a validator. HRTH used as generating balance, owned or leased-in under a non-custodial lease with the lessor paid an agreed share of the validator's rewards, earns the HRTH block reward of §2 (amplified by workBoost for serving nodes), and earns no Creds.

The exclusivity is load-bearing because the two yields price different things: Creds price inference demand, emission prices security provision. Forcing the choice creates an internal opportunity-cost equilibrium, with stake flowing toward whichever yield is undervalued until the two risk-adjusted returns meet, which is the coupling loop below stated at the level of a single holder's decision. It also partitions the security analysis: workBoost and the forging pool concern only position 3, the Cred-issuance stream only position 2, so an attacker must split capital to farm both channels.

Entering position 2 or 3 is subject to an activation window: the stake counts, for the Cred stream or as generating balance, only at the minimum balance held over the preceding N_{act} blocks, so freshly moved coin cannot instantly claim either yield. Exiting is immediate, the coin is spendable at once and simply stops earning ($\tau_{\text{unb}} = 0$). At the genesis value $N_{\text{act}} = 1000$ blocks (≈ 17 h at 60 s), repositioning friction is bounded by hours, so the self-correction of §6.2 operates on a day timescale, not an unbonding-queue timescale.

6.2 The coupling loop

There is no explicit Cred-to-HRTH peg. The two assets are coupled through staking, and the coupling self-corrects:

1. Demand: users pay USD for Creds (any stablecoin, swapped to the model's Cred at point of sale, §9). Real inference demand becomes Cred purchases.

2. Yield: because Creds are emitted to Cred-stakers (§3.3), Cred demand is the USD yield on HRTH staked for Creds.

3. Value: a real USD yield makes staking HRTH worthwhile, creating HRTH demand.

4. Supply floor: validators must hold generating balance $\geq b_{\min}$ (owned or leased-in) to serve, so HRTH's value sets the serving floor: if HRTH

falls, serving turns unprofitable, nodes exit, Cred scarcity lifts the price → yield → HRTH value, and serving resumes.

The single health metric governance watches is the ratio of organic USD Cred revenue to HRTH subsidy value: rising, the market is self-sustaining; flat, the network is renting volume. The loop is reflexive with lag and degrades gracefully: Creds stay redeemable at true cost, so there is no peg to break.

Tagline: Creds exist to make HRTH staking viable.

7 Forging boost and self-dealing security

The quantitative security claims of this section are conditional on two mechanisms specified in the Hearth whitepaper and used here as a trust root: the payer gate, so only retirements paid with external funds through the gateway count, and verification, so a retired Cred corresponds to real, attested output, with fraud detected with near-certainty at the audit sampling rate. Every payer-gated quantity in this document ($C_m(E)$, $r_i(E)$, k_V, $\mathcal{D}$) inherits these assumptions.

7.1 workBoost (Proof of Inference)

Verified work amplifies a staker's forging weight, capped, never replacing stake. For node i with generating balance b_i (owned plus leased-in stake):

$$b_{\text{eff}}(i) = b_i\,(1 + \text{workBoost}_i), \qquad b_i \geq b_{\min}$$

Let $r_i(E)$ be the USD value of the burned fraction of payer-gated Creds retired on node i's verified jobs in epoch E (§4.1). Smooth it with an exponential moving average:

$$w_i(E) = (1-\alpha)\,w_i(E-1) + \alpha\,r_i(E)$$

Fix a governance-set dust floor $\tau_w > 0$ and let $S(E) = \{\, j : w_j(E) \geq \tau_w,\ b_j \geq b_{\min} \,\}$ be the eligible set. Normalize against the stake-weighted median work level

$$m(E) = \arg\max_{w \geq 0} \Big\{ \sum_{j \in S(E)} b_j\,\mathbf{1}[\,w_j(E) \geq w\,] \ \geq\ q \sum_{j \in S(E)} b_j \Big\}, \qquad q = \tfrac{1}{2},$$

the retired-work level reached by the median unit of stake rather than the median identity, with the stake quantile q (default $1/2$) governance-tunable and distinct from the half-saturation κ. If $S(E)$ is empty, no node clearing τ_w (as at bootstrap, before any gated demand accrues), $m(E)$ is undefined and the protocol sets $\text{workBoost}_i = 0$ for all i, running as vanilla FPoS; the boost switches on automatically in the first epoch some node clears the floor. Then $g_i(E) = w_i(E)/m(E)$ and

$$\text{workBoost}_i = B_{\max} \cdot \frac{g_i(E)}{g_i(E) + \kappa}$$

The map is monotone, concave, zero at zero demand (recovering vanilla FPoS), and bounded by $B_{\max}$ (governance, e.g. 2–3).

Remark 1 (Dust-Sybil resistance)**.** *A plain count median over positive-work nodes is cheap to move: an adversary registering many dust servers with w_j just above zero drags it down, so a genuine large-work node reaches the concave cap with far less real work. The dust floor τ_w and stake-weighting close this, since each eligible node enters $m(E)$ weighted by its stake b_j rather than as one identity, so shifting the stake-weighted median past a challenger requires controlling roughly half the eligible stake mass. Sybil identity count is then irrelevant without stake behind it.*

7.2 Bounded amplification

Lemma 2 (Bounded amplification)**.** *Take the worst case, an attacker fully boosted while the honest set is unboosted. Since $0 \le \text{workBoost}_i < B_{\max}$, a party with stake share σ then has forging-weight share*

$$s = \frac{(1 + B_{\max})\sigma}{(1 + B_{\max})\sigma + (1 - \sigma)},$$

so controlling a forging majority ($s > \frac{1}{2}$) requires

$$\sigma > \frac{1}{2 + B_{\max}}.$$

Proof. In the worst case the attacker's multiplier is $1 + B_{\max}$ and every honest stakeholder's is 1, which gives the share s above. Requiring $s > \frac{1}{2}$ gives $(1 + B_{\max})\sigma > 1 - \sigma$, i.e. $\sigma > 1/(2 + B_{\max})$. □

At $B_{\max} = 2$, an attacker still needs more than $\frac{1}{4}$ of stake (vs. $\frac{1}{2}$ for pure FPoS), the disclosed, bounded cost of coupling work to consensus.

7.3 Self-dealing deterrence

A node can pay itself to fabricate demand. Because it recovers $\phi_n + \phi_v$ on the round trip, its only real loss is the burned fraction plus wash friction ω on the payer-gate swap. Per dollar self-dealt, the forfeit rate is

$$f = \phi_b + \omega\,(\phi_n + \phi_v) \approx \phi_b$$

The analysis conservatively sets $\omega = 0$, assuming the attacker recycles the recovered node and verifier shares through the payer gate at zero cost. Since $f \geq \phi_b$ always, every deterrence figure below rests on the burn alone, and any real-world swap friction only strengthens it.

Let V be the self-served volume needed to saturate the boost and G the maximum boost gain. Deterrence is

$$\mathcal{D} = \frac{f \cdot V}{G} \approx \frac{\phi_b\, V}{G}$$

At launch ($\phi_b = 0.60$), against a 5%-stake attacker at $B_{\max} = 2$, $\mathcal{D} \approx 4.9\times$, self-dealing is strictly loss-making. Note the deterrent is the burn fraction only; the node share, being recovered, contributes nothing, which is why ϕ_n cannot be raised freely (§7.4).

7.4 The split as a governed function of network maturity

Write the boost-gain fraction $g_{\text{frac}} = G/\text{Pool}$, which from the Lemma's worst case has the closed form

$$g_{\text{frac}}(\sigma, B_{\max}) = \frac{(1 + B_{\max})\sigma}{1 + \sigma B_{\max}} - \sigma \quad (\approx 0.086 \text{ at } \sigma = \sigma_{\text{adv}} = 5\%,\ B_{\max} = 2),$$

evaluated at σ_{adv}, the assumed worst-case adversary stake share against which the ceiling is calibrated. Write $k_V = V/\text{Pool}$ for the maturity ratio, the self-serve volume needed to saturate the boost per unit of security budget, made on-chain-computable below. Requiring deterrence $\mathcal{D} \geq M$ gives a minimum burn:

$$\phi_b^{\min} = \frac{M \cdot g_{\text{frac}}}{k_V} \implies \phi_n^{\max} = 1 - \phi_v - \frac{M\, g_{\text{frac}}}{k_V}$$

The pool cancels. The safe node-share ceiling does not depend on the size of the security budget, only on the margin M and the ratio k_V. k_V is a network-maturity proxy: low when median honest work is small relative

to the pool (a thin, early network is cheaply farmed), rising as real usage deepens.

Concretely, k_V is computed from measured quantities each epoch. Saturating the boost to a fraction θ of $B_{\max}$ requires sustaining $g_i = \chi\,\kappa$ with $\chi = \theta/(1-\theta)$ (inverting workBoost $= B_{\max}\,g/(g+\kappa)$), hence a per-epoch burned gated volume of $\chi\,\kappa\,m(E)$. Annualized against the security budget,

$$k_V(E) = \frac{\chi\,\kappa\,m(E)\,E_{\mathrm{yr}}}{R_{\mathrm{yr}}(E)\,\bar{P}(E)},$$

where $E_{\mathrm{yr}} = 26$ is epochs per year, $R_{\mathrm{yr}}(E)$ the annualized HRTH emission at the current height (§2.2), and $\bar{P}(E)$ the epoch TWAP of the HRTH price. Every input is measured or protocol-known: $m(E)$ from §7.1, κ and $B_{\max}$ constants, $\bar{P}(E)$ an oracle TWAP like $\bar{p}_m(E)$. The convention $\theta = 0.8$ (so $\chi = 4$) is a protocol constant.

k_V (maturity)	$\phi_n^{\max}$ at $M = 3\times$	$\phi_n^{\max}$ at $M = 5\times$
0.71 (early)	54%	29%
1.00 (growing)	64%	47%
1.50 (deep)	73%	61%

Table 3: Safe node-share ceiling as a function of network maturity k_V and deterrence margin M.

Rule: launch node share low (30%, giving ≈5× margin at $k_V \approx 0.71$) and let governance raise it toward the $M \geq 3\times$ ceiling as measured k_V climbs, never past it. The ceiling is protocol-enforced, not advisory: the protocol recomputes $\phi_n^{\max} = 1 - \phi_v - M\,g_{\mathrm{frac}}/k_V$ each epoch from the measured k_V and rejects any governance-set split with ϕ_n above it. If a falling k_V pushes the ceiling below the current ϕ_n, the split is clamped to the ceiling from the next epoch, giving it the same mechanical status as the capacity clamp of §3.3: no vote can make self-dealing profitable, a constraint, not a policy. The protocol measures k_V each epoch; the split is a governed function, not a constant. Attempting to cover node costs by pushing ϕ_n toward 90% drives $\mathcal{D}$ below 1 (self-dealing becomes profitable) and is prohibited: cost coverage is the subsidy's job (§5.3).

8 Parameter reference

Illustrative economics: HRTH ≈ \$0.20, node cost ≈ \$0.17/1M tok, break-even Cred ≈ \$0.58/1M tok, market band \$0.12–0.90/1M, year-1 pool ≈ 6.4M HRTH (≈\$1.27M).

9 Optional: the stablecoin settlement rail and HRTH buyback

Creds are the default and only consensus-bearing settlement instrument. Two roles for external stablecoins are distinguished, because only one is safe by default.

The on-ramp is the default and always safe: a buyer pays in USDC/USDT, the gateway atomically swaps to the model's Cred and settles by burning it. The buyer experiences "pay a dollar, get an answer"; the protocol sees a burned Cred, so §3–§7 are untouched. This is the recommended universal payment experience.

The parallel rail with buyback is a governance option: nodes may also accept stablecoins directly, with a protocol cut ϕ_{dao} routed to the DAO in stablecoin and used to buy back HRTH on the open market. Because only external money can buy back HRTH (purchasing one nascent protocol token with another is circular) the buyback is the one mechanism that converts stablecoin inference demand into HRTH price support, complementing (not replacing) the burn's scarcity. The binding constraint: if stablecoin jobs earn workBoost, the DAO cut is the only deterrent on that rail, so it must be burn-equivalent,

$$\phi_{\text{dao}} \geq \phi_b^{\min} = \frac{M\, g_{\text{frac}}}{k_V} \quad (\approx 36\% \text{ at } M = 3,\ k_V = 0.71),$$

which is, conveniently, also the size that makes the buyback material. Annual buyback volume is $\phi_{\text{dao}} \cdot$(stablecoin revenue); whether it is significant against HRTH float is the empirical test that decides if the rail earns its complexity, or whether the network stays burn-and-on-ramp only until scale.

All figures illustrative and mutually consistent with §8. Formulas are the specification; constants are for governance and genesis to set. Cross-references (FPoS forging, verification committee, payer-gate provenance) are detailed in the Hearth whitepaper §4–§6.

References

[1] Aggelos Kiayias, Philip Lazos, and Paolo Penna. Single-token vs two-token blockchain tokenomics. In *6th Conference on Advances in Financial Technologies (AFT 2025)*, 2025.

[2] Helium. Helium documentation: Tokens and data credits. `https://docs.helium.com`, 2024.

[3] Render Network Foundation. Render network: Burn-and-mint equilibrium. `https://render.network`, 2023.

[4] DFINITY Foundation. Tokenomics of the internet computer. `https://learn.internetcomputer.org`, 2025.

[5] Aggelos Kiayias, Philip Lazos, and Jan Christoph Schlegel. Would Friedman burn your tokens? In *Financial Cryptography and Data Security (FC 2024)*, 2024.

Symbol	Meaning	Value	Kind
$C_{\max}$	HRTH hard cap	$B/0.03$ (illustrative 100,000,000)	derived (freeze)
B	total burn credits at freeze	turnout-dependent (illustrative 3,000,000)	derived (freeze)
P_{gen}	genesis premine (burn claim + DAO + team)	5% (illustrative 5,000,000)	genesis
C_{emit}	emission (forged)	95% (illustrative 95,000,000)	genesis
τ_b	block time	60 s	protocol
$T_{1/2}$	reward half-life	10 yr	genesis
N_{yr}	blocks per year	525,960	protocol
R_0	initial block reward	≈ 12.52 HRTH	derived
N_{ep}	Cred epoch	2 wk (20,160 blocks)	protocol
λ	Cred consumption-EMA rate	$(0, 1]$	DAO-set constant
γ	Cred growth allowance	≥ 0	DAO-set constant
$Q_m(0)$	initial Cred supply per model	DAO-set	governance
$\bar{p}_m(E)$	epoch TWAP of Cred-m price	measured	protocol
$\bar{P}(E)$	epoch TWAP of HRTH price	measured	protocol
$\phi_b/\phi_n/\phi_v$	retirement split	0.60 / 0.30 / 0.10	governance (clamped by $\phi_n^{\max}$)
$B_{\max}$	forging-boost cap	2 (2–3)	governance
α	workBoost EMA rate	$(0, 1]$	governance
κ	workBoost half-saturation	> 0	governance
τ_w	workBoost dust floor	> 0	governance
q	stake quantile for the median	1/2	governance
$b_{\min}$	minimum stake to serve	–	governance
N_{act}	position activation window	1000 blocks	protocol
τ_{unb}	exit delay	0	protocol
M	target deterrence margin	$\geq 3\times$	governance
σ_{adv}	assumed adversary stake share (ceiling calibration)	5%	governance
θ	boost-saturation convention for k_V	0.8	protocol
ω	wash friction on the recycled node share	≥ 0, taken as 0 (worst case)	market
penalty	response to proven cheating	generation-suspension	protocol

Table 4: Parameter reference. Constants marked genesis or governance are tunable; the formulas are the specification.