HashKinetics Yellowpaper

Formal Protocol Specification, v1.0

HashKinetics · companion to the whitepaper; where prose and specification disagree, this document governs

2026-08-18

Abstract

This document formally specifies the HashKinetics ledger: world state and its commitment; transaction syntax, authentication, and the state transition function Υ; MandateTree accounting; PayWord channels; the shielded pool including the mint and spend proof relations; recursive aggregation and per-block coverage; block structure, consensus signing layouts, and processing order; wire and digest encodings; the confidentiality layer and the offline disclosure verification algorithm; verifying-key pinning; and the consensus-critical invariants. Normative keywords follow RFC 2119. The reference implementation is the Rust workspace `chain/` together with the circuit crate `zkvm-bakeoff/circuit` — the *single hashing truth*, compiled into prover guests and linked by the state machine alike.

1 Notation

$\mathbb{B}$: byte strings; $a \,\|\, b$: concatenation. $\mathsf{H32}(tag, m_1, \ldots, m_k)$: SHAKE-256, 32-byte output, over $tag \,\|\, m_1 \,\|\, \cdots \,\|\, m_k$ with *tag* an ASCII string from the registry (§16); all chain-side hashing. $\mathsf{Hc}(d, m_1, \ldots, m_k)$: SHA-256 over $d \,\|\, m_1 \,\|\, \cdots$ with d a single domain byte; all in-circuit hashing (pool v1, rotatable per §13). $\mathsf{u64}^{\mathsf{LE}}, \mathsf{i64}^{\mathsf{LE}}$: 8-byte little-endian; $\mathsf{u32}^{\mathsf{BE}}, \mathsf{u64}^{\mathsf{BE}}$: big-endian where stated. `H256`: 32 bytes; identifiers (accounts, mandates, channels, assets) are `H256`; `Amount` = `u128` micro-units (10^6 per token); `Timestamp` = `u64` seconds. $\mathsf{JSONc}(x)$: canonical JSON of a fixed-field structure (declaration order, no maps, byte fields as lowercase hex) — used *only* for signing digests and transaction identifiers, never on the wire. $\Upsilon(\Sigma, T) \to (\Sigma', r)$ produces a post-state and a typed receipt $r \in \{\mathsf{ok}(events), \mathsf{rejected}(reason)\}$; a rejected transaction MUST leave Σ unmodified.

2 Cryptographic primitives

SHAKE-256 (FIPS 202) instantiates H32; distinct tags yield independent oracles; untagged hashing is forbidden. **SLH-DSA-SHAKE-192s** (FIPS 205): stateless root signatures (pk 48 B, sig 16,224 B; deterministic keygen from a 32-byte seed; non-hedged); roots sign only certificates. **LMS/HSS** (RFC 8554, SHAKE-256 sets): stateful consensus signatures, profile $[H10, H5]$ (2^{15} signatures); state advancement is durable before release (§10). **WOTS** (w=16): a 32-byte digest parses to 64 base-16 digits m_i plus checksum $C = \sum_i (15 - m_i)$ (3 digits; 67 chains). Signing reveals $\sigma_j = \text{chain}_j(sk_j, m_j)$; verification *recovers* the key: $\sigma'_j = \text{chain}_j(\sigma_j, 15 - m_j)$, $pk = \mathsf{Hc}(3, \sigma'_1 \,\|\, \cdots \,\|\, \sigma'_{67})$; the checksum blocks digit inflation; no public key is transmitted. **PayWord**: $w_0 = \mathsf{H32}(\texttt{payword-seed}, s)$, $w_i = \mathsf{H32}(\texttt{payword-link}, w_{i-1})$, tip w_n; revealing w_{n-k} proves k payments. **Lamport ratchet** (account tier v0): `auth_commit` = $\mathsf{H32}(\texttt{lamport-pk-commit}, pk)$ at index = nonce; a transaction reveals pk, signs the digest of §5, installs `next_auth` — leaf index = nonce as a ledger rule. **ML-KEM-768** (FIPS 203): ek 1184 B, ct 1088 B; epoch keygen $(ek_e, dk_e) = \text{KeyGen}(\mathsf{H32}(\texttt{mlkem-note-seed}, master \,\|\, \mathsf{u32}^{\mathsf{BE}}(e)))$; deterministic encapsulation; implicit-rejection decapsulation. **SHAKE-AEAD**: keystream $\mathsf{H32}(\texttt{note-enc}, k \,\|\, \nu \,\|\, ctr)$, tag $\tau = \mathsf{H32}(\texttt{note-mac}, k \,\|\, \nu \,\|\, c)$ (32 B), nonce ν = the note commitment; keys one-time, $k = \mathsf{H32}(\texttt{note-key}, ss \,\|\, ctx)$. **STARKs**: SP1 proofs over the circuit crate; modes core (≈2.7 MB) and compressed (≈1.24 MB). *F1 (normative): pairing-based wraps MUST NOT be emitted, transmitted, or accepted.* Interface: $\text{Verify}(vk, \pi, P) \in \{0, 1\}$ with P the byte-exact committed public string.

3 Protocol constants

Constant	Value	Meaning
M	10^6	micro-units per token
`POOL_DEPTH`	32	commitment tree depth (capacity $2^{32}-1$)
`ANCHOR_WINDOW`	128	blocks of valid anchors
`SPEND_TREE_DEPTH`	10	statement-side spend tree ($2^h \le 2^{10}$ wallet capacity)
`EPOCH_BLOCKS`	1,000	KEM-key epoch length
`MAX_TXS_PER_BLOCK`	256	batch capacity (configuration tier)
`EK_LEN/CT_LEN/TAG_LEN`	1184/1088/32	ML-KEM sizes; AEAD tag
WOTS	w=16, 67 chains	§2
HSS profile	$[H10, H5]$	≈32.7k signatures per operational key
SLH-DSA-192s	pk 48 B, sig 16,224 B	root tier

4 World state

$\Sigma = (\mathrm{Acc}, \mathrm{Man}, \mathrm{Chan}, \mathrm{Pool}, \mathrm{Val})$ with

- $\mathrm{Acc} : \mathbb{A} \to (\mathit{bal} : \mathit{Asset} \to \mathit{Amount},\ \mathit{nonce},\ \mathit{auth\_commit})$;
- $\mathrm{Man} : \mathbb{A} \to (\mathit{parent}?, \mathit{holder}, \mathit{asset}, \mathit{rate\_per\_sec}, \mathit{buffer}, \mathit{buffer\_max}, \mathit{per\_tx\_max}, \mathit{last\_accrual}, \mathit{expiry}, \mathit{revoked}, \mathit{tier})$; roots bind a funding account;
- $\mathrm{Chan} : \mathbb{A} \to (\mathit{payer}, \mathit{payee}, \mathit{asset}, \mathit{mandate}, \mathit{tip}, \mathit{unit\_price}, \mathit{max\_steps}, \mathit{highest\_step\_settled}, \mathit{escrow\_remaining}, \mathit{expiry})$;
- $\mathrm{Pool} = (F : \mathrm{Frontier}_{32},\ A : \text{ring of} \le 128 \text{ anchors},\ N \subseteq \mathtt{H256},\ \mathit{total\_shielded},\ \mathit{asset}?,\ \mathit{version})$;
- Val: per validator, stable address, SLH-DSA `root_pk`, operational `op_pk`, epoch, power.

Transient, excluded from commitment: `agg_cover` (§9). **State commitment** $C(\Sigma) = \mathsf{H32}(\mathtt{state\text{-}commit}, \mathrm{enc}(\Sigma))$ over a canonical encoding of the non-transient components (v1 whole-state; merkleization is a scheduled same-interface replacement).

5 Transactions

Envelope. `SignedTx` $= \{\mathit{sender}, \mathit{nonce}, \mathit{payload}, \mathit{next\_auth}, \mathit{lamport\_pk}, \mathit{sig}\}$. Signing digest $d(T) = \mathsf{H32}(\mathtt{hk/v1/tx}, \mathsf{JSONc}(T \setminus \mathit{sig}))$; $\mathrm{txid}(T) = \mathsf{H32}(\mathtt{hk/v1/txid}, \mathsf{JSONc}(T))$ — both wire-codec-independent (Invariant 9).

Syntax.

```
Tx ::= Transfer      { to, asset, amount }
     | MandateCreate { id, parent?, holder, asset, rate_per_sec, buffer_max,
                       per_tx_max, initial_buffer, expiry, tier }
     | MandateSpend  { leaf, to, amount }
     | MandateRevoke { target }
     | ChannelOpen   { id, mandate, payee, asset, tip, unit_price, max_steps, expiry }
     | ChannelSettle { id, word, step }
     | ChannelRefund { id }
     | MintToPool    { asset, value, commitment, proof, stealth_ct }
     | ShieldedSpend { anchor, nullifier, out_commitment, out2_commitment,
                       fee, credit?, mandate?, proof, stealth_ct, stealth_ct2 }
```

`stealth_ct*` are *advisory*: stored and gossiped, never interpreted by consensus (Invariant 8).

Envelope validity (all payloads): (E1) Acc[*sender*] exists; (E2) *nonce* = Acc[*sender*].*nonce*; (E3) $\mathsf{H32}(\mathtt{lamport\text{-}pk\text{-}commit}, \mathit{lamport\_pk}) = \mathit{auth\_commit}$; (E4) *sig* verifies over $d(T)$. Success advances the ratchet: *nonce*+=1, *auth_commit* ← *next_auth*.

6 Mandate semantics

Accrual (drip-not-reset).

$$\mathrm{avail}(n, t) \;=\; \min\big(\mathit{buffer\_max},\ \mathit{buffer} + \mathit{rate\_per\_sec} \cdot (t - \mathit{last\_accrual})\big).$$

Authorization walk. For leaf ℓ, amount a, time t, with $\mathrm{chain}(\ell) = [\ell, \mathrm{parent}(\ell), \ldots, \mathrm{root}]$, $\mathrm{check}(\ell, a, t)$ fails with the *first* violation at its depth k (0 = leaf): $\mathsf{Revoked}(k)$; $\mathsf{Expired}(k)$ if $t \ge \mathit{expiry}$; $\mathsf{PerTxCap}(k)$ if

$a > \mathit{per\_tx\_max}$; $\mathsf{Buffer}(k)$ if $a > \mathrm{avail}(n_k, t)$ — rendered `insufficient buffer at depth k from leaf (have ..., need ...)`.

Rules. *Create*: root ⇒ sender becomes holder and funding account; child ⇒ sender MUST hold the parent and $\mathit{per\_tx\_max}$(child) ≤ $\mathit{per\_tx\_max}$(parent) (attenuation at creation); sibling oversubscription of buffers is legal by design. *Spend*: sender MUST hold the leaf; check first; funds move from the *root's* funding account; every node on the chain draws down; two-phase — no partial effects. *Revoke*: sender MUST hold the target's parent (root: itself); cascade is implicit.

7 Channels

Open: sender = leaf holder; $\mathit{escrow} = \mathit{unit\_price} \times \mathit{max\_steps}$ drawn through the walk once, at open; `id` MUST equal the derived channel id; `tip` anchors the chain. *Settle* (proof-carrying, any sender): for $s > \mathit{highest}$, verify $s - \mathit{highest}$ PayWord links from `word` to the last settled word (or tip); pay $(s - \mathit{highest}) \cdot \mathit{unit\_price}$ from escrow. *Refund*: after expiry, payer reclaims the remainder. Reference measurement: 1,000 steps settled by one 32-byte word in one transaction.

8 The shielded pool

Note and commitment. $\mathit{note} = (v : \mathsf{u64}, \mathit{tag}, \rho, \mathit{rcm})$; $\mathit{cm} = \mathsf{Hc}(1, \mathsf{u64}^{\mathsf{LE}}(v) \,\|\, \mathit{tag} \,\|\, \rho \,\|\, \mathit{rcm})$.

Frontier insertion. With empty ladder $E_0 \ldots E_{32}$ ($E_{l+1} = \mathsf{Hc}(2, E_l \,\|\, E_l)$), inserting at $\mathit{next\_index}$ updates one frontier node per level and recomputes the root in ≤ 32 steps of $\mathsf{Hc}(2, L \,\|\, R)$. Every block end (and genesis) seals the root into A (dedup; evict beyond 128).

Nullifiers. $\mathit{nf} = \mathsf{Hc}(4, \mathit{nk} \,\|\, \rho)$ with nk secret to the owner (public fingerprint $\mathsf{Hc}(10, \mathit{nk})$). N is append-only forever.

Address tag and spend tree. $\mathit{tag} = \mathsf{Hc}(9, \mathit{spend\_root} \,\|\, \mathsf{Hc}(10, \mathit{nk}))$; the spend tree is depth-10 over WOTS pk digests ($\mathsf{Hc}(8, \cdot \,\|\, \cdot)$ inner nodes), wallet capacity 2^h with empty-ladder padding.

The spend relation. Public $P = (\mathit{anchor}, \mathit{nf}, \mathit{cm}_1, \mathit{cm}_2, \mathit{fee}, b)$; the witness w comprises the input note and its Merkle path, the WOTS signature with spend-tree path and leaf index, the nullifier key nk, the two output notes, and the credit account. $R_{\mathrm{spend}}(P, w) = 1$ iff

$$\mathit{cm}_{\mathrm{in}} = \mathsf{Hc}(1, \mathit{note}_{\mathrm{in}}), \qquad \mathrm{Fold}_{32}(\mathit{cm}_{\mathrm{in}}, \mathit{path}) = \mathit{anchor}; \tag{1}$$

$$\mathit{pk} = \mathrm{Recover}(\sigma, d_P), \qquad \mathrm{Fold}_{10}(\mathsf{Hc}(3, \mathit{pk}), \mathit{ots\_path}) = \mathit{spend\_root}; \tag{2}$$

$$\mathit{note}_{\mathrm{in}}.\mathit{tag} = \mathsf{Hc}(9, \mathit{spend\_root} \,\|\, \mathsf{Hc}(10, \mathit{nk})), \qquad \mathit{nf} = \mathsf{Hc}(4, \mathit{nk} \,\|\, \mathit{note}_{\mathrm{in}}.\rho); \tag{3}$$

$$\mathit{cm}_1 = \mathsf{Hc}(1, \mathit{out}_1), \quad \mathit{cm}_2 = \mathsf{Hc}(1, \mathit{out}_2), \qquad \mathit{note}_{\mathrm{in}}.v = \mathit{out}_1.v + \mathit{out}_2.v + \mathit{fee}; \tag{4}$$

$$b = \mathsf{Hc}(7, \mathit{credit} \,\|\, \mathit{fee}). \tag{5}$$

The mint relation. $R_{\mathrm{mint}}((\mathit{cm}, v), (\mathit{tag}, \rho, \mathit{rcm})) = 1$ iff $\mathit{cm} = \mathsf{Hc}(1, \mathsf{u64}^{\mathsf{LE}}(v) \,\|\, \mathit{tag} \,\|\, \rho \,\|\, \mathit{rcm})$ — the inflation guard.

Expected publics (chain-derived). $\hat{P}$(`MintToPool`) = ($\mathit{commitment}, \mathit{value}$); $\hat{P}$(`ShieldedSpend`) = ($\mathit{anchor}, \mathit{nullifier}, \mathit{cm}_1, \mathit{cm}_2, \mathit{fee}, \mathsf{Hc}(7, \mathit{credit} \,\|\, \mathit{fee})$). Verification compares the proof's committed bytes to bincode($\hat{P}$) *byte-exactly*, then checks the STARK (Invariant 4).

Transition rules. *MintToPool*: (M1) capacity; (M2) asset matches (first mint pins); (M3) balance ≥ value; (M4) proof valid for $\hat{P}$ **or** coverage (§9). Effects: debit; insert; $\mathit{total\_shielded} += \mathit{value}$. *ShieldedSpend*: (S1) $\mathit{anchor} \in A$; (S2) $\mathit{nf} \notin N$; (S3) $\mathit{fee} > 0 \Rightarrow \mathit{credit}$ present; (S4) if `mandate`: envelope sender = leaf holder and check($\mathit{leaf}, \mathit{fee}, t$) (public-skeleton mode); (S5) proof valid for $\hat{P}$ or coverage. Effects, in order: mandate draw-down → $N \cup= \{\mathit{nf}\}$ → insert $\mathit{cm}_1, \mathit{cm}_2$ → if $\mathit{fee} > 0$: credit transparently, $\mathit{total\_shielded} -= \mathit{fee}$.

9 Aggregation and coverage

With kinds `KIND_SPEND` = 1, `KIND_MINT` = 2 and vkbytes(vk) the verifying key's 8 hash words big-endian:

$$\mathit{leaf}_i = \mathsf{Hc}\big(\mathit{kind}_i \,\|\, \mathrm{vkbytes}(\mathit{vk}_i) \,\|\, \mathsf{u32}^{\mathsf{BE}}(|P_i|) \,\|\, P_i\big), \qquad \mathit{digest} = \mathsf{Hc}\big(\mathsf{u32}^{\mathsf{BE}}(n) \,\|\, \mathit{leaf}_1 \,\|\, \cdots \,\|\, \mathit{leaf}_n\big),$$

binding kind, key, content, order, and count. The **aggregator statement** verifies, in-zkVM, verify(vk_i, SHA256(P_i)) for each of n compressed proofs and commits *digest*; the emitted aggregate is itself a compressed raw STARK (F1 applies). **Block rule**: at commit, a present `agg_proof` is verified once against the digest recomputed from the block's proof-less pool transactions in order, with each $P_i = \hat{P}(\mathit{tx}_i)$; success installs `agg_cover` = {$\mathsf{H32}$(`agg-cover`, $\mathit{kind} \,\|\, \mathrm{bincode}(\hat{P})$)}. A proof-less pool transaction is valid iff its key ∈ `agg_cover`; the set clears at block end (Invariant 6); the per-proof path stays legal.

10 Blocks and consensus interface

Batch. {*parent_app_hash*, *txs*, *rotations*, *agg_proof*}, bincode-encoded; the consensus value id is H32(`block-value`, *batchby*
— *a vote for a block is a vote for its parent state* (Invariant 2).

Processing order (normative). (1) *parent_app_hash* = $C(\Sigma)$ or consensus-fatal; (2) verify `agg_proof`, install coverage; (3) apply `txs` in order; (4) apply rotations; (5) seal the anchor; (6) clear coverage; (7) publish $C(\Sigma')$.

Sign-bytes. Votes (`DOM_SIGN_VOTE`):

```
tag || 0x00 || type(1B: 0=prevote,1=precommit) || u64LE(height) || i64LE(round)
    || (0x00 | 0x01 || value_id) || validator_address
```

Proposals (`DOM_SIGN_PROPOSAL`): `tag||0x00||u64LE(height)||i64LE(round)||i64LE(pol_round)||value_id`. Parts (`DOM_SIGN_PART`): `tag||0x00||` part tag 0 (Init: height, round, pol_round, proposer) / 1 (TxBatch chunk: u64LE length || bytes) / 2 (Fin: value_id). The Fin carries only the value id: proposal authenticity is the engine's signed proposal — *one key, one signer* (Invariant 3).

Rotation. `RotationCert` = $\{root\_pk, new\_op\_pk, epoch, valid\_from, \sigma_{\text{root}}\}$: valid iff the root is the validator's registered root, the epoch strictly increases, and the SLH-DSA signature verifies; on commit all nodes swap `op_pk` and the owner swaps its live signer in the same commit.

Reserve-then-sign (normative). A signer MUST durably persist *used* $\|$ *state* (write, fsync, atomic rename) before releasing any signature, and MUST resume from it on restart. Observed one-time-leaf reuse is equivocation evidence.

11 Wire and digest encodings

Consensus wire and WAL: bincode over fixed-field structures (deterministic, no maps). Byte fields are *format-aware*: raw in binary formats; lowercase hex in human-readable surfaces (RPC, receipts, genesis). $d(T)$, txid, and all sign-bytes are codec-independent (Invariant 9); a codec change alters wire/WAL compatibility only.

12 Confidentiality layer

Seal (sender), for output o to (tag, ek_e): $(ss, ct_{\text{kem}}) = \text{Encaps}(ek_e)$; $k = \mathsf{H32}(\texttt{note-key}, ss \,\|\, ctx)$; $c \,\|\, \tau = \text{Seal}(k, \nu{=}cm(o), o \,\|\, memo)$; publish $ct_{\text{kem}} \,\|\, c \,\|\, \tau$. **Scan** (recipient), per entry $(i, cm, blob)$: split at `CT_LEN`; decapsulate (implicit rejection); derive k'; if Open succeeds, parse the note and *require* $\mathsf{Hc}(1, note) = cm$ — the lying-ciphertext rule. $\mathbf{IVK}_e = dk_e$ (+ context): one epoch's discovery+decryption; no spend authority, no *nk*. **Disclosure package** $D = \{chain\_id, k, payload, cm, index, path, anchor\}$; Verify($D$), offline and pure: (1) Open($k, cm, payload$); (2) $\mathsf{Hc}(1, note) = cm$; (3) $\text{Fold}_{32}((cm, index), path) = anchor$. One key opens exactly one payment.

13 Verifying-key pinning and version pools

Genesis MAY (mainnet: MUST) carry `vk_pins` = {spend, mint, agg}, each = hex(H32(`vk-pin`, vkbytes)); a verifying node MUST refuse to run on mismatch. Statement changes ship as new pool versions with fresh pins; old-version notes migrate by spending into the new pool.

14 Consensus-critical invariants

Invariant 1 (Determinism). Υ *and block processing are pure in* $(\Sigma, block)$*; honest replicas replay to identical* $C(\Sigma)$.

Invariant 2 (Parent binding). *The value id commits to parent_app_hash; divergence cannot gather votes.*

Invariant 3 (Single signer). *One consensus signer per node; one-time-leaf reuse is equivocation.*

Invariant 4 (Chain-derived statements). *Proofs verify only against* $\hat{P}$ *computed from transaction fields.*

Invariant 5 (Conservation). *Transparent supply +* `total_shielded` *is preserved by every rule.*

Invariant 6 (Coverage locality). `agg_cover` *never survives its block nor enters* $C(\Sigma)$.

Invariant 7 (Nullifier permanence). N *is append-only; anchor eviction never touches it.*

Invariant 8 (Advisory blobs)**.** *No consensus rule depends on stealth-ciphertext contents.*

Invariant 9 (Codec independence)**.** *Signatures and identifiers are invariant under wire-codec change.*

Invariant 10 (Atomic refusal)**.** *A rejected payload mutates nothing; nullifiers burn only on success.*

15 Receipts (normative strings, excerpt)

`ok: <n> event(s)` · `rejected: mandate: insufficient buffer at depth <k> from leaf (have <a>, need <b>)` · `rejected: mandate: per-tx cap at depth <k>` · `rejected: mandate: revoked at depth <k>` · `rejected: nullifier already spent (double spend)` · `rejected: unknown anchor` · `rejected: pool proof rejected (invalid, or no verifier wired)` · `rejected: fee requires a credit account` · channel and envelope analogues per the reference implementation's typed error set.

16 Domain registries

Chain-side string tags (SHAKE-256; normative list in `hk-crypto::hash`): `hk/v1/tx`, `hk/v1/txid`, `hk/v1/state-commit`, `hk/v1/account-id`, `hk/v1/mandate-id`, `hk/v1/channel-id`, `hk/v1/delegation-cert`, `hk/v1/payword-seed`, `hk/v1/payword-link`, `hk/v1/lamport-{sk,leaf,pk-commit}`, `hk/v1/mlkem-note-seed`, `hk/v1/note-{key,enc,mac}`, `hk/v1/agg-cover`, `hk/v1/vk-pin`, plus block-value and consensus signing tags.

In-circuit byte domains (SHA-256): 1 note commitment · 2 commitment-tree node · 3 WOTS pk digest/owner · 4 nullifier · 5 WOTS chain step · 6 WOTS sk derivation (host) · 7 tx binding · 8 spend-tree node · 9 address tag · 10 nk commitment · 11 nk seed (host) · 12 per-leaf WOTS seed (host).

HashKinetics yellowpaper v1.0 (2026-08-18). Normative companions: the reference implementation and its test suite; `docs/SHIELDED-POOL-SPEC.md` (narrative protocol detail); the whitepaper (motivation, economics, history).