

 Rand Protocol

Rand Protocol

Private computation, hash-based proofs

Shielded notes, a RISC-V zkVM proved by a hash-based STARK with no trusted setup, and chained HotStuff BFT. For due diligence.

Technical overview · Version 6 · October 2026

Live public test network (chain 20, release v0.6.8, since 1 October 2026).
Not a mainnet; all validators are run by the core team.

Rand makes privacy the default and proves every transaction with hash-based STARKs

A layer-1 chain whose transactions are zero-knowledge proofs; live as a public test network (chain 20) since 1 October 2026.

Private by default

Value exists only as notes in a shielded pool, with no accounts or balances. A transfer publishes nullifiers and commitments, not sender, recipient or amount. **Cost:** fees, timing and bridge amounts stay public.

General programs, private inputs

A RISC-V (RV32IM) zkVM runs programs off-chain; the chain checks the proof and eight public output words. A transfer is such a program too, with a small auth proof. **Cost:** calls are limited to tier 14, a size class of under 16,383 cycles (slide 6).

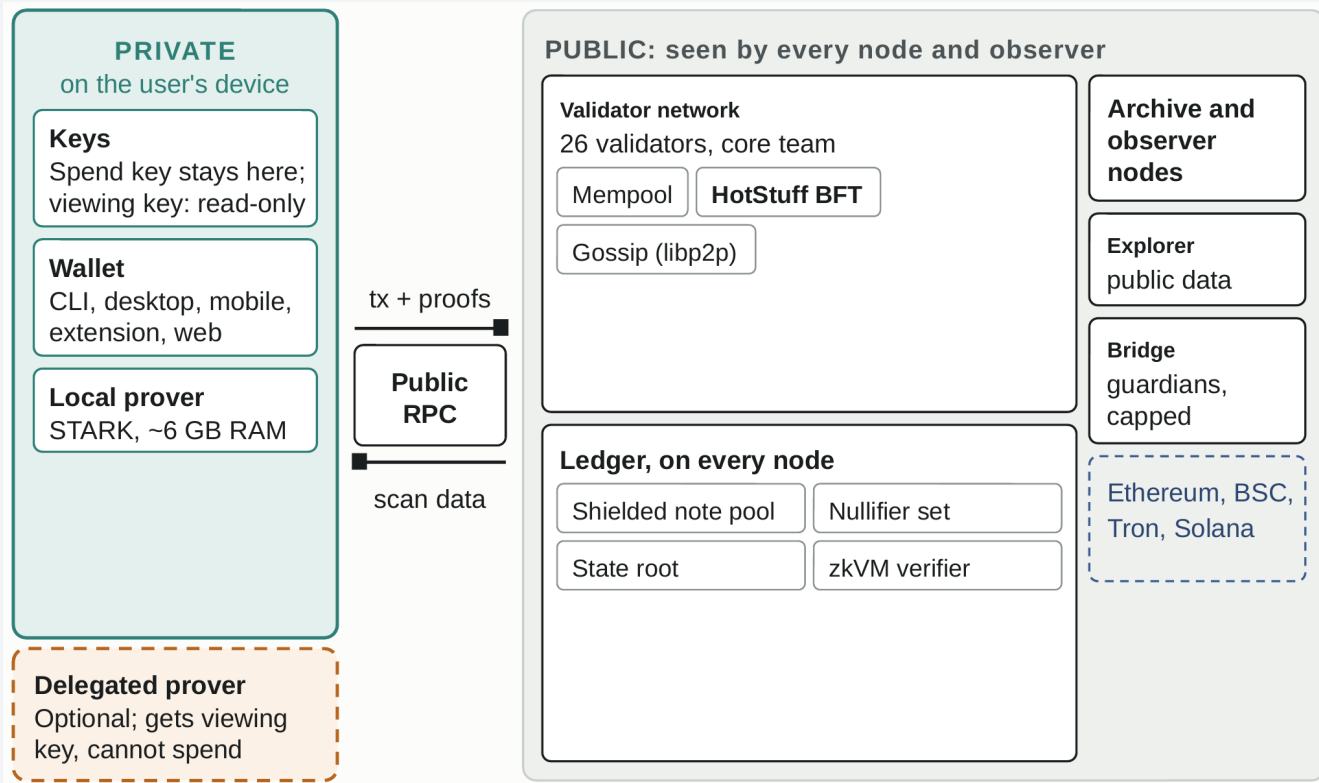
Hash-based proofs

A Plonky3 batch STARK over the Goldilocks field with FRI and Poseidon2: hashes and field arithmetic, with no elliptic-curve or pairing assumption in the proof system. **Cost:** about 1.5 MB per proof, about 2.9 MB per transfer.

No trusted setup

No ceremony and no secret parameters: every verifier key derives from public values. **Cost:** the proof rules are still changing, and each change so far shipped as a new chain (a hard fork).

Private work happens on the user's device; the public network only verifies proofs



READING THE DIAGRAM

- The spend key never leaves the wallet; an optional delegated prover (slide 16) gets the viewing key and cannot spend.
- The RPC passes proofs to the mempool; HotStuff BFT orders them; every node verifies each proof.
- All 26 validators are run by the core team today.
- A capped guardian bridge links to Ethereum, BSC, Tron and Solana; the core team runs it, administrator keys included.
- Observers feed the explorer, a public index of chain data.

Teal: on the user's device. Grey: visible to every node and observer. Chain 20, 1 October 2026; 6 GB peak measured on one laptop.

The ledger hides owner, amount and the note a spend consumes; fees, timing and bridge amounts stay public

No accounts, balances or nonces exist: value is notes, and a note is only a commitment until its holder spends it.

HIDDEN ON THE LEDGER

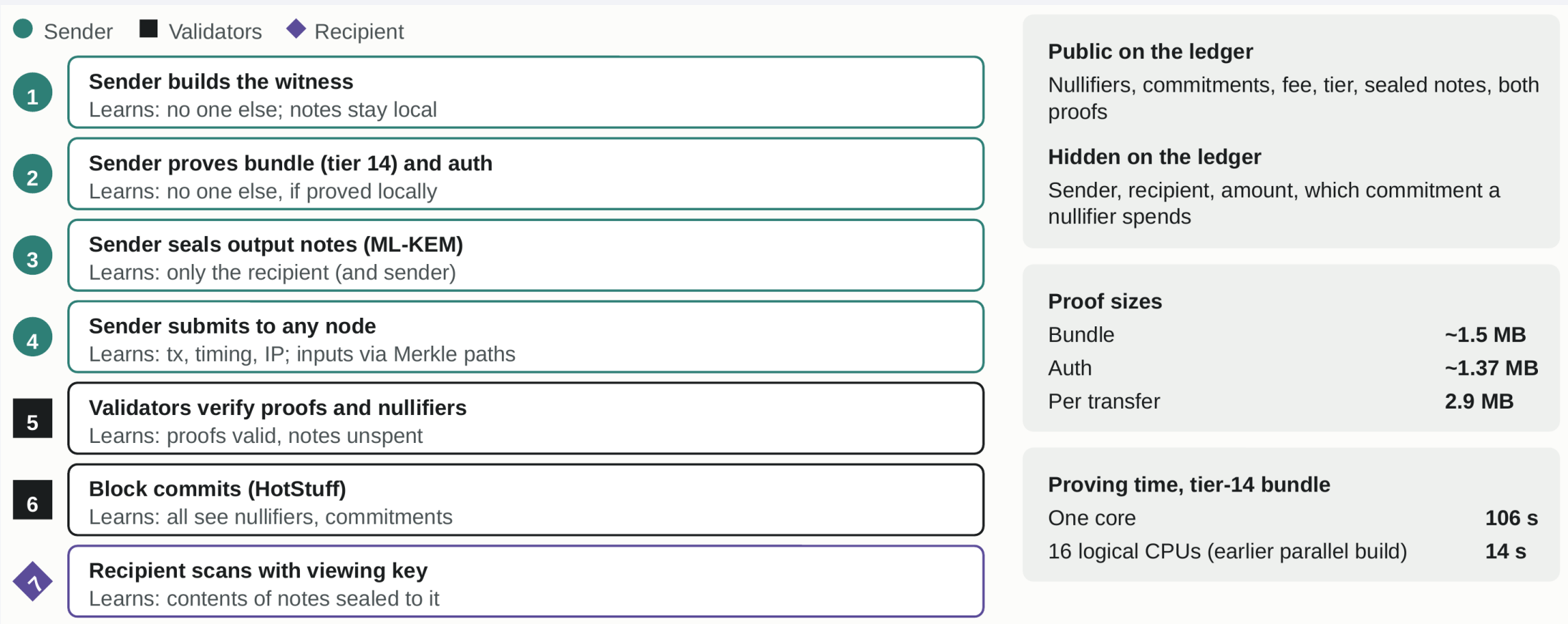
- **Owner, amount and asset of each note:** only a Poseidon2 commitment in a depth-32 tree is stored.
- **Which note a spend consumes:** a published nullifier does not link to its commitment without the viewing key.
- **Sender, recipient and amount:** a node's view of a transaction has no from, to, nonce or amount field.
- **Shape and memos:** every bundle has four inputs and four outputs; every note envelope is a fixed 1,860 bytes on chain 20.
- **Key holders only:** the recipient opens amount, asset and the sender's owner field; a viewing key opens a whole history, read-only.

PUBLIC, OR VISIBLE TO THE NODE A WALLET USES

- **Nullifiers**, which stop double spends, and **note commitments**, which feed the tree.
- **The fee**, whose floor differs by action: a transfer differs from a program deploy or call.
- **Submission timing**, and the network address of a wallet that submits or scans.
- **Merkle paths:** a node that serves them sees which note a wallet spends; the command-line wallet computes them locally.
- **Bridge amounts**, the validator register and stake bonds, program code, and program payouts (recipient, amount).
- **A call proof's header:** its tier and, roughly, whether the hash tables were used.

The sender proves a private transfer before submitting it; every validator then checks it

Hidden means hidden on the ledger. Times measured on one 16-core laptop-class machine, not optimised; the 14 s figure is from an earlier parallel build.



The zkVM proves RISC-V programs in padded tiers, so cost follows the tier and not the cycles used

Chain 20 settings, live public test network, 1 October 2026.

- **Guests.** A guest is an RV32IM program (32-bit RISC-V with integer multiply and divide). Its identity is a Poseidon2 digest computed inside the proof; a run ends in eight public output words.
- **Tiers.** Execution is padded to a tier t , from 10 to 20 in steps of 2, which allows up to $2^t - 1$ cycles. The proof shows t , not the number of cycles used.
- **Two proofs per transfer.** The bundle guest (tier 14, four inputs and four outputs) and a small auth guest (tier 10), which is the only proof that carries the spend key.
- **Calls.** Calls to deployed programs are limited to tier 14 today (under 16,383 cycles, digest rows included), so only small programs are admitted.
- **Constraint sets.** The proof rules are versioned. Set 8 runs on chains 18 to 20; each of seven changes in September 2026 needed a new chain.

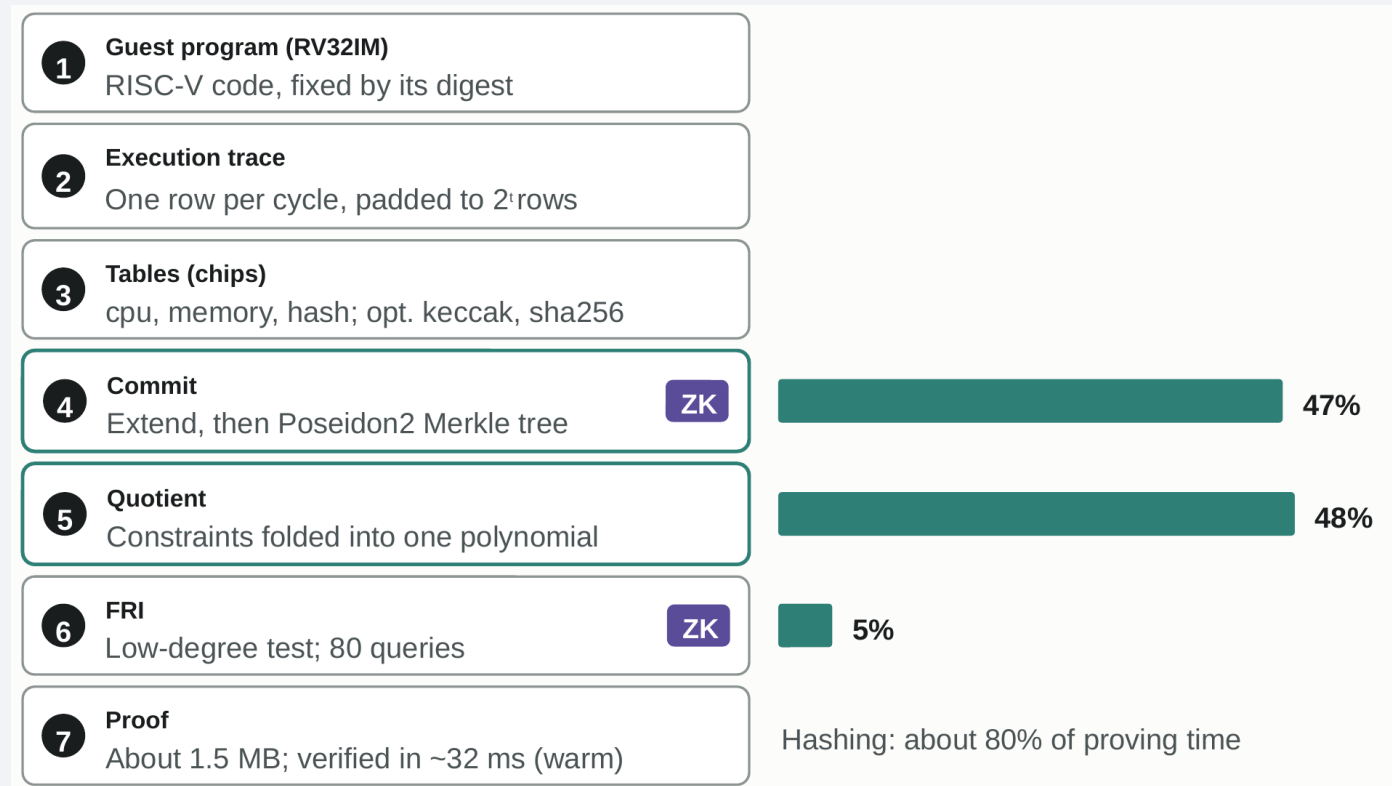
TIERS · MAX CYCLES

TIER	MAX CYCLES	USED BY
10	1,023	Auth guest
12	4,095	
14	16,383	Bundle guest; today's call cap
16	65,535	
18	262,143	
20	1,048,575	

Why padded tiers

Padding hides how much work a run did: the proof shows only its tier. A short program pays for its whole tier.

Transparent STARK proofs need no trusted setup, at the cost of megabyte-sized proofs



Plonky3 0.7.0, Goldilocks, blowup 8, 80 queries, 20 grinding bits, Poseidon2. Bars: share of one-core proving time of a tier-14 bundle, one 16-core laptop-class machine, not optimised.

GIVES, AND DOES NOT GIVE

- **Gives:** no trusted setup; hashes and field arithmetic only; verifier keys derive from public values.
- **Gives:** zero knowledge from salted, hiding commitments; a warm check of about 32 ms per proof.
- **Does not give:** perfect zero knowledge. It is statistical; each proof shows its tier and table heights.
- **Does not give:** small proofs. About 1.5 MB each, set by 80 queries and trace width.
- **FRI stage, classical:** about 86 bits proven (66 from queries, 20 grinding), about 100 conjectured. No overall level is claimed.

MEASURED COST

A proof takes seconds to minutes to make, is about 1.5 MB and takes about 32 ms to verify

Measured on one 16-core laptop-class machine, not optimised. Parallel row: earlier build that could stall (1 run in 20), not re-measured.

MEASURE	BUNDLE PROOF (TIER 14)	AUTH PROOF (TIER 10)	ONE TRANSFER (BOTH)
Proof size	1.50 MB	1.37 MB	about 2.9 MB
Prove, one core	106 s	6.8 s	about 113 s
Prove, parallel prover, 16 logical CPUs (earlier build)	13.9 s	about 1.3 s	about 15 s
Peak prover memory, one core	5.98 GB	0.39 GB	not summed
Verify, warm key	about 32 ms	about 29 ms	about 61 ms (sum)
Verify, first use (key build)	4.5 s one core, 1.5 s parallel	0.33 s	once per node start
Earlier records (8-vCPU cloud machine, older prover and constraint sets; not comparable)	217–237 s single-threaded, 69–73 s parallel	n/a	n/a
GPU proving	Design, not yet active; no GPU time measured	n/a	n/a

Consensus is chained HotStuff: stake-weighted votes, a three-chain commit and a timeout pacemaker

One round: propose, vote, certify

Leader of view v
validators[$v \bmod 26$]:
round robin; proposes
 $B(v)$ with its parent's
QC

1
block

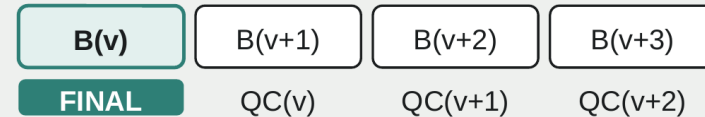
Every validator
Checks the vote rule,
flushes state to disk,
signs a Dilithium2 vote

2
votes

QC(v)
Votes of over 2/3 of
stake: 18 of 26; 18
signatures, about 68
KB

3 QC(v) lets the leader of $v+1$ propose $B(v+1)$

Commit: three consecutive QCs



Consecutive views: $B(v)$ is final. The lock moves to QC($v+1$). A skipped view breaks the chain.

Leader fails: the pacemaker

- a** Timer fires: 3 s by default, doubling up to 8x.
- b** Each validator signs a timeout vote for $v+1$ with its highest QC.
- c** Over 2/3 of stake (a timeout certificate): all enter view $v+1$.
- d** That leader proposes from its highest QC.

Safety rules per validator

One vote per view.
Lock rule: vote only if the block extends the locked block, or its QC is newer than the lock.
Safety state is flushed to disk before each vote leaves the node.

This test network, 1 Oct 2026

Validators 26, equal stake
Quorum 18 of 26 (over 2/3)
Safety Holds with under 1/3 Byzantine: at most 8
Progress Needs 18 online, timely network
Run by Core team; no slashing active

The core team runs all 26 validators today; the code supports an epoch register, admission votes and slashing

TRUE TODAY (CHAIN 20, 1 OCTOBER 2026)

- **26 validators of equal stake**, all run by the core team. A quorum is 18 of 26, so the core team alone can finalise any block.
- **The fault bound** (under one third Byzantine stake) holds only through the core team's custody of all 26 keys.
- **New validators** need a signed vote of more than two thirds of the current set: the register is permissioned in practice.
- **Slashing is off** on chain 20: a misbehaving validator loses no stake; the remedy is operator action.
- **No date** is given for a wider set; slashing is advised only once stake is held by more than one operator.

WHAT THE CODE SUPPORTS

- **A public validator register.** Each epoch of 1,000 blocks recomputes the set from it: up to 100 validators, sorted by address.
- **Stake is bonded by burning notes;** new weight activates two epochs later and unbonding takes two epochs.
- **Weight caps:** one third per validator; new weight entering at an epoch boundary is capped at 25 per cent of active weight.
- **A leader's second block** for one view is refused; slashing and jailing for it exist behind a genesis switch, off today.
- **Leaders rotate** round robin in address order. A sortition beacon is a research direction, not built.

Nodes gossip over libp2p, verify every proof themselves and keep consensus state in one hashed root

Network

libp2p 0.57 over TCP with Noise and Yamux. Gossipsub topics per chain for consensus, transactions, status and peer bindings; Kademlia discovery from a bootstrap list. Connections, gossip rates and the mempool have configured limits.

Verification and sync

Cheap checks come first (size, anchor window, nullifiers); proofs are verified last, on 4 workers. A lagging node fetches up to 100 blocks with their certificates and re-executes them, proofs included.

Consensus state

Commitment-tree frontier, nullifier set, validator register and program set, plus genesis-gated sections, under one BLAKE3 state root. The pool's leaves sit in node storage, outside consensus state.

Roles and history

Validator, observer, archive or public RPC front. RocksDB with one fsynced batch per block. Most test-network validators keep one day of blocks and two archives keep all (chain 18 record).

By the block cap, proof size limits a block to about seven transfers; no sustained-load test exists

Chain 20 settings, 1 October 2026. Derived ceilings; proof times and 32 ms measured on one 16-core laptop-class machine, not optimised.

QUANTITY	VALUE	BASIS
Bytes per transfer	Bundle 1.50 MB + auth 1.37 MB = about 2.9 MB; rest about 8 KB	Measured (proofs); derived (rest)
Block cap, proof cap	20 MiB per block, 4 MiB per proof	Chain 20 genesis (testnet settings)
Transfers per full block	About 7 (20,971,520 / 2,869,328 = 7.3)	Derived
Transfer ceiling	Arithmetic, not a measured rate: about 5 to 7 per second (1 s minimum spacing; 1.2–1.4 s recorded on chains 14 and 18, none for 20). Every node must receive 20 MiB a block; bandwidth untested	Derived, not measured
Verifying a full block	14 proofs × 32 ms = about 0.45 s	Derived; not the limit
Proving one transfer	About 15 s, parallel prover on 16 logical CPUs (earlier build, not re-measured)	Measured; done by senders, in parallel
Data if every block were full	About 1.5 TB per day at about 1.2 s a block (1.8 TB at 1 s)	Derived; why pruning matters
Design target	Block-level aggregation: not active, and not yet feasible at scale on the memory measured	Specified; see next slide

The scaling path is designed, not active: proof bytes bound a block today and aggregation is off

	Today: live public test network	Designed future: none of it active yet
Prove	Wallet, or a delegated prover ACTIVE TODAY Bundle 1.50 MB (about 14 s*, earlier build), auth 1.37 MB. Core-team pool live on chain 19; sees history, cannot spend	GPU provers DESIGNED CUDA crate: transforms and hashing, about 87% of CPU time (hashing alone 80%). No GPU timing yet
Combine	No aggregation on any chain NONE TODAY Measured limit: one aggregate over one bundle proof took 8,131 s and 377 GB (one thread)	Aggregators: permissionless, GPU hosts DESIGNED One recursive proof over N bundle proofs; larger N needs a design change; validators stay CPU-only
History	Archive nodes keep both proofs ACTIVE TODAY About 2.9 MB of proof bytes per transfer; nothing is sealed yet	Stage 1: sealing DESIGNED Nodes could drop covered bundle proofs: 2.9 MB to about 1.4 MB. Not demonstrated; the one aggregate measured was 1.56 MB
Block	Block cap 20 MiB: about 7 transfers ACTIVE TODAY 2.9 MB per transfer; every validator checks every proof (about 32 ms* each)	Stage 2: aggregate before inclusion PLANNED Only the aggregate enters the block: a future decision; block bytes unchanged until then

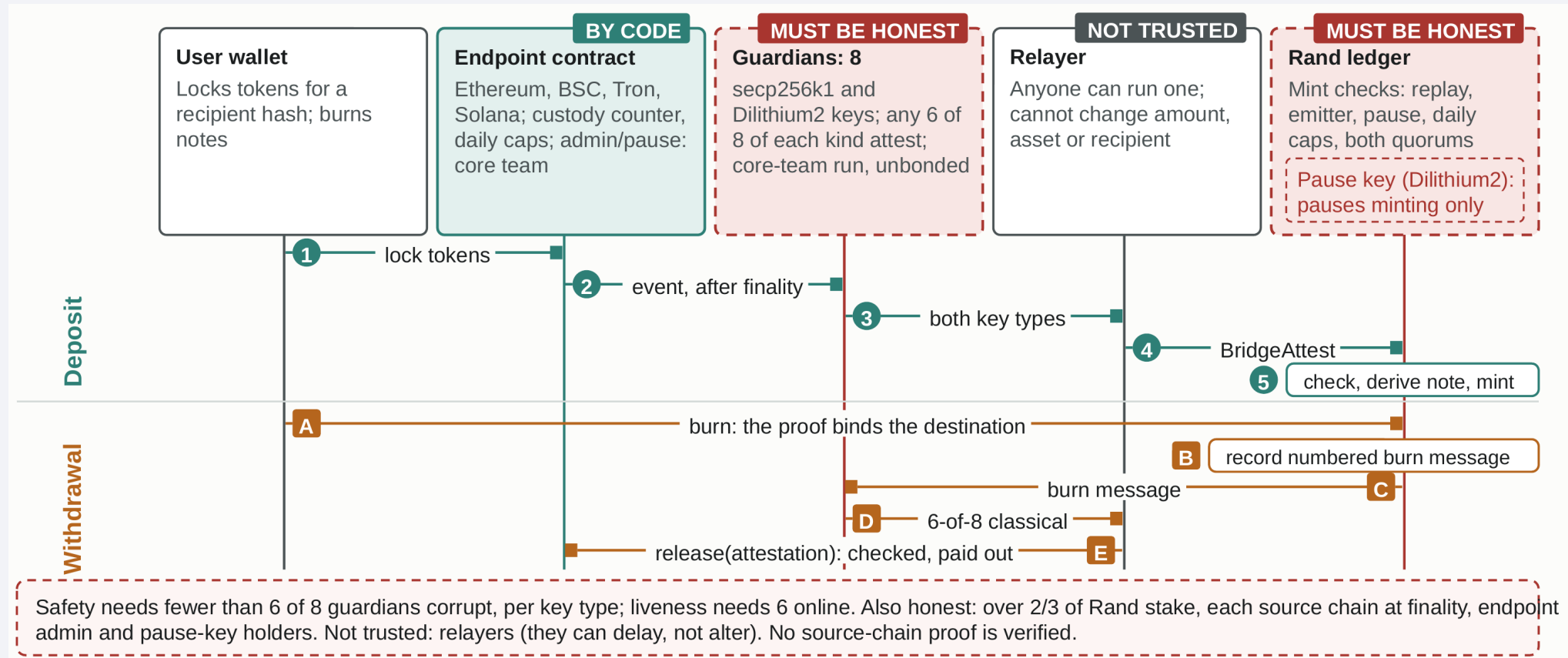
* Measured on one 16-core laptop-class machine, not optimised.

Programs, translators and the token standard are technical capabilities of the zkVM and the ledger

Capabilities on chain 20, the public test network, as recorded on 1 October 2026.

CAPABILITY	HOW IT WORKS	STATUS
Programs	RV32 code deployed once and named by its content hash. The caller runs it privately and proves the run; the chain stores the proof and eight public output words. Calls are limited to tier 14 (under 16,383 cycles).	Implemented
Programs with state (Invoke, RPL-2)	Public cells and value vaults per program. An Invoke carries a declared state change and the proof of the run (at most 8 reads, 8 writes, 4 payouts). A completed run is accepted: a program must refuse what it forbids.	New in v0.6.8. Enabled in chain 20's genesis; early, no program recorded in use
Translators (EVM and Solana bytecode)	Two off-chain tools turn a contract into an RV32 image; the chain sees only the image and its hash. A translated ERC-20 was deployed and called on an earlier test chain.	Partial. Calls above tier 14 are refused, so translated token calls are not admissible today
RPL token standard	A registry entry, not a contract. Tokens move as shielded notes in the same pool as the native coin RAND. Mint authority is set at registration: none, one key (can hand it on or renounce it) or, since v0.6.8, a program.	Implemented

The bridge trusts a core-team guardian committee: any 6 of 8 can attest a mint, and daily caps limit exposure



As of 1 October 2026.

WALLETS

Wallets keep the spend key on the device; a prover is used only when the device cannot hold the proof

Design of each client. At the chain 20 cut only the command-line wallet could send; app, extension and web builds follow the node release.

CLIENT	BUNDLE PROOF (~1.5 MB)	SPEND KEY AND AUTH PROOF	SHARED WITH A PROVER
Command-line wallet	On the device, or a paired prover	On the device	Viewing key and salt, only if the user opts in to a prover
Desktop app (Windows, macOS, Linux)	On the device (~6 GB peak, one laptop-class machine); can prove for the user's other devices	On the device	Nothing, unless a prover is paired
iOS and Android	On a device with ~8 GB of memory, otherwise a paired prover	On the device	Viewing key and salt, if a prover is paired
Browser extensions and web wallet	A paired prover: WebAssembly memory is capped at 4 GB	On the device	Viewing key and salt, on every send
Core-team prover pool (chain 19 record)	Viewing-key jobs only, no fee, one job per member at a time; the wallet verifies every proof	Never receives the spend key	Operators can read the history of any wallet using it: a trusted third party for privacy

Each property rests on a stated assumption; the heaviest today are one operator and one committee

Threat model from the project's papers: classical and quantum attackers, under 1/3 of stake Byzantine, partial synchrony.

PROPERTY	DESIGN BASIS AND ASSUMPTION	LIMIT STATED IN THE PROJECT'S DOCUMENTS
No conflicting commits	Quorum over 2/3 of stake, three-chain commit, one vote per view, persisted lock; under 1/3 Byzantine	On chain 20 one operator, the core team, holds all 26 keys and slashing is off; the safety argument is on paper
Progress	Rotating leaders, timeout certificates; over 2/3 of stake online	Assumes messages arrive; connection exhaustion is outside the argument. Leaders are predictable and the mempool is plaintext, so the leader orders
Soundness and quantum margin	Hash-based STARK, hiding FRI, no trusted setup; Dilithium2 and ML-KEM-768. Elliptic curves only in peer transport and source-chain contracts	FRI, classical: ~86 bits proven (66 queries, 20 grinding), ~100 conjectured. Quantum, by the project's analysis: ~43 proven, 50 conjectured; transcript hash ~85. No overall level is claimed
Note privacy	Hiding commitments, fixed 4-in, 4-out shape, ML-KEM-768 envelopes; statistical zero knowledge	Fees, timing, nullifiers, proof tier, deposit and burn amounts, program code and payouts are public; a node serving Merkle paths sees the note spent
No double spend or replay	Nullifier set; transactions and proofs bound to the chain id and, on chain 20, the genesis hash	Chains 14 to 19 bound the chain id only; bridge messages have their own replay rules (slide 15)
Bridge	Any 6 of 8 guardians in both signature sets, replay floors, custody counters, a pause key	A corrupt quorum could mint, undetected; releases stay classical; admin and pause roles are core-team accounts

State on 1 October 2026: a live test network; aggregation and GPU proving are designed, not active

From the project’s release records of 1 October 2026; directions are stated by the project and carry no dates.

AREA	IMPLEMENTED	DESIGNED, NOT YET ACTIVE	STATED DIRECTION
Network	Chain 20, live since 1 October 2026; chains 14 to 19 were test chains; each hard fork so far started a new chain	None	A main network, later, with a fresh genesis
Consensus	HotStuff-style BFT; 26 equal-stake validators, all run by the core team; quorum 18	Slashing for leader equivocation: in code, off on chain 20	A wider validator set; more slashing; a leader beacon
Proving	80-query FRI profile, parallel CPU prover (patched, not re-measured), split authorisation, core-team prover pool (chain 19 record)	Block aggregation; a CUDA backend crate; sealing of covered proofs	An aggregator machine class; GPU measurement
Programs	zkVM programs, token registry, program state and Invoke (new in v0.6.8, early)	None	Cross-program calls; block context for programs
Bridge	Four source chains, 8 guardians, small caps; round trip exercised on chain 19 on 1 October 2026	None	Distinct guardian operators; timelock control; later, a STARK proof of source-chain consensus
Clients	Wallets for CLI, desktop, iOS, Android, browser and web; at the chain-20 cut only the CLI carried v0.6.8	A CI release workflow with provenance attestation	Rebuilt wallet apps; store listings; more public RPC endpoints

CLOSE

Rand runs on a public test network, limits stated

SUMMARY

Shielded notes, a RISC-V zkVM and a hash-based STARK with no trusted setup, finalised by HotStuff-style consensus, on a live public test network (chain 20, 1 October 2026).

TRADE-OFFS

About 2.9 MB of proofs per transfer bounds a 20 MiB block to about seven transfers; the core team runs all 26 validators and the bridge's guardian committee.

NOT YET ACTIVE

Block aggregation and GPU proving.
Directions, without dates: a wider validator set, bridge decentralisation, a main network.

TO DISCUSS

1

From one operator to a wider validator set, and what must hold before slashing is enabled.

2

How proof size falls (aggregation memory limit, sealed history, the auth proof) and what that does to block capacity.

3

Reducing bridge trust: from guardian operators and key hardware to proof-based verification of source-chain consensus.

A technical description of technology under development on a public test network; not an offer or solicitation of any security, token or financial product, and not investment advice. Directions may change; figures are not guarantees.