Rldcoin: A Peer-to-Peer Payment System Across Delayed Regions

Runlai Deng
dengrunlai@gmail.com
www.rldcoin.com
27 September 2026 · Version 1.4

Abstract. Rldcoin is a peer-to-peer payment design for communities separated by long or intermittent communication. Digital signatures authorize ownership; regional ledgers establish local order; proof of work issues a bounded currency and selects a valid history. Prefunded channels create a path to fast, verifiable local receipts. To move value between regions, the source records an export, a signed checkpoint commits its history, and the destination verifies the proof before a unique import. Delay-tolerant relays carry this evidence without gaining authority over it. Explicit pending states, durable receipts, recovery rules, and a fixed global supply prevent communication delay from silently becoming a second spend. The design accepts a physical lower bound on remote credit and makes progress conditional on future contact, storage, and honest validation. This paper presents the currency, consensus, payment, cross-region, privacy, and security principles as one system. Current public operating status is reported separately at www.rldcoin.com/network.

1. Introduction

Bitcoin showed how signatures, public transaction history, and proof of work can order electronic payments without a central transaction processor [1]. Rldcoin begins with the same need to reject double spending, then asks a further question: how can an asset move between regions whose messages may be delayed, interrupted, or carried physically? Delay-tolerant networking research uses persistent store-carry-forward delivery across scheduled or opportunistic contacts [2, 3]. Neither a synchronous global vote across distant regions nor a promise of immediate remote arrival follows from that transport. Treating a sent message as a received payment is unsafe.

The architecture separates local ownership from inter-region delivery. Each Zone has its own chain and validation rules. A source export removes an amount from that Zone's spendable state. A destination may create corresponding spendable value only after verifying an authenticated export and a finalized source checkpoint. A missing receipt or expired delivery deadline cannot authorize a second spend. Earth provides the first regional instance; the same principles can be evaluated for more distant Zones.

This is an engineering design and a statement of its trust assumptions, not a claim that the present Earth deployment has an interplanetary route. A local channel receipt, source finality, destination import, and a returned receipt are different events. The recipient must distinguish the evidence available at each event. Formal proof-of-work analyses rely on bounded communication and adversary assumptions within a chain [4]; they do not make one globally synchronized chain practical when signal propagation between Zones takes minutes or longer.

The official Earth identity began on 27 September 2026 with an empty height-zero history and zero issued RLD. Retired trial chains have different identities, and their balances do not migrate. The genesis, signed rule adoptions, and source commitment identify the network. The parameter values and algorithms below define Rldcoin; the bibliography cites external research and standards.

2. Participants and system invariants

The network has owners, miners, full verifiers, checkpoint signers, proof couriers, and optional channel watchtowers. Owners hold payment keys. Miners propose proof-of-work blocks within a Zone. Verifiers independently execute ledger transitions. Checkpoint signers commit a sufficiently buried source block for inter-Zone import. A courier stores and forwards an export proof or receipt across one or more contacts but cannot create value. A watchtower can challenge a stale channel close with a newer jointly signed state. These roles may be held by one operator today; separating their names does not establish independent control.

Four invariants govern every transition. Only an owner signature or an expressly authorized protocol rule may move an output. Value can be spendable in one Zone or locked in one journey, never spendable in two places. Accepted state must be reproducible from a pinned genesis and ordered history. Total issued RLD must remain within the cap. Invalid, missing, or contradictory evidence stops the affected transition; elapsed time cannot manufacture authorization.

3. Ownership and transactions

The source ledger represents value in unspent outputs. A local transfer names mature, unspent inputs, authorizes them with the owner's signature, and creates recipient and optional change outputs. The input value must cover outputs and a nonnegative fee; execution is atomic. A failed command cannot partially update balances, escrow, exports, or the state root. Signatures establish authority to spend an output, while the shared chain resolves which conflicting spend is accepted.

The smallest accounting unit is the runlai: 1 RLD = 10^24 runlai. Every amount is an exact integer. Address, transaction, and proof validation bind the applicable chain or Zone identity; an object from a retired genesis has no authority on Earth. A recipient should verify the selected chain, input maturity, signatures, and confirmation state rather than relying on a broadcast acknowledgment. A signed payment request can bind the recipient, destination Zone, amount, purpose, expiry, and nonce, but the payer still verifies its fields and signs the resulting transfer.

Figure 1 traces two mature outputs owned by one key into a signed Earth transfer. Its amounts are illustrative; the essential checks are network binding, input ownership, maturity, signature validity, non-reuse, and exact conservation. A fee output has reward maturity rather than immediate spendability.

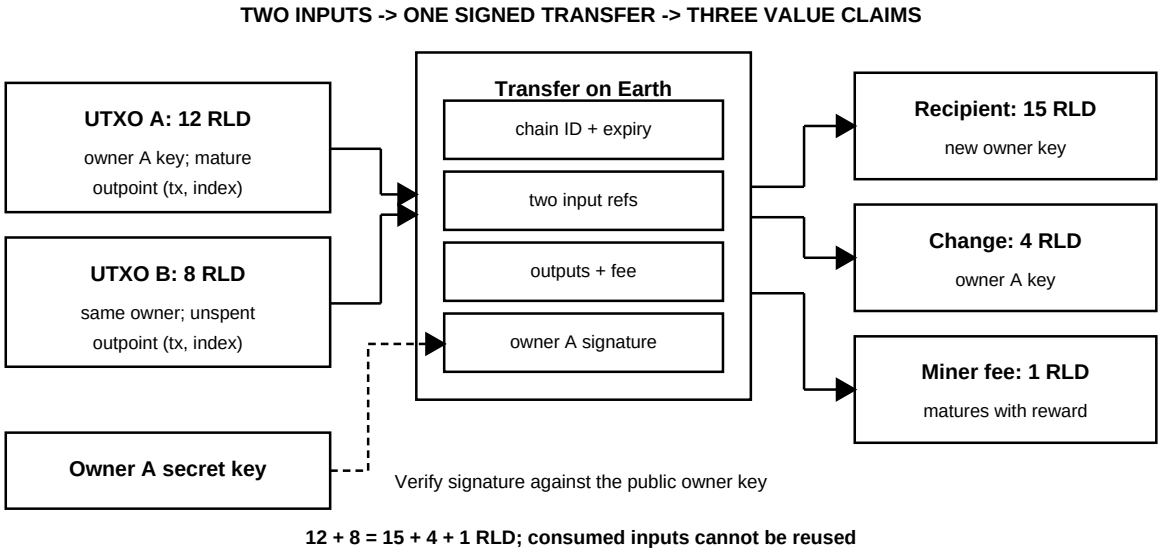

Figure 1. Two input outpoints, one owner signature, and three value claims: 12 + 8 = 15 + 4 + 1 RLD.

4. Genesis and chain identity

Earth's height-zero anchor commits an empty block history, zero issuance, the Zone identity, an initial target, a genesis manifest, and an exact implementation-source commitment. Four sorted genesis validator keys sign the purpose-separated proof-of-work adoption. A separate unanimous value adoption binds that anchor and authorizes the first monetary block at height one. No retired balance, reward, escrow, export, or service reserve is copied into this state.

The current source chain ID is 6831f7a95fec6e52e77034cc57592f01a83e8f663dd5520d7c93d8cfe101daa8. The separately authorized Earth destination starts with zero native issuance and binds the source identity. These identifiers are network pins, not substitutes for verifying the signed records and replaying the published rules. The release commits the implementation source as 3dbdc83228ac06049ee077db5c95436273ac6784cfc964bdb678e603669e1974.

5. Proof of work and network selection

Earth value blocks use double SHA-256 proof of work. A block header binds its parent, chain, height, timestamp, target, miner, commands root, resulting state root, and nonce. The commands root commits the exact ordered command body; it is not a Bitcoin transaction Merkle root. A node validates all commands and state transitions before extending a branch. Subject to an installed source-finality checkpoint, it selects the valid branch with greatest cumulative work. Network messages may arrive out of order; a node can request missing blocks and replay from the pinned genesis.

The initial target was fixed in the signed birth context. Difficulty adjusts every 144 blocks toward a 600-second average interval, with bounded adjustment. The initial target was calibrated from an operator measurement for faster early reward maturation; it is not a guaranteed block time. Mining is probabilistic. An unfinalized transaction may be displaced by a competing branch, so acceptance at the API and inclusion in one block must be described separately from finality.

Figure 2 opens two connected blocks: the header hash links them, while the commands root and state root bind the body and replay result. Proof of work only qualifies a block after the independent command and state checks pass. An installed checkpoint adds a separate branch constraint.

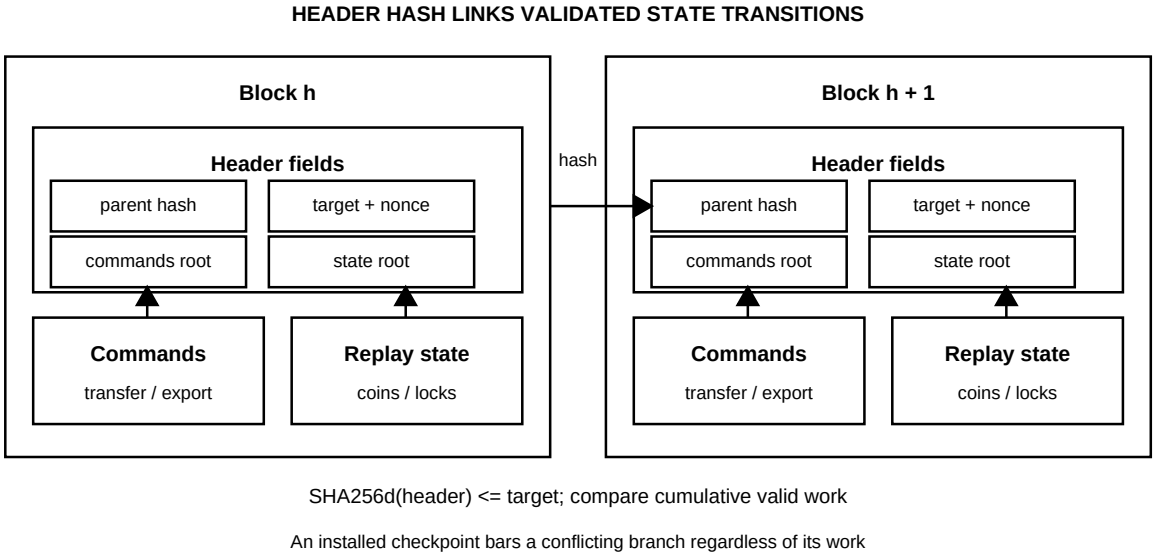

Figure 2. Consecutive Earth blocks bind parent hash, target, nonce, commands root, and replayed state root.

The essential source-block validation algorithm is:

```
ValidateBlock(B, P, C):
  require B.chain == pinned_chain and B.parent == hash(P)
  require SHA256d(B.header) <= B.target
  require B.target == expected_target(P) and B.height == P.height + 1
  require B extends installed checkpoint C
  S = replayed_state(P)
  for command in B.commands:
      S = apply_atomically(command, S) or reject B
  require commands_root(B.commands) == B.commands_root
  require root(S) == B.state_root and supply(S) <= 100 billion RLD
  return valid block with cumulative_work(P) + work(B)
```

A node compares only valid branches that extend its checkpoint; a hash meeting the target does not excuse an invalid command or incorrect state root.

6. Issuance and incentives

The maximum supply is 100,000,000,000 RLD. All of it was unissued at genesis; no person or service received an initial allocation. The first selected block's subsidy is 250,000 RLD. Each 200,000-block era releases half of the remaining unissued reserve under exact integer accounting. Rewards and fees become spendable after 100 additional source blocks. Only valid blocks on the selected branch earn rewards; a discarded branch does not retain its issuance.

For a valid source transfer, the conservation condition is: mature inputs = recipient outputs + change + miner fee. Each uniquely issued unit can appear in source spendable state, source channel escrow, an in-transit export, or a destination claim under the adopted transition rules; an export record remains as evidence after import but is not a second unit of value. The source debit must precede destination credit, and one export ID cannot produce two destination claims. The cap bounds distinct source-issued units, not the sum of historical proof records. A destination import does not mine new RLD. A new Zone cannot increase the supply by copying records or assigning itself a new reserve. The early mining period can concentrate ownership even without a founder allocation; work and access are not guaranteed to be evenly distributed.

7. Prefunded local payment channels

An opening command places a mature source coin into a channel escrow; a separate mature coin can reserve a challenge fee. A receiver checks both the funded channel and its reservation before accepting a payment state. Each successive state has an increasing sequence number, identifies the same channel, and requires both parties' signatures. The receipt binds that state to an invoice.

A unilateral close exposes its state for 2,016 blocks. During that interval, a higher jointly signed sequence can replace a stale state; an on-chain challenge consumes its reserved fee atomically. Settlement conserves value and returns an unused reservation. Payer, receiver, and watchtower records must survive exact retries and restart. A reliable fast-payment receipt depends on verified funding, data availability, independent observation of stale closes, and tested recovery. It is evidence of a funded channel state, not a claim that an on-chain block has appeared.

Figure 3 separates the funded off-chain state from the on-chain dispute. The reserved fee is a separate mature coin. A later jointly signed sequence can defeat a stale close during the 2,016-block window, provided the newer state remains available to a party or watchtower.

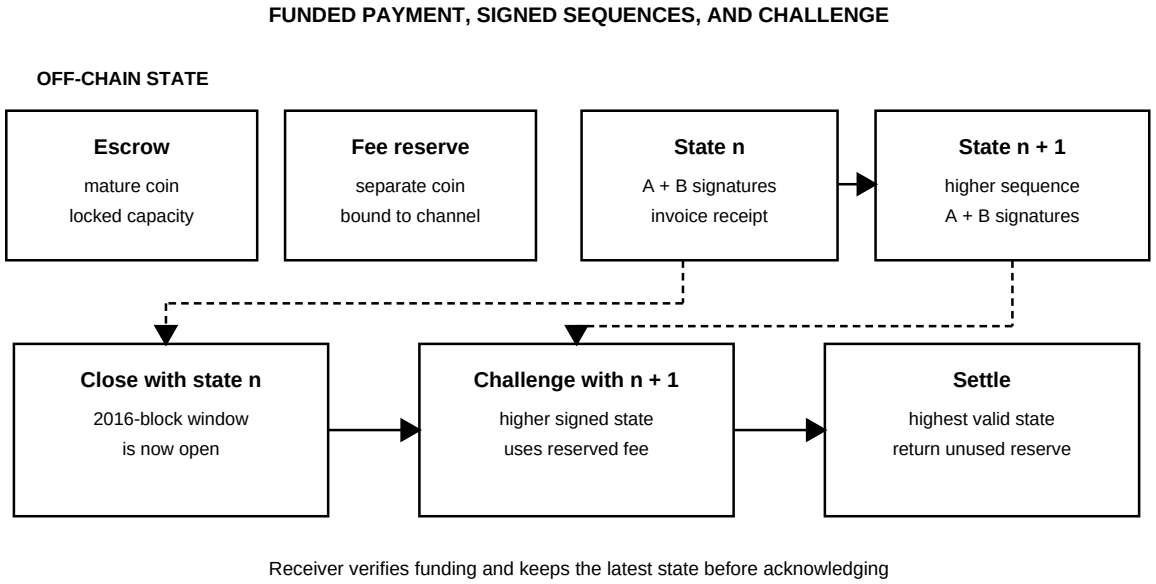

Figure 3. Mature escrow and a separate challenge reserve support signed payment states, stale-close challenge, and final settlement.

```
AcceptChannelState(new, old, funding, reservation):
  require funding is mature and locked to the exact channel
  require reservation is mature and bound to its challenge fee
  require both signatures valid for new.channel and new.sequence
  require new.sequence > old.sequence
  require sum(new.participant_balances) == funding.value
  persist new and its invoice-bound receipt before acknowledging
```

8. Export, checkpoint, and destination import

An export removes value permanently from the source's spendable state and creates an ordered, unique export record. Delivery timeout alone never refunds it: the destination might already have imported the proof. The export records form a hash tree in sorted export-ID order. A membership proof carries the record, its leaf index and count, and sibling hashes; the verified export root is part of the source commitment and resulting state root. The proof establishes membership in a *claimed* source state, not that the state belongs to the selected source chain or has reached finality. Those checks require replay and a signed checkpoint. Replaying an export ID cannot mint a second destination output.

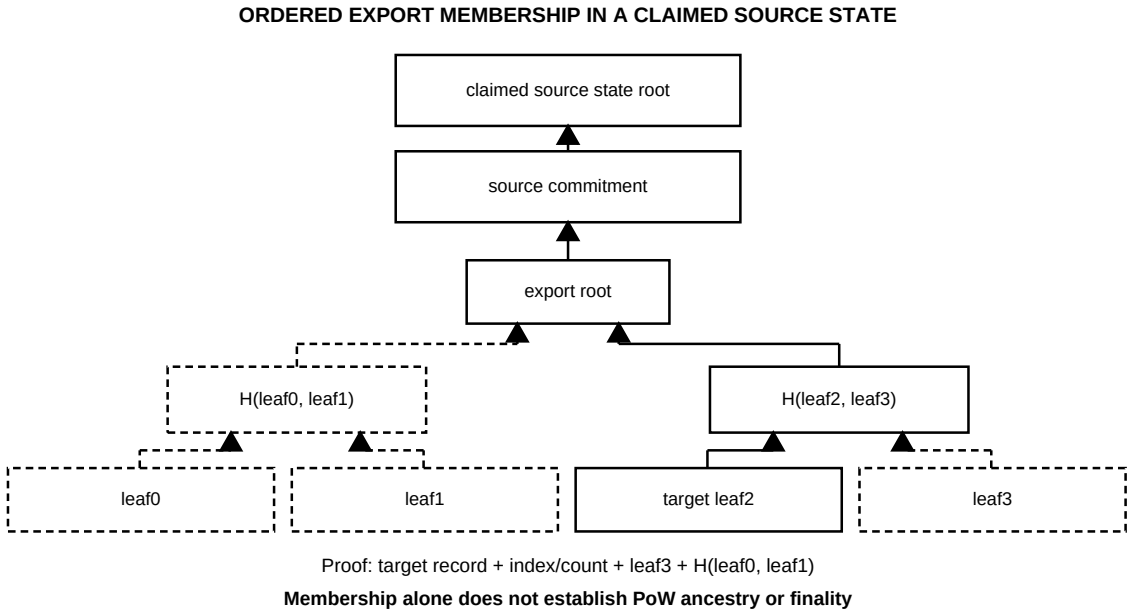

Figure 4. A target export, its index and count, and two sibling hashes recompute the ordered export root; dashed outlines also show surrounding tree context.

A source checkpoint can be signed only after its selected block has at least 12 source confirmations. All four adopted keys sign the same statement, which binds the chain, adoption, block, height, state root, cumulative work, and preceding certificate. Source nodes durably install the certificate and reject any later branch that does not descend from the highest installed checkpoint, even if that branch has more proof of work. The destination checks the certificate against replayed source history and requires six destination blocks before an imported coin can be spent. Source finality does not finalize destination blocks.

The destination records the export ID once: exact retransmission is idempotent. An acknowledgment or later spend record may return as an asynchronous receipt. Loss of that receipt does not reverse the import or make the original source output spendable. Clients should expose separate states for export, source finality, proof delivery, destination import, destination maturity, and spend. A transport delivery report is not a destination import receipt; a return receipt is not proof that the destination's later blocks cannot reorganize. The four keys are currently controlled by one owner; their unanimity is an operator-controlled finality boundary whose custody assumptions must be considered separately.

Figure 5 shows the independent checks on each side of the communication boundary. A courier carries evidence but has no authority to issue value. Finalized source ancestry, matching signatures, unique import ID, and destination maturity each answer a different question.

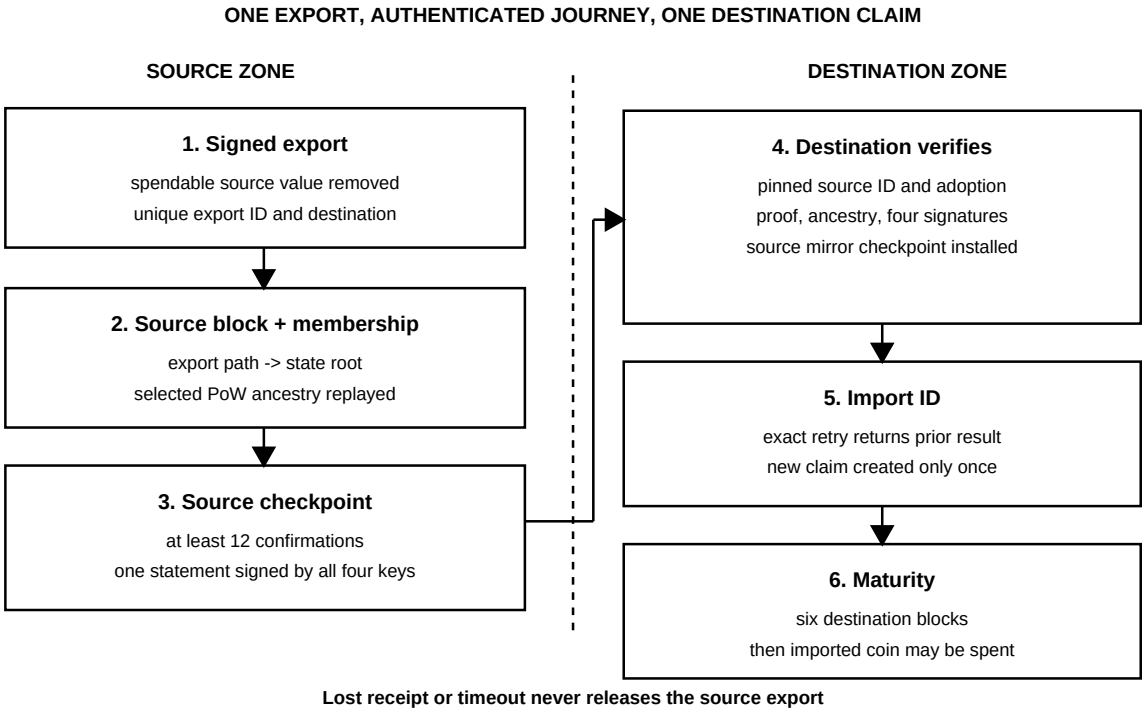

Figure 5. Exported value crosses the Zone boundary only through a verified membership path, source checkpoint, unique import, and destination maturity.

The checkpoint and import decisions are separate algorithms:

```
SignCheckpoint(block, source_history, signer_lock):
  require block is on selected source ancestry
  require at least 12 selected source confirmations after block
  statement = bind(chain, adoption, block, state_root,
                   cumulative_work, signer_lock.previous_id)
  require statement does not conflict with signer's durable lock
  persist the lock before releasing the signature
  return signature(statement)
Import(bundle, certificate, destination_state):
  require bundle.destination == pinned_destination
  require all four signatures on one valid source certificate
  require bundle.export is in the checkpoint's source ancestry
  require membership proof and source replay agree on value and ID
  if bundle.export_id already imported: return prior import result
  record exactly one imported output under bundle.export_id
  allow spending only after six destination blocks
```

9. Network operation and delayed communication

Owners broadcast signed commands, miners collect valid commands into blocks, and peers relay blocks or request missing ancestors after reconnecting. A Zone can process local transactions while its link to another Zone is down. It cannot declare a remote transfer complete merely because it sent a message. Each receiving Zone pins the source genesis, adoption, proof format, and signer set it accepts. Wrong-domain and unsupported-version messages fail closed.

A long-distance transfer follows an explicit state machine: locally authorized, source-locked, source-finalized, proof in transit, destination-imported, destination-matured, and receipt returned. Messages may be delayed, reordered, duplicated, or lost. A stable export ID, authenticated ancestry, and deterministic retry behavior make a courier replaceable without letting it mint value. Relays, intermittent networks, and carried media affect availability, not the authority to spend. The recipient can verify spendability on a local destination full node without waiting for a return signal to Earth. Earth can verify that outcome only after destination history and a receipt reach an Earth-side verifier. A timeout can trigger

investigation or a second courier, but it cannot prove the destination did not import. This distinction resembles asynchronous cross-ledger packet protocols [5], although Rldcoin's adopted source-finality and no-silent-refund rules are its own.

For a candidate route with hops i, let L_i be one-way propagation distance divided by signal speed, W_i the wait until a usable contact, and X_i its transmission and relay time. Let T_source include export inclusion and source finality, and let T_dest include destination verification, import inclusion, and the six-block spend maturity. Even without processing failures, a useful lower-bound decomposition is:

```
T_remote_spend >= T_source + T_route + T_dest
T_route >= sum over hops i of (L_i + W_i + X_i)
T_maturity_receipt >= T_remote_spend + T_return_route
```

These are durations, not promises of arrival. A scheduled contact plan can estimate W_i and capacity; an unscheduled or failed contact may make it unbounded [2, 6]. Mars links illustrate that even one-way light time varies with geometry and communications can be disrupted [7]. The return path may be different or absent. A local funded-channel receipt can be fast because its parties share a local validation context; it cannot certify remote destination maturity before evidence returns.

An export proof is an application payload for a persistent queue. Each relay should bind payload bytes to the export ID and intended destination, authenticate its peer or bundle where appropriate, retain the payload until durable custody transfer or a documented retention limit, and record receipt, forwarding, and deletion events. Contact scheduling must account for payload size, link capacity, competing traffic, and storage exhaustion [2, 3, 6]. Bundle Protocol lifetime expiration or a courier's deletion only ends that copy's delivery attempt; it does not reverse the ledger export. With imperfect clocks, transport lifetime and ledger validity must not be inferred from one another [2]. Bundle-layer integrity or confidentiality can protect payloads in transit but cannot substitute for source replay, checkpoint signatures, or the destination's unique-ID check [8].

The Earth implementation now has an application-layer evidence carriage profile for the adopted source and destination nodes. A versioned canonical JSON frame contains one of five kinds: source sync page, source finality certificate, finalized import command, destination sync page, or destination receipt. It binds both 32-byte chain IDs, the 32-byte export ID, the exact payload bytes, their SHA-256 digest, and a domain-separated message ID. Each payload is at most 3 MiB; a durable queue accepts at most 4,096 frames or 256 MiB and fails closed at capacity. Repeated receipt of identical bytes is idempotent, while carrying a frame leaves the original queue copy intact. These limits are transport controls, not consensus limits. The message ID detects changed bytes but is not a signature or proof of sender identity. Operators must authenticate peers or use bundle security for an actual route [8]. The current tool can hand frames to a file, contact medium, or future BPv7 payload; it does not implement BPv7, a contact scheduler, custody acknowledgment between independent operators, or a physical interplanetary link [2, 6].

The receiver must replay source blocks from its pinned source anchor, install a valid four-signature certificate on its local source copy, then submit the finalized import to a destination node that independently mirrors that source. The destination node validates the proof and unique ID before local inclusion. To report the result back, a courier must return selected destination blocks and an import receipt; an Earth-side destination mirror must replay those blocks against the same source identity and verify the receipt on its selected branch at or above the receipt's recipient spendable height. The adopted rule adds six successor blocks after import, so an import in block 91 becomes spendable at height 97; six confirmations counted *including* block 91 reach only height 96 and are insufficient. A receipt alone is an index into history, not an independent finality proof. A source-region verifier must not interpret its own stale destination mirror as evidence of non-import.

```
Frame(kind, source, destination, export_id, exact_payload):
  require pinned source != pinned destination and payload <= 3 MiB
  body = canonical_json(version, kind, source, destination,
                        export_id, SHA256(exact_payload), base64(exact_payload))
  message_id = SHA256(ASCII("RLD-INTERREGION-EVIDENCE-V1") || 0x00 || body)
  persist frame and message_id before a contact copies exact bytes
  return message_id  # integrity and deduplication, never value authority
ReceiveAndSettle(frames, local_source, local_destination):
  require each frame's route, digest, and message_id match pinned IDs
  replay ordered source pages and verify every block and state root
  install source certificate only if signatures and ancestry verify
  submit finalized import; require destination independently accepts it
  observe selected import plus six-block maturity on destination
  return destination history and receipt; replay on Earth side
  require selected destination height >= receipt.recipient_spendable_height
  never infer import, non-import, or refund from contact silence
```

Figure 7 makes the epistemic limit explicit. After the source finalizes an export, silence is consistent with both a proof that never reached the destination and a completed import whose receipt was lost. Those worlds are indistinguishable at the source without new authenticated evidence. A source-only time-based refund would release spendable value in the second world and violate conservation. Atomic-swap timelocks can work under their own bounded-time and observation assumptions [9]; they do not make an unverified absence of import a proof in this model.

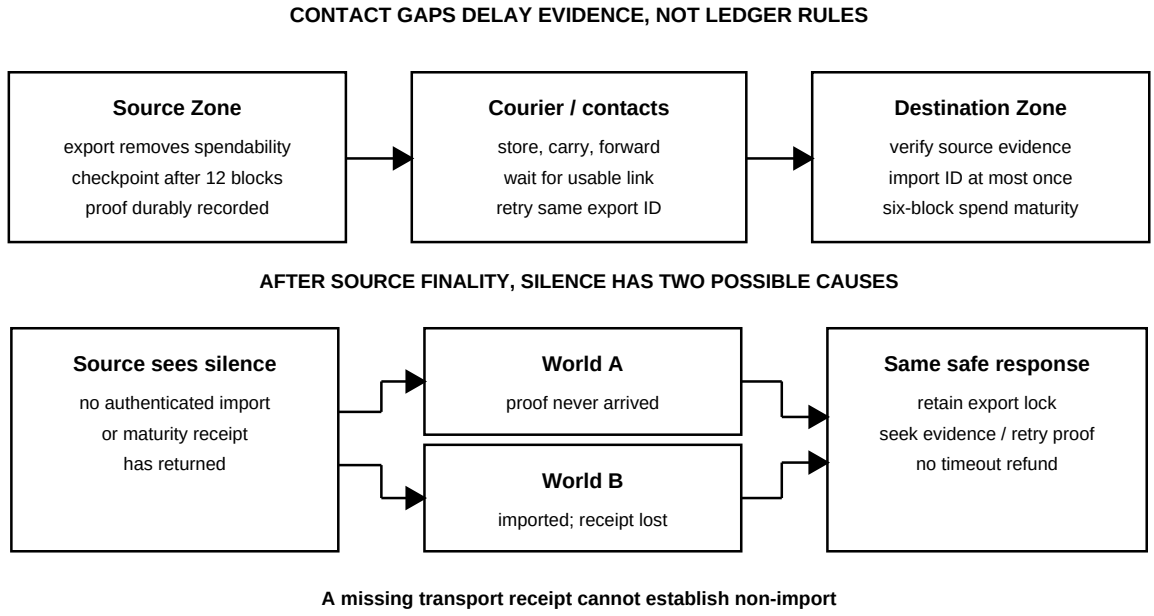

Figure 7. Store-carry-forward proof delivery and two indistinguishable silent outcomes; neither a contact gap nor a missing return receipt authorizes a refund.

```
RelayExport(proof, export_id, destination):
  require proof bytes and destination match the queued export ID
  persist payload, hash, and forwarding state before acknowledging custody
  for each usable contact until a documented retention limit:
      send identical payload; record peer receipt and retry safely
  report delivery evidence separately from destination import evidence
  never issue or refund value as a consequence of transport status
```

The Earth nodes and external queue implement the local replay and bounded carriage interfaces described above; the relay procedure remains an operational requirement for a future long-distance deployment. The current Earth export, import, maturity, and operator receipt were exercised under one owner's control. No distant independent destination or physical route has been verified. A live interstellar claim requires a pinned distant Zone, independent operators and key

custody, a tested forward and return contact path, durable capacity, and disconnection, restart, reorganization, and contradictory-certificate drills. Safety does not imply liveness: without such a route, storage and capacity, available verifiers, and non-conflicting checkpoints, the system cannot guarantee a finite time to remote spend. In a fully asynchronous system even a single crash prevents a deterministic consensus protocol from guaranteeing termination in all executions [10]; this result motivates explicit conditional progress rather than an unconditional cross-Zone deadline. It does not by itself prove the safety of Rldcoin's checkpoint design.

10. Verification and storage

A full verifier pins the exact genesis and adopted rules, checks signatures and block work, replays commands in order, and compares resulting state roots. Startup and replay reject wrong identities, corrupt durable records, unsupported history, or noncanonical inputs. Source and destination can be checked from their public records and published release rather than from a website badge. A lightweight client that trusts reported roots or a remote API inherits that operator's availability and honesty assumptions.

Old blocks and proofs may be indexed or compacted for convenience only if the commitments and sufficient data remain available for independent replay, destination import, channel challenge, and dispute resolution. An index accelerates lookup but is not an authority independent of the committed history. A light client that accepts a header and membership path without executing every command relies on full verifiers to detect invalid state and on its chosen source of the best chain. That trust assumption must be visible to the user. For a delayed route, archival policy must outlive the longest intended journey and possible dispute, including the source ancestry, export membership path, signed checkpoint, adoption records, import record, and receipts. A hash commitment cannot reconstruct data that every custodian has deleted. Independent archival copies and tested recovery are therefore part of availability, not a relaxation of the verification rule.

11. Keys, privacy, and user interfaces

Owner keys authorize payments; genesis and checkpoint keys sign only their narrowly defined network statements. These roles need separate custody, backups, and purpose-separated signing domains. A wallet should display the actual genesis, recipient, amount, fee, destination, expiry, and state before signing. A payment request can help bind these terms but cannot replace transaction validation. Secret keys should never be sent to a website, courier, or mining peer.

Journeys and archives spanning years also require cryptographic and operational migration planning. A future signature change must bind the same network and intent, preserve verification of old records, define activation and dual-verification rules, and test how in-flight exports are interpreted across versions. NIST's standardized ML-DSA is a candidate signature family to evaluate for such a change [11]; the current Earth protocol has not adopted it, and existing signatures should not be described as post-quantum secure. Key compromise during a long disconnection can be discovered only after evidence crosses that gap, so withdrawal and incident procedures must preserve conflicting signed evidence rather than rewrite history.

Public keys are pseudonyms, not anonymity. On-chain amounts, timing, and output relationships are visible; reused keys and multi-input transfers can reveal further links. New keys can reduce casual linkage, while cross-Zone proofs necessarily reveal enough history to justify an import. Privacy must account for the ledger and network metadata rather than assume that a cryptographic address hides identity.

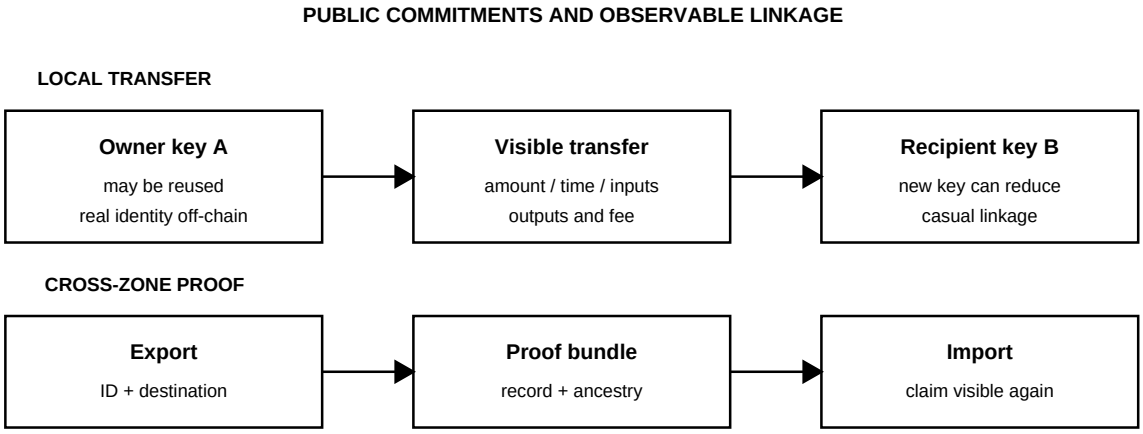

Figure 6. Public local transfers and cross-Zone proofs expose different links; a fresh key reduces simple reuse but does not hide value or timing.

12. Recovery and protocol evolution

Nodes recover by authenticating the original genesis, durable history, installed checkpoints, and executed state before serving value. Recovery must retain per-key signer locks and the most recent signed channel states. Restoring a checkpoint key without its last signed statement can permit equivocation; restoring a channel without its newest state can permit a stale close. A restart cannot turn an in-transit export into a refund or a pending receipt into a settled balance.

An upgrade requires an explicit version, activation rule, exact rule and code commitments, replayable migration, and preserved asset history. A software release alone cannot change another verifier's accepted genesis or supply. New Zones require their own signed identity and source bindings. Unknown formats must fail closed instead of falling back to an older interpretation. A public rule change must state how prior signatures, outputs, channels, exports, imports, and checkpoints remain valid or are resolved.

For a stranded export, recovery starts by querying independently held source and destination evidence and replaying both histories. If the destination has imported, resend or reconstruct the receipt. If import is not observed, keep the export pending and retry a valid proof through another route. Absence from an incomplete destination view is not non-import proof. Any future cancellation design would require an authenticated destination non-import commitment with a specified finality and no-conflict rule; that rule does not exist in the current adopted protocol. Recovery must preserve the possibility of later evidence rather than making a temporary outage into a new coin.

13. Security analysis and economic recovery

An attacker may double-spend an unfinalized local transfer by outworking the observed branch, censor or withhold a contact, replay an export, substitute another Zone's proof, exhaust relay storage, present a stale channel state, steal a key, or cause signers to authorize conflicting checkpoints. Work, signature, domain, maturity, unique-ID, and state-root checks address different parts of this model. Proof of work is probabilistic; channel safety needs available evidence during its challenge window; source checkpoint safety requires adopted signers not to authorize conflicting histories and all value-serving source nodes to enforce the lock. Redundant couriers and contact paths can improve delivery odds but do not prevent signer equivocation or invalid destination validation.

For an idealized proof-of-work race, let p be the honest share of work, q the attacking share, and z a current block deficit. With independent block discoveries, the probability that the attacker ever catches up is (q/p)^z when p > q, and 1 when p <= q [1]. This is a catch-up model from an observed deficit, not an end-to-end payment guarantee. It excludes network partition, signer equivocation, channel failure, and destination reorganization. Those conditions must be evaluated separately.

If z instead counts honest confirmations observed while an attacker mines privately, the attacker may already have found k blocks. Under the idealized independent-discovery and Poisson approximation used in [1], let lambda = z*q/p. For q <

p, the approximate eventual catch-up probability is:

```
P(z, q) = 1 - sum(k = 0..z) [ exp(-lambda) * lambda^k / k!
                            * (1 - (q/p)^(z-k)) ]
lambda = z*q/p,  p = 1-q,  q < p
```

Illustrative values, rounded from that model, are:

Honest confirmations z	q = 0.10	q = 0.30
1	0.204587	0.627749
3	0.013172	0.324584
6	0.000243	0.132111
12	0.000000089	0.023584

Figure 8 uses a logarithmic axis to display that decline. Neither this Poisson calculation nor its illustrative inputs measure Rldcoin's live hashrate or guarantee an import, because the source checkpoint's signer custody and the destination chain add separate risks.

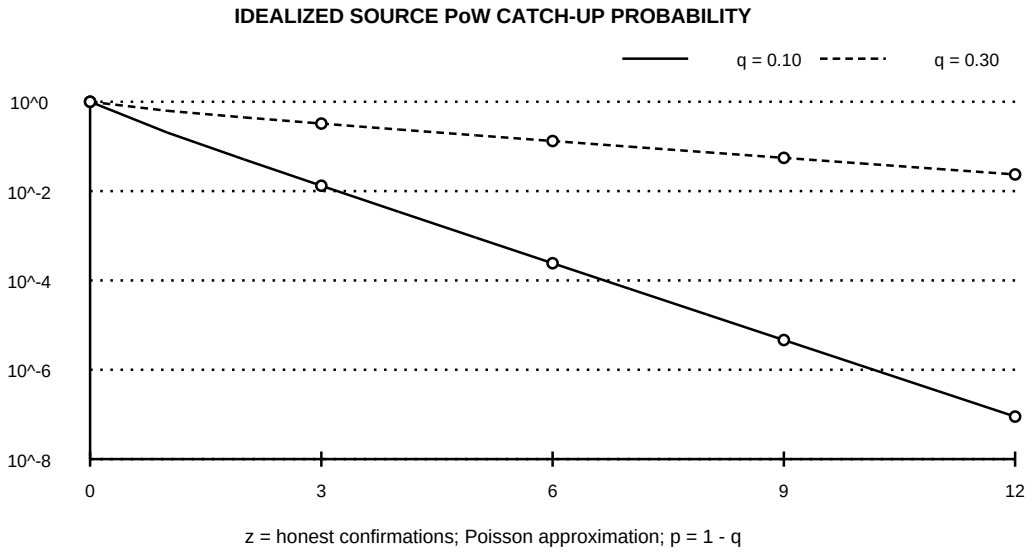

Figure 8. Idealized source proof-of-work catch-up probability after z confirmations, for illustrative attacker work shares of 10% and 30%.

The protocol can reject an ordinary source reorganization after a unanimous checkpoint, but cannot automatically restore value already spent at the destination after signer equivocation or compromise. If contradictory certificates appear, affected imports and redemptions should stop, both certificates and liabilities should be published, and any compensation should follow an auditable rule. An external value service needs a cap on outstanding credit, matured source collateral, an incident ledger, and a reproducible allocation process. A shortfall must be disclosed rather than covered by silent extra issuance.

Independent miners and operators, separate key custody, outside review, capacity testing, and long-running recovery improve assurance. Four signatures controlled by one person do not establish independent governance. An operator status page helps users observe a service but cannot prove custody, consensus safety, or availability of a distant route. Each new Zone and route needs evidence for delay, disconnection, conflicting proofs, and operator failure. No protocol can make a message arrive before physical communication permits.

14. Conclusion

Rldcoin combines local proof-of-work ownership, bounded issuance, funded payment states, and verifiable movement between delayed regions. Value changes location only through an authenticated source export and a unique destination import; lost time and lost messages do not create a second spend. Signed checkpoints, durable locks, independent replay, and explicit user-facing states make the remaining assumptions inspectable. The architecture can be evaluated on Earth before adapting it to habitats, spacecraft, or distant settlements, where communication delay is a physical constraint rather than a software defect.

References

[1] S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2008. https://bitcoin.org/bitcoin.pdf

[2] S. Burleigh et al., *Bundle Protocol Version 7*, RFC 9171, 2022. https://www.rfc-editor.org/rfc/rfc9171.html

[3] K. Fall, *A Delay-Tolerant Network Architecture for Challenged Internets*, SIGCOMM, 2003. https://people.eecs.berkeley.edu/~sylvia/papers/dtn.pdf

[4] J. Garay, A. Kiayias, and N. Leonardos, *The Bitcoin Backbone Protocol: Analysis and Applications*, EUROCRYPT, 2015. https://doi.org/10.1007/978-3-662-46803-6_10

[5] C. Goes, *The Interblockchain Communication Protocol: An Overview*, 2020. https://arxiv.org/abs/2006.15918

[6] CCSDS, *Schedule-Aware Bundle Routing*, 734.3-B-1, 2019. https://ccsds.org/Pubs/734x3b1.pdf

[7] NASA, *Mars Communications Disruption and Delay*, 2023 Moon to Mars Architecture Concept Review. https://www.nasa.gov/wp-content/uploads/2024/01/mars-communications-disruption-and-delay.pdf

[8] E. Birrane and K. McKeever, *Bundle Protocol Security (BPSec)*, RFC 9172, 2022. https://www.rfc-editor.org/rfc/rfc9172.html

[9] M. Herlihy, *Atomic Cross-Chain Swaps*, 2018. https://arxiv.org/abs/1801.09515

[10] M. Fischer, N. Lynch, and M. Paterson, *Impossibility of Distributed Consensus with One Faulty Process*, JACM, 1985. https://groups.csail.mit.edu/tds/papers/Lynch/jacm85.pdf

[11] NIST, *Module-Lattice-Based Digital Signature Standard*, FIPS 204, 2024. https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.204.pdf