BunkerCoin: A Low Bandwidth, Shortwave Radio-Compatible Blockchain Protocol

Anatoly Yakovenko[*]   BunkerCoin Team

July 31st, 2025
Version 0.0.5

— WORKING DOCUMENT —

Abstract

The rapid evolution of blockchain technology has demanded innovative solutions that extend beyond conventional digital landscapes. This paper introduces BunkerCoin, a groundbreaking blockchain protocol designed to operate under the constraints of low bandwidth networks, specifically through shortwave radio channels. At the heart of BunkerCoin is the Alpenglow consensus mechanism, which is adapted to the unique challenges of shortwave radio.

The protocol's network architecture is defined by two key components: a dynamic networking layer that leverages both broadcast and reliable P2P links, and a location-aware routing service (called Sherpa). Sherpa maintains a map of the nodes and peer information, allowing the system to route data efficiently across the globe. One central challenge lies in using this setup to aggregate the BLS multi-signatures required for voting within Alpenglow's consensus flow efficiently (routing and aggregating via optimal paths), minimizing the amount of data that needs to be transmitted for voting. The propagation of blocks and votes over shortwave radio is meticulously engineered to accommodate the protocol's 300-byte Maximum Transmission Unit (MTU). Each transmission is either disseminated through a series of 32:96 erasure coded frames, ensuring reliability and redundancy in the face of packet loss, or transferred via reliable (TCP-like) ARQ links.

BunkerCoin's architecture not only challenges traditional blockchain paradigms but also paves the way for secure, decentralized communications in bandwidth-constrained environments worldwide, marking a significant leap forward in the field of distributed ledger technology.

[*]Parts of the abstract and most core concepts by Anatoly Yakovenko. This paper is currently expanded and maintained by the BunkerCoin Team and external contributors, i.e. do not automatically expect content (especially beyond the abstract) to be endorsed by Anatoly.

1 Physical (Network) Layer

This section outlines the low-bandwidth communication substrate that enables BunkerCoin to propagate blocks, votes and other data across the globe, even under contested or spectrum-constrained conditions.

1.1 Reliability over Throughput: P2P vs. Broadcast

A fundamental requirement for BunkerCoin's transport layer is a reliable peer-to-peer (P2P) network that serves as a robust foundation. This mesh of acknowledged, 1:1 links guarantees that any node can eventually retrieve any piece of data it needs. Upon this foundation, two distinct strategies for block and vote propagation can be implemented, presenting a trade-off between simplicity and peak performance.

The first strategy is P2P-first. In this model, blocks and other data (such as votes) are disseminated primarily through the reliable P2P links. It is inherently predictable and robust. Opportunistic broadcasts can be introduced as a throughput-accelerating optimization, but they are not required for the network's core operation. This offers a simpler implementation path.

The second strategy is broadcast-first. This model leverages wide-area, connectionless broadcasts for the initial, high-speed dissemination of erasure-coded block and vote data. The P2P network serves purely as a fallback mechanism for nodes to request specific missing fragments needed for reconstruction ("repair traffic"). While this approach promises higher raw throughput under ideal conditions, it is operationally more complex, as it requires a seamless and difficult-to-manage interplay between multiple broadcasts and P2P modes, as well as better optimization of frequencies chosen for local broadcasts (there's far more free frequencies for a narrow link compared to a wide broadcast). It is also more susceptible to volatile ionospheric conditions, where widespread packet loss could create network storms as many nodes fall back to the P2P network for repairs simultaneously.

Given these considerations, the current BunkerCoin specification prioritizes the P2P-first strategy. This ensures a baseline of predictable and robust performance. Unreliable, opportunistic broadcasts are therefore treated as an enhancement for throughput acceleration (see Section 1.1.2), not as the primary means of dissemination.

This preliminary choice was made based on pessimistic assumptions about the availability of global spectrum, and will be validated through real-world testing.

1.1.1 Sherpa: A Location-Aware Routing Layer

To manage the complexity of routing data across a dynamic global network, both block and vote propagation protocols rely on an underlying service called

Sherpa. This layer is responsible for maintaining a map of the network topology, including peer locations and connections. By abstracting this logic into a shared service, higher-level protocols like Votor and Radiotor can simply request optimal routing paths without needing to implement this complex state management themselves.

1.1.2 Opportunistic Broadcasts as a Throughput Accelerator

While the P2P mesh provides a reliable baseline, network throughput can be significantly increased by layering opportunistic broadcasts on top. When spectrum conditions permit, a node can choose to broadcast erasure-coded block data to all listening peers in its vicinity instead of initiating multiple, sequential P2P links.

This hybrid model allows a node to significantly accelerate propagation. After broadcasting the erasure-coded data, the node can check in with its designated P2P peers. If a peer does not respond promptly, it can be assumed to have successfully received the broadcast and is already busy forwarding it to its own neighbors. If a peer does respond, it signifies that it requires repair traffic via a standard ARQ link. This "silence is golden" approach minimizes latency, as nodes do not have to wait for explicit acknowledgements before continuing the propagation chain.

Crucially, this allows for dynamic optimization at the individual node level. Each node can independently assess network conditions and decide whether broadcasting or initiating direct P2P transmissions is the more efficient strategy at any given moment, creating a resilient and adaptive data propagation fabric.

1.2 Proof of Location

Accurate validator locations are essential for Sherpa's routing decisions. To prevent location spoofing, validators must obtain cryptographic attestations from neighboring nodes:

1. **Location Claim:** A validator announces its approximate location (precision 50km).
2. **Neighbor Attestations:** At least min(3, total validators) existing validators within radio range must sign attestations. Attesters must have held stake for at least one prior epoch to prevent collusion attacks. Each attestation confirms:
    - Successful bidirectional radio contact
    - Propagation delay consistent with claimed distance
    - Signal characteristics matching expected ionospheric conditions
3. **Verification:** The attestations are submitted during epoch transitions and verified by the network.

Radio physics makes location fraud extremely difficult—signal delay, strength, and skip patterns are inherently tied to geographic distance. This natural constraint provides strong sybil resistance while ensuring Sherpa has reliable topology data.

1.3 Radio Data Protocols

1.3.1 PACTOR-IV

PACTOR-IV represents the state-of-the-art in high-frequency (HF) digital modems. It is a hardware-based solution requiring a dedicated SCS Dragon modem, but offers the highest publicly documented throughput, particularly under challenging and variable ionospheric conditions. A key advantage of this platform is its mature "2G ALE" functionality, which enables fully automated link establishment and management, allowing nodes to connect on autopilot. While the modem can be programmatically controlled for all essential operations, it does not offer the same low-level access as a software-defined radio (SDR) solution. PACTOR protocols have historically supported broadcast via Forward Error Correction (FEC), though documentation suggests this is primarily a feature of older, lower-speed versions (e.g., PACTOR-I), and its utility at PACTOR-IV speeds is not well-documented.

1.3.2 VARA HF

VARA HF offers a compelling alternative as a flexible, software-based modem. Its primary advantage is accessibility; it is significantly cheaper to deploy and can run on standard computing hardware. VARA natively supports both reliable ARQ links and an unreliable "unproto" broadcast mode, making it well-suited for a hybrid P2P/broadcast network architecture. When paired with an SDR, VARA provides extensive low-level control over the physical layer, but this flexibility comes with increased responsibility. For instance, a custom ALE solution would need to be developed and integrated, as VARA's native link management is not as field-proven as PACTOR's. While highly capable, VARA is generally not considered as performant or robust as PACTOR-IV under the most demanding HF conditions.

1.3.3 Summary and Outlook

The choice between PACTOR-IV and VARA HF presents a classic trade-off. PACTOR-IV is expected to deliver superior throughput and reliability across the board, backed by its robust hardware and mature ALE system, but at the cost of higher expense and less flexibility. Conversely, VARA offers a cheaper, more flexible software-based path, but is likely less performant in adverse conditions and places a greater implementation burden on the project.

Ultimately, a definitive decision requires data. To this end, the project will establish a testnet incorporating nodes that utilize both PACTOR-IV and VARA

HF stacks. A series of real-world benchmarks must be conducted to compare the two protocols, specifically evaluating their performance in both reliable P2P (ARQ) and broadcast (FEC/unproto) modes under a variety of ionospheric conditions. The results of these tests will inform the final choice for BunkerCoin's physical layer.

2 Consensus Layer

BunkerCoin adopts Alpenglow, an advanced consensus protocol originally proposed for the Solana blockchain, to achieve rapid block finality under constrained network conditions. Alpenglow replaces legacy mechanisms like Proof of History and Tower BFT with a streamlined dual-component design, achieving deterministic finality in record time.

2.1 Staking and Validator Selection

BunkerCoin employs a stake-weighted validator system where the natural barrier of expensive radio equipment limits participation. All validators with sufficient stake participate in consensus, with voting power proportional to their total stake (self-bonded plus delegations). Stake changes only activate at epoch boundaries to maintain a stable validator set throughout each epoch.

2.1.1 Stake Lifecycle

Stake (either self-bonded or delegated) follows a simple two–step lifecycle that is tightly coupled to the fixed–length epoch schedule:

- **Bonding:** A *bond* transaction locks the indicated amount immediately, but the stake becomes *active* only at the next epoch boundary ($T + 1$). Until then it is counted as "pending stake" and is not eligible for rewards or voting power.
- **Unbonding:** A *retire* transaction flags the stake for withdrawal. It becomes *inactive* at the next epoch boundary and then enters a fixed unbonding delay of U epochs (parameterised, default $U = 2$). During the delay the tokens remain locked and subject to slashing. After U epochs the owner may withdraw them with a *withdraw* transaction.

Validators must maintain a minimum self-bond (parameter `MIN_SELF_STAKE`). Falling below this threshold—whether by withdrawal or slashing—causes automatic deactivation at the next epoch boundary.

2.1.2 Reward Accrual and Distribution

Epoch rewards are *fee-driven*: the native protocol does not mint inflationary rewards; instead it redistributes the contents of the three fee pools described in Section 2.2. The distribution procedure at the end of each epoch is:

1. Split each fee pool *pro-rata by active stake* across all validators who are `ACTIVE` at the instant the epoch ends. Validators that were jailed at any time during the epoch receive no reward.
2. Apply the validator's self-declared commission rate (bounded to $\leq 20\%$ and changeable only at epoch boundaries). Delegators receive the remainder proportionally to their stake.
3. Rewards below a dust threshold of 1 μBUNKER are rolled over to the next epoch to avoid radio transmissions for micro-rewards.

2.1.3 Slashing and Jailing

Misbehaviour is deterred through stake slashing and temporary exclusion (*jailing*). Only two slashable offences are recognised to keep the protocol lightweight:

Offence	**Slash**	**Jail**
Double-vote / equivocation	100%	until re-stake
Excessive downtime (>20% votes missed in an epoch)	0%	1 epoch

Slash evidence is a pair of conflicting vote signatures in the case of double-signing. Upon inclusion of valid evidence, the protocol burns the slashed stake and moves the validator to the `JAILED` state. After the jail period elapses the operator may send an *un-jail* transaction at any time; their stake becomes active again at the following epoch boundary provided the minimum self-stake is satisfied.

2.2 Fee Distribution

Network fees are collected into three separate pools, each rewarding different network contributions:

- **Transaction Pool:** Fees from regular transactions, distributed to all active validators proportionally by stake.
- **Message Pool:** Fees from the messaging service, distributed only to validators who participated in message relay during the epoch.
- **Bridge Pool:** Fees from bridge operations, preferentially rewarding validators with stable internet connectivity who can produce Bridge Certificates.

2.3 Votor: Voting and Finalization Engine

Votor handles block notarization and finalization through a concurrent dual-path voting system. If a block receives $\geq 80\%$ stake approval in the first round, it is immediately finalized with a Fast-Finalization Certificate. If it secures only $\geq 60\%$ approval, a second round begins; upon reaching $\geq 60\%$ again, a Finalized Certificate is issued. Both paths run in parallel, ensuring rapid consensus even under partial network participation. To adapt this to a high-latency environment, Votor relies on the Sherpa service for the "in-flight" aggregation of BLS signatures across geographically dispersed radio links, minimizing the amount of data that needs to be transmitted for voting.

2.4 Radiotor: Location-Aware Block Propagation

Radiotor is BunkerCoin's adaptation of Alpenglow's Rotor protocol, tailored specifically for block dissemination over shortwave radio. It follows an approach similar to Solana's Turbine, where each node forwards erasure-coded block shreds to multiple downstream peers. Radiotor leverages the Sherpa service to select geographically distinct paths, enabling multiple, parallel data transfers across different HF links and frequencies to maximize throughput. While opportunistic broadcasts (as described in Section 1.1.2) can accelerate this process, Radiotor's multi-path forwarding is the primary mechanism for robust block propagation.

2.5 Epoch Transitions

At each epoch boundary, the network pauses block production to execute critical maintenance tasks. During this window, submitted transactions remain queued at their originating nodes while the following operations occur in sequence:

1. **State Snapshot:** Validators compute and sign the Merkle root of the current state (see Section 3).
2. **Validator Set Update:** Pending stake changes are processed, and the new validator set is finalized. A Validator Set Update Certificate (VSCert) is generated for the bridge (see Section 5).
3. **Location Attestations:** New validators submit proof-of-location claims, and existing validators may need to refresh their attestations (see Section 1.2).
4. **Fee Distribution:** The three fee pools are distributed according to each validator's contributions during the ending epoch.
5. **Epoch Transition Block:** A special block containing all transition metadata is produced and voted on.

Once the transition block achieves consensus, normal block production resumes with the new validator set and updated network state.

2.6 Fault Tolerance

To be added.

3 Transaction and State Model

3.1 Account Model

BunkerCoin uses a simplified account model with no smart contracts. Each account is identified by a public key and contains:

- **Native Balance:** Amount of native BunkerCoin tokens for fees and staking.
- **Token Balances:** A sparse array mapping token IDs to amounts. Only non-zero balances are stored.
- **Nonce:** Transaction replay prevention counter.

3.2 Token System

Tokens are created via special mint transactions that assign a unique 4-byte token ID. Each token stores minimal on-chain data:

- **Token ID:** Unique 4-byte identifier
- **Supply:** Current and maximum supply
- **Ticker:** 3-8 character symbol (e.g., "USDC")
- **Metadata Hash:** SHA-256 of off-chain JSON file containing name, description, logo, etc.

This design keeps account entries compact—most accounts will hold only 1-5 different tokens, requiring just 8 bytes per token type (4 for ID, 8 for amount when packed efficiently).

3.3 State Snapshots for Network Bootstrapping

Due to the low-bandwidth constraints of shortwave radio, full blockchain synchronization is infeasible for new nodes joining the network. To address this, BunkerCoin implements epoch-end state snapshots.

At the conclusion of each epoch, validators collectively compute the current state and construct a Merkle tree over it. The Merkle root is included in a special "snapshot block," which is then voted on and multi-signed using Votor's BLS aggregation mechanism via Sherpa routing.

This signed Merkle root ensures network-wide agreement on the state.

New nodes bootstrap by first obtaining the signed snapshot block containing the Merkle root. They then request portions of the state data from peers, verifying each subset against the root using Merkle proofs. This allows efficient, verifiable synchronization without transmitting the entire history.

Additionally, zero-knowledge proofs could be employed to further compress these Merkle proofs, reducing bandwidth requirements for state verification (e.g. when new nodes joining load chunks of the state with the related Merkle proofs).

4 Verified Offline Messaging

Beyond its primary function as a financial ledger, the BunkerCoin network is designed to support verified, asynchronous, and censorship-resistant messaging, even when direct, real-time connections (such as internet) are not possible. This is achieved through a deprioritized store-and-forward protocol that leverages the existing node infrastructure.

4.1 Store-and-Forward Transport

Messaging data is treated as lower-priority traffic compared to consensus votes and block/transaction data. When a node has available bandwidth after handling its critical consensus duties, it must accept and relay message packets as part of its core network responsibilities. Messages are passed from node to node opportunistically, stored temporarily at each hop until a suitable forward path becomes available.

This store-and-forward model ensures that messages can eventually reach their destination even in highly partitioned or intermittent networks. To enable cryptographic verification of the relay path, each handoff between nodes generates a signed BLS receipt. These receipts are designed to be efficiently combined into a single, compact aggregate signature, which forms the basis for the final proof of delivery. This creates a verifiable chain of custody for the message as it traverses the network.

4.2 On-Chain Anchoring and Verification

To provide an immutable, on-chain proof of a message's existence and size, the sender broadcasts an anchor transaction. This transaction contains two key components: (1) the public key of the intended destination, and (2) a Zero-Knowledge (ZK) proof. The ZK proof is critical for the fee mechanism, as it cryptographically attests to the message's exact byte-length without revealing the content itself. This allows the protocol to calculate a fair, size-based fee. For efficiency, a compact proof system like Groth16 (288 bytes) can be used.

4.3 Fee and Incentive Structure

To be viable in a low-throughput environment, the messaging system's economic incentives must minimize on-chain traffic. The protocol therefore uses a streamlined two-transaction model: one to anchor the message and one to wrap up delivery.

1. **Message Anchor Transaction:** The sender broadcasts the anchor transaction, containing the destination's public key and the ZK proof of length. The protocol verifies the proof and charges a messaging fee proportional to the proven length, which is then deposited into the epoch reward pool.

2. **Delivery Wrap-up Transaction:** Once the message is delivered, any node can submit a wrap-up transaction containing: (1) an aggregated BLS signature encompassing all handoff receipts from the relay chain, and (2) the destination's acknowledgement signature. The protocol verifies the destination's signature against the public key from the anchor transaction, and upon success, marks the message as complete. This design prioritizes verification simplicity and low computational overhead while maintaining cryptographic integrity.

This model treats messaging as a public good that strengthens the entire network. By automatically pooling messaging fees, it incentivizes all nodes to maintain good connectivity and participate in forwarding. To ensure accountability, one could have the epoch reward pool distributing only to those validators who can prove they participated in at least one valid message relay during the epoch, preventing free-riders from benefiting without contributing.

5 Bridging and Interoperability

One of BunkerCoin's primary value propositions is serving as a resilient custody layer for digital assets during periods of internet instability or outright failure. To fulfill this role, the protocol must support trustless asset transfers between BunkerCoin and high-throughput networks like Solana or similar chains. This section describes a cryptographically secure bridge architecture that leverages BunkerCoin's existing consensus mechanisms, offering security guarantees superior to federated or multisig-based models.

5.1 A Native Consensus Bridge

The bridge is not an external or secondary system; it is a native extension of the core protocol. Its security model is two-tiered, separating the slow, deliberate process of validator set authentication from the high-frequency task of attesting to individual asset transfers. This ensures the bridge's root of trust is always anchored to the full, radio-capable consensus. The system comprises:

1. **The Full Validator Set**, which uses the core radio-based consensus mechanism at each epoch boundary to produce a signed *Validator Set Update Certificate* (VSCert). This certificate is the ultimate source of authority for the bridge.

2. **An Internet-Connected Validator Subset**, composed of validators with stable IP connectivity. This dynamic subset is authorized—as per its stake recorded in the most recent VSCert—to produce and sign *Bridge*

Certificates (BCerts) for individual asset transfers, keeping this high-volume traffic off the radio network.

3. **An Immutable On-Chain Program** on the target chain (e.g., Solana) that verifies both certificate types. It only accepts a BCert if it is signed by a super-majority of the internet-connected validators authorized by the most recent VSCert.
4. **A Permissionless Relayer Network**, composed of off-chain actors who submit these certificates to the destination chain. Relayers are trusted for liveness only.

5.2 Certificate-Based Verification and Versioning

To allow for future upgrades without compromising the security of existing deployments, all certificates are versioned. An on-chain program can support multiple certificate versions, allowing for a seamless transition to new bridge contracts over time while ensuring old, immutable contracts remain fully functional and secure.

5.2.1 Validator Set Update Certificates (VSCerts)

The bridge's root of trust is established by VSCerts. At the end of each epoch, as part of the main state snapshot process, the entire validator set signs a message containing the version, the complete list of public keys, and stake weights for the upcoming epoch. This aggregated signature and validator list form the VSCert. Because this process is part of the core, radio-based consensus, it is as secure and censorship-resistant as BunkerCoin itself. A relayer submits this VSCert to the on-chain program, which verifies it against the previous validator set, thus securely rotating the authorized set for the new epoch.

5.2.2 Bridge Certificates (BCerts)

BCerts authorize individual asset transfers, primarily to facilitate the movement of external assets onto BunkerCoin for secure custody. For example, when a user locks native `SOL` in the bridge program on Solana, internet-connected validators observe the event and produce a BCert to authorize the minting of wrapped `SOL` on BunkerCoin. Conversely, when a user burns wrapped `SOL` on BunkerCoin, the validators generate a BCert to authorize the release of the native `SOL` on Solana. This signed message contains the certificate version and transfer details (amount, asset, source, destination, nonce). The on-chain program verifies the BCert against the public keys of the currently authorized internet-connected validators. If valid, the mint or release is executed. If fewer than 2/3 of the total stake has internet connectivity, no BCerts can be produced, and the bridge pauses until connectivity is restored.

5.3 Superiority over Federated Models

This native, two-tiered design is fundamentally more secure than common trust-based bridge models. Systems like Wormhole rely on a fixed, permissioned set of so-called Guardian nodes, while many Layer 2 rollups secure their bridges with a small multisig. In these models, the security of the bridge is delegated to a committee that is external to and significantly smaller than the full validator set of the chains they serve.

BunkerCoin's bridge suffers from no such security degradation. Although only validators with stable IP connectivity can physically transmit their signatures, the BCert itself must carry an aggregated signature representing at least two-thirds of *all* staked weight. If this threshold is not met—because insufficient stake is online—the bridge simply pauses rather than falling back to a weaker committee. The authority to produce BCerts therefore remains identical to that of the base chain; the internet-reachable validators are merely the portion of the full set that can currently participate. The VSCerts that ratify this arrangement each epoch are signed by the entire validator set over the radio network, so no smaller group can hijack the bridge. Consequently, an attacker's only viable path to stealing funds is to corrupt $\geq 2/3$ of BunkerCoin's total stake—the exact same condition required to break main-chain consensus.

5.4 Merkle Root Anchoring for Enhanced Security

Beyond basic asset transfers, the bridge could *optionally* leverage BunkerCoin's existing state snapshot mechanism (Section 3) to post periodic commitments on external chains. These Merkle root anchors serve a dual purpose:

1. **Checkpoint Finality:** Once BunkerCoin's state root is included in the consensus of Solana or a similar chain, reverting BunkerCoin's history would require reorganizing the external chain—a massive increase in attack cost.
2. **Generalized State Proofs:** Applications can verify arbitrary BunkerCoin state claims using Merkle proofs against these anchored roots, enabling complex cross-chain protocols beyond simple transfers.

The same internet-connected validator subset that produces BCerts can generate these checkpoint certificates, reusing the existing BLS aggregation infrastructure without additional complexity.

6 Call for Contribution

This paper serves as a working foundation for the BunkerCoin protocol and is under active development. We welcome contributions from the community to expand, refine, and challenge the concepts outlined here. For further information on how to contribute, and to join the ongoing technical discussions, please join our community on Discord: `https://discord.gg/DGBqm92VQm`.