What Bitcoin's Stalled Proposals Tell You

Josh · Secure Sovereign

Published June 4, 2026 · Updated September 20, 2026

Why Bitcoin's most wanted improvements haven't shipped, and what it looks like when they do.

Contents

Bitcoin's consensus rules are sound, covering a small, well-defined set of operations: signature validation, UTXO accounting, block structure, and script execution. **Bitcoin Core is not those rules.** It is a 300,000-line C++ application that happens to carry them. The sacredness of the consensus layer does not extend to the monolith surrounding it, and **conflating the two is the primary mechanism by which the status quo defends itself.**

Bitcoin’s long-term value proposition depends on it being genuinely ungovernable by any single entity, whether a state, a corporation, or a development team. That property requires infrastructure diversity, meaning multiple independent implementations, multiple funding sources, and governance that no single party can capture. What exists today is the opposite: one implementation, a handful of maintainers, and funding concentrated in a small number of grant organizations. The infrastructure surrounding them is a single point of failure, and it has been producing the same failure mode for years.

Caution is appropriate for consensus changes. **It is not an explanation for why networking and privacy improvements with no consensus implications sit unmerged for years.** The features below are not controversial. They are not dangerous. They are research-complete improvements that a functional governance structure would have shipped. **The fact that they haven't shipped is the data.** For governance evidence behind the stall, see *Who Controls Bitcoin*. For the blocksize-war paralysis narrative, see *Governance Paralysis Was The Victory*. For numbered arguments and

monopoly-defense refutations, see *Bitcoin Governance: Argument Map* and *Bitcoin Core: The Biggest Fallacies*. For what consensus can close on blockspace, see *The Achievable Floor* and the three-BIP stack (Permanent Data Channel Closure, Static Per-Output Miner Fee, Dynamic Escalation).

The Governance Reality

When one implementation controls the defaults for the entire network, every improvement has to clear one gatekeeping process with a thin reviewer pool, a thin funding base, and institutional incentives that do not always align with node operators.

Across 17 years of Core commit history, the authorship Gini is ~0.851 historical / ~0.834 recent, where 0 is perfectly equal and 1 means one person controls everything. The top three merger accounts control 81.1% of historical merges and 82.2% since 2022. In a single recent year, over half of all merges flowed through one Brink-funded individual, a figure Brink itself published. Bitcoin Core has never had more than a handful of active maintainers in its entire history. That concentration is not a distribution that ships a long backlog of improvements. **It is a distribution that produces exactly the list below.**

Stalled Proposals

Dandelion (2017): research-complete, never merged

UTXO Commitments (~2014): research-complete, never merged

Erlay (~2019): years of research, still in the queue

Formal Specification: never written down; Core's behavior is the de facto spec

Wallet/Node Separation (2016): still bundled after a decade of universal agreement

Formal Verification: recommended by Quarkslab, not implemented

Package relay / BIP331: designed on the list as policy, not consensus; implementation PRs spent years under `[NO MERGE]`

and more...

Privacy and Relay

Dandelion has been research-complete since 2017. The mechanism is a two-phase propagation model: transactions travel a random path through the network in the stem phase before diffusing outward, making it significantly harder for a passive observer to determine which node originated a transaction. It was never merged.

The close comment on the Core implementation PR (#13947) is not a vote and not a "false sense of privacy" dismissal. The author closed it because stem routing is non-trivial against the wallet's own RBF and CPFP without opening DOS vectors, and pointed at a thinner "Dandelion Light" first step. That is a bundled wallet-and-node problem, not a privacy-philosophy problem. Dandelion still reduces leakage to passive network observers. The stall is that the monolith will not take a privacy hop that collides with policy it already shipped. See `findings/ARCHIVE_GEMS.md`.

What eventually landed in Core was private broadcast, merged years after Dandelion was first proposed. Private broadcast protects your IP from the specific peer you connect to. That is a narrower threat model. Both are useful. They are not substitutes, and private broadcast landing does not close the Dandelion argument.

Erlay would cut node bandwidth on transaction relay using set reconciliation rather than flooding. The author's measurements on the full-protocol PR sit in the 20–50% range; a NACK in the same thread called the patch premature on efficiency grounds. Full-protocol Erlay is **0/7** merged. Scaffolding merges are not delivery. The stall is a measurement fight that never closed, not an empty objection docket. See `findings/STALLED_PROPOSALS_REPORT.md`.

Package relay was announced on bitcoin-dev as mempool policy, "not consensus or P2P protocol changes." The large implementation PR is titled `[NO MERGE]` until maintainers have rough consensus on approach (#27742). That is the 2022+ informal-before-PR flow of 0.002 in prose. The list still names the topic. GitHub refuses to treat the first large PR as a decision.

Sync and Verification

UTXO commitments date to at least 2014. The idea is to commit the current state of all unspent outputs in a form a new node can verify against the proof-of-work chain, rather than replaying over a decade of transaction history from genesis. Initial block download currently takes days to weeks. A working commitment scheme could cut that to hours or less while keeping the trust anchor in the chain itself. The research has been complete for over a decade. **Core has not shipped it.** What that unrecovered IBD burden costs operators is modeled in *Full Cost of Running a Bitcoin Node*.

Formal Verification. The Quarkslab audit, the first external professional security audit in Core's history, was scoped primarily to the P2P networking layer over 100 man-days. Quarkslab states directly that existing harnesses "do not fully exercise the complete range of consensus checks" and that "most existing harnesses target narrow, well-scoped portions of the code in isolation." Based on their documented scope and the gaps they identify, the audit covered an estimated 20 to 30% of consensus-critical code paths. Finding no critical issues against that baseline is a different claim from confirming the consensus code is secure.

Quarkslab explicitly recommended formal verification as the path forward. Not an external critic making that call, but the auditors who spent 100 man-days inside the codebase, commissioned by Brink, telling the Core development community what the next step should be. It has not happened.

Structural Issues

The Missing Specification. Bitcoin Core has never produced an independent mathematical specification of its consensus rules that the network treats as a pass-or-fail standard sitting above any one program. BIPs record proposals. They are not that spec. A specification written outside Core does not dissolve the moat until operators and implementations actually use it. Every mature protocol at the infrastructure level has a formal specification that exists independently of any single implementation: TCP/IP, HTTP, TLS, SMTP. The specification defines the protocol. Implementations conform to it. When an implementation diverges from the spec it is wrong, and you can say so precisely. When the network has no adopted spec, the dominant implementation is the spec by default, which means whoever controls the dominant implementation controls the protocol definition. Not through any explicit authority, but through the structural fact that there is nothing else to appeal to.

This is not an accident that nobody got around to fixing but a condition that has been actively maintained. The response to calls for a formal specification has been consistent. The code is the spec. Bitcoin is too complex to specify formally. Any written spec would inevitably diverge from the implementation. Each of these objections has the same practical effect, which is keeping the no-spec condition in place, keeping Core as the sole authority on what the rules actually are, and making it structurally impossible for any independent implementation to prove consensus compatibility without deferring to Core.

Without a specification, any alternative implementation must reverse-engineer undocumented behavior from Core itself. Core stays the thing you check against, even when the other program is built differently. When a consensus-critical behavior changes in Core, an independently written client that wasn't tracking that specific change becomes a chain split risk. Following Core closely is the safe path, which is why most serious attempts stay Core forks. Independently written clients exist. btcd keeps diverging. libbitcoin exists and the ecosystem cannot use it without being rebuilt.

This is why the “just fork it” response to governance criticism is a non-answer. A Core fork inherits the same undocumented behavior, the same 300,000 lines of technical debt, and the same capture vectors. The governance problem follows the code.

This is why no independently written client has been a real check for 17 years, not because writing one is impossible but because the no-spec condition makes proving consensus compatibility structurally impossible without deferring to Core. The moat protecting Core’s position is not technical excellence. It is the absence of a document. libbitcoinkernel, the project intended to extract Core’s consensus logic into a reusable library, confirms rather than refutes this. The proposed solution to implementation monoculture is having every other implementation run Core’s code. That is not diversity. That is a franchise.

Wallet and Node Separation has had universal conceptual agreement since GitHub issue 7525 in 2016. Running with -disablewallet is not separation. The wallet code is still present in the binary, still compiled, still part of the same release process. The governance problem follows the dependency regardless of which version of Core’s code you ship.

Policy Monoculture and Revealed Preference

When one implementation sets policy defaults for the entire network, every policy disagreement becomes an all-or-nothing fight for control of what that implementation ships. The OP_RETURN debate was not about 80 bytes. The mempool policy fights, the dust limit arguments, the RBF debates are structurally identical, civil wars over a throne that exists only because there is one throne to capture.

The OP_RETURN debate is worth dwelling on because it illustrates the mechanism precisely. Bitcoin Core merged a change in 2025 raising the default OP_RETURN data limit, a relay policy decision with no consensus implications. Operators who disagreed had no meaningful recourse inside the Core process. For many, the only way to enforce a different policy was to run different software. That is what they did. In 2025 Knots, a Core fork with stricter defaults, went from under 2% to roughly 20% of the reachable network in the months following the merge. That was a policy switch on Bitcoin, still Core's code. A five-fold move in reachable nodes driven by a relay policy disagreement.

Policy does not bind miners who bypass relay. The 83-byte cap belongs at consensus; that is *Permanent Data Channel Closure*.

The demand for policy plurality was always there. The monoculture was suppressing it. When a conservative alternative presented itself, a significant fraction of the network moved to it immediately.

What Better Looks Like

Bitcoin Commons is a from-scratch Rust client built from a human-readable mathematical specification called the Orange Paper. Same chain, same 21 million. It is not a Core fork or a thin wrapper around Core's consensus logic. Cryptographic merge authorization is designed and stays off until security review, key management, and community validation. The design is meant to make the merge path harder to capture than informal GitHub authority.

The Orange Paper and the BLVM

The Orange Paper is a formal mathematical specification of Bitcoin's consensus rules. It defines signature validation, UTXO accounting, script execution, and block structure as mathematical objects with precise definitions rather than as the emergent behavior of a C++ codebase. The BLVM consensus layer is a pure implementation of that spec: deterministic, side-effect-free functions that directly implement the mathematical definitions with no room for undocumented behavioral drift.

The spec is a public good that extends beyond Commons. Any implementation can verify against it. Consensus rule disputes can be resolved by appealing to a document rather than deferring to whoever controls the dominant codebase. The no-spec moat dissolves the moment the spec is published, for everyone.

The BLVM Specification Lock

The BLVM Specification Lock uses Z3 to check annotated implementation paths against the Orange Paper. Coverage will expand. That is not a claim every consensus property is proven on every merge, or that every CI job has zero-divergence proof to tip. When a locked check fails, the merge fails because the contract failed, not because a reviewer happened to notice.

That is the direction Quarkslab named as the next step for Bitcoin consensus code. Commons is building it.

The Specification Lock also changes who can safely contribute. The primary risk with AI-generated code is subtle behavioral changes that pass code review but alter consensus-critical behavior in ways that are hard to catch. A lock that actually covers the changed path makes that tractable. The barrier shifts from years of pipeline navigation toward writing code that satisfies the contracts.

UTXO Commitments Without a Consensus Change

The Commons UTXO commitments implementation achieves fast sync without modifying block headers, without a soft fork, without any consensus change. It operates entirely at the P2P layer using a sparse Merkle tree commitment verified against the existing proof-of-work chain.

The trust model is an honest majority of diverse peers, not a single snapshot publisher. The mechanism requires 80% agreement across peers spread by ASN, country, and subnet, so no one peer or one hosting region can feed you a fake set. You download headers, select a checkpoint, query those peers for their UTXO commitment at that height, check it against the header chain and accumulated proof of work, then sync forward with full incremental validation. A hybrid mode does this while verifying the full chain from genesis in the background.

The result is fast sync with no hardcoded hashes, no dependency on any release team, and no new consensus rules required, under that peer-majority assumption rather than under a baked-in release hash.

Networking and Protocol

Dandelion++ is in the build, covering the threat model that private broadcast does not. The networking layer runs on Iroh and QUIC with NAT traversal alongside the standard Bitcoin P2P transport, so nodes behind residential connections that currently struggle to maintain reliable peer connections get real improvement. High-performance block relay uses Fibre with UDP transport and Reed-Solomon forward error correction. Package relay per BIP331 is implemented. The mempool supports four configurable RBF modes.

Policy is modular. You can configure your own relay and mempool defaults without affecting anyone else's consensus participation. The dynamic that turns policy disagreements into civil wars under the monolith loses most of its energy when there is no single default to capture.

Architecture

All consensus logic lives in blvm-consensus, fully separated from storage, networking, and RPC. The architecture is six clean layers from the Orange Paper down to governance enforcement, with explicit interfaces between them. A developer working on the consensus layer cannot accidentally break the networking layer because the boundary is enforced architecturally, not by convention. That separation is what makes formal verification tractable. You cannot formally verify 300,000 lines of entangled C++. You can formally verify a bounded, well-defined consensus module.

The BLVM's secp256k1 implementation is pure Rust and benchmarks 10 to 22% faster than libsecp256k1 across signature verification workloads.

Governance

Cryptographic merge authorization is designed and stays off until security review, key management, and community validation. The design has no self-merge at protocol level. Maintainers cannot simultaneously propose and merge their own code. Crate-scoped voting weights contributions proportionally to the subsystem being changed, conflict of interest disclosure is a required part of the process, and the reviewer pool is not gated by years of pipeline navigation through fellowship programs that select for ideological alignment as much as technical capability. The barrier to safe contribution is writing code that satisfies the contracts, which is a technical bar, not a social one.

The Proof

The bottleneck was never engineering. The Dandelion papers are published. The UTXO commitment schemes are fully specified. Erlay has years of development behind it. Formal verification tooling exists and works. The objections raised against these proposals, when they exist at all, fall into three kinds. Some are strawmen that substitute stronger claims for the ones actually being made. Some describe the status quo as if that were an argument against changing it. Some concede the research is sound and say the problem is prioritization.

What they share is a governance structure that cannot process improvements when the coordination cost exceeds the threshold the structure can clear, or when the improvement threatens the institutional position of the people whose funding depends on the status quo. An authorship Gini of 0.851 across 17 years of commit data is not an accident. It is the predictable output of a system where access to merge authority is controlled by a small group with aligned institutional interests and no formal accountability to the node operators running the software.

The conservative governance argument has merit for consensus changes. It has no application to a P2P privacy proposal sitting unmerged for years because it collided with wallet policy the same tree already shipped. At some point the distinction between intentional conservatism and structural paralysis requires evidence, and the evidence is the list.

Bitcoin is the only monetary network in history not ultimately controlled by a state. That property is fragile. It depends on the network remaining genuinely decentralized at every layer, including the software layer. A network where one development team controls the only production-grade node implementation, where that team’s funding flows from a handful of grant organizations, and where the protocol definition exists only as the emergent behavior of that team’s codebase is not decentralized at the software layer. It is a single point of failure dressed in the language of decentralization.

Commons demonstrates the technical problem is solvable as a program, not as a finished certificate. The Orange Paper exists. The Specification Lock checks annotated paths. Full-chain differential replay against Core is the scale test, operator-driven, and not a CI-to-tip claim. UTXO commitments and Dandelion++ are in the Commons build as engineering answers to the stalled list. That is not a claim the money already runs on those tools. The features that sat unmerged for years can be built from a specification rather than reverse-engineered from a 300,000-line monolith.

The argument is the artifact: a spec, a second client, and a differential program, not a speech about Core.

Sources

- Fanti, G. et al., "Dandelion: Redesigning the Bitcoin Network for Anonymity", arXiv:1701.04439, 2017
- BIP 156: Dandelion, Privacy Enhancing Routing
- BIP 330: Transaction Announcements Reconciliation: Naumenko, G. and Wuille, P., created 2019-09-25
- Bitcoin Optech: Erlay
- Bitcoin Core Issue #7525: Separate Node and Wallet Functions, filed February 12, 2016
- Rusty Russell, "Pettycoin Revisited Part I: UTXO Commitments": Friedenbach, M., Todd, P., Miller, A. et al., 2014
- Quarkslab, "Bitcoin Core Security Audit", November 2025: full report
- Bitcoin Governance Research: 16-year commit history / merge concentration
- Brink Engineering Impact Report 2025
- Bitcoin Core PR #32359: Remove OP_RETURN size limits, 2025
- Bitnodes.io: Bitcoin Knots node statistics

SECURESOVEREIGN

Markdown is the canonical source: https://secsov.com/articles/what-bitcoins-stalled-proposals-tell-you/index.md

Formatted snapshot of https://secsov.com/articles/what-bitcoins-stalled-proposals-tell-you

Josh · Secure Sovereign