Composable External Verification
Bitcoin & Beyond as Verifiable Inputs Under Bounded Resources

Purpose: Define a composable model for verifying external facts (starting with Bitcoin) inside a bounded verifier environment, without turning any system into a full cross-chain execution engine.

Non-goal: This is not a bridge product specification, not a custody claim, not a guarantee of canonical history, and not a promise that any particular chain feature exists today.

1. Motivation

Most chains treat "external truth" as something you ask an oracle for. Composable External Verification (CEV) treats external truth as something you can prove within explicit bounds:

- Verify a Bitcoin event (e.g., inclusion of a transaction output) without trusting a third party.
- Reduce dependency on RPC providers and off-chain intermediaries.
- Compose verified external facts into application logic (escrow, attestations, settlement triggers) in a manner that is bounded in cost, explicit in assumptions, and refusal-forward under uncertainty.

CEV is not "make one system run another." CEV is: accept external commitments as inputs when they can be validated under bounded verification.

2. Definitions

2.1 External System

An external chain or system X with:

- a commitment sequence (e.g., headers),
- a consensus rule (e.g., PoW),
- inclusion commitments (e.g., Merkle roots).

Examples: Bitcoin, other PoW chains, checkpointed ledgers, signed transparency logs.

2.2 External Fact

A statement about X that can be represented as a predicate:

$\phi_X(\pi, z) \in \{0,1\}$

Where:

- z is the claim (e.g., "TXID t included in block h with confirmations ≥ d").
- π is the proof bundle required to validate z under a verification rule-set.

2.3 Verifier

A bounded client that:

- does not enumerate global histories,
- retains only a finite window of commitments,
- refuses when assumptions are not met.

3. Core Claim

CEV is a verification interface:

"Given a proof bundle π, accept the external fact z only if it is cryptographically consistent with a commitment sequence retained (or retrievable) under bounded rules."

It composes cleanly with:

- Header-only verification (commitment tracking)
- Bounded inclusion (effect proofs, not transaction enumeration)
- Minimal frontier verification (temporal coherence under a window k)
- Genesis-anchored lineage (offline-resilient trust roots)

CEV does not require that every node re-verify everything about Bitcoin. Only verifiers that care about a Bitcoin fact need to validate its proof bundle.

CEV composes with genesis-anchored lineage verification and bounded verifier lifecycles—it does not replace them. External facts become verifiable inputs to local state machines that already maintain their own temporal coherence.

4. Bitcoin as the Reference External System

Bitcoin is the canonical starting point because it provides:

- a globally recognized commitment sequence (headers),
- a deterministic work function (PoW),
- inclusion commitments (Merkle roots),
- widely available header distribution.

Bitcoin's multi-decade operational history and broad social finality make it a practical foundation for verifiable external inputs—not because its consensus is formally verified by all observers, but because its commitment sequence is widely replicated and its fork-choice rule converges under standard network conditions.

CEV begins with Bitcoin SPV-style verification, then generalizes.

5. Bitcoin Verification Interface

5.1 Proof Bundle for a Bitcoin Inclusion Fact

Claim z:

• z := "Transaction t is included in Bitcoin block B and buried by depth d."

Proof bundle π contains:

1. Header sequence $H_{start..tip}$
2. Merkle inclusion path $p(t \rightarrow merkleRoot(B))$
3. Confirmation rule parameters (d)
4. Optional checkpoint anchor (trusted or genesis-anchored, depending on mode)

Verification predicate:

$\phi_{BTC}(\pi, z) = ValidHeaders(H) \wedge ValidPoW(H) \wedge Inclusion(t, p, merkleRoot(B)) \wedge Depth(B, tip) \geq d$

5.2 What This Proves (and What It Doesn't)

Provides:

• Cryptographic evidence that t is included under a header sequence satisfying Bitcoin's PoW rule and a depth threshold.

Does not provide:

• A universal guarantee that this is the globally canonical Bitcoin chain (network partitions exist).
• Censorship detection.
• A guarantee that all users see the same chain unless they share trust roots / checkpoints.

CEV is refusal-forward: if it cannot validate under configured assumptions, it returns Unknown, not "probably true."

6. Composability: From Facts to Contracts

CEV becomes powerful when external facts are inputs to local state machines.

6.1 Oracle Example (Verified External Price Feed Wrapper)

Instead of "trust this oracle value," CEV encourages:

• Oracle publishes a signed statement s plus evidence of provenance.
• Verifier accepts s only if it is anchored to an external transparency log or externally verifiable chain event.

This yields a verifiable oracle pattern:

$\text{Accept}(s) \Leftarrow \text{SigValid}(s) \wedge \phi_X(\pi, z)$

This is still not "truth," but it upgrades oracle integrity from "trust me" to "prove your provenance." Unlike committee-based oracles that rely on honest-majority assumptions, this pattern allows any observer to independently verify the attestation chain.

CEV is not an oracle system: it does not aggregate off-chain observations into on-chain attestations. It validates that a claim about an external system is consistent with that system's own cryptographic commitments. Where oracles say "we observed X," CEV says "X is provable under this commitment sequence."

6.2 Escrow Example (Bitcoin Payment Unlock)

A local escrow state machine can require:

"Release asset A iff BTC UTXO u is spent to script S with depth ≥ d."

The contract logic consumes:

• a verified spend fact z,
• validated by ϕ_{BTC}.

Escrow becomes:

• deterministic,
• verifiable,
• and auditable by any verifier that can validate the same proof bundle.

6.3 Bridge Example (Trust-Minimized Mint on Proof)

A "bridge" here means a fact-triggered mint, not a generalized cross-chain execution environment.

• Deposit BTC to a known script.
• Produce SPV proof of inclusion + depth.
• Mint representation inside the local system.

This is powerful but requires strong boundaries:

• You must explicitly specify redemption rules, fraud handling, and operational assumptions.
• You must state that "canonical Bitcoin history" is not automatically determined for all observers.
• Reorg sensitivity warning: Proof-triggered minting is exposed to chain reorganizations near the specified depth threshold. A conservative d reduces reorg risk but does not eliminate it. Systems must define explicit refusal or rollback behavior when reorganizations invalidate previously accepted proofs.

CEV can minimize trust, but cannot remove all coordination problems from bridging. Unlike multisig federations or validator committees, CEV shifts verification responsibility to proof construction—but the verifier must still define what happens when proofs fail or conflict.

7. Threat Model

CEV explicitly models:

7.1 Eclipse / Isolation

A verifier may be fed a consistent-but-false view of headers.

Mitigations:

- multi-source header gossip,
- pinned checkpoints,
- diversity requirements.

7.2 Proof Withholding

A fact might be true but proofs are unavailable.

Correct behavior:

- return Unknown,
- do not "green check" a missing proof.

7.3 Reorg Near Threshold

A depth d is probabilistic finality.

Mitigation:

- choose conservative d,
- define refusal behavior during instability,
- optionally require stronger checkpoints.

8. Operational Modes

CEV should be presented as modes, not a single claim.

Mode A: Pure SPV Mode (Best-effort, decentralized)

- validate headers from peers,
- validate PoW and depth,
- accept facts under configured d,
- refuse when insufficient header corroboration exists.

Mode B: Checkpoint-Pinned Mode (Stronger offline consistency)

- pin a checkpoint header hash (or multiple),
- accept only chains extending that checkpoint,
- higher resistance to eclipse at cost of assuming checkpoint correctness.

Mode C: Genesis-anchored Composition Mode

- treat a genesis-anchored lineage root as the "time origin" for local state,
- compose external proofs as inputs whose acceptance is bounded and scoped,
- do not claim this replaces full data availability or full execution verification.

9. Minimal Interface Spec

A CEV module exposes:

9.1 Inputs

- chain identifier X
- claim z
- proof bundle π
- verification policy P (mode, thresholds, sources)

9.2 Outputs

- Proven (accepted under explicit scope)
- Consistent-but-expiring (accepted but nearing retention horizon)
- Unknown (insufficient data / policy not satisfied)
- Rejected (invalid proof / contradiction / rule violation)

9.3 Determinism Requirement

For any (π, z, P), all conforming implementations must return identical results:

$V_X(\pi, z; P) \rightarrow$ {Proven, Expiring, Unknown, Rejected}

10. Extending Beyond Bitcoin

Bitcoin is the reference implementation because it is well-defined.

"Beyond" means: other external systems that provide:

- commitments,
- verifiable inclusion,
- a stable fork-choice rule (even if checkpointed),
- and a practical proof distribution path.

Examples:

- other PoW chains (similar SPV model),
- signed transparency logs (inclusion proofs),
- checkpointed ledgers (proof-of-signature committees),
- data availability layers (proof-of-publication claims).

Each system X requires:

- a specific predicate ϕ_X,
- explicit non-guarantees,
- and explicit operational assumptions.

CEV is a pattern, not a claim that all external systems are equally verifiable.

11. Limits and Non-Guarantees

CEV cannot guarantee:

- Canonical global history for all observers without additional trust roots or coordination.
- Censorship resistance in proof distribution.
- Global data availability of external systems.
- Transaction identity semantics beyond what the external proof system itself provides.
- Universal agreement between independent verifiers.

CEV is not magic. It is bounded verification with explicit scope.

12. Why This Matters

Most distributed systems fail at the same place:

- users rely on RPC endpoints they do not control,
- applications rely on oracle committees with opaque selection,
- bridges rely on multisig federations with underspecified slashing.

CEV shifts the default posture from:

"trust this external statement"

to:

"accept this external statement only when it is provably consistent with a commitment sequence under bounded rules."

It does not eliminate trust from all systems. It makes trust explicit, scoped, and composable.

13. Summary Statement

Composable External Verification is a bounded verification interface for importing external facts (starting with Bitcoin SPV-style proofs) into local application logic.

It should be viewed as:

- a trust-minimizing primitive for verifiable inputs,
- refusal-forward under uncertainty,
- composable with bounded verification and lineage anchoring,
- and explicitly not a replacement for execution verification or data availability guarantees.

Appendix A: Minimal Bitcoin Inclusion Proof Sketch

Given:

- txid t
- Merkle path p (hashes + left/right positions)
- block header B with merkle root R

Compute:

$h_0 = t$
$h_{i+1} = H(h_i \,||\, p_i)$ or $H(p_i \,||\, h_i)$ depending on position

Accept inclusion if:

$h_n = R$

Then require depth rule over valid PoW headers.

Appendix B: Practical Reader Guidance

If you are evaluating CEV, focus on:

- what is proven vs what is assumed,
- how refusal is handled,
- how header sources are diversified,
- how checkpoints are chosen (if any),
- and how proof unavailability is surfaced.

A system that hides uncertainty is not a verifier. It is an oracle with a UI.