Bridging UTXO Chains to EVM:

A Federated Multisig Bridge with a Roadmap to Trustless Verification

The Wojak Bridge Project

https://bridge.wojakcoin.cash

Version 1.0 · June 2026

Abstract

We present the design and security model of a two-phase cross-chain bridge connecting Proof-of-Work UTXO chains—specifically Wojakcoin (WJK) and Junkcoin (JKC)—to EVM-compatible networks, with Base (an Ethereum Layer-2 rollup) as the initial deployment target. In Phase 1, the bridge operates under a federated *2-of-3 multisig* custody model where an independent relayer set collectively controls both the UTXO-chain P2SH custody wallet and the EVM-side mint/burn authority. We analyse the threat surface, liveness guarantees, and economic security of this design and enumerate its trust assumptions explicitly. In Phase 2 we outline a roadmap toward a *trustless* bridge that removes operator trust through light-client proof verification, succinct non-interactive proofs (SNARKs/STARKs), and on-chain SPV relay, drawing on recent advances in zero-knowledge cryptography. Our staged approach allows the bridge to be operational and auditable today while providing a credible path to removing all privileged operators in the future.

Disclaimer: This document is provided for informational purposes only. It does not constitute financial advice. Smart contract usage involves risk. Always do your own research.

0. Contents

1. Introduction

Proof-of-Work UTXO blockchains such as Bitcoin, Litecoin, and their derivatives represent some of the most battle-tested distributed ledgers in existence. Yet their scripting systems are intentionally limited: arbitrary programmability, DeFi composability, and token standards remain out of reach on native UTXO chains without significant protocol modification.

EVM-compatible networks, by contrast, provide a rich smart-contract environment but lack direct access to the economic security and monetary policy of established UTXO chains. Bridging the two worlds unlocks compelling use cases: holders of WJK or JKC can participate in decentralised exchanges, liquidity provision, and lending markets on Base, while the native chains retain their security properties and monetary supply caps.

Cross-chain bridges are, however, among the most exploited components in the broader blockchain ecosystem [1]. The fundamental challenge is establishing *shared state knowledge*: an EVM contract must learn—without trusting any single party—that a UTXO transaction paying into a custody address has been confirmed to a sufficient depth.

This paper makes the following contributions:

1. We present the *Wojak Multisig Bridge*, a production-deployed 2-of-3 federated bridge, detailing its on-chain and off-chain components, security assumptions, and economic model (Section 3).
2. We provide a formal threat model and enumerate the residual trust assumptions of the current design (Section 4).
3. We outline a concrete roadmap from the current federated model to a fully trustless bridge using zero-knowledge proofs and on-chain light clients (Section 5).

2. Background

2.1 UTXO Model and P2SH Multisig

In the Unspent Transaction Output (UTXO) model, the blockchain maintains a set of unspent outputs rather than account balances. Each output is locked by a *scriptPubKey*; a spending transaction must provide a *scriptSig* that satisfies the locking condition.

Pay-to-Script-Hash (P2SH, BIP 16 [2]) allows the locking condition to be expressed as the hash of an arbitrary *redeem script*. For an m-of-n multisig, the redeem script is:

$$\texttt{OP\_}m\ \texttt{<pubkey}_1\texttt{>}\ \ldots\ \texttt{<pubkey}_n\texttt{>}\ \texttt{OP\_}n\ \texttt{OP\_CHECKMULTISIG}$$

Spending requires m valid signatures from distinct keys. Keys are sorted lexicographically (BIP 67 [3]) to ensure a canonical, deterministic redeem script across independent signers. The bridge uses $m = 2$, $n = 3$.

2.2 EVM Mint/Burn Token Model

On the EVM side, bridged assets are represented as ERC-20 tokens with a controlled mint/burn mechanism. The `wWojakcoin` and `wJunkcoin` contracts implement the following invariant:

Definition 1 (Reserve Invariant)**.** *At any block b, let S_b be the total supply of the wrapped ERC-20 token and R_b be the number of native coins held in the P2SH custody wallet with at least k confirmations. The bridge targets $S_b \leq R_b$.*

A surplus ($S_b < R_b$) arises from bridge fees or unclaimed refunds. A deficit ($S_b > R_b$) is impossible under correct operation and would indicate a theft or bug.

2.3 The Cross-Chain Communication Problem

Neither blockchain has native awareness of the other. An EVM contract cannot directly query whether a UTXO transaction has been confirmed. All existing solutions introduce some trust assumption:

Approach	**Trust Model**	**Maturity**
Federated multisig	Honest majority of operators	Production
Optimistic relay	Fraud-proof window + watchers	Research/early prod
Light-client relay	Proof-of-Work header chain	Research
ZK proof of inclusion	Cryptographic soundness	Early research

Table 1: Cross-chain communication approaches and their trust models.

3. Phase 1: The Federated Multisig Bridge

3.1 Architecture Overview

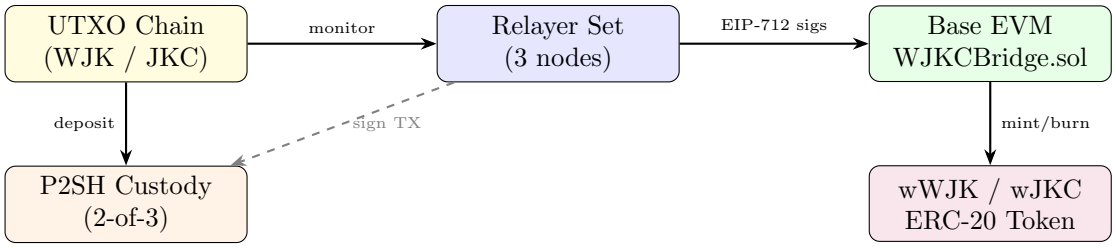

Figure 1: High-level architecture of the Wojak Multisig Bridge.

The bridge consists of three logical layers:

1. **UTXO Custody Layer.** A 2-of-3 P2SH address controlled jointly by the three relayer public keys. Users deposit native coins here to initiate a bridge-in (wrap) operation.

2. **Relayer Coordination Layer.** Three independent relayer processes monitor both chains. For deposits they aggregate EIP-712 signatures off-chain before submitting a single `registerDepositClaim` call. For withdrawals one leader builds and broadcasts a multisig UTXO transaction; co-signers independently verify and countersign.

3. **EVM Smart Contract Layer.** `WJKCBridge.sol` verifies relayer signatures on-chain and gates all minting/burning. The `wWojakcoin`/`wJunkcoin` ERC-20 tokens implement a strict single-minter model; the bridge is the only authorised minter.

3.2 Deposit Flow (UTXO → EVM)

1. User sends native coins to the P2SH custody address, embedding their EVM recipient address in an `OP_RETURN` output.

2. Each relayer's `DepositProcessor` detects the UTXO transaction via an Electrs REST endpoint once $k_{\min}$ confirmations are reached (default: 2).

3. Relayers sign an EIP-712 structured message encoding the deposit:

$$\textsc{DepositClaim}(\mathit{txid}, \mathit{vout}, \mathit{amountSats}, \mathit{recipient}, \mathit{chainId})$$

4. Once ≥ 2 signatures are aggregated, any relayer submits `registerDepositClaim` to `WJKCBridge.sol`, which verifies the signatures, deduplicates by *txid*, and emits an event.

5. The recipient calls `claimDeposit`, triggering an ERC-20 mint for $\lfloor amountSats \cdot (1 - f) \rfloor$ units (where $f = 0.5\%$ is the bridge fee).

3.3 Withdrawal Flow (EVM → UTXO)

1. User calls `initiateWithdrawal(amountSats, jkcAddress)` on the bridge, burning their wrapped tokens immediately.

2. The `WithdrawalProcessor` on each relayer detects the `WithdrawalInitiated` event.

3. A deterministic leader is elected: leader $= H(evmTxHash) \bmod 3$, rotating by round if the leader is unresponsive.

4. The leader constructs a UTXO transaction paying the user minus fee, with an `OP_RETURN` embedding the EVM tx hash for auditability.

5. Co-signers independently verify the proposed outputs against the EVM event before signing. The leader broadcasts once threshold is met.

6. Any relayer calls `confirmWithdrawal` with the UTXO txid, closing the on-chain withdrawal record.

3.4 Smart Contract Design

3.4.1 WJKCBridge.sol

Key design decisions:

- **Signature verification.** EIP-712 typed-data signatures prevent cross-chain and cross-domain replay. The domain separator includes `chainId` and `verifyingContract`.
- **Deduplication.** A `processedTxids` mapping prevents double-minting from the same UTXO output.
- **Ownership.** A Gnosis Safe (2-of-3 threshold) holds the `owner` role from block 0. The deployer EOA has no residual admin power.
- **Parameter timelock.** All parameter changes (signers, threshold, fee) are subject to a 24-hour timelock enforced on-chain.
- **Fee accounting.** Fees accumulate in the bridge as `wjkc` tokens, sweepable only by the owner to a designated address.

3.4.2 wWojakcoin / wJunkcoin ERC-20

- **Single minter.** Only the bridge contract can call `mint()` or `burn()`. The minter role is transferred via a 2-step timelocked handoff (2-day delay) to prevent abrupt authority changes.
- **Hard cap.** `MAX_SUPPLY` is set at deploy time to the native chain's monetary cap (42 M WJK; 54 M JKC) and is immutable.
- **8 decimals.** Matching native satoshi precision to avoid rounding on conversion.
- **ERC-20 Permit.** Gasless approvals via EIP-2612 permit signatures.

3.5 Relayer Design

3.5.1 Signer Isolation

Each relayer holds its own EVM private key (for EIP-712 signing) and UTXO private key (for P2SH multisig). Private keys never leave the relayer node. Peer-to-peer communication uses a lightweight Flask API over WireGuard / ZeroTier overlay networks; no relayer exposes a public API surface.

3.5.2 Double-Spend Prevention

The withdrawal pipeline implements three independent layers of protection:

1. **On-chain deduplication.** The `PayoutCoordinator` (optional module) requires the leader to claim a withdrawal before soliciting co-signatures; followers reject proposals that do not match the active on-chain claim.
2. **Local sign-once registry.** Each relayer maintains a persistent SQLite store that records the exact UTXO inputs+outputs tuple signed for each withdrawal ID. A relayer will never sign two different tuples for the same ID.
3. **Payout existence check.** Before constructing or co-signing, the relayer queries the custody address for an existing `OP_RETURN` matching the EVM tx hash. If found, the withdrawal is already complete.

3.5.3 Liveness and Leader Rotation

The system is live if at least 2-of-3 relayers are online. Leader election is anchored to the EVM block timestamp of the `WithdrawalInitiated` event (a value shared by all relayers). If the round-0 leader fails to broadcast within `LEADER_TIMEOUT_S`, leadership passes to the next relayer sequentially—exactly one leader is eligible at any instant.

4. Threat Model and Trust Assumptions

4.1 Explicit Trust Assumptions (Phase 1)

Assumption	Consequence if violated
≥ 2 of 3 relayers are honest and online	A colluding majority can drain the P2SH custody wallet to arbitrary addresses, or censor deposits/withdrawals.
EVM smart contracts are correct	A bug could allow unauthorised minting or burning. Mitigated by audit and immutable `MAX_SUPPLY`.
UTXO chain has no deep reorg past $k_{\min}$	A reorg deeper than confirmation threshold could cause a deposit to be counted twice if the spending tx is re-observed.
Safe multisig signers are independent	Collusion of Safe signers allows parameter changes (after timelock) but cannot directly steal custody funds.

Table 2: Trust assumptions in the Phase 1 federated design.

4.2 Attack Vectors and Mitigations

Relayer collusion. If ≥ 2 relayers collude, they can sign arbitrary UTXO withdrawal transactions. Mitigation: geographic and organisational separation of relayer operators; transparent on-chain event log allows post-hoc detection; reserve monitoring with automated alerts.

Smart contract exploit. A reentrancy or arithmetic bug could allow over-minting. Mitigations: checks- effects-interactions pattern; `MAX_SUPPLY` hard cap; OpenZeppelin audited base contracts; no owner upgrade path (immutable logic).

Front-running / MEV. `claimDeposit` can be front-run since the deposit event is public. Mitigation: the claim is bound to a specific recipient address embedded in the `OP_RETURN`; a front-runner would mint tokens to the legitimate user (no financial gain).

Eclipse attack on relayer. A network-level attacker isolating a relayer from the canonical chain tip could cause it to act on stale data. Mitigation: relayers use multiple redundant RPC endpoints and cross-validate tip height.

5. Phase 2: Roadmap to a Trustless Bridge

The federated model's critical weakness is its reliance on operator honesty. We outline a three-stage path to removing this trust assumption entirely.

5.1 Stage A: Optimistic Verification with Fraud Proofs

As an intermediate step, we can introduce an *optimistic relay*:

1. Any party may submit a UTXO block header to an on-chain `HeaderRelay` contract, staking a bond.
2. A challenge window (e.g., 7 days) allows anyone to submit a fraud proof if the header is invalid or does not extend the heaviest chain.
3. After the window, a deposit claim referencing a confirmed-depth inclusion proof against the relayed header is accepted without relayer signatures.

This approach reduces trust to: no coalition of adversaries can sustain censorship of fraud proofs for the full challenge window. It has precedent in optimistic rollup designs [4].

5.2 Stage B: On-Chain Light Client (SPV Relay)

Bitcoin-style Simple Payment Verification (SPV) allows verifying transaction inclusion with $O(\log n)$ data using a Merkle proof against the block header. An on-chain light client maintains a running chain of PoW block headers and exposes a `verifyInclusion(txid, proof, blockHeight)` function.

Key challenges for UTXO chains with GPU-friendly PoW:

- **Header validation gas cost.** Verifying a SHA-256d block header on EVM costs approximately 20 000–60 000 gas. At 1 block per minute, this is feasible but not cheap.
- **Chain selection.** The contract must implement the heaviest-chain rule. Storing the full header chain on-chain is prohibitively expensive; a checkpoint-based approach with fraud proofs is more practical.
- **Difficulty adjustment.** The contract must replicate the target recalculation algorithm of the specific chain.

With a live header relay, deposit claims become trustless: the EVM contract itself verifies the Merkle proof of inclusion at sufficient depth.

5.3 Stage C: Zero-Knowledge Proof of Chain State

The ultimate form of trustless bridging uses a succinct proof (SNARK or STARK) that a specific transaction is included in the canonical chain at block b with $\geq k$ confirmations [5].

$$\pi = \text{ZKProof}(\exists\, b_1, \ldots, b_k \text{ s.t. } b_1 \text{ includes } \mathit{txid} \wedge \text{chain}(b_1, \ldots, b_k) \text{ valid} \wedge \text{work}(b_k) \geq W)$$

The on-chain verifier is a fixed Solidity function; proof generation is done off-chain by any party (including the user). Proof generation for a Bitcoin- style PoW chain currently takes seconds to minutes on consumer hardware using systems such as Groth16 [6] or PLONK [7].

Recent work (zkBridge [5], BTC Relay successors) demonstrates practical ZK verification of Bitcoin-compatible headers. We intend to track this literature and integrate a ZK verifier contract once proving costs for UTXO inclusion fall within acceptable on-chain gas bounds.

5.4 Phased Migration

Phase	Name	Trust model	Timeline
1	Federated Multisig	2-of-3 honest majority	**Live (2026)**
A	Optimistic Relay	Censorship-resistant challengers	2026–2027
B	SPV Light Client	Honest full nodes + EVM correctness	2027–2028
C	ZK Proof Bridge	Cryptographic soundness only	2028+

Table 3: Bridge evolution roadmap.

Importantly, phases A, B, and C are *additive*: the multisig path can remain as a fallback during transition, with the trustless path becoming the canonical path once it is proven in production.

6. Economic Model

6.1 Bridge Fee

A fee of $f = 50\,\text{bps}$ (0.5%) is applied to each bridge operation. Fees accumulate in the bridge contract as wrapped tokens and are periodically swept to a protocol treasury controlled by the owner Safe. Fees serve two purposes:

1. Covering relayer gas costs and operational overhead.

2. Creating a treasury for funding the ZK research roadmap (Phase C).

6.2 Reserve Health

The *reserve ratio* is defined as:

$$\rho = \frac{S_b}{R_b}$$

where S_b is the ERC-20 total supply and R_b is the UTXO custody balance (both in satoshi units). The bridge targets $\rho \leq 1$. A publicly accessible reserve dashboard publishes ρ, S_b, R_b, and the UTXO set in real time, enabling independent verification of solvency.

7. Security Considerations

7.1 Key Management

Relayer private keys are stored encrypted at rest on isolated server instances. The UTXO custody keys are used only to sign withdrawal transactions and are never exported or used for any other purpose. The Safe multisig owner keys are held on hardware wallets by independent parties.

7.2 Upgrade Policy

The `WJKCBridge.sol` and token contracts are *non-upgradeable*. All parameter changes (signer set, fee, threshold) are subject to a 24-hour on-chain timelock, during which any user can observe the pending change and exit the bridge before it takes effect. Logic changes require deploying a new contract and a governance-approved minter handoff with a 2-day timelock.

7.3 Monitoring and Incident Response

An automated reconciliation module (`reconciliation.py`) runs continuously on each relayer, computing:

- Proof-of-reserve: $R_b \geq S_b$ at every block.
- Stuck deposit/withdrawal detection: alerts if an operation is pending beyond a configurable deadline.
- UTXO health: alerts if the custody UTXO count indicates consolidation is needed.

Alerts are emitted with a `[reconcile] ALERT` log prefix and can be forwarded to any monitoring backend.

8. Conclusion

We have presented the Wojak Multisig Bridge, a production-ready federated bridge connecting Wojakcoin and Junkcoin UTXO chains to Base EVM. The design prioritises practical deployability today while making explicit every trust assumption it relies on.

The federated model is not the end state. By publishing this roadmap alongside the bridge itself, we commit to a research and engineering trajectory that progressively removes operator trust—first through optimistic fraud proofs, then on-chain SPV verification, and ultimately zero-knowledge proofs of chain state. Each stage is auditable, each transition is backwards-compatible, and the final state requires no trusted party whatsoever.

We invite the community to audit our contracts, run independent relayer nodes, and contribute to the ZK research effort that will make trustless UTXO-to-EVM bridging a reality.

Contracts (Base Mainnet):

`wWojakcoin`	`0x867340cFC92a771cd3cFFCfF056a84490cAde7C0`
`WJKCBridge (WJK)`	`0x91bB8225b5b5fEA61A8638B4897141219cd1451E`
`wJunkcoin`	`0x613352f33E8900246f619B7CBD6C3dC80E919122`
`WJKCBridge (JKC)`	`0x22BEc33348037cF4200813A96f1981A70e37A489`

8. References

[1] Rekt News, *Bridge Hacks: A Running Total*, 2023. https://rekt.news

[2] G. Maxwell, *BIP 16: Pay to Script Hash*, 2012. https://github.com/bitcoin/bips/blob/master/bip-0016.mediawiki

[3] T. Harding, *BIP 67: Deterministic Pay-to-script-hash multi-signature addresses through public key sorting*, 2015. https://github.com/bitcoin/bips/blob/master/bip-0067.mediawiki

[4] Optimism Contributors, *Optimism: Bedrock and the Road to Decentralisation*, 2021. https://community.optimism.io

[5] Q. Xie et al., *zkBridge: Trustless Cross-chain Bridges Made Practical*, CCS 2022.

[6] J. Groth, *On the Size of Pairing-Based Non-interactive Arguments*, EUROCRYPT 2016.

[7] A. Gabizon, Z.J. Williamson, O. Ciobotaru, *PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge*, 2019.