Multisig Digital Bearer Instruments: Trustless Digital Cash

Asensio Arias
keys@multisigbearer.org multisigbearer.org

Abstract

A trustless bearer instrument for Bitcoin would allow value to transfer directly between parties without network connectivity, fees, or custodial dependency at each hop. Digital signatures and multisignature scripts provide part of the solution, but the core benefit is lost if the issuer retains a key capable of unilateral redemption after issuance. All prior multisignature bearer schemes have positioned the issuer at or above the spending threshold. We propose a system that inverts this configuration: the bearer holds the two keys constituting the spending threshold of a 2-of-3 multisignature script, and the issuer holds one key that is arithmetically below it. The issuer's key is deleted immediately after address construction, before the instrument is funded. No copy of the issuer's key exists at any point during the instrument's funded lifetime. Transfer is key handover requiring no on-chain transaction, no network connection, and no fee. Only two on-chain transactions occur across the instrument's lifetime regardless of the number of transfers. We also describe a buyer-generated key issuance protocol that closes the malicious issuer attack; a receipt mechanism using the Nostr protocol to achieve cryptographically verifiable key deletion upon confirmed receiver possession; a bounded UTXO set enabling offline verification from the mainnet inception block; and a transport-agnostic payment protocol supporting fully offline operation. A proof-of-concept on Bitcoin Signet demonstrates the core security properties on live hardware.

---

1 Introduction

Commerce on Bitcoin requires an on-chain settlement event for every transfer of value. Fees, confirmation latency, and network dependency follow necessarily from this. While the base layer provides strong finality guarantees, these properties make it unsuitable for small, frequent, or physically proximate payments. The Lightning Network reduces fees and latency through payment channels but requires channel liquidity, online counterparties, and network connectivity at payment time. A gap remains for payments, like buying a coffee, that are small, frequent, or made without network access, where the costs and dependencies of both mechanisms become impractical.

Physical bearer instruments for Bitcoin have been attempted in several forms since the early years of the protocol. In each case, the issuer held a private key at some point during manufacture or issuance. Single-key designs create an unresolvable dependency on the issuer's non-retention. Multisignature bearer schemes have addressed part of this by distributing keys between parties, but in every prior implementation the issuer has been positioned at or above the spending threshold, either as a sole keyholder or as a required co-signer. The bearer's claim on the funds has always been contingent on the issuer's behaviour or continued availability.

The present proposal departs from all prior art on this specific point. We invert the key distribution: the bearer holds both keys required to meet the spending threshold, and the issuer holds one key that is arithmetically below it. The issuer cannot spend, cannot block a spend, and cannot co-sign a spend, because no combination of the issuer's keys alone reaches the threshold. This is not a policy claim: it is a consequence of the signature threshold enforced by every validating node

on the Bitcoin network. The issuer's key is further deleted immediately after the multisignature address is constructed and before any Bitcoin is sent to the address. A later compromise of the issuer's device finds no key to extract.

Once funded, the instrument transfers through bearer key handover. No transaction is broadcast, no fee is paid, no network connection is required. The UTXO remains at the same address through any number of transfers. The final bearer sweeps to their own address at their discretion. That sweep is the second and last on-chain event in the instrument's lifetime.

2 Background and Prior Work

2.1 Physical Bitcoin Bearer Instruments

Several approaches to physical Bitcoin bearer instruments have been developed since the early years of the protocol. Single-key designs encode a private key in a physical medium and rely on the physical form as a signal that the key has not been accessed since manufacture. The issuer necessarily generates and handles the private key before delivery. Hardware-enforced write-once designs improve on this by generating the key on-device and revealing it only on physical destruction, but remain limited by single-key architecture. In each case the security claim rests on a statement about the issuer's behaviour rather than on a verifiable property of the key distribution.

2.2 Multisignature Bitcoin

Bitcoin's scripting system supports m-of-n multisignature addresses, which require m valid signatures from a designated set of n keys before funds can be spent. Pay-to-Script-Hash encapsulates the spending conditions such that the redeemScript is revealed only at spend time. Multisignature has been applied to collaborative custody and exchange cold storage. Prior multisignature bearer schemes have positioned the issuer as a required co-signer, preserving custodial dependency through the instrument's settled lifetime. The present proposal keeps the issuer permanently below the spending threshold and deletes the issuer's key before funding. The issuer is not a co-signer. The issuer is not required for settlement. The issuer cannot act on the instrument after creation.

2.3 Off-Chain Payment Schemes

Chaumian ecash systems issue bearer tokens using blind signatures and allow off-chain transfer without per-transaction settlement costs. Both issuance and redemption require the mint as a trusted and online participant. Lightning Network channels allow off-chain payments between nodes with established channels but require liquidity management, channel funding transactions, and online counterparties at payment time. The mechanism proposed here is complementary to Lightning: instruments can be swept into Lightning channels at settlement, and this protocol serves payment contexts where Lightning cannot operate.

3 The MBI Mechanism

3.1 Core Architecture

We define a Multisig Digital Bearer Instrument as a funded unspent transaction output locked to an m-of-n P2SH multisignature address where the bearer holds exactly m keys and the issuer holds $(n - m)$ keys, with $m > (n - m)$. In the recommended configuration, $n = 3$ and $m = 2$.

Key A is held by the issuer. Keys B and C are held by the bearer. Any two of the three keys can authorise a spend. The bearer, holding both B and C, can spend unilaterally. The issuer, holding

only A, cannot spend under any combination: A constitutes one signature where two are required. Public keys are sorted lexicographically per BIP 67 before script construction, making the resulting address deterministic given any ordering of the three public keys and allowing independent address verification from the public keys alone.

This threshold configuration is the central departure from all prior multisignature bearer schemes. In prior art, the spending function includes at least one issuer key as a required input. In the present scheme, spend $= f(k_B, k_C)$. The issuer's key is absent from the spending path entirely. It exists only to participate in address construction, after which it is deleted.

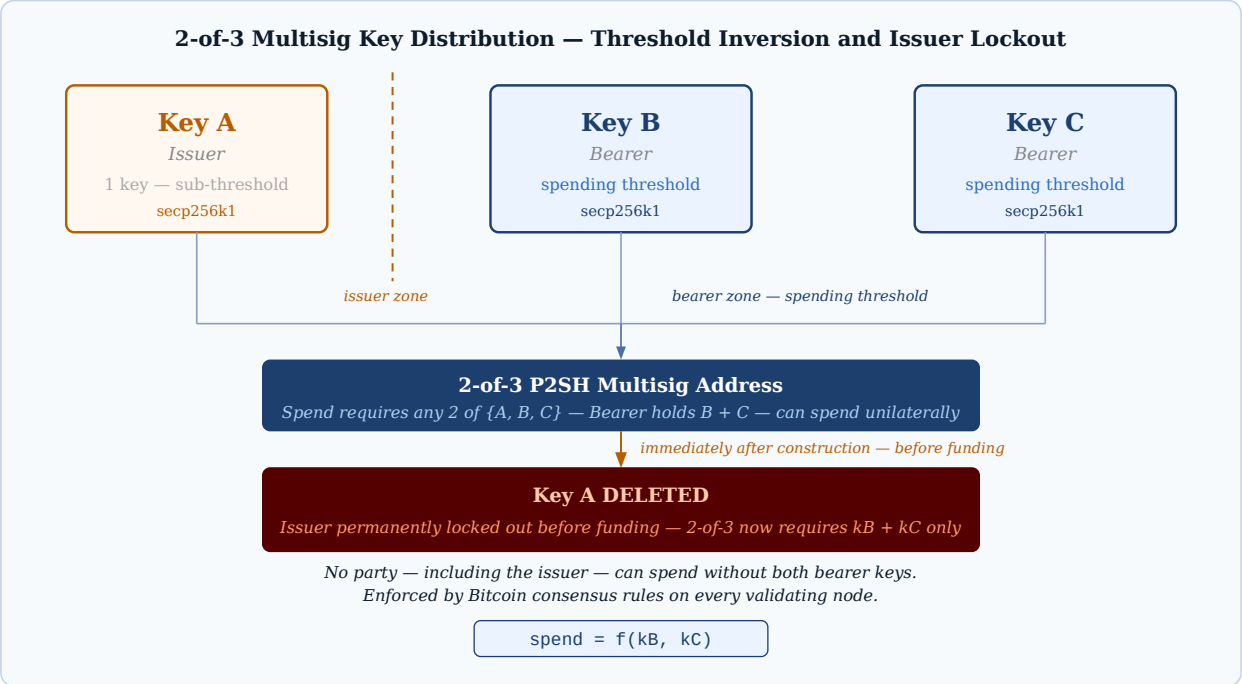

Figure 1: Key distribution architecture showing the threshold inversion. The bearer holds Keys B and C (spending threshold). The issuer holds only Key A (sub-threshold), deleted immediately after address construction before any Bitcoin is sent to the address.

3.2 Key Generation

All three keypairs are generated using `SecureRandom.getInstanceStrong()`, which on Android maps to `/dev/urandom` seeded from hardware entropy sources. Rejection sampling ensures the generated scalar is strictly below the secp256k1 group order n. Keys exist as byte arrays in application memory during generation before being encrypted and written to hardware-backed storage.

Private keys are stored in `EncryptedSharedPreferences` using AES-256-SIV for key encryption and AES-256-GCM for value encryption. The wrapping key for this preference store is held in the Android Keystore hardware security module, backed by StrongBox (Titan M2 security chip) on qualifying devices or TEE otherwise. The wrapping key never leaves the hardware security module. Access to it is gated by biometric authentication with a 30-second session validity window. Private keys are never written to the plain SQLite Room database; only the public keys K_A, K_B, and K_C appear in plain storage for receipt verification purposes.

All private key byte arrays are zeroed using `sodium_memzero()` immediately after use. This function uses platform-specific memory barrier instructions to prevent compiler elision of the zeroing operation. Hex encoding during key storage is performed through a mutable `CharArray` zeroed in a `finally` block immediately after the write completes. Private key objects are scoped to short-lived lambdas and never returned to callers. A known limitation exists: bitcoin-kmp's

`PrivateKey.value` is an immutable JVM `ByteVector` that cannot be zeroed. Its lifetime is minimised by confining it within the scoped lambda and relying on GC eligibility immediately on return.

3.3 Key A Deletion

Key A is deleted from encrypted storage immediately after the multisignature address is constructed, before the instrument is funded and before any Bitcoin is sent to the address. After deletion no copy of Key A remains on the issuer's device. A later compromise of the device, including access to the device seed phrase, does not recover a key that has been removed from encrypted storage. Because Key A is gone before funding, the instrument is locked to the bearer keys from the moment any Bitcoin arrives at the multisignature address.

The proof that a single key is insufficient to spend can be demonstrated directly: a transaction signed with Key A alone is a valid PSBT but is rejected by every Bitcoin node on the network, which requires two valid signatures against the redeemScript. This is arithmetic enforced by consensus, not a software policy.

3.4 Buyer-Generated Key Issuance

If the issuer generates all three keypairs, the issuer has access to Keys B and C at generation time. A compromised or malicious issuer could retain copies and later spend outstanding instruments. This attack is closed by having the buyer generate Keys B and C on their own device. Only the public keys K_B and K_C are transmitted to the issuer. The issuer generates Key A independently, constructs the 2-of-3 P2SH address from K_A, K_B, and K_C, and funds it. Recovering a private key from its corresponding public key requires solving the elliptic curve discrete logarithm problem over secp256k1, for which no efficient algorithm is known.

3.5 Transfer, Intermediate State, and Settlement

Transfer begins when the sender constructs the sealed QR payload. At this point the instrument is marked with a `pending_transfer` status in local storage, which prevents the same instrument from being presented for payment again during the transfer window. If the transfer is cancelled or the application terminates before the receiver scans, a revert operation restores the instrument to active status. No keys have been deleted and no funds are at risk. Keys are deleted by one of three paths depending on transfer mode. In the online path, keys are deleted only after the Nostr receipt proof passes the NC1 gate described in Section 5.2. In the offline P2P path, keys are deleted after the sender taps an explicit confirmation button, `kB` and `kC` are removed from `EncryptedSharedPreferences` and the instrument is marked transferred. On the merchant receive side, key deletion does not occur on the merchant's device; the merchant receives the keys and sweeps on network recovery.

Once the NC1 gate is passed and keys are deleted, the transfer is irreversible. The previous bearer, no longer holding both Keys B and C, cannot produce two valid signatures. The new bearer, holding B and C, can spend unilaterally at any time.

Settlement occurs when the bearer broadcasts a sweep transaction signed by Keys B and C. The scriptSig must contain two valid ECDSA signatures from keys enumerated in the redeemScript, followed by the redeemScript itself. Every node on the network validates this independently. The sweep is the second and final on-chain event in the instrument's lifetime.

3.6 Verification and Attack Window

A recipient verifies an instrument by confirming the UTXO exists and carries the stated satoshi value; decoding the redeemScript to confirm a 2-of-3 P2SH structure with public keys in BIP 67 sorted order; confirming Key A is one of three and the bearer holds both remaining keys; confirming the timelock expiry, if present, meets the minimum standard of Section 6; and querying nodes for unconfirmed spending transactions referencing the instrument address.

The full protocol specification calls for querying multiple geographically diverse nodes simultaneously, reducing the risk that a pre-broadcast sweep transaction has reached some nodes but not the one queried. The current proof-of-concept queries a single endpoint. Multi-node querying is a specified design property noted as not yet implemented.

In the digital transfer path, private key bytes are zeroed via `sodium_memzero()` immediately after the NaCl decryption operation completes. The window between decryption and zeroing is bounded by the NaCl open operation, typically under one millisecond on current hardware. An attacker who intercepts the sealed QR payload cannot extract the keys without the recipient's X25519 private key. The primary defence against the pre-broadcast sweep race is immediate sweep by the recipient.

4 The Electronic Banknote Model

A physical banknote changes hands many times between printing and destruction. Each transfer is free, instantaneous, and requires no network connection. Two events involve the issuing authority: issuance and final redemption. Between them the note circulates freely at no cost.

An MBI instrument replicates this property. Two on-chain transactions bound the instrument's lifetime. Between them it circulates through key handover at zero cost, zero latency, and with no network requirement. An instrument that has transferred forty times carries the same value as one that has transferred once, less the eventual sweep fee. This differs from all prior digital payment mechanisms, in which settlement costs accumulate with each transfer.

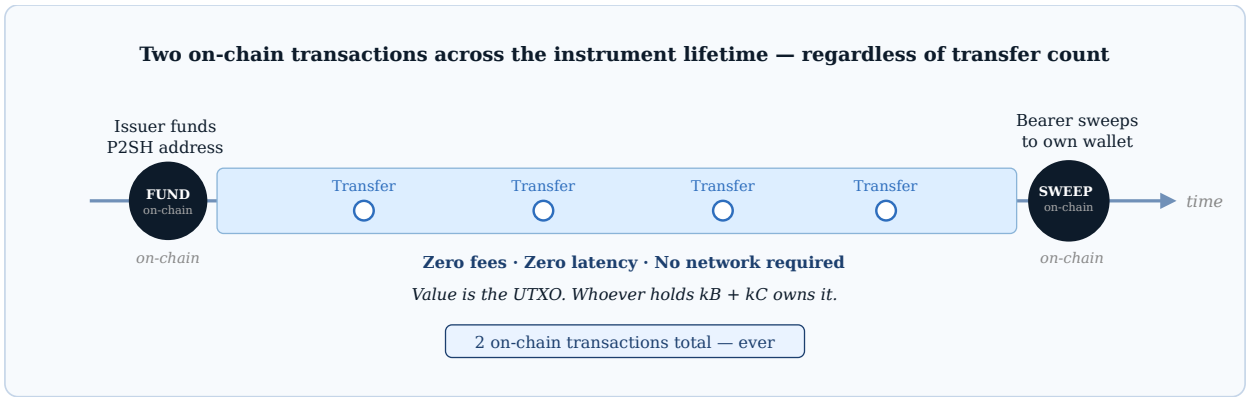

Figure 2: Instrument lifecycle. Two on-chain transactions bound the instrument's entire existence. Between issuance and redemption, any number of free, instant, offline transfers occur with no on-chain footprint.

4.1 Denomination and Fee Economics

Instruments are denominated in satoshis. Fiat denomination is not recommended because Bitcoin price movement changes the real value of outstanding instruments over time. Batch issuance funds many instruments in a single transaction with one input and multiple outputs. Transaction fees scale primarily with byte size rather than output count, so per-instrument issuance cost at batch

scale is low. At 5 sat/vbyte, a batch of 100 instruments at 10,000 sat costs approximately $0.81 in total fees, or 0.16% of float value. A P2SH multisig sweep transaction is approximately 339 virtual bytes; at the same fee rate the sweep costs approximately $0.085 regardless of denomination. This establishes a practical denomination floor near $2 to $3 at normal fee rates. Recipients who accumulate multiple instruments and batch their sweeps reduce the per-instrument settlement cost proportionally.

Table 1: Fee comparison across payment methods for 100 × $5 payments.

Payment method	Per-transfer fee	100 × $5 total
MBI (batch issue + batch sweep)	$0.00	∼$1.66
Lightning Network	∼$0.001 routing	∼$3.00
Bitcoin on-chain	∼$0.50 per tx	∼$100.00
Visa / Mastercard	2.9% + $0.30 per tx	∼$44.50

5 Digital Transfer and MBI-PRP

5.1 QR Payload Construction

For digital transfer, the sender constructs a payload containing the P2SH address, the satoshi value, the issuer public key K_A, and the bearer private keys k_B and k_C. The payload is compressed and sealed using the recipient's X25519 public key. This function performs an ephemeral X25519 key exchange and encrypts under XSalsa20-Poly1305 with the resulting shared secret. The sender's identity is not embedded in the ciphertext. Only the holder of the corresponding X25519 private key can open it. The sealed ciphertext is placed in a JSON envelope alongside a 32-byte random nonce, a Nostr routing identifier, and the sender's X25519 public key. This envelope is encoded as a QR code. A party who photographs the QR code obtains only ciphertext; the bearer keys are not recoverable without the recipient's private key.

5.2 Nostr Receipt and Key Deletion

The sender cannot delete bearer keys on QR display without risking fund loss if the receiver never scans. The sender cannot retain keys indefinitely without maintaining a double-spend capability. We require a mechanism by which the sender can delete keys with cryptographic certainty that the receiver holds them.

On scanning and decrypting the QR payload, the receiver's device automatically computes an ECDSA signature over the nonce using k_B: $\text{sig} = \text{sign}(k_B, \text{nonce})$. This signature can only be produced by the holder of k_B. The device constructs a receipt containing the signature, the instrument address, the routing identifier, and the public key K_B; encrypts it to the sender's X25519 public key using `crypto_box_seal`; and publishes it as a Nostr text note of kind 1. Custom event kinds were evaluated and found to be identifiable by relay operators, subject to silent rejection by relay whitelists, and susceptible to protocol fingerprinting; kind 1 is universally indexed and delivered by all public relays. The Nostr event is signed with a throwaway secp256k1 keypair that is zeroed immediately after the Schnorr signature is computed. Relay selection is a configurable deployment parameter.

Routing uses a `#p` subscription filter keyed to the sender's public key. A `#e` filter based on the event identifier was evaluated but found to be rejected by relay indexes that do not recognise the channel identifier as a valid stored event reference when used with kind 1. The `#p` filter is universally supported.

The sender maintains a subscription with a retrospective time filter, ensuring events stored be-

fore the subscription opens are delivered immediately on connection. On receiving the encrypted receipt, the sender decrypts it and verifies the ECDSA signature against K_B. The NC1 gate is then applied: key deletion proceeds only if the instrument address is present in `onChainVerifiedAddresses`, a set populated only after independent UTXO confirmation by the receiver's device. This gate prevents an adversary from triggering key deletion by publishing a receipt for an unfunded address. When both conditions are satisfied, the sender removes k_B and k_C from `EncryptedSharedPreferences`. The keys do not exist on the sender's device after this operation.

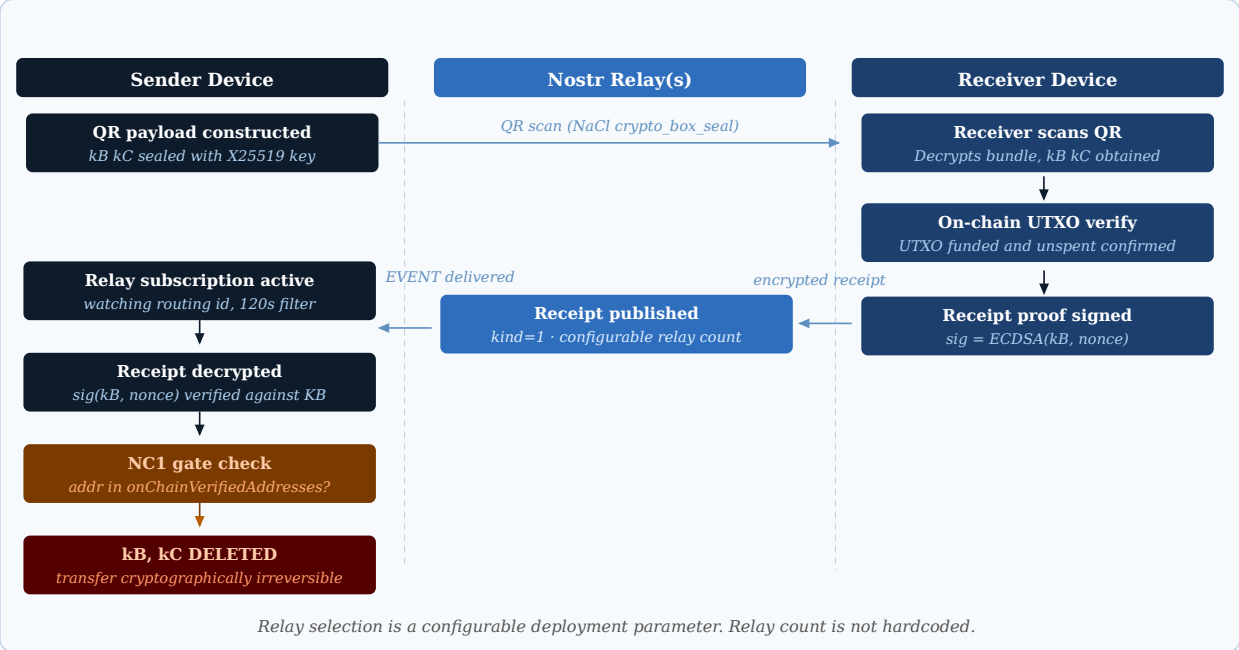

Figure 3: MBI-PRP payment flow and Nostr receipt mechanism. A single QR scan on the receiver's device triggers the full automatic sequence. The sender applies the NC1 gate before irrevocably deleting k_B and k_C. Relay selection is a configurable deployment parameter.

5.3 MBI Payment Request Protocol

MBI-PRP is a payment request standard for app-to-app and app-to-website payments using MBI instruments. No prior relationship, account, or registration between payer and payee is required. A payment request contains the payee's NaCl public key, the requested satoshi amount, a one-time request identifier, an expiry timestamp, and a callback endpoint. The payer selects instruments covering the requested amount, seals the bearer keys to the payee's public key, and transmits the bundle. The payee decrypts, verifies the UTXO balance and script structure, and returns confirmation.

The protocol is transport-agnostic by design. HTTPS serves web checkout. QR codes serve mobile scan payments. The full specification includes NFC NDEF for in-person tap payments and similar near-field communication transports for fully offline peer-to-peer payments; these transport bindings are specified design and are not yet implemented in the proof-of-concept. No existing standardised payment protocol occupies the position of requiring no account, operating fully offline, and incurring zero fee per payment.

Table 2: Comparison of standardised payment protocols.

Standard	Account required	Works offline	Fee per payment	Open
Apple Pay	Yes	No	0.15%	No
Lightning BOLT11	Yes	No	Routing fee	Yes
BIP 21 URI	Yes	No	On-chain fee	Yes
MBI-PRP (this paper)	**No**	**Yes**	**Zero**	**Yes**

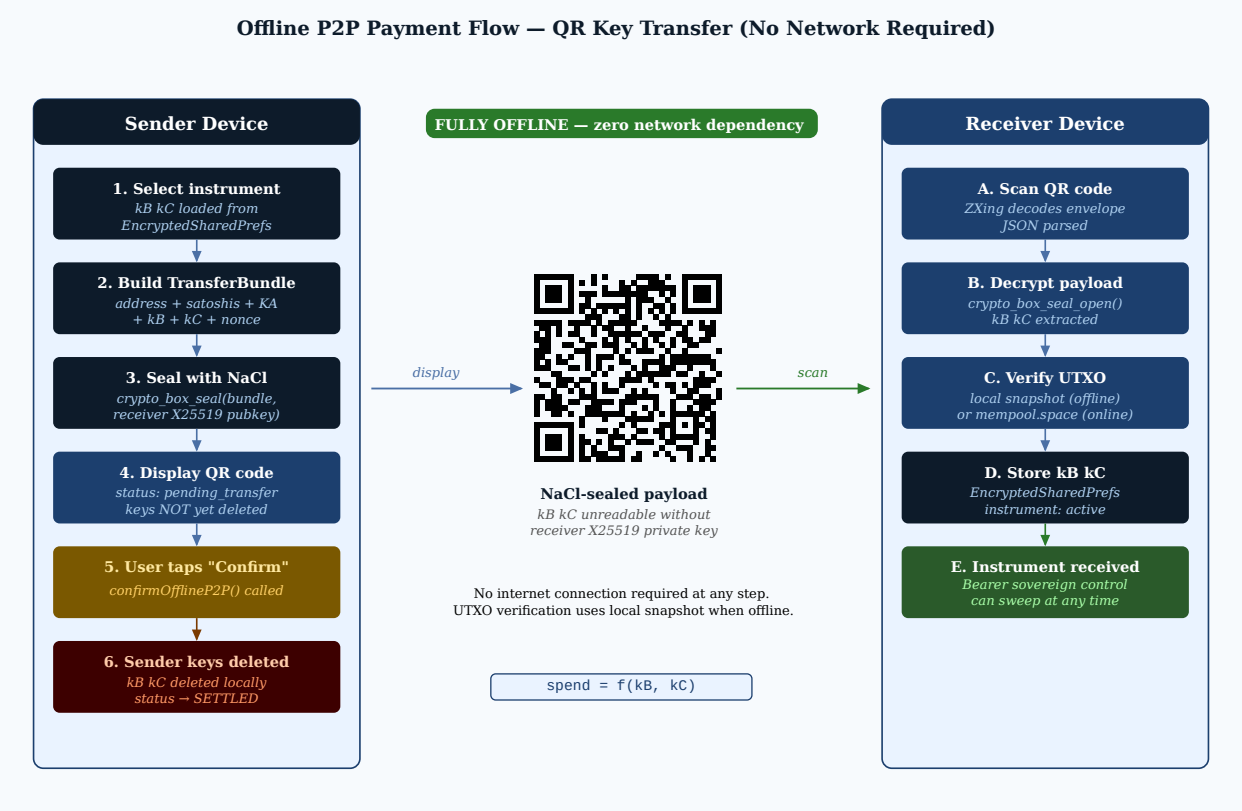

Figure 4: Offline P2P transfer protocol. Scan 1 obtains the receiver's X25519 public key. Scan 2 transfers the NaCl-sealed instrument payload. Key deletion follows one of three paths: immediate on verified Nostr receipt (online); after explicit user confirmation and 15-second countdown (P2P offline); or not at all on the merchant side, where the receiver holds the keys and sweeps on network recovery.

6 Timelock-Based Issuer Recovery

An instrument whose bearer keys are permanently lost is permanently unspendable. To address this, the multisig redeemScript may optionally encode a recovery path using `OP_CHECKLOCKTIMEVERIFY`. The script encodes two spending branches via `OP_IF/OP_ELSE/OP_ENDIF`: the normal 2-of-3 multisig path, accessible at any block height, and a recovery path requiring only Key A, accessible after a specified block height. The timelock is visible in the redeemScript and can be verified independently by any party. Instruments with no timelock encoded are valid and carry no expiry.

Timelock recovery is a specified design feature. It is not implemented in the current proof-of-concept, which constructs standard 2-of-3 P2SH scripts with no recovery branch.

6.1 Short Timelock Attack and Minimum Standard

A sender who encodes a short timelock can transfer the instrument, allow the recipient to accept it in good faith, wait for expiry, and recover the funds using Key A. Verification software must enforce a minimum remaining timelock before accepting an instrument. The minimum is a configurable parameter dependent on instrument denomination and use case. For low-denomination instruments in everyday payment contexts, a minimum of 4,320 blocks (approximately one month) is appropriate. For higher-value instruments or institutional applications, longer minimums are warranted. Instruments with fewer blocks remaining than the configured minimum must be rejected regardless of funded state.

7 Adversarial Security Analysis

We consider a technically capable adversary with full knowledge of the protocol. Eleven attacks are identified. Nine are closed by the protocol design. Two require true hardware Secure Enclave key generation and are noted as open in the current proof-of-concept, which generates keys in application code under hardware-backed encrypted storage.

7.1 Critical Attacks

Malicious issuer records Keys B and C. If the issuer generates all three keypairs, the issuer has access to Keys B and C at generation time. Closed by buyer-generated key issuance as described in Section 3.4.

Copied key multi-sale. A bearer copies Keys B and C before transfer and sells the same instrument to multiple parties. On qualifying devices the proof-of-concept implements hardware-backed key storage via StrongBox, which confines key material within the hardware security module. Full Secure Enclave enforcement with per-operation attestation remains a further hardening step for production deployment. The primary practical mitigation for merchant receipts is immediate sweep, which reduces the exploitable window to the mempool propagation time of the sweep transaction.

7.2 High-Exploitability Attacks

Pre-broadcast sweep. The attacker pre-constructs a sweep transaction and broadcasts it during the recipient's verification window. Bitcoin mempool propagation takes 30 to 60 seconds to reach a supermajority of nodes. Primary closure: the recipient sweeps within seconds of accepting the instrument. Secondary measure: the full protocol specification calls for querying multiple geographically diverse nodes; the current proof-of-concept queries a single endpoint.

NC1: forged receipt triggering premature key deletion. An adversary publishes a Nostr receipt for an instrument whose on-chain verification has not yet completed. Closed by the NC1 gate: key deletion requires the instrument address to be present in `onChainVerifiedAddresses`, populated only after independent UTXO confirmation by the receiver's device. Both the cryptographic receipt and on-chain confirmation must be satisfied before deletion proceeds.

Memory extraction. A rooted device or modified application extracts key bytes from memory. On qualifying devices the proof-of-concept stores keys within the Android Keystore hardware security module, backed by StrongBox (Titan M2), such that key material does not exist in application memory. Devices without StrongBox fall back to TEE-backed storage. Per-operation Secure Enclave attestation with transfer marking remains a further hardening step.

Timing attack on merchant verification. Closed by re-querying immediately before release and by immediate sweep.

Network partition and eclipse attack. Closed in the full protocol by querying multiple geographically diverse nodes simultaneously. Partially mitigated in the current proof-of-concept by TLS on the single queried endpoint.

MITM public key substitution. Closed by TLS on all network transport.

Unfunded instrument. Closed by UTXO verification against a confirmed funded state before accepting the instrument.

Receipt replay. Closed by the per-payment 32-byte random nonce; a receipt signature produced over one nonce is invalid for any other.

Table 3: Attack surface summary.

Attack	Severity	Status
Malicious issuer copies B+C	Critical	Closed: buyer-gen. keys
Copied key multi-sale	Critical	Requires Secure Enclave
Mempool race / pre-broadcast sweep	High	Closed: immediate sweep
NC1: forged receipt / premature deletion	High	Closed: NC1 gate
Memory extraction	High	Requires Secure Enclave
Timing / partition / MITM	Moderate	All Closed
Fake instrument / receipt replay	Low	All Closed

8 Implementation Considerations

8.1 Software Stack

Full implementation requires P2SH 2-of-3 multisig address generation, available in Bitcoin Core, libsecp256k1, BitcoinJS, and all major Bitcoin libraries. PSBT as defined in BIP 174 supports collaborative transaction construction and atomic change-making. `OP_CLTV` script encoding per BIP 65 supports optional timelock recovery. libsodium provides the NaCl primitives used in digital transfer. The Android Keystore system or Apple CryptoKit provides hardware-backed key storage.

8.2 Key Storage in the Proof-of-Concept

Bearer private keys k_B and k_C are stored in hardware-backed encrypted storage using AES-256-SIV for key encryption and AES-256-GCM for value encryption. The wrapping key for this store is held in the Android Keystore hardware security module, backed by StrongBox (Titan M2 security chip) on qualifying devices or TEE otherwise. The wrapping key never leaves the hardware security module. Access is gated by biometric authentication with a 30-second session validity window. Private keys are not stored in the plain SQLite Room database; only the public keys K_A, K_B, and K_C appear there for receipt verification. All private key byte arrays are zeroed via `sodium_memzero()` immediately after use. Key generation uses `SecureRandom.getInstanceStrong()` with rejection sampling. The AES-256-GCM wrapping layer retains 128-bit effective security against an adversary with access to a quantum computer running Grover's algorithm.

The full production architecture specifies key generation inside a hardware Secure Enclave with a per-operation biometric gate, such that private key bytes never exist in application memory. Transition to this model requires Secure Enclave support for secp256k1 operations, available on recent Apple hardware via CryptoKit and on Android via StrongBox on qualifying devices at API level 31 and above.

8.3 Bounded UTXO Set and Offline Verification

The full protocol specification describes a bounded UTXO set indexed from the mainnet inception block: the first Bitcoin mainnet block in which a funded MBI instrument appears at launch. MBI instruments are identifiable by script template and issuer public key, so the verification application indexes only matching UTXOs from that block forward. No prior blockchain history is relevant. Swept instruments are removed from the index. The set grows in proportion to active instrument circulation, not cumulative issuance.

At 100,000 active instruments the set occupies approximately 15 MB. At 1,000,000 active instruments approximately 150 MB. Initial synchronisation uses an issuer-signed snapshot verified against block headers via SPV. Ongoing synchronisation uses BIP 157/158 compact block filters to identify relevant blocks without downloading full block data. A device with a synchronised snapshot can verify any previously indexed instrument without network access.

This offline verification capability is specified design. The current proof-of-concept verifies instruments by querying mempool.space over HTTPS and does not implement local UTXO indexing. The offline transport bindings in MBI-PRP depend on this capability and are similarly specified design rather than current implementation.

9 Comparison with Related Systems

Prior off-chain bearer systems achieve some of the relevant properties but not all simultaneously. Ecash systems using blind signatures allow off-chain transfer at zero per-transfer cost but require the mint as a trusted participant at redemption. Payment channel networks allow off-chain settlement at low fees but require online counterparties and channel liquidity. Physical single-key instruments allow off-chain transfer by key handover but depend on trust in the issuer's non-retention. Multisignature bearer schemes that position the issuer as a co-signer preserve custodial dependency through settlement.

The present proposal achieves issuer lockout through the signature threshold itself. No coalition of issuer keys meets the spending threshold. The issuer is not required at settlement, is not required to be online, and cannot be compelled to act on the instrument. The issuer's key does not exist after address construction. This combination of properties has not been achieved by any prior system. The property is verifiable by any party who can decode a P2SH redeemScript and confirm the key count against the spending threshold.

Table 4: Comparison with related systems.

System	Issuer lockout	Off-chain transfer	Fee	No trust
Single-key instrument	No	Yes	None	No
Write-once hardware	Partial (hw)	Yes	None	Partial
Issuer co-sign multisig	No	Yes	None	No
Ecash (blind sig)	Yes (blind sig)	Yes	None	No (mint)
Lightning Network	N/A	Yes	Routing	No (LSP)
MBI (this paper)	**Yes (by threshold)**	**Yes**	**None**	**Yes**

10 Privacy Analysis

At issuance and settlement, MBI instruments are visible on-chain. The funding transaction reveals the P2SH address and denomination. The sweep transaction reveals that a 2-of-3 multisig has been spent. Between these two events the instrument leaves no on-chain trace. Key handover between

bearer and recipient produces no blockchain record.

When a recipient queries an external endpoint to verify an instrument, the instrument address and the recipient's network address are disclosed to the endpoint operator. Verification against a local UTXO snapshot eliminates this exposure. Fixed denominations allow chain analysts to identify probable MBI outputs by amount. Issuers can mitigate this by adding a small random satoshi offset at funding time, varying each instrument's value within a defined range above the base denomination.

Migration from P2SH to Pay-to-Taproot with MuSig2 key aggregation removes the on-chain multisig fingerprint. Keys A, B, and C aggregate into a single public key. The resulting address is indistinguishable from a standard single-signature P2TR address. The bearer produces a single aggregate Schnorr signature using Keys B and C. No chain analyst can determine from the sweep transaction that a threshold scheme was involved. This upgrade requires no Bitcoin protocol changes beyond Taproot, activated in November 2021 under BIP 340 through 342. It is a specified design property and is not yet implemented in the proof-of-concept.

11 Quantum Safety Evaluation

The asymmetric primitives used in this system, secp256k1 ECDSA, secp256k1 Schnorr, and X25519, are vulnerable to Shor's algorithm on a cryptographically relevant quantum computer. This vulnerability is common to Bitcoin generally and is not specific to this protocol. No quantum computer capable of breaking 256-bit elliptic curve cryptography exists or is considered imminent; published estimates place such capability at a minimum of ten or more years.

The symmetric layer is not vulnerable in the same sense. AES-256-GCM, used in the Android Keystore wrapping key, retains 128-bit effective security against Grover's algorithm. SHA-256, used in Bitcoin transaction hashing and Nostr event identifiers, retains 128-bit collision resistance. These components do not require modification.

The asymmetric components are isolated in two classes: `NaclIdentity`, which handles X25519 key exchange for QR payload encryption, and `ReceiptProof`, which handles ECDSA signing and verification. Replacing these with ML-KEM per NIST FIPS 203 and ML-DSA per NIST FIPS 204 constitutes the post-quantum migration path for this protocol. The Bitcoin multisig layer depends on a network-level upgrade that applies to all Bitcoin applications equally.

12 Generalisation to Other Settlement Layers

The mechanism described in this paper requires three properties of the underlying settlement layer: threshold multisignature spending enforced by consensus rules; a verifiable funded state equivalent to a UTXO; and off-chain key transmission capability. Bitcoin-derived UTXO-based networks that support P2SH multisig can adopt this mechanism directly. Stablecoin-denominated instruments on Layer 2 networks are a commercially significant application: a fixed-value instrument eliminates the denomination volatility while retaining all bearer instrument properties.

The threshold inversion principle applies to any context in which post-issuance issuer control is a structural problem. Digital bearer bonds, commercial paper, trade finance instruments, and wholesale CBDC designs share the same limitation in their current electronic form: transfer requires a registry update and the issuing institution retains the ability to freeze, reassign, or cancel. A threshold-configured instrument restores negotiability at the cryptographic layer. KYC enforcement at issuance addresses the regulatory concerns that eliminated physical bearer instruments in most jurisdictions. For regulated deployments where blockchain settlement is not appropriate, the same threshold configuration is implementable using hardware security modules as key custodians with settlement on a permissioned ledger.

13 Conclusion

We have proposed a system for transferring Bitcoin value without on-chain settlement at each transfer, without custodial dependency on the issuer, and without network connectivity at payment time. The central contribution is a specific inversion of the key distribution used in all prior multisignature bearer schemes: the bearer holds the full spending threshold and the issuer holds a sub-threshold key that is arithmetically insufficient to authorise any transaction. The issuer's key is deleted before the instrument is funded. No copy of it exists during the instrument's funded lifetime.

The Nostr receipt mechanism provides the sender with cryptographic proof that the receiver holds the bearer keys before deletion proceeds. The NC1 gate ensures key deletion cannot be triggered by a forged receipt for an unfunded instrument. The `pending_transfer` intermediate state allows cancellation without fund loss up to the point of irrevocable key deletion. Two on-chain transactions bound the instrument's lifetime regardless of transfer count. Between them it circulates as bearer value with no fees, no latency, and no network requirement.

A working implementation on Bitcoin Signet demonstrates these properties on live hardware. A batch funding transaction confirmed in block 297,032 on 2026-03-24:

`cdd394e8f059d80daa06b09816ce253fe916edc082dfd2190f4f71a67ca35c68`

demonstrates batch issuance of multiple denominated P2SH multisig instruments in a single on-chain transaction. A peer-to-peer transfer between two Android devices, including Nostr receipt, key deletion, and on-chain sweep confirmed in block 297,198 on 2026-03-26:

`549017b2b6b3556a58ac80cd502fc0e75d56dfa95135361e762f509d2c1bd692`

confirms the full transfer lifecycle including the 2-of-3 multisig script path. Both transactions are verifiable on the Bitcoin Signet explorer.

Directions for further work include formal verification of the multisig redeemScript and timelock script constructions; implementation of the bounded UTXO set and offline verification capability; the P2TR/MuSig2 upgrade path. The system requires no changes to the Bitcoin protocol and no trust in any party beyond the mathematics of threshold signatures and the consensus rules that enforce them.

鍵音ひびく
扉の影集う
会の朝霧

References

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

[2] Greg Maxwell. *BIP 16: Pay to Script Hash*. Bitcoin Improvement Proposal. 2012.

[3] Nils Schneider and Matt Bengsson. *BIP 21: URI Scheme*. Bitcoin Improvement Proposal. 2012.

[4] Peter Todd. *BIP 65: OP_CHECKLOCKTIMEVERIFY*. Bitcoin Improvement Proposal. 2014.

[5] Pieter Wuille. *BIP 67: Deterministic Pay-to-script-hash multi-signature addresses through public key sorting*. Bitcoin Improvement Proposal. 2015.

[6] Andrew Towns et al. *BIP 174: Partially Signed Bitcoin Transaction Format*. Bitcoin Improvement Proposal. 2017.

[7] Olaoluwa Osuntokun, Alex Akselrod, and Jim Lamarque. *BIP 157/158: Client-Side Block Filtering*. Bitcoin Improvement Proposal. 2017.

[8] Pieter Wuille, Jonas Nick, and Andrew Poelstra. *BIP 340–342: Schnorr Signatures, Taproot, Tapscript*. Bitcoin Improvement Proposal. 2020.

[9] Jonas Nick, Tim Ruffing, and Yannick Seurin. "MuSig2: Simple Two-Round Schnorr Multi-Signatures". In: *Advances in Cryptology — CRYPTO 2021*. Springer, 2021.

[10] Daniel J. Bernstein, Tanja Lange, and Peter Schwabe. *NaCl: Networking and Cryptography Library*. `https://nacl.cr.yp.to`. 2012.

[11] David Chaum. "Blind Signatures for Untraceable Payments". In: *Advances in Cryptology — CRYPTO 1982*. Springer, 1983, pp. 199–203.

[12] Matthew Green and Giuseppe Ateniese. "Identity-Based Proxy Re-Encryption". In: *Applied Cryptography and Network Security — ACNS 2007*. Springer, 2007, pp. 288–306.

[13] Cashu Contributors. *Cashu: Chaumian Ecash for Bitcoin*. `https://cashu.space`. 2022.

[14] Fedimint Contributors. *Fedimint: A Federated E-Cash Mint Protocol*. `https://fedimint.org`. 2022.

[15] Nostr Protocol. *NIP-01: Basic Protocol Flow*. `https://github.com/nostr-protocol/nostr`. 2022.