BITTRUST

Sovereign Bitcoin Inheritance

Technical Whitepaper v1.0

"*Not your keys, not your coins.*"

"*Not your plan, not your legacy.*"

January 2026

Executive Summary

Bitcoin represents the most significant monetary innovation in human history. For the first time, individuals can hold wealth that cannot be seized, censored, or debased by any government or institution. Yet this revolutionary self-sovereignty creates an unprecedented challenge: how do you transfer this wealth to your heirs without compromising the very principles that make Bitcoin valuable?

The current inheritance landscape forces Bitcoin holders into an impossible choice. Custodial solutions require trusting third parties with your private keys—a fundamental betrayal of Bitcoin's core promise. Traditional estate planning exposes seed phrases to lawyers, executors, and probate courts. The alternative—hoping your heirs can find and access your Bitcoin after you're gone—is a recipe for permanent loss.

BitTrust solves this problem using Bitcoin's native scripting capabilities. By leveraging Miniscript, timelocks, and multisignature configurations, BitTrust creates inheritance vaults that execute automatically without any third-party involvement. Your Bitcoin remains under your exclusive control until predetermined conditions are met. No custody. No trust. No compromise.

This whitepaper presents the technical architecture, security model, and operational procedures for trustless Bitcoin inheritance. It is written for Bitcoin maximalists who understand that sovereignty is not negotiable—not in life, and not in death.

1. The Inheritance Crisis

1.1 The Sovereignty Paradox

Bitcoin's fundamental value proposition is self-sovereignty: the ability to hold and transfer wealth without permission from any intermediary. This is achieved through cryptographic key pairs—whoever controls the private key controls the Bitcoin. Simple. Elegant. Revolutionary.

But this same property creates what we call the Sovereignty Paradox: the mechanisms that protect your Bitcoin during your lifetime become obstacles to transferring it after your death. A private key that dies with its owner locks away that Bitcoin forever. The very security that made Bitcoin valuable becomes a liability.

Estimates suggest that 3-4 million Bitcoin are already permanently lost, many due to failed inheritance planning. As early adopters age and the value of Bitcoin holdings grows, this problem will only accelerate. Without a solution, generational wealth transfer—one of humanity's oldest institutions—becomes impossible for Bitcoin holders.

1.2 The Custodial Trap

The most common "solution" offered by the industry is custodial inheritance services. Companies like Casa, Unchained, and various trust services offer to hold your keys (or portions of them) and release Bitcoin to your heirs upon your death. This approach has several fatal flaws:

Counterparty Risk: Any entity holding your keys can be compromised, coerced, or corrupted. Exchanges have been hacked. Custodians have committed fraud. Governments have seized assets. The history of financial custodians is a history of failures.

Regulatory Capture: Custodians operate under legal jurisdictions. They can be compelled to freeze assets, report holdings, or deny access. Your Bitcoin becomes subject to the same state control you sought to escape.

Ongoing Dependency: Custodial services require ongoing relationships and recurring fees. If the company fails, pivots, or is acquired, your inheritance plan fails with it. You're betting your family's financial future on a startup's longevity.

Philosophical Betrayal: Most importantly, custodial solutions violate the principle that led you to Bitcoin in the first place. "Not your keys, not your coins" doesn't include an asterisk for inheritance. If you're willing to give up custody to plan for death, why not give it up for convenience during life?

1.3 The Traditional Estate Planning Failure

Traditional estate planning instruments—wills, trusts, powers of attorney—were designed for a world of physical assets and institutional intermediaries. They fail catastrophically when applied to Bitcoin:

Probate Exposure: Wills are public documents. A seed phrase or wallet instructions in a will become visible to courts, clerks, and potentially anyone with access to probate records. Your Bitcoin holdings become public knowledge.

Executor Risk: Executors must access and distribute assets. Giving an executor access to your seed phrase gives them the ability to steal your Bitcoin before distribution. Even honest executors may not understand proper security practices.

Timing Mismatch: Probate can take months or years. During this time, your Bitcoin sits in limbo—vulnerable to theft if keys are compromised, inaccessible to heirs who may need funds immediately.

Jurisdictional Complexity: Estate law varies by jurisdiction. Bitcoin exists outside jurisdictions. Courts struggle to apply property law designed for physical assets to cryptographic bearer instruments.

2. The BitTrust Solution

2.1 Design Principles

BitTrust is built on four non-negotiable principles:

Zero Third Parties: No entity—not BitTrust, not any company, not any government—ever has access to your private keys or the ability to move your Bitcoin. The software runs locally on your machine. Your keys never leave your hardware wallet.

Bitcoin-Native Security: Inheritance conditions are enforced by Bitcoin's consensus rules, not by contracts, companies, or courts. A timelock doesn't require a judge's order to execute. A multisig doesn't need a custodian's permission.

Complete Transparency: BitTrust is open source. Every line of code is auditable. The Miniscript policies governing your vaults are human-readable. You don't trust BitTrust—you verify it.

User Responsibility: BitTrust shifts responsibility to where it belongs: you. You control your keys. You distribute key shares to heirs. You define the conditions for inheritance. This is not a bug—it's the entire point.

2.2 Technical Architecture Overview

BitTrust leverages three Bitcoin scripting capabilities to create trustless inheritance:

Miniscript: A structured subset of Bitcoin Script that enables complex spending conditions while remaining analyzable and composable. Miniscript allows us to express inheritance logic in a way that's both machine-verifiable and human-readable.

Timelocks (OP_CHECKLOCKTIMEVERIFY): Bitcoin's native ability to make outputs unspendable until a specific block height or timestamp. Timelocks enable dead man's switches and delayed inheritance without any trusted timekeeper.

Multisignature (OP_CHECKMULTISIG): Requiring multiple keys to authorize a transaction. Multisig enables quorum-based inheritance where heirs must cooperate, preventing any single party from unilaterally accessing funds.

3. Inheritance Vault Types

3.1 Timelock Vault (Dead Man's Switch)

The Timelock Vault implements a dead man's switch: funds become accessible to heirs after a specified period unless the owner resets the timer. This is the simplest and most powerful inheritance mechanism.

How It Works: You create a vault with a timelock of, say, one year. As long as you're alive and active, you periodically "refresh" the vault by moving funds to a new vault with a reset timer. If you become incapacitated or die, the timer expires and your designated heir can spend the funds using their key.

Miniscript Policy:

```
or(pk(owner),and(pk(heir),after(locktime)))
```

This policy says: the owner can spend at any time, OR the heir can spend after the locktime expires. Simple. Trustless. Unstoppable.

Recommended For: Single-heir scenarios, users who can commit to regular vault maintenance, those who want the simplest possible inheritance mechanism.

3.2 Multisig Inheritance Vault

The Multisig Vault requires multiple heirs to cooperate in order to access funds. This prevents any single heir from unilaterally claiming the inheritance and enables more complex family structures.

How It Works: You create a vault requiring, for example, 2-of-3 heir signatures to spend. Each heir receives their own key share. Only when the timelock expires AND sufficient heirs cooperate can the funds be moved.

Miniscript Policy:

```
or(pk(owner),and(thresh(2,pk(heir1),pk(heir2),pk(heir3)),after(locktim
e)))
```

Recommended For: Multiple heirs, situations requiring checks and balances, larger estates where collusion risk must be mitigated.

3.3 Hybrid Vault

The Hybrid Vault combines multiple conditions with fallback mechanisms, enabling sophisticated inheritance logic that adapts to different scenarios.

Example: Owner can spend anytime. After 6 months, spouse can spend alone. After 1 year, any 2 of 3 children can spend together. After 2 years, any single child can spend (emergency fallback).

Miniscript Policy:

```
or(pk(owner),or(and(pk(spouse),after(6_months)),or(and(thresh(2,pk(child1),pk(child2),pk(child3)),after(1_year)),and(thresh(1,pk(child1),pk(child2),pk(child3)),after(2_years)))))
```

Recommended For: Complex family structures, tiered inheritance, scenarios requiring both immediate and delayed access paths.

4. Key Distribution Strategy

4.1 The Distribution Imperative

A trustless inheritance system requires that heirs possess their keys BEFORE the inheritance event occurs. This is non-negotiable. You cannot distribute keys from beyond the grave, and any system that promises to do so for you reintroduces the third-party trust you're trying to eliminate.

This creates what some perceive as a security tradeoff: heirs have keys before you die. However, the timelock mechanism ensures they cannot spend until the lock expires. As long as you regularly refresh your vaults, possession of heir keys grants no spending power.

4.2 Hardware Wallet Key Ceremony

The recommended approach for key distribution is a formal Hardware Wallet Ceremony. This in-person process ensures heirs properly generate and secure their own keys.

Step 1: Gather heirs in a secure, private location. No phones. No cameras. No digital devices except the hardware wallets being configured.

Step 2: Each heir initializes their own hardware wallet, generating their own seed phrase. They write down and verify their seed phrase. They NEVER share their seed phrase with anyone, including you.

Step 3: Each heir exports their extended public key (xpub). The xpub allows vault creation but does not enable spending. You collect these xpubs to configure the inheritance vault.

Step 4: You create the vault using BitTrust, inputting your key and all heir xpubs. The software generates the Miniscript policy and vault address.

Step 5: You fund the vault and provide each heir with their Encrypted Heir Kit containing vault details, spending instructions, and verification information.

4.3 Shamir's Secret Sharing Alternative

For users who cannot conduct in-person ceremonies or who want additional redundancy, BitTrust supports Shamir's Secret Sharing (SSS) for key distribution.

SSS splits a secret into N shares, of which any K are sufficient to reconstruct the original. For example, a 3-of-5 split means any 3 shares can recover the key, but 2 shares reveal nothing.

Important: SSS should be used for backup redundancy, not as a replacement for proper multisig. If you split a single key using SSS, heirs must reconstruct that key before spending—a single point of failure. True multisig keeps keys separate and never requires reconstruction.

5. Security Model

5.1 Threat Analysis

BitTrust's security model addresses the following threat vectors:

Third-Party Compromise: Eliminated. No third party ever holds keys or has spending authority. BitTrust software is a tool, not a custodian.

Software Supply Chain Attack: Mitigated. BitTrust is open source and reproducibly built. Hardware wallet signing ensures malicious software cannot steal funds—transactions must be verified and signed on the hardware device.

Premature Heir Access: Prevented by timelocks. Heirs possess keys but cannot spend until the timelock expires. Regular vault refresh resets the timer.

Single Heir Collusion: Prevented by multisig. In a 2-of-3 configuration, no single heir can access funds alone.

Key Loss: Mitigated by redundancy. Multiple vault types, backup procedures, and Shamir splitting provide recovery options. However, user responsibility means key loss is possible if procedures are not followed.

Legal/Regulatory Seizure: Mitigated. No central point of control exists. Vaults are self-custodied. Timelocks and multisig are enforced by Bitcoin consensus, not legal agreements.

5.2 User Responsibilities

Sovereignty requires responsibility. BitTrust users must:

Secure Their Own Keys: Hardware wallet seed phrases must be stored securely. BitTrust cannot recover lost keys.

Maintain Vault Freshness: Timelock vaults must be refreshed before expiration to prevent premature heir access. BitTrust provides reminders but cannot force compliance.

Distribute Keys Properly: Heirs must receive their keys through secure channels. Improper distribution creates single points of failure.

Verify Everything: Trust, but verify. Check vault addresses on hardware wallets. Verify Miniscript policies. Test recovery procedures.

Keep Backups: Vault configuration data (policies, public keys, locktimes) must be backed up. BitTrust provides encrypted export, but storage is your responsibility.

6. Getting Started

6.1 Requirements

Hardware Wallet: BitTrust requires a hardware wallet for key security. Supported devices include Coldcard, Trezor, Ledger, and other devices supporting PSBT (Partially Signed Bitcoin Transactions).

Desktop Computer: BitTrust runs as a desktop application on Windows, macOS, or Linux. Internet connection required only for blockchain queries—all sensitive operations occur offline.

Heir Coordination: Heirs must be able to generate and secure their own keys, either through hardware wallets or secure software wallets.

6.2 Recommended Configuration

For most users, we recommend starting with a simple Timelock Vault:

Timelock Period: 6-12 months. Long enough that you won't forget to refresh, short enough that heirs don't wait years.

Refresh Schedule: Set calendar reminders for 1 month before expiration. BitTrust will also alert you.

Heir Keys: Hardware wallets for heirs whenever possible. Software wallets acceptable for lower-value vaults.

Backup: Export encrypted vault configuration. Store backup in a separate physical location from your hardware wallet.

6.3 Vault Creation Workflow

1. Launch BitTrust and connect your hardware wallet.

2. Create New Vault and select your vault type (Timelock, Multisig, or Hybrid).

3. Configure Parameters: Set timelock duration, add heir public keys, define spending conditions.

4. Review Policy: BitTrust displays the generated Miniscript policy. Verify it matches your intentions.

5. Generate Address: The vault address is derived from the policy. Verify on your hardware wallet.

6. Fund Vault: Send Bitcoin to the vault address. Start small to test before funding with significant amounts.

7. Export Heir Kits: Generate encrypted packages for each heir containing everything they need to claim their inheritance.

8. Distribute Securely: Provide heir kits through secure channels. Verify heirs can decrypt and access their information.

7. Conclusion

Bitcoin represents a fundamental shift in how humans can store and transfer value. For the first time in history, an individual can hold wealth that cannot be taken by force, frozen by decree, or inflated away by monetary policy. This is not merely a technical achievement—it is a civilizational one.

But sovereignty is not inherited by default. Without deliberate planning, your Bitcoin dies with you. Your family is left with nothing—not because Bitcoin failed, but because you failed to extend your sovereignty across generations.

BitTrust exists to solve this problem without compromise. No custodians. No third parties. No trust. Just Bitcoin, doing what Bitcoin does best: enforcing rules without permission.

The technology is here. The tools are ready. The only question is whether you'll use them.

Your Bitcoin. Your keys. Your legacy.

Don't let your Bitcoin die with you.

Appendix A: Glossary

Miniscript: A structured representation of Bitcoin Script that enables composition, analysis, and optimization of spending conditions.

OP_CHECKLOCKTIMEVERIFY (CLTV): A Bitcoin Script opcode that makes an output unspendable until a specified block height or timestamp.

Multisig: A Bitcoin script pattern requiring M-of-N signatures to spend. E.g., 2-of-3 requires any 2 of 3 designated keys.

PSBT (Partially Signed Bitcoin Transaction): A standard format for passing unsigned or partially-signed transactions between wallets and signing devices.

Hardware Wallet: A dedicated device for storing private keys and signing transactions, isolated from internet-connected systems.

Seed Phrase: A human-readable representation of a wallet's master private key, typically 12 or 24 words.

xpub (Extended Public Key): A public key that can derive child public keys, enabling watch-only wallet functionality without spending capability.

Shamir's Secret Sharing: A cryptographic algorithm that splits a secret into N shares, requiring K shares to reconstruct.

Dead Man's Switch: A mechanism that triggers automatically when a user fails to perform a regular action (like refreshing a vault).

Timelock: A spending condition that prevents an output from being spent until a specified time or block height.

Appendix B: Resources

BitTrust Website: https://bittrust.io

Source Code: https://github.com/bittrust/bittrust

Miniscript Reference: https://bitcoin.sipa.be/miniscript/

BIP-65 (CHECKLOCKTIMEVERIFY):
https://github.com/bitcoin/bips/blob/master/bip-0065.mediawiki

PSBT Standard (BIP-174):
https://github.com/bitcoin/bips/blob/master/bip-0174.mediawiki