Bitcoin Lending Protocol

Product Requirements Document

Phase 2: Bitcoin-Native Custody with Threshold Signatures

Version: 1.0
Date: January 2026
Status: Draft for Review
Author: Jamie (Project Lead)
Builds On: Phase 1 (Stacks/sBTC Implementation)

Table of Contents

---

1. Executive Summary

Phase 2 removes the dependency on sBTC by implementing **Bitcoin-native custody** using threshold signature schemes. This advancement enables the protocol to hold actual Bitcoin directly, eliminating trust in the sBTC peg mechanism while maintaining the oracle-free competitive bidding auctions proven in Phase 1.

Key Innovation

Threshold signature custody where Bitcoin collateral is held in a multi-signature address controlled by a decentralized network of validators. No single entity can access funds, and the protocol operates trustlessly on native Bitcoin without wrapped tokens.

Phase 2 Objectives

Objective	Description
Eliminate sBTC Dependency	Direct Bitcoin custody removes trust in sBTC peg
Decentralized Validators	Network of independent validators secure collateral
Threshold Signatures	M-of-N multisig using Schnorr/FROST or MuSig2
Maintain Stacks Benefits	Smart contract logic remains on Stacks for low fees
Backwards Compatible	Phase 1 loans continue operating; gradual migration
Enhanced Security	No custodian, no peg risk, pure Bitcoin security

Phase 2 Targets

Metric	Target
Launch Timeline	7 months (Months 7-13 after Phase 1)
Validator Network	15+ independent validators at launch
Threshold Scheme	10-of-15 multisig (67% threshold)
Migration Volume	50%+ of Phase 1 loans migrate to native BTC
New Loan Volume	$5M+ in native BTC loans by Month 13
Validator Uptime	99.5%+ availability
Total Budget	$312,000

Strategic Positioning

Phase 2 positions the protocol as the **first truly trustless Bitcoin lending platform** with: - No wrapped tokens (eliminates WBTC/sBTC trust assumptions) - No centralized custody (eliminates Celsius/BlockFi risk) - No oracles (maintains Phase 1's competitive bidding innovation) - Pure Bitcoin security (leverages Bitcoin's $2T security budget)

This makes the protocol **the gold standard** for Bitcoin-backed lending, setting up Phase 3's multi-chain liquidity integration.

---

2. Problem Statement

2.1 Limitations of Phase 1 (sBTC Dependency)

While Phase 1 successfully demonstrates oracle-free lending with competitive bidding auctions, it relies on **sBTC** as a Bitcoin representation on Stacks. This introduces several trust assumptions:

Trust in sBTC Peg Mechanism

- **Current sBTC Design**: Decentralized signers maintain 1:1 peg
- **Trust Assumption**: Signers must be honest and available
- **Risk**: If signers collude or fail, peg could break
- **Scale Limit**: sBTC capacity limited by signer network

Wrapped Token Risks (Historical Context)

- **WBTC**: Centralized custody by BitGo (single point of failure)
- **renBTC**: Project shut down, holders scrambled to redeem
- **hBTC**: Low liquidity, trust in centralized bridge
- **General Issue**: All wrapped BTC requires trusting custodians

Market Perception

- Users prefer **actual Bitcoin** over representations
- “Not your keys, not your coins” applies to wrapped tokens
- Institutional users require custody transparency
- Maximum TVL limited by comfort with sBTC peg

2.2 The Native Bitcoin Challenge

Holding actual Bitcoin in a trustless protocol is technically challenging:

Bitcoin’s Limited Scripting: - No Turing-complete smart contracts - Limited programmability compared to Ethereum/Stacks - Multisig requires on-chain transaction for setup - Complex business logic not feasible on Bitcoin L1

The Dilemma: - Smart contracts on Stacks (flexible, low-cost, programmable) - Bitcoin custody on Bitcoin (trustless, secure, native) - Need: **Bridge these two worlds without centralization**

2.3 Existing Solutions Are Insufficient

BitVM: - Theoretical framework for Bitcoin computation - Not production-ready (highly experimental) - Requires optimistic verification (complex) - No mature implementation available

Federated Sidechains (Liquid, RSK): - Require trusting federation of signers - Not meaningfully more decentralized than sBTC - Add another layer of abstraction

Lightning Network: - Designed for payments, not lending collateral - Requires channels and liquidity management - Not suitable for long-term collateral locks

What We Need: Direct Bitcoin custody (no wrapped tokens)
Decentralized validator network (no federation trust)

Threshold signatures (no single point of failure)
Integration with Stacks (leverage smart contract logic)
Production-ready today (not experimental)

---

3. Solution Overview

3.1 Architecture: Hybrid Bitcoin + Stacks

Phase 2 uses a **hybrid architecture** that leverages the strengths of both chains:

Bitcoin Layer (Custody): - Holds actual BTC collateral in threshold multisig addresses - Provides ultimate security through Bitcoin's consensus - Uses Taproot/Schnorr for efficient multisig - No trust assumptions beyond Bitcoin's security model

Stacks Layer (Logic): - Maintains smart contract logic for auctions, loans, NFTs - Coordinates validator actions through consensus - Provides user interface and state management - Keeps transaction costs low (~$1-5 vs Bitcoin's $10-50)

Validator Network (Bridge): - Decentralized network of 15+ independent operators - Run both Bitcoin and Stacks nodes - Hold threshold signature key shares - Sign Bitcoin transactions based on Stacks contract state

3.2 Threshold Signature Scheme

Technology: FROST (Flexible Round-Optimized Schnorr Threshold) or MuSig2

Configuration: 10-of-15 multisig - 15 total validators - 10 signatures required (67% threshold) - 5 validators can be offline without stopping operations - 6+ colluding validators needed to steal (highly unlikely with good incentives)

Key Generation (Distributed):

```
No single party ever has full private key!

Each validator i generates:
- Secret share: s_i (kept private, never shared)
- Public key share: P_i = s_i * G

Combined public key:
- P = P_1 + P_2 + ... + P_15

Bitcoin Address:
- Taproot address derived from P
- Looks like any normal Bitcoin address
- No on-chain indication of multisig (privacy!)
```

Signature Generation (Threshold):

```
To spend Bitcoin:
1. Stacks contract reaches consensus on action
2. 10+ validators see the consensus
3. Each validator creates partial signature
4. Coordinator aggregates 10+ partial signatures
5. Combined signature is valid for public key P
6. Broadcast to Bitcoin network
```

3.3 Loan Lifecycle with Native Bitcoin

1. Borrower Creates Loan Request:

```
User → Stacks Contract
- Specify: loan amount, max repayment, duration
- Contract generates unique Bitcoin address (threshold multisig)
- Borrower sends BTC to this address
- Bitcoin tx confirmed → Stacks contract activated
```

2. Auction & Bidding (Unchanged from Phase 1):

```
- Competitive bidding auction on Stacks
- Lenders place bids (total repayment amounts)
- Lowest bid wins when auction ends
- All logic on Stacks (cheap, fast)
```

3. Loan Finalization:

```
Stacks Contract → Validators
- Contract finalizes auction with winning bid
- Winning lender provides Stablecoin on Stacks
- Contract signals validators to release BTC
- 10+ validators sign Bitcoin transaction
- BTC transfers to borrower
- Lender receives NFT position
```

4. Repayment:

```
Borrower → Stacks Contract
- Borrower repays stablecoin on Stacks (fixed amount from auction)
- Contract signals validators to release collateral
- 10+ validators sign Bitcoin transaction
- BTC returns to borrower
- Both NFTs burned
```

5. Default Scenario:

```
If borrower doesn't repay by maturity:
- Stacks contract enters "defaulted" state
- Lender can claim collateral
- Contract signals validators
- 10+ validators sign Bitcoin transaction
- BTC transfers to lender
```

3.4 Validator Network Design

Validator Roles Responsibilities: - Monitor both Bitcoin and Stacks blockchains - Maintain synchronized state view - Participate in threshold signing when required - Monitor other validators for liveness - Submit signed transactions to Bitcoin network

Requirements: - Run full Bitcoin node (verify all Bitcoin state) - Run full Stacks node (verify all Stacks state) - Maintain >99% uptime - Secure key management (HSM recommended) - Network connectivity (low latency) - Stake/bond collateral (economic security)

Incentives: - Earn fees: 0.1% of loan volume processed - Distributed proportionally to all 15 validators - Slashing for downtime or misbehavior - Reputation system for reliable validators

Validator Selection & Rotation **Phase 2 Launch** (Months 7-13): - **Permissioned set**: 15 curated validators - Selection criteria: reputation, uptime history, geographic diversity - Includes: protocol team (3), community members (5), partners (7) - Fixed set for initial 6 months (stability focus)

Future Phase (Post-Phase 2): - **Permissionless entry**: Anyone can apply to be validator - Stake requirement: e.g., $50,000-100,000 in STX or BTC - Rotation mechanism: validators can be voted in/out - Governance: STX holders or future DAO

Geographic Distribution Target distribution for censorship resistance: - North America: 5 validators (33%) - Europe: 5 validators (33%) - Asia: 3 validators (20%) - Other: 2 validators (14%)

No more than 3 validators in same jurisdiction.

3.5 Security Properties

Trustlessness: No single validator can steal funds (need 10/15)
No custodian (pure threshold cryptography)
No federation (open validator set long-term)
No wrapped token (actual Bitcoin)

Liveness: 5 validators can be offline without issue
System continues with 10+ online validators
Graceful degradation (not binary failure)
Automatic failover if validators drop

Censorship Resistance: Geographic distribution prevents single-jurisdiction attack
Multiple validator entities (individuals, companies, DAOs)
6+ colluding validators needed to censor (highly unlikely)

Attack Scenarios:

Attack	Difficulty	Mitigation
Steal funds	Need to compromise 10/15 validators	High threshold, diversity, monitoring
Censor transaction	Need to control 6/15 validators	Geographic distribution, incentives
Network partition	Need to split validators	Multiple network paths, redundancy
Key extraction	Need to compromise validator HSMs	Hardware security, secure boot
Social engineering	Need to trick 10/15 operators	Strict procedures, multi-approval

3.6 Benefits vs Phase 1

Aspect	Phase 1 (sBTC)	Phase 2 (Native BTC)
Collateral	sBTC (wrapped)	Bitcoin (native)
Trust Assumptions	sBTC signers	Validator threshold
Peg Risk	Yes (sBTC peg)	No (actual BTC)
Bitcoin Security	Indirect	Direct
Market Perception	"Wrapped token"	"Real Bitcoin"
TVL Ceiling	Limited by sBTC supply	Limited by validator capacity
User Confidence	Medium-High	Very High
Custody Transparency	sBTC multisig	Observable threshold

Why This Matters: - Institutional users prefer actual Bitcoin custody - Eliminates entire category of risk (peg breaks) - No dependency on sBTC project success - Scales independently - More credible for large loans ($500K+)

---

4. User Personas

4.1 Primary Persona: Large-Scale Bitcoin Miners (Borrowers)

Phase 2 targets **larger mining operations** that Phase 1's sBTC might not accommodate:

Demographics

- **Operation size**: 100-5,000 ASICs (medium to large miners)
- **Geography**: USA (30%), Canada (20%), Kazakhstan (15%), Russia (10%), Norway (10%), Other (15%)
- **Technical level**: Very high - infrastructure operators
- **Bitcoin philosophy**: Bitcoin maximalists, only use BTC
- **Monthly revenue**: $100,000 - $5,000,000+

Needs and Goals

- **Primary**: Large working capital loans ($100K-$2M)
- **Secondary**: Trustless custody (no wrapped tokens acceptable)
- **Tertiary**: Institutional-grade security for large amounts
- **Financial**: Competitive rates for large loans (5-8% APR)

Pain Points with Phase 1

- sBTC supply might be insufficient for large loans
- Uncomfortable with wrapped token for large amounts
- Prefer direct Bitcoin custody
- Institutional policies may prohibit wrapped assets
- Higher perceived risk with sBTC peg for multi-million dollar amounts

User Story

> *"I run a 1,000 ASIC operation in Texas. My monthly electricity bill is $300,000. When BTC dropped from $100K to $85K, I needed $1M to cover operations without selling my mined Bitcoin.*
>
> *Phase 1 with sBTC was interesting, but for $1M+ I need actual Bitcoin custody. My CFO won't approve wrapped tokens for that size. We need to see our BTC locked in an on-chain multisig address.*
>
> *With Phase 2's threshold custody, I can borrow $1M against 15 BTC collateral. I can verify the multisig address on Bitcoin. I can see the validator signatures. That's institutional-grade security.*
>
> *I'm willing to pay 6-7% APR if it means trustless custody of actual Bitcoin."*

— **Marcus, 42, Mining Operation Director, Houston TX**

Success Metrics for This Persona

- 20+ large miners using Phase 2 by Month 13
- Average loan size: $250,000 - $1,000,000
- Total volume: $5M+ from large miners
- 80%+ of these miners prefer Phase 2 over Phase 1
- Repeat borrowing rate: >60%

4.2 Primary Persona: Institutional Lenders (Liquidity Providers)

Demographics

- **Profile**: Family offices, crypto hedge funds, high net worth individuals
- **Experience level**: Advanced - understand Bitcoin/DeFi deeply
- **Capital**: $500,000 to $50M+
- **Investment thesis**: Conservative crypto yield with Bitcoin collateral
- **Risk tolerance**: Low to medium - want best security

Needs and Goals

- **Primary**: Institutional-grade custody transparency
- **Secondary**: Large position sizes ($100K-$1M per loan)
- **Tertiary**: Verifiable Bitcoin custody (not wrapped tokens)
- **Trust**: Proof that collateral is in decentralized threshold custody

Pain Points with Phase 1

- sBTC limits confidence for large positions
- Wrapped tokens not acceptable to compliance teams
- Peg risk unacceptable for multi-million portfolios
- Want to see actual Bitcoin multisig on-chain

User Story

> *"I manage a $20M family office crypto portfolio. We're interested in Bitcoin-collateralized lending at 7-9% APR - much better than tradfi bonds.*
>
> *But we won't touch wrapped Bitcoin. We've seen WBTC centralization, we've seen renBTC shut down. For us, it's actual Bitcoin or nothing.*
>
> *Phase 2's threshold custody is exactly what we need. We can verify the Bitcoin multisig address. We can see the validator signatures on every transaction. We can audit the whole process.*
>
> *We're prepared to deploy $2-5M into Phase 2 if security is rock-solid."*

— **Elizabeth, 55, Family Office Manager, Singapore**

Success Metrics for This Persona

- 10+ institutional lenders in Phase 2
- Average position size: $200,000+
- Total institutional capital: $3M+
- 90%+ cite "native Bitcoin custody" as key reason
- Zero security incidents

4.3 Secondary Persona: Bitcoin Maximalists (Both Roles)

Demographics

- **Profile**: Hardcore Bitcoin believers
- **Philosophy**: Bitcoin only, no altcoins, no wrapped tokens
- **Technical level**: Very high - run own nodes
- **Values**: Decentralization, trustlessness, censorship-resistance

Needs and Goals

- **Primary**: Pure Bitcoin solution (no compromises)
- **Secondary**: Verifiable decentralization (no trust required)
- **Tertiary**: Support Bitcoin-native infrastructure

Why Phase 2 Appeals to Them

- Actual Bitcoin (not wrapped)
- Threshold custody (no custodian)
- Decentralized validators (no single point of control)
- Verifiable on Bitcoin blockchain
- Aligns with Bitcoin ethos

User Story

> *"I don't touch wrapped Bitcoin. I don't touch altcoins. Bitcoin is the only real money.*
>
> *Phase 1 with sBTC? That's an altcoin token. Not interested.*
>
> *Phase 2 with threshold signatures holding actual Bitcoin? Now that's interesting. That's how Bitcoin lending should work. Pure Bitcoin security."*

— **Max, 38, Bitcoin Developer, Remote**

---

5. Feature Requirements

5.1 Threshold Custody Features (Must Have - P0)

FR1.1: Distributed Key Generation (DKG) **Description**: Generate threshold multisig keys without any single party knowing the full private key.

Functional Requirements: - 15 validators participate in DKG ceremony - Each validator generates secret share (never leaves their system) - Combined public key computed from all shares - No validator ever has full private key - Verifiable Secret Sharing (VSS) ensures correctness - Public key shares published on-chain for transparency

DKG Process:

```
Round 1: Commitment Phase
- Each validator commits to their secret polynomial
- Commitments published and verified

Round 2: Share Distribution
- Each validator sends encrypted shares to others
- Shares transmitted over secure channels

Round 3: Verification
- Each validator verifies received shares
- Complaints raised if shares invalid

Round 4: Finalization
- If no complaints: DKG succeeds
- Combined public key computed
- Key shares ready for threshold signing
```

Acceptance Criteria: - [ ] DKG ceremony completes with 15 validators - [ ] Combined public key is Taproot-compatible - [ ] Each validator can prove their key share is valid - [ ] Full private key is never reconstructed - [ ] Process completes in <10 minutes - [ ] Ceremony can be audited on-chain

Security Properties: - No single validator learns full key - No subset <10 validators can reconstruct key - Malicious validators detected during verification - Byzantine fault tolerant (up to 5 malicious validators)

FR1.2: Threshold Signature Generation **Description**: Generate valid Bitcoin signatures using threshold cryptography (10-of-15).

Functional Requirements: - Any 10+ validators can create valid signature - Partial signatures combined into final signature - Final signature indistinguishable from single-key signature (Taproot privacy) - No interactive rounds needed (non-interactive threshold signing) - Signature valid for Bitcoin consensus rules

Signing Process:

```
Step 1: Request Initiated
- Stacks contract determines action needed (e.g., release collateral)
- Contract publishes signing request with:
  - Transaction details (outputs, amounts, fees)
  - Verification proof (e.g., loan was repaid)
  - Nonce for this signing session

Step 2: Validators Verify
- Each validator independently verifies:
  - Stacks contract state is correct
  - Action is authorized per contract rules
  - Bitcoin transaction matches contract intent
  - No double-signing or replay

Step 3: Partial Signatures
- Each validator that agrees creates partial signature
- Partial signature based on their key share
- Partial signatures published to coordinator

Step 4: Aggregation
- Once 10+ partial signatures collected
- Coordinator aggregates into final signature
- Final signature is standard Schnorr signature

Step 5: Broadcast
- Bitcoin transaction with final signature broadcast
- Confirms in Bitcoin block
- Stacks contract updated with confirmation
```

Acceptance Criteria: - [ ] 10 validator signatures sufficient to sign - [ ] Signing completes in <5 minutes typically - [ ] Failed signing doesn't lock system (retry possible) - [ ] Signature passes Bitcoin consensus validation - [ ] Process works even with 5 validators offline - [ ] All partial signatures logged for audit

Edge Cases: - Exactly 10 validators online → Should still work - 11+ validators but disagree on action → Majority rules - Validators sign conflicting transactions → First-seen-valid wins - Network partition splits validators → Larger partition continues

---

FR1.3: Bitcoin Transaction Construction **Description**: Build valid Bitcoin transactions that validators will sign.

Functional Requirements: - Parse Stacks contract state to determine Bitcoin action - Construct appropriate Bitcoin transaction: - Collateral lock (P2TR output to threshold address) - Collateral release (spend from threshold address to borrower) - Collateral claim (spend from threshold address to lender on default) - Calculate appropriate fees (dynamic fee estimation) - Handle UTXO management (coin selection) - Support RBF (Replace-By-Fee) for stuck transactions

Transaction Types:

Type 1: Collateral Lock (Borrower → Threshold Address)

```
Inputs:
  - Borrower's UTXOs (sufficient BTC + fees)

Outputs:
  - Threshold address: [collateral_amount]
  - Change address: [remaining if any]

Note: Borrower signs this themselves (not threshold)
```

Type 2: Collateral Release (Threshold Address → Borrower)

```
Inputs:
  - Threshold address UTXO: [collateral_amount]

Outputs:
  - Borrower address: [collateral_amount - fee]

Signature: Threshold signature (10-of-15 validators)
Trigger: Stacks contract confirms repayment received
```

Type 3: Collateral Claim (Threshold Address → Lender)

```
Inputs:
  - Threshold address UTXO: [collateral_amount]

Outputs:
  - Lender address: [collateral_amount - fee]

Signature: Threshold signature (10-of-15 validators)
Trigger: Stacks contract confirms default (maturity passed, no repayment)
```

Fee Management: - Query mempool for current fee rates - Target 1-2 block confirmation (medium priority) - Allow manual fee bumping if transaction stuck - Reserve small amount of BTC for fee buffer

Acceptance Criteria: - [ ] Transactions are valid per Bitcoin consensus - [ ] Fees are competitive (not overpaying) - [ ] All outputs are correct addresses and amounts - [ ] Change handling works correctly - [ ] RBF enabled for all threshold-signed transactions - [ ] Transaction monitoring until confirmation

---

FR1.4: Validator Monitoring & Coordination **Description**: Monitor validator health and coordinate threshold signing sessions.

Functional Requirements: - Real-time monitoring of all 15 validators: - Online/offline status - Last heartbeat timestamp - Bitcoin node sync status - Stacks node sync status - Partial signature latency - Coordinator selects best validators for signing - Automatic failover if validators drop during signing - Alert system for validator issues - Dashboard showing validator network health

Monitoring Metrics:

Metric	Green	Yellow	Red
Uptime	>99%	95-99%	<95%
Response Time	<30s	30-60s	>60s
Bitcoin Sync	<1 block behind	1-5 blocks	>5 blocks
Stacks Sync	<1 block behind	1-5 blocks	>5 blocks
Signing Success	>99%	95-99%	<95%

Coordinator Role: - Aggregates partial signatures - Requests retries if some validators timeout - Publishes final signature to Bitcoin network - Updates Stacks contract with Bitcoin tx confirmation - Not trusted (anyone can verify signatures are correct)

Acceptance Criteria: - [ ] All validators monitored in real-time - [ ] Dashboard shows current network health - [ ] Alerts fire for validator issues (email, Discord) - [ ] Coordinator successfully aggregates signatures - [ ] System continues with 10+ validators online - [ ] Graceful degradation if validators drop

---

5.2 Smart Contract Integration (Must Have - P0)

FR2.1: Bitcoin Event Monitoring **Description**: Stacks contracts must monitor Bitcoin blockchain for collateral deposits and confirmations.

Functional Requirements: - Stacks contract listens for Bitcoin transactions - Detects deposits to threshold addresses - Requires 3+ Bitcoin confirmations before activating loan - Updates contract state when Bitcoin events occur - Handles Bitcoin reorgs gracefully

Bitcoin → Stacks Flow:

```
1. Borrower sends BTC to threshold address
2. Bitcoin transaction confirms (1 block)
3. Validators observe transaction
4. Validators submit proof to Stacks contract
5. After 3 confirmations, contract activates loan
6. Auction begins on Stacks
```

Proof Structure:

```
Bitcoin Transaction Proof {
  txid: Bitcoin transaction ID
  block_height: Block containing transaction
  merkle_proof: Proof transaction is in block
  outputs: List of outputs with amounts
  confirmations: Number of confirmations
}
```

Acceptance Criteria: - [ ] Contract correctly parses Bitcoin transaction data - [ ] 3+ confirmations required before activation - [ ] Reorgs handled (contract waits for finality) - [ ] Invalid proofs rejected - [ ] Multiple validators can submit same proof (deduplicated)

FR2.2: Validator Action Requests **Description**: Stacks contract publishes requests for validators to sign Bitcoin transactions.

Functional Requirements: - Contract emits "action required" events: - `release-collateral` (on successful repayment) - `claim-collateral` (on default) - `refund-collateral` (if auction fails) - Events include all necessary transaction details - Validators automatically respond to valid requests - Multiple validators verify before signing - Prevents unauthorized Bitcoin spends

Action Request Format:

```
{
  action: "release-collateral" | "claim-collateral" | "refund-collateral",
  loan-id: uint,
  bitcoin-txid: (buff 32),          ;; Collateral UTXO
  recipient: (string-ascii 62),      ;; Bitcoin address
  amount: uint,                      ;; Satoshis
  fee: uint,                         ;; Satoshis for miner fee
  nonce: uint                        ;; Unique per request
}
```

Validator Verification Checklist:

```
Before signing, each validator checks:
 Stacks contract state matches request
 Action is authorized per contract rules
 Recipient address matches loan participant
 Amount matches contract records
 No duplicate/replay request (nonce check)
 Bitcoin UTXO actually exists and is spendable
```

Acceptance Criteria: - [ ] Events correctly formatted and complete - [ ] Validators receive events in real-time (<10s) - [ ] All validators independently verify before signing - [ ] Malicious requests rejected by honest validators - [ ] Partial signature aggregation works correctly - [ ] Bitcoin transaction broadcasts after 10+ signatures

---

FR2.3: Cross-Chain State Synchronization **Description**: Keep Stacks and Bitcoin states synchronized despite different block times.

Functional Requirements: - Bitcoin block time: ~10 minutes (variable) - Stacks block time: ~10 minutes (anchored to Bitcoin) - Handle timing differences gracefully - Maintain consistency even during reorgs - Provide UI with accurate status across both chains

State Machine:

```
Loan Status Flow (spans both chains):

1. "Created" (Stacks) → Waiting for BTC deposit
2. "Collateral-Pending" (Bitcoin tx seen, <3 confirmations)
3. "Auction" (Bitcoin confirmed, auction active on Stacks)
4. "Active" (Auction finalized on Stacks)
5a. "Repay-Pending" (Repayment on Stacks, BTC release initiated)
5b. "Released" (BTC release confirmed on Bitcoin)
6. "Completed" (Both chains finalized)

OR

5a. "Default" (Maturity passed on Stacks)
5b. "Claim-Pending" (BTC claim initiated)
5c. "Claimed" (BTC claimed confirmed on Bitcoin)
6. "Defaulted" (Both chains finalized)
```

Acceptance Criteria: - [ ] State transitions happen correctly across chains - [ ] No state desync even during reorgs - [ ] UI reflects current state accurately - [ ] Timing edge cases handled (e.g., Stacks faster than Bitcoin) - [ ] Recovery possible if synchronization fails

---

5.3 User Interface Updates (Must Have - P0)

FR3.1: Native Bitcoin Deposit Flow **Description**: Users must be able to lock actual Bitcoin as collateral.

User Flow:

```
1. User creates loan request on Stacks
```

```
2. UI generates unique threshold multisig Bitcoin address
3. UI displays address as:
   - Text (copyable)
   - QR code (scannable)
   - Verification info (multisig details)
4. User sends BTC from their wallet
5. UI monitors Bitcoin mempool for transaction
6. Shows "Pending confirmation" (0-3 blocks)
7. After 3 confirmations: "Collateral locked, auction starting"
```

Address Display:

```
Send Bitcoin Collateral


Amount: 1.5 BTC

[QR CODE]

bc1p8xj2f7a...k3m9tn2 [Copy]

   Do NOT send from exchange!
    You must control the address

   This is a 10-of-15 multisig address
    View validators →

Status: Waiting for your deposit...
[View on Bitcoin Explorer]
```

Acceptance Criteria: - [ ] Address displayed clearly (text + QR) - [ ] Address is valid Bitcoin address (bech32) - [ ] Clear instructions to avoid exchange deposits - [ ] Real-time monitoring of Bitcoin mempool - [ ] Progress indicator (0/3, 1/3, 2/3, 3/3 confirmations) - [ ] Link to Bitcoin explorer for transparency

---

FR3.2: Validator Transparency Dashboard **Description**: Users must be able to verify the decentralization and security of the validator network.

Dashboard Sections:

1. **Validator List**:

```
Active Validators (15 total)
```

```
 ID        Location   Uptime     Last      Status
                                 Seen

 Val-01    USA        99.8%      5s ago      Up
 Val-02    Germany    99.9%      3s ago      Up
 Val-03    Japan      99.7%      8s ago      Up
 Val-04    Canada     99.6%      15s ago     Up
 Val-05    UK         97.2%      2m ago      Slow
 ...       ...        ...        ...       ...
```

2. Geographic Distribution Map: - World map showing validator locations - Color-coded by status (green/yellow/red) - Hover for details (name, uptime, last activity)

3. Network Health:

```
Overall Status:  Healthy

Active Validators: 14/15 (93%)
Signing Threshold: 10 required
Current Capacity: 14 available (140% of minimum)
Average Response Time: 12 seconds
Last Signature: 3 minutes ago (Loan #157)
```

4. Recent Signatures:

```
  Recent Threshold Signatures

 Loan    Action     Sigs      Bitcoin Tx

 #157    Release    12/15     3a8f2c...
 #155    Release    11/15     7b3d91...
 #154    Claim      13/15     2f8a34...
 #152    Release    10/15     8c2f74...
```

Acceptance Criteria: - [ ] All 15 validators listed with current status - [ ] Geographic map shows distribution - [ ] Real-time updates (<30 second refresh) - [ ] Historical uptime data available - [ ] Links to validator profiles (if public) - [ ] Clear explanation of threshold security model

---

FR3.3: Bitcoin Transaction Monitoring **Description**: Users must be able to track Bitcoin transactions related to their loans.

Loan Detail View:

```
   Loan #42 - Bitcoin Transactions


   Collateral Deposit
   1.5 BTC → bc1p8xj2f7a...k3m9tn2
   Txid: 3a8f2c91...
   Confirmations: 87
   [View on Bitcoin Explorer]



   Collateral Release (pending)
   1.5 BTC → bc1qxy8w2n3...r5t7m9
   Status: Waiting for 10 validator signatures
   Current: 8/10 signatures received
   ETA: ~5 minutes
```

Acceptance Criteria: - [ ] All Bitcoin transactions displayed clearly - [ ] Confirmation count shown and updated - [ ] Links to Bitcoin explorer for verification - [ ] Threshold signature progress visible - [ ] Clear status indicators (pending, confirmed, failed)

5.4 Migration Features (Should Have - P1)

FR4.1: Phase 1 → Phase 2 Loan Migration **Description**: Allow existing Phase 1 (sBTC) loans to migrate to Phase 2 (native BTC).

Functional Requirements: - User can opt-in to migrate active loan - Process: 1. Repay Phase 1 loan (get sBTC back) 2. Convert sBTC → BTC (via sBTC redemption) 3. Deposit BTC to Phase 2 threshold address 4. Create equivalent Phase 2 loan - Preserve favorable terms from Phase 1 if possible - Migration window: Months 7-13 (during Phase 2 rollout)

Migration UI:

```
Your Phase 1 Loan: #123
Collateral: 1.5 sBTC
Status: Active


   Migrate to Phase 2?

   Benefits:
     Native Bitcoin custody
     Eliminated sBTC peg risk
     Institutional-grade security

   Process:
   1. Repay current loan (automatic)
   2. Convert sBTC to BTC
   3. Deposit to Phase 2

   [Migrate Now] [Learn More]
```

Acceptance Criteria: - [ ] Migration process is seamless (1-click if possible) - [ ] Clear explanation of benefits - [ ] No loss of value during migration - [ ] Migration tracking in UI - [ ] Support and documentation available

---

6. Architecture Requirements

6.1 Validator Node Requirements

Hardware: - CPU: 8+ cores (Intel/AMD x86_64) - RAM: 32GB minimum (64GB recommended) - Storage: 2TB NVMe SSD (Bitcoin full node ~600GB, Stacks ~100GB, growth room) - Network: 100 Mbps+ bandwidth, <50ms latency to other validators - Uptime: >99.5% target

Software: - Bitcoin Core v25.0+ (full node, txindex enabled) - Stacks Node v2.5+ (full node) - Validator software (custom, written in Rust) - HSM support (optional but recommended): YubiHSM 2, Ledger, etc.

Security: - Key material stored in HSM or secure enclave - Firewall: Only p2p ports open (not RPC) - Regular security updates - Intrusion detection system (IDS) - Encrypted communications between validators

Monitoring: - Prometheus metrics exported - Grafana dashboards for node health - Alert manager for issues - Logging aggregation (e.g., ELK stack)

6.2 Threshold Cryptography Stack

Library: Secp256k1-zkp (Schnorr threshold)

Alternative: FROST Implementation (production-ready)

Key Features Needed: - Schnorr signature support (Bitcoin Taproot) - Threshold signature generation (M-of-N) - Distributed key generation (DKG) - Verifiable secret sharing (VSS) - Non-interactive signing (no interactive rounds)

Integration Points: - Rust validator software calls library - Bitcoin transaction signing pipeline - Key share management (encrypted storage) - Partial signature aggregation

---

6.3 Network Communication

Validator-to-Validator: - Protocol: libp2p (peer-to-peer networking) - Transport: TCP with TLS encryption - Gossip protocol for partial signatures - DHT for peer discovery - NAT traversal support

Validator-to-Coordinator: - REST API for submitting partial signatures - WebSocket for real-time updates - Authentication via validator public keys - Rate limiting per validator

Coordinator-to-Bitcoin: - Bitcoin Core RPC API - Transaction broadcasting via multiple nodes - Mempool monitoring - Fee estimation queries

---

6.4 Disaster Recovery

Validator Failure Scenarios:

Scenario 1: Single Validator Down - Impact: None (need 10/15, have 14 remaining) - Response: Monitor, contact operator, wait for recovery - Threshold: Alert if down >24 hours

Scenario 2: 5 Validators Down - Impact: Minimal (still have 10/15) - Response: Urgent investigation, activate backup validators - Threshold: P1 alert, all hands on deck

Scenario 3: 6+ Validators Down - Impact: CRITICAL (below 10/15 threshold) - Response: Emergency protocol: - Pause new loans immediately - Prioritize bringing validators back online - Activate backup validators - Consider emergency validator recruitment - Allow existing loans to complete without new signatures - Threshold: P0 alert, 24/7 response

Key Recovery: - Each validator backs up encrypted key share - Recovery requires validator identity proof - No single backup location (distributed) - Regular backup testing (quarterly)

Network Split: - Monitor for network partitions - Larger partition (8+ validators) continues - Smaller partition waits for reconnection - No conflicting signatures across partitions

---

7. Non-Functional Requirements

7.1 Security (Critical - P0)

NFR1.1: Threshold Security **Requirements**: - 10-of-15 threshold enforced cryptographically - No single validator can steal funds - No subset <10 validators can reconstruct key - Key generation ceremony auditable - All signatures verified before Bitcoin broadcast

Acceptance Criteria: - [ ] Cryptographic proof that threshold holds - [ ] Independent security audit of threshold implementation - [ ] Penetration testing of validator network - [ ] Bug bounty program ($100K+ rewards) - [ ] No critical vulnerabilities in audit

---

NFR1.2: Validator Security **Requirements**: - Validator keys stored in HSMs (hardware security modules) - No remote access to validator key material - Multi-factor authentication for validator access - Regular security updates within 48 hours - Intrusion detection on all validator nodes

Acceptance Criteria: - [ ] 100% of validators use HSMs or secure enclaves - [ ] Security audits pass for all validator setups - [ ] No successful attacks on validator keys - [ ] Incident response plan tested quarterly

---

7.2 Reliability (High Priority - P0)

NFR2.1: Validator Uptime **Requirements**: - Individual validator uptime: >99.5% - Network capacity: Always >10 validators online - Maximum signing latency: <5 minutes (typical: <2 minutes) - Zero downtime for user-facing operations

Monitoring: - Real-time validator health checks - Alert if <12 validators online - Automated failover to backup validators - Geographic redundancy

Acceptance Criteria: - [ ] Network uptime: 99.9%+ over 6 months - [ ] No periods with <10 validators - [ ] Average signing time <3 minutes - [ ] Zero failed signatures due to validator unavailability

---

NFR2.2: Bitcoin Transaction Reliability **Requirements**: - 100% of legitimate transactions must eventually confirm - Failed transactions must be retryable - RBF (Replace-By-Fee) for stuck transactions - Fee estimation accuracy: ±20% of optimal

Acceptance Criteria: - [ ] <0.1% transaction failure rate - [ ] All failures are recoverable - [ ] Median confirmation time: <30 minutes - [ ] Maximum confirmation time: <6 hours (even with low fees)

7.3 Performance (Medium Priority - P1)

NFR3.1: Signing Performance Targets: - Threshold signature generation: <2 minutes (typical) - Maximum signing latency: <5 minutes - Support 10+ concurrent signing sessions - No performance degradation with load

Acceptance Criteria: - [ ] 95th percentile signing time <3 minutes - [ ] 99th percentile signing time <5 minutes - [ ] Can handle 50 loans per day - [ ] No bottlenecks under load testing

---

7.4 Decentralization (High Priority - P0)

NFR4.1: Geographic Distribution Requirements: - No more than 3 validators in same country - No more than 5 validators in same region - Minimum 3 continents represented - No cloud provider hosts >40% of validators

Initial Distribution: - North America: 5 validators - Europe: 5 validators - Asia: 3 validators - Other: 2 validators

Acceptance Criteria: - [ ] Geographic diversity meets targets - [ ] No single jurisdiction can censor - [ ] Cloud provider diversity maintained - [ ] Validators in politically diverse regions

---

NFR4.2: Validator Independence Requirements: - No single entity controls >2 validators (13%) - Validators have different operators/owners - No conflicts of interest (validators can't be borrowers/lenders) - Public disclosure of validator identities (optional anonymity)

Acceptance Criteria: - [ ] All validators are independent entities - [ ] No undisclosed common ownership - [ ] Conflicts of interest documented and mitigated - [ ] Validator registry publicly available

---

8. Success Metrics

8.1 Primary Success Metrics (Phase 2)

Metric	Month 10	Month 13	Measurement
Native BTC Loan Volume	$2M	$5M+	Sum of all Phase 2 loans

Metric	Month 10	Month 13	Measurement
Active Validators	15	15	Count of online validators
Network Uptime	99.5%+	99.9%+	% time 10 validators online
Large Loans (>$100K)	5	15+	Count of loans over threshold
Institutional Lenders	3	10+	Lenders with $100K+ deployed
Migration Rate	30%	50%+	% of Phase 1 loans migrated
Average Signing Time	<3 min	<2 min	Median threshold signature time
Transaction Success Rate	>99%	>99.5%	Bitcoin tx confirmed successfully

8.2 Secondary Success Metrics

Metric	Target	Measurement
Validator Uptime (individual)	>99.5%	Average across all validators
Geographic Distribution	Meets target	5/5/3/2 (NA/EU/Asia/Other)
Signature Participation	>10/15 always	Minimum signatures per tx
Zero Key Compromises	0	Security incidents
Code Open-Sourced	100%	Validator software public
Community Validators	7/15	Non-team operators
Security Audit Complete	Pass	No critical issues
Bug Bounty Paid	$0 (no bugs)	Security program active

8.3 Comparison to Phase 1

Metric	Phase 1 (sBTC)	Phase 2 (Native BTC)	Improvement
Total Volume	$1M	$5M+	5x
Average Loan Size	$50K	$250K	5x
Large Loans (>$100K)	~10%	~50%	5x
Institutional Users	2-3	10+	3-5x
User Trust	Medium-High	Very High	Qualitative

---

9. Phase 2 Deliverables

9.1 Deliverable Timeline

Code	Deliverable	Timeline	Budget	Dependencies
D2.1	Threshold Crypto Library Integration	Month 7-8	$45,000	None
D2.2	Validator Node Software (Rust)	Month 7-9	$62,000	D2.1
D2.3	DKG Ceremony Implementation	Month 8-9	$28,000	D2.1, D2.2
D2.4	Stacks Bitcoin Bridge Logic	Month 8-10	$45,000	D2.2
D2.5	Validator Network Launch (15 validators)	Month 9-10	$38,000	D2.2, D2.3
D2.6	Security Audit (Threshold + Validators)	Month 10-11	$52,000	All above
D2.7	Frontend Updates (Native BTC UI)	Month 9-11	$25,000	D2.4
D2.8	Migration Tools (Phase 1 → Phase 2)	Month 11-12	$12,000	D2.7
D2.9	Launch & First $1M Native BTC Volume	Month 11-13	$5,000	All above

Total Phase 2 Budget: $312,000
Total Phase 2 Duration: 7 months (Months 7-13)

9.2 Detailed Deliverable Descriptions

D2.1: Threshold Crypto Library Integration ($45,000) **Scope**: - Research and select optimal threshold signature library - Options: FROST, secp256k1-zkp MuSig2, custom implementation - Integrate library with Rust validator software - Implement Bitcoin Taproot signature generation - Test suite for threshold signature correctness

Deliverables: - Library integrated and tested - Benchmark results (signing performance) - Documentation on cryptographic approach - Test vectors for signature verification

Success Criteria: - Signatures valid per Bitcoin consensus - Signing time <5 seconds on standard hardware - 100% test coverage of cryptographic functions - Independent cryptographer review

---

D2.2: Validator Node Software ($62,000) **Scope**: - Build Rust application for validator nodes - Features: - Monitor Bitcoin and Stacks blockchains - Participate in threshold signing - Communicate with other validators (libp2p) - Manage encrypted key shares - Export Prometheus metrics - Configuration management - Logging and monitoring - Documentation and deployment guides

Deliverables: - Rust binary (Linux x86_64) - Docker image for easy deployment - Configuration files and examples - Setup documentation - Monitoring dashboards (Grafana)

Success Criteria: - Binary runs stably for 7+ days without restart - Resource usage acceptable (<4GB RAM, <10% CPU idle) - All functionality tested (unit + integration) - Documentation clear enough for operators

---

D2.3: DKG Ceremony Implementation ($28,000) **Scope**: - Implement Distributed Key Generation protocol - Support 15-validator ceremony - Byzantine fault tolerance (handle malicious validators) - Verification and commitment rounds - Export combined public key - Ceremony auditability (all messages logged)

Deliverables: - DKG ceremony software (part of validator binary) - Ceremony coordinator scripts - Audit logs from test ceremonies - Documentation on ceremony process

Success Criteria: - DKG completes successfully with 15 honest validators - DKG detects and excludes malicious participants - Combined public key is Taproot-compatible - Ceremony takes <10 minutes

---

D2.4: Stacks Bitcoin Bridge Logic ($45,000) **Scope**: - Stacks smart contract updates: - Monitor Bitcoin deposits - Emit validator action requests - Track Bitcoin transaction confirmations - Bitcoin transaction builder - UTXO management - Fee estimation and RBF support - Cross-chain state sync

Deliverables: - Updated Stacks contracts (Clarity) - Bitcoin transaction construction library - Integration tests (cross-chain scenarios) - API for frontends to query bridge status

Success Criteria: - Deposits detected within 2 blocks - 100% of valid action requests result in Bitcoin txs - No funds ever stuck due to bridge logic - State sync works even during Bitcoin reorgs

---

D2.5: Validator Network Launch ($38,000) **Scope**: - Recruit and onboard 15 validators - Coordinate DKG ceremony - Deploy all validator nodes - Establish network communication - Initial testing on testnet - Mainnet launch

Deliverables: - 15 active validators on mainnet - Validator registry (public) - Network health dashboard - Validator operating procedures - Emergency contact list

Success Criteria: - All 15 validators online and signing - Network uptime >99% from day 1 - Geographic distribution met - No validator downtime >1 hour in first week

---

D2.6: Security Audit ($52,000) **Scope**: - Third-party audit of threshold implementation - Validator node software review - Bridge logic security assessment - Economic attack vectors - Operational security review - Fix critical and high-severity findings

Deliverables: - Audit report from reputable firm - List of findings with severity - Remediation for all criticals and highs - Re-audit confirmation - Published report

Success Criteria: - Zero critical vulnerabilities - All high-severity issues fixed - All medium issues addressed or documented - Public disclosure of final report

---

D2.7: Frontend Updates ($25,000) **Scope**: - Native Bitcoin deposit UI - Threshold multisig address generation - Bitcoin transaction monitoring - Validator transparency dashboard - Migration from Phase 1 interface

Deliverables: - Updated React frontend - Native BTC loan creation flow - Validator dashboard page - Migration wizard - User documentation

Success Criteria: - Users can create native BTC loans easily - Bitcoin addresses displayed clearly (QR + text) - Validator status visible and transparent - Migration tool works end-to-end - Mobile responsive

---

D2.8: Migration Tools ($12,000) **Scope**: - Tools to assist Phase 1 → Phase 2 migration - sBTC → BTC conversion automation - Loan term preservation - User communication and education - Analytics on migration rates

Deliverables: - Migration smart contracts - Migration UI workflow - User guides and FAQs - Email notifications - Migration tracking dashboard

Success Criteria: - 50%+ of Phase 1 users migrate - Migration completes in <24 hours - No value lost during migration - High user satisfaction with process

---

D2.9: Launch & Volume Milestone ($5,000) **Scope**: - Marketing for Phase 2 launch - Target institutional users - Content: case studies, security documentation - Conference presentations - Community engagement

Deliverables: - $1M+ in native BTC loan volume - 15+ large loans (>$100K) - 10+ institutional lenders - Security and trust documentation - Launch announcement materials

Success Criteria: - $5M total volume by Month 13 - Strong initial adoption from target users - Positive feedback on native BTC custody - Media coverage and social buzz

---

10. Technical Constraints

10.1 Bitcoin Constraints

Bitcoin Script Limitations: - No smart contracts (simple scripting only) - Multisig requires on-chain setup (expensive) - Taproot helps but still limited - Must use threshold signatures (off-chain aggregation)

Block Time Variability: - Bitcoin blocks: 10 min average (can vary 1-60 min) - Confirmation delays impact user experience - Must design UX around variable timing - Can't guarantee exact loan activation time

Transaction Costs: - Bitcoin fees spike during high activity - Median fee: $5-20 per transaction - Spikes: $50-100+ per transaction - Must reserve fee budget for each loan - RBF allows fee bumping if stuck

Mitigation: - Use SegWit and Taproot (lower fees) - Batch transactions where possible - Dynamic fee estimation - Allow manual fee bumping - Buffer for fee reserves

---

10.2 Stacks Constraints

Block Time Coupling: - Stacks blocks anchored to Bitcoin - Stacks block Bitcoin block 10 minutes - Cross-chain coordination slower than single-chain - Must design for ~10 min latencies

Contract Upgradability: - Phase 2 contracts should be immutable - If bugs found, must deploy new version - Migration process needed for users - No emergency pause (by design)

10.3 Threshold Cryptography Constraints

Signing Complexity: - Need 10/15 validators to sign - Requires network coordination - Latency: 1-5 minutes typical - If validators slow/down, delays possible

Key Management: - Each validator must securely store key share - Lost key shares = reduced threshold (bad) - Backup and recovery critical - No key rotation (would require new DKG)

Threshold Fixed: - 10-of-15 set at launch - Changing threshold requires new DKG - Can't easily add/remove validators - Future: need validator rotation mechanism

---

11. Migration from Phase 1

11.1 Backwards Compatibility

Phase 1 Continues Operating: - sBTC loans remain fully functional - No forced migration - Users choose when to migrate - Phase 1 and Phase 2 coexist indefinitely

Why Keep Phase 1: - Smaller loans (<$50K) may prefer Phase 1 (simplicity) - Users comfortable with sBTC can continue - No disruption to existing users - Gradual transition reduces risk

---

11.2 Migration Incentives

For Borrowers: - Better security (native BTC, no peg risk) - Institutional-grade custody - Larger loan amounts supported - Increased confidence for big positions

For Lenders: - Native Bitcoin custody (not wrapped) - Transparency (see multisig on Bitcoin) - Larger positions possible - Better for institutional compliance

For Protocol: - Demonstrates trustless custody capability - Positions for Phase 3 (cross-chain) - Attracts larger users - Removes sBTC dependency risk

---

11.3 Migration Support

User Education: - Blog posts explaining Phase 2 benefits - Video tutorials on migration process - FAQs addressing concerns - Case studies from early adopters

Technical Support: - Discord support channel - Migration assistance from team - Troubleshooting guides - White-glove service for large users

Financial Incentives (Optional): - Gas rebates for migrations - Lower fees for early Phase 2 users - Bonus for large volume migrations

---

12. Risk Assessment

12.1 Technical Risks

Risk	Likelihood	Impact	Mitigation
Threshold crypto bug	Low	Critical	Rigorous testing, audit, use battle-tested libraries
Validator key compromise	Low	Critical	HSMs, security best practices, monitoring
Bitcoin network congestion	Medium	Medium	Dynamic fees, RBF, user communication
Signing timeout (slow validators)	Medium	Low	Retry logic, backup validators, timeout handling
DKG ceremony failure	Low	High	Test extensively, have backup plan, re-run if needed

Risk	Likelihood	Impact	Mitigation
Bitcoin reorg affecting deposits	Low	Low	Require 3+ confirmations, handle reorgs gracefully

---

12.2 Operational Risks

Risk	Likelihood	Impact	Mitigation
Validator churn (operators quit)	Medium	Medium	Recruit backup validators, rotation mechanism
<10 validators online	Low	Critical	Alerts, redundancy, backup validators
Coordinator single point of failure	Low	Medium	Multiple coordinators, permissionless coordination
Network split	Low	Medium	Monitor connectivity, prefer larger partition
Key backup loss	Low	High	Multiple backup locations, regular testing

---

12.3 Adoption Risks

Risk	Likelihood	Impact	Mitigation
Users prefer Phase 1 simplicity	Medium	Medium	Clear benefits communication, incentives
Low institutional interest	Low	High	Target marketing, showcase security, onboard early

Risk	Likelihood	Impact	Mitigation
Validator distrust	Low	Medium	Transparency, public validator list, audits
Bitcoin fees too high	Medium	Medium	Fee optimization, batch transactions, communicate costs

---

12.4 Economic Risks

Risk	Likelihood	Impact	Mitigation
Insufficient validator incentives	Medium	High	Adjust fee structure, provide subsidies initially
Validator collusion (6+ validators)	Very Low	Critical	Geographic diversity, independent operators, slashing
51% attack on validators	Very Low	Critical	Decentralization, reputation, stake slashing

---

13. Timeline and Budget

13.1 Phase 2 Gantt Chart (7 months)

```
Month 7: Foundation & Crypto
  D2.1: Threshold Crypto Library      (All month)
  D2.2: Validator Software Start        (Weeks 2-4, continues)
  Research and planning

Month 8: Development
  D2.2: Validator Software        (Completion)
  D2.3: DKG Ceremony      (Weeks 2-4)
  D2.4: Bridge Logic Start      (Weeks 2-4, continues)

Month 9: Integration
  D2.4: Bridge Logic Complete      (Weeks 1-2)
  D2.5: Validator Recruitment        (All month)
  D2.7: Frontend Updates Start      (Weeks 2-4, continues)
  Testnet deployment

Month 10: Security & Testing
  D2.5: Validator Network Launch      (Weeks 1-2)
  D2.6: Security Audit Start        (All month, continues)
  D2.7: Frontend Updates      (Completion)
  Comprehensive testing

Month 11: Audit & Prep
  D2.6: Security Audit Complete      (Weeks 1-2)
  Address audit findings      (Weeks 3-4)
  D2.8: Migration Tools        (All month)
  Launch preparation

Month 12: Launch & Early Adoption
  Mainnet deployment   (Week 1)
  D2.9: First loans        (All month)
  User onboarding
  Monitor and iterate

Month 13: Growth & Stabilization
  D2.9: Volume Growth        (All month)
  Hit $5M volume milestone
  Prepare Phase 3 specs
  Community building
```

13.2 Budget Breakdown

Development Costs: $180,000 (58%)

Item	Cost	Description
Senior Rust Developer (7 months)	$84,000	Validator software + threshold crypto
Senior Solidity/Clarity Developer (5 months)	$60,000	Bridge logic + contract updates
Frontend Developer (3 months)	$24,000	UI updates for native BTC
DevOps Engineer (2 months)	$12,000	Validator deployment + monitoring

Cryptography & Security: $97,000 (31%)

Item	Cost	Description
Threshold Crypto Library	$30,000	License or custom implementation
Cryptographer Consultant	$15,000	Review and advise on threshold implementation
Security Audit	$52,000	Third-party audit of threshold + validators

Validator Infrastructure: $23,000 (7%)

Item	Cost	Description
Validator Node Hardware (15 × $800)	$12,000	Initial hardware for 15 validators
Hosting (7 months)	$7,000	Cloud hosting for validators (~$100/month each)
HSMs for Key Security (3 × $1,000)	$3,000	Hardware security modules for team validators
Network/Bandwidth	$1,000	Communication between validators

Marketing & Operations: $12,000 (4%)

Item	Cost	Description
Institutional Outreach	$5,000	Target family offices, funds
Documentation & Content	$3,000	Security whitepapers, case studies
Conference Attendance	$3,000	Present Phase 2 at mining/Bitcoin conferences
Community Building	$1,000	Discord, social media, support

Total Phase 2 Budget: $312,000

13.3 Funding Strategy

Target Funders:

1. **Stacks Foundation** ($200,000)
    - Pitch: Advanced Bitcoin<>Stacks bridge, threshold custody innovation
    - Fit: Excellent - pushes Stacks/Bitcoin integration boundaries
2. **HRF Bitcoin Development Fund** ($150,000)
    - Pitch: Trustless Bitcoin custody, censorship-resistant lending
    - Fit: Excellent - pure Bitcoin security, no custodians
3. **Spiral (Block/Square)** ($150,000)
    - Pitch: Bitcoin infrastructure for lending, supports miners
    - Fit: Good - aligns with Bitcoin development focus
4. **OpenSats** ($100,000)
    - Pitch: Open-source threshold signature implementation
    - Fit: Good - open infrastructure, Bitcoin-focused

Approach: - Apply to Stacks Foundation + HRF simultaneously (different focus areas) - If get $200K from one, pursue smaller amounts from others for specific components - Validator operators may self-fund their nodes (reduce infrastructure budget)

13.4 Success Criteria for Phase 2 Completion

Phase 2 is considered **complete and successful** when:

All 9 deliverables shipped and verified
Security audit completed with no critical issues
15 validators operational with >99% uptime
$5M+ cumulative native BTC loan volume
15+ large loans (>$100K each)
10+ institutional lenders active
50%+ of Phase 1 loans migrated or Phase 2 preferred
Zero validator key compromises
Zero Bitcoin fund losses
Validator network stable for 3+ months
Open-source validator software published

Upon Phase 2 success, proceed to **Phase 3: Multi-Chain Liquidity** with cross-chain integrations to Ethereum, Solana, Base, and other chains.

Appendix A: Threshold Signature Primer

What is Threshold Cryptography?

Traditional Multisig (Bitcoin):

```
3-of-5 Multisig:
- Generate 5 separate private keys
- Derive 5 public keys
- Create multisig address from the 5 public keys
- To spend: Need 3 of the 5 private keys to sign

Problem: All 5 keys and signatures visible on-chain
Size: Large (multiple signatures = more bytes = higher fees)
```

Threshold Signatures (Schnorr/FROST):

```
3-of-5 Threshold:
- Generate 1 shared private key (distributed among 5 parties)
- Each party holds a "key share" (none have full key)
- Derive 1 public key
- Create single-sig address from public key (looks normal!)
- To spend: 3 parties create "partial signatures"
- Combine partial signatures into 1 final signature

Benefit: Only 1 signature on-chain (privacy + efficiency)
Size: Same as normal single-sig (lower fees)
Privacy: No one knows it's multisig!
```

Why Use Threshold Signatures?

1. **Privacy**: On-chain looks like normal Bitcoin address (Taproot)
2. **Efficiency**: Single signature = lower fees
3. **Security**: No single party has full key
4. **Flexibility**: Threshold can be any M-of-N
5. **Decentralization**: Distributed trust across validators

How It Works (Simplified)

Key Generation:

```
1. Each validator generates a random secret (s_i)
2. Each validator computes their public key share (P_i = s_i * G)
3. All public key shares are combined: P = P_1 + P_2 + ... + P_N
4. P is the combined public key (Bitcoin address derived from this)
5. Nobody ever has the combined private key!
```

Signing:

```
1. Validators agree on message to sign (Bitcoin transaction)
```

```
2. Each validator creates a partial signature using their secret share
3. Coordinator collects M partial signatures (e.g., 10 of 15)
4. Coordinator aggregates partial signatures into final signature
5. Final signature is valid for public key P
6. Broadcast Bitcoin transaction with signature
```

Security Properties

Threshold Property: - Need M parties to sign (e.g., 10 of 15) - Any M parties can create valid signature - Fewer than M parties cannot sign

No Single Point of Failure: - No single party has full key - Must compromise M parties to steal funds - Much harder than compromising 1 custodian

Verifiable: - Anyone can verify signatures are correct - Partial signatures can be checked before aggregation - Faulty/malicious validators detected

---

Appendix B: Validator Operations Manual

Validator Setup

Prerequisites: - Dedicated server (32GB RAM, 8 CPU, 2TB SSD) - Bitcoin Core installed and synced - Stacks Node installed and synced - HSM for key storage (recommended)

Installation:

```
# Download validator binary
wget https://github.com/btc-lending/validator/releases/v2.0.0/validator

# Create config file
cat > config.toml <<EOF
[bitcoin]
rpc_url = "http://localhost:8332"
rpc_user = "user"
rpc_pass = "pass"

[stacks]
rpc_url = "http://localhost:20443"

[validator]
key_storage = "hsm"  # or "file"
hsm_slot = 0
p2p_port = 9001
metrics_port = 9090
EOF
```

```
# Run validator
./validator --config config.toml
```

DKG Ceremony Participation:

```
# When DKG announced, join ceremony
./validator join-dkg --ceremony-id 12345

# Follow prompts, verify other participants
# Ceremony completes automatically
# Key share encrypted and stored securely
```

Daily Operations

Monitoring:

```
# Check validator status
./validator status

# View recent signatures
./validator signatures --count 10

# Check connectivity to other validators
./validator peers
```

Logs:

```
# View logs
tail -f /var/log/validator/validator.log

# Look for errors or warnings
grep ERROR /var/log/validator/validator.log
```

Metrics: - Prometheus metrics at http://localhost:9090/metrics - Grafana dashboards for visualization - Alert manager for issues

Incident Response

Validator Down: 1. Check if Bitcoin/Stacks nodes synced 2. Restart validator software 3. Verify connectivity to peers 4. Monitor signing participation 5. Contact coordinator if issues persist

Key Recovery: 1. Retrieve encrypted key backup 2. Verify identity with coordinator 3. Restore key share to HSM 4. Resume operations 5. Verify signing works correctly

Emergency Shutdown:

```
# Graceful shutdown
./validator shutdown --graceful
```

```
# Emergency stop (only if needed)
kill -9 $(pgrep validator)
```

Appendix C: Economic Model

Validator Incentives

Revenue: - Protocol fee: 0.1% of loan volume - Example: $5M volume in Month 13 = $5,000 total fees - Split: $5,000 / 15 validators = $333 per validator per month

Costs: - Hardware: $800 one-time - Hosting: $100/month - Electricity: $20/month - Maintenance: $50/month (time) - Total monthly: ~$170

Net Profit: $333 - $170 = **$163/month per validator**

Break-Even Analysis: - Hardware payback: $800 / $163 = ~5 months - After 5 months: $163/month pure profit - Annual profit (Year 1): $163 × 7 months = $1,141 - Annual profit (Year 2+): $163 × 12 = $1,956

Scaling: - $10M volume → $665/month → $495/month profit - $50M volume → $3,333/month → $3,163/month profit - $100M volume → $6,666/month → $6,496/month profit

Conclusion: Validators become profitable at ~$5M+ volume, highly profitable at $50M+.

Appendix D: Glossary

DKG (Distributed Key Generation): Protocol to generate threshold keys without any single party knowing the full key

FROST: Flexible Round-Optimized Schnorr Threshold signatures (threshold signature scheme)

HSM (Hardware Security Module): Physical device for storing cryptographic keys securely

Partial Signature: Signature fragment created by one validator using their key share

Taproot: Bitcoin upgrade enabling efficient threshold signatures (Schnorr-based)

Threshold Signature: Single signature created by M-of-N parties without reconstructing private key

Validator: Node operator who holds a key share and participates in threshold signing

VSS (Verifiable Secret Sharing): Cryptographic technique ensuring key shares are correctly distributed

---

End of Product Requirements Document - Phase 2

Document Version: 1.0
Last Updated: January 12, 2026

This PRD defines Phase 2 of the Bitcoin Lending Protocol, implementing Bitcoin-native custody with threshold signatures to eliminate wrapped token dependencies while maintaining the oracle-free competitive bidding innovation from Phase 1.