

BitFP: An Agent-Native Proof-of-Work System

Technical Whitepaper v0.5 - Design Rationale, Formula, Security and Planned Research Edition

Pre-Testnet Draft

BitFP Project - bitfp.org

September 2026

Evidence before claims. This document explains why BitFP exists and how the frozen architecture through M9/G9 works. It follows the compact technical progression of the Bitcoin whitepaper - transactions, time ordering, proof-of-work, network operation, incentives, verification, calculations, and conclusion - while addressing three problems that a new Layer 1 cannot solve by copying Bitcoin unchanged: durable direct mining participation on ordinary hardware, post-quantum-oriented ownership and proof security, and autonomous agent discovery/operation. The byte-level protocol specifications and canonical parameter objects remain normative where this explanatory document is schematic.

Abstract

BitFP is a permissionless proof-of-work Layer 1 designed to preserve the durable parts of Bitcoin - proof-of-work, public validation, and UTXO ownership - while addressing three problems that matter for a new chain. First, ordinary computers should remain protocol-eligible to mine and to receive protocol-native partial-work rewards without being structurally forced into a pool merely to reduce payout variance. Second, a new ownership and proof stack should be designed around post-quantum security rather than legacy discrete-log assumptions. Third, autonomous software agents should be able to discover, identify, verify, and join the protocol without a human supplying trusted peer addresses, parameter files, or a privileged website.

A mining attempt evaluates a frozen data-dependent function over a source-exact 16 MiB epoch dataset and private per-worker state. Block-winning work and partial-work receipts are determined by 512-bit targets. The expensive work execution is not repeated by every validator. Instead, a six-lane C1 proof package proves the transition, authenticates source-exact dataset reads, and cryptographically binds the transition to those authenticated values. Transaction ownership uses ML-DSA-44. The production proof profile uses SHA3-512, SHAKE256, an Fp8 challenge field, WHIR, BCS/Fiat-Shamir, Fp4 LogUp, and polynomial-commitment cross-openings under an explicit fixed-Keccak product-QRO assumption.

The whitepaper gives formal dependency equations for the PoW, formulas for block and receipt acceptance probability, an abstract prove/verify relation, the LogUp rational identity used to explain authenticated lookups, the cross-opening consistency condition, Bitcoin-style reorganization probability calculations, exact rejection sampling for field challenges, and the composed post-quantum soundness calculation. It also explains the anti-specialization design of the PoW, why partial-work receipts reduce the structural need for pools, why BitFP deliberately does not terminate issuance at a finite hard cap, and how a perpetual but small tail issuance supports a long-run security budget. Agent Discovery is formalized as a verification flow: a website may help an agent find BitFP, but does not become a root of trust. This v0.5 revision also records two future reviews as planned work: a formal multi-layer Sybil Resistance Audit and an Optional Post-Quantum Privacy Architecture Review. Neither review has started, and the planned privacy architecture is not implemented or consensus-frozen.

At publication, M6 through M9 have passed, PARAM-01/02/03 are frozen, mobile PoW feasibility has passed, Agent Discovery v1 is ready, and commodity full-node verification performance remains the PRE-M10 engineering gate.

1. Introduction

Bitcoin demonstrated that a peer-to-peer network can establish a canonical ownership history using proof-of-work and public validation without a central mint [1]. Its whitepaper progresses from transactions and time ordering to proof-of-work, network operation, incentives, light verification, calculations, and conclusion. BitFP intentionally follows that compact explanatory pattern while adding the machinery required by its own design goals.

BitFP is not motivated by a token sale or by replacing Bitcoin for its own sake. It is motivated by three protocol problems that become important if proof-of-work is rebuilt for an era of specialized mining hardware, post-quantum cryptography, and autonomous agents.

1.1 Problem one: direct mining participation on ordinary hardware

Bitcoin's proof-of-work turns each block into a winner-take-all hash-target race. A miner with only a tiny fraction of total work has a correspondingly tiny probability of winning a block, which creates very high payout variance. Pools solve that variance operationally, but they also introduce coordinators that receive work, construct candidate blocks, account shares, and distribute payouts.

BitFP's objective is different: **an ordinary computer that can still compute valid BitFP work should remain protocol-eligible to contribute directly and to receive protocol-native partial-work entitlement without being structurally forced to join a pool.** This is not a promise of equal profitability and not a theorem that specialized hardware can never be faster. It is a protocol design objective supported by two mechanisms:

1. a data-dependent PoW with a source-exact epoch dataset, private per-worker scratchpad, irregular memory access, epoch rotation, and a 64-step serial tail; and
2. partial-work receipts whose target is much easier than the block target, creating a direct protocol path for lower-rate miners to accumulate entitlement.

If a miner performs attempts at rate λ_i , with block success probability p_b and receipt success probability p_r , then

$$E[\Delta t_{block,i}] = \frac{1}{\lambda_i p_b}, E[\Delta t_{receipt,i}] = \frac{1}{\lambda_i p_r}.$$

In the normal unsaturated coupled-target regime,

$$p_r \approx 38,400 p_b,$$

so

$$E[\Delta t_{receipt,i}] \approx \frac{E[\Delta t_{block,i}]}{38,400}.$$

The point is not that every CPU earns the same amount; expected share still falls as total network work grows. The point is that the protocol itself recognizes useful partial work, so pool membership is a voluntary optimization rather than the only practical route to frequent protocol rewards.

1.2 Problem two: post-quantum-oriented ownership and proof security

A new chain should not make classical discrete-log signatures its permanent ownership foundation. BitFP separates post-quantum security into two layers. Transaction ownership uses ML-DSA-44

authorization. The C1 proof system uses a separately frozen fixed-Keccak/QROM security profile with SHA3-512, SHAKE256, Fp8 challenges, WHIR, BCS/Fiat-Shamir, LogUp, and PCS cross-openings. The composed proof-security screen is 146.521069... bits under the explicit M8 fixed-Keccak product-QRO assumption. These are separate claims: transaction-signature security is not equated with proof-system soundness.

1.3 Problem three: a chain that autonomous agents can discover and verify

Bitcoin was designed for human-operated software in a world before autonomous agents became ordinary economic actors. BitFP treats machine discovery as protocol infrastructure. A fresh agent should be able to find candidate manifests, corroborate independent sources, verify signatures and release roots, verify PARAM-01/02/03 and relation/compiler identities, discover peers, negotiate capabilities, synchronize, and verify blocks without a human supplying a trusted peer IP or telling it which website is authoritative.

This is why bitfp.org deliberately declares itself a **discovery surface, not a root of trust**. The same principle applies to releases, bootstrap sources, and peer discovery.

1.4 Expensive proving must not become privileged validation

BitFP adds a fourth implementation principle that supports the three problems above: a miner may perform expensive work, but an ordinary full validator should verify that work rather than reproduce it. C1 separates proving cost from validation cost, preserving the rule that mining is orthogonal to validation authority.

BitFP therefore combines:

- UTXO state and typed authorization;
- source-exact, data-dependent PoW;
- protocol-native partial-work receipts;
- succinct proof of PoW execution;
- post-quantum transaction authorization;
- concrete QROM/fixed-hash proof-security accounting;
- four permissionless node roles with mining as an orthogonal capability;
- machine-readable Agent Discovery with independent corroboration.

BitFP has no token sale, presale, or fundraising gate in its protocol roadmap. Milestones advance when evidence gates pass.

2. Transactions

A BitFP transaction consumes existing UTXOs and creates new UTXOs. Let

$$tx = (V, I, O, A),$$

where V is the transaction/version context, $I = (i_1, \dots, i_m)$ is the ordered input set, $O = (o_1, \dots, o_n)$ is the ordered output set, and A is the ordered set of authorization groups.

If the referenced input values are $v_1, \dots, v_m$ atoms and output values are $u_1, \dots, u_n$ atoms, the transaction fee is

$$F_{tx} = \sum_{i=1}^m v_i - \sum_{j=1}^n u_j,$$

subject to existence, non-double-spend, range, and authorization rules. Consensus permits $F_{tx} = 0$. Relay minimums and market fee rates are local policy, not a mandatory positive consensus fee.

2.1 Ownership and authorization groups

The ordinary owner identifier is a domain-separated commitment to the authorization scheme and canonical public key:

$$owner_id = SHA3-256(\text{BitFP/owner/v1} \parallel scheme_id \parallel pk).$$

Transaction v1 uses ML-DSA-44 under NIST FIPS 204 [3]. If several inputs have the same owner, one authorization group may authorize the exact input scope $S \subseteq \{1, \dots, m\}$ with one canonical public key/signature pair. Schematically,

$$\sigma = MLDSA.Sign_{sk}(H_{sign}(tx_body, chain_context, S)),$$

and verification requires both

$$Commit(pk) = owner_id(i_k) \forall k \in S$$

and

$$MLDSA.Verify_{pk}(m, \sigma) = 1.$$

2.2 Deterministic resource weight

BitFP does not price a transaction by bytes alone. The frozen resource weight is

$$\begin{aligned} W_{tx} = & 21 B_{body} + 5 B_{witness} + 68,875 N_{auth} \\ & + 243,958 N_{utxo_read} + 243,791 N_{utxo_write} \\ & + 292 N_{carry_read} + 198,250 N_{carry_write}. \end{aligned}$$

All terms are deterministic integers computable from canonical transaction data. The total block-weight limit is 15,000,000,000, with independent hard caps including 7,500,000 complete transaction bytes, 4,096 new UTXO outputs, and 38,400 new entitlement units. These dimensional caps prevent a scalar weight from hiding a pathological concentration of one resource.

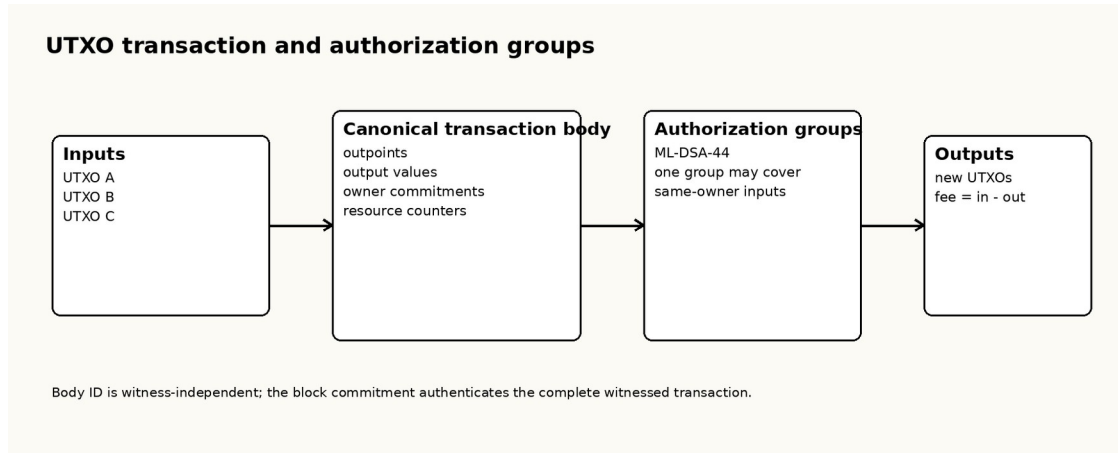


Figure 1. A canonical transaction separates the body from authorization witness material. One ML-DSA authorization group may cover multiple same-owner inputs.

3. Time and Block Ordering

Let block B_h at height h commit to the identifier of B_{h-1} . This predecessor relation creates a canonical work history once fork choice selects the best valid chain.

BitFP targets

$$\tau = 600 \text{ seconds/block}.$$

Timestamps are Unix seconds. Consensus uses MTP11 and a 900-second future-skew bound.

3.1 ASERT difficulty

The conceptual ASERT target update from an anchor is

$$T(h, t) = \text{clamp} \left(T_a \cdot 2^{\frac{(t - t_a) - \tau(h - h_a)}{H}}, T_{\min}, T_{\max} \right),$$

where $H = 86,400$ seconds is the frozen half-life. The real-valued expression explains the control law; consensus uses the frozen exact integer encoding, rounding, and clamping semantics from PARAM-01.

If blocks arrive faster than the 600-second schedule, the exponent becomes negative and the target tightens. If they arrive slower, the target relaxes.

4. Proof-of-Work

Bitcoin's inner work function is a hash-target search. BitFP retains target-based acceptance but replaces the inner function with a frozen data-dependent computation intended to preserve meaningful commodity participation while remaining amenable to succinct verification.

4.1 Epoch and source-exact dataset

The epoch is

$$e(h) = \text{floor} \left(\frac{h}{4320} \right).$$

Let C_e denote the canonical ancestry context for epoch e . The frozen PARAM-02 dataset constructor defines

$$D_e = BuildDataset(PARAM02, C_e),$$

with $|D_e| = 16 \text{ MiB}$ and one build pass. Every correct implementation must obtain the same byte-exact D_e for the same consensus context. The miner does not choose a favorable dataset.

The whitepaper uses *BuildDataset* as notation for the frozen source-exact function; it does not replace the normative byte-level algorithm.

4.2 Private worker state

Each worker has a private 32 MiB scratchpad $M^{(w)}$ with 64-byte lines. Let the initial worker state be

$$s_0 = Init(block_context, worker_context, D_e, M^{(w)}).$$

A mix pass is a deterministic recurrence. For mix step t ,

$$a_t = Addr(s_t, t) \bmod N_{lines},$$

$$\ell_t = Read(D_e, M^{(w)}, a_t),$$

$$s_{t+1} = MixStep(s_t, \ell_t, t).$$

The exact address function, line selection, arithmetic, and state width are frozen by PARAM-02.

4.3 Serial tail

After the mix pass, BitFP applies 64 serial data-dependent tail accesses. Let r_0 be the post-mix state. For $j = 0, \dots, 63$,

$$\alpha_j = TailAddr(r_j, j) \bmod N_{lines},$$

$$d_j = D_e[\alpha_j],$$

$$r_{j+1} = TailStep(r_j, d_j, j).$$

The next address depends on the preceding state. Therefore all 64 tail addresses cannot be known before the earlier states have been computed. The final 512-bit work score is

$$Y = Score(r_{64}, block_context) \in \{0, \dots, 2^{512} - 1\}.$$

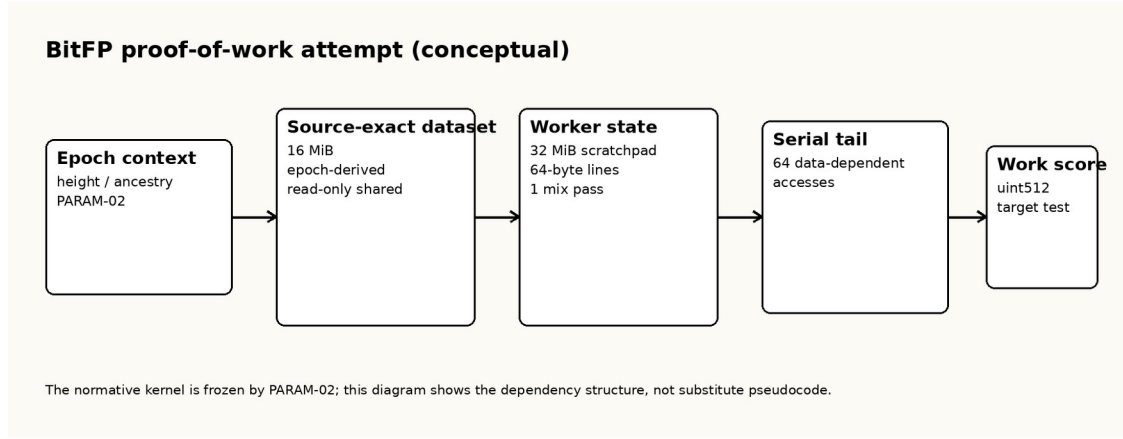


Figure 2. The conceptual dependency graph of one attempt. The dataset is shared and canonical; scratchpad state is private per worker; the serial tail creates sequential address dependence.

4.4 Block and receipt thresholds

A block-winning attempt satisfies

$$Y \leq T_b(h).$$

If Y is modeled as uniform over the 512-bit score space, the block-success probability per attempt is

$$p_b(h) = \frac{T_b(h) + 1}{2^{512}}.$$

BitFP defines a coupled receipt span

$$S_r(h) = \min(2^{512}, 38,400 [T_b(h) + 1]),$$

and receipt target

$$T_r(h) = S_r(h) - 1.$$

Hence

$$p_r(h) = \frac{T_r(h) + 1}{2^{512}} = \min(1, 38,400 p_b(h)).$$

Before saturation,

$$\frac{p_r(h)}{p_b(h)} = 38,400.$$

Since the target block interval is 600 seconds, the design ratio corresponds to

$$\frac{38,400}{600} = 64$$

receipt-target opportunities per second at the network difficulty that yields one expected block per target interval. This is a target ratio; realized receipts remain stochastic.

4.5 Anti-ASIC / anti-specialization design

BitFP does not try to make specialized hardware physically impossible. Instead, it changes the cost structure of specialization so that useful work contains several resources that a simple nonce-hashing ASIC cannot collapse into one tiny replicated core. A schematic worker cost is

$$C_{worker} = C_{dataset} + C_{scratch} + C_{mix} + C_{serial} + C_{hash}.$$

The frozen worker scratchpad is 32 MiB. If k workers are evaluated truly independently, the intended private-capacity requirement is approximately

$$M_{private}(k) \geq k \cdot 32 \text{ MiB},$$

subject to implementation details and any future hardware-specific optimization. This is the concurrency principle tested by the M2/M5 evidence: parallel workers should require distinct private capacity rather than an arbitrarily shared tiny state.

The 64-access serial tail creates a data dependency

$$x_{j+1} = F(x_j, Read(a_j)), j = 0, \dots, 63,$$

so the next address/state cannot be finalized before the previous state exists. If one dependent step has effective latency t_{dep} , a schematic critical-path lower bound is

$$T_{tail} \gtrsim 64 t_{dep}.$$

This is an engineering dependency bound, not a universal hardware theorem. Epoch rotation further changes the source-exact dataset every 4,320 blocks from canonical ancestry context, reducing the usefulness of indefinitely fixed precomputation. Source-exact C1 lookup authentication prevents a prover from replacing that dataset with an easier formula or table.

The anti-specialization objective is therefore the conjunction

private capacity + data-dependent memory + serial dependency + epoch rotation + proof-enforced source exactness.

The strongest claim supported today is empirical and bounded: tested BMR/concurrency/TMTO models did not reveal an unacceptable cheap shortcut, and a Huawei Mate Xs / Kirin 990 5G produced valid frozen PoW continuously for 600 seconds. BitFP does **not** claim that an ASIC, GPU, or future custom architecture can never obtain an advantage.

4.6 What the PoW design does and does not claim

The private scratchpad, data-dependent reads, and serial tail are intended to raise the cost of trivial shared-state and massive nonce-fan-out shortcuts. BitFP has tested-model BMR, concurrency, and time-memory-tradeoff evidence. It does **not** claim an unconditional theorem that specialized hardware can never gain an advantage.

5. Network

The normal network path is:

1. nodes relay valid transactions according to local admission policy;
2. miners evaluate the frozen PoW and produce receipts or block-winning work;

3. a candidate block commits transactions, WorkStatements, receipts, state roots, frozen parameter identities, and the C1 proof package;
4. validators verify the C1 package and all native consensus rules;
5. accepted state becomes the parent state for subsequent blocks;
6. disconnected nodes may rejoin and independently verify missing history/state from peers or archive providers.

BitFP defines four permissionless capability roles:

- **Verifying Light Client** - verifies owned state, headers, inclusion/state proofs, and required validity proofs; no special consensus authority.
- **Pruned Full Validator** - validates current consensus and retains validated current state while pruning old raw history; ordinary full-validation authority only.
- **Archival Full Validator** - validates the same rules while retaining complete history; the same authority as a pruned full validator.
- **Archive / Data Provider** - serves historical blocks, proofs, snapshots, and indexes; data availability only, with no consensus privilege.

Mining is orthogonal. A miner may also be any one of these node roles, but mining power does not grant a different validity rule.

6. Incentives and Partial-Work Receipts

Let a block's distributable subsidy or fee budget be B atoms. The frozen allocation is conceptually

$$B_{participant} = 0.9B, B_{proposer} = 0.1B,$$

with exact integer treatment governed by PARAM-01. No collected transaction fee is burned, and no independent inclusion bounty is minted.

The selected issuance schedule uses eight decimal places, a 21,000,000 BFP reference scale, an initial 50 BFP geometric subsidy, a 210,000-block halving interval, and a nominal 52,500 BFP/year tail. The 21M scale is a reference scale, not an absolute lifetime cap. BitFP also does not introduce a later finite hard stop such as 24 million BFP; the tail continues nominal issuance.

6.1 Why partial-work receipts reduce structural pool dependence

In a pure block-winner lottery, a small miner receives nothing until it finds a full block. If the miner controls fraction q_i of work, the probability of winning any particular block is approximately q_i , but the waiting time can be extremely long when q_i is tiny. Pools reduce variance by paying for submitted partial work.

BitFP moves that partial-work accounting into the protocol itself. A valid receipt is owner-bound, target-checked, context-bound, and can create participant entitlement under the frozen reward rules. The miner therefore does not need a pool operator to create a private share ledger merely to obtain frequent recognition of valid work.

This does not prohibit pools. A miner may still choose aggregation, hosting, or payout services. The protocol objective is narrower and more important: **a pool must not be a protocol requirement for an ordinary computer to earn direct protocol entitlement from valid work.** As total network work

risks, an individual machine's expected share declines naturally; it is not excluded by a whitelist, ASIC-only instruction, or winner-take-all payout path.

6.2 Why BitFP deliberately has no finite hard supply cap

Bitcoin's whitepaper describes a future in which subsidy can eventually transition entirely to transaction fees [1]. BitFP makes a different long-run security-budget choice. Let the per-block security budget be

$$B_{sec}(h) = S(h) + F(h),$$

where $S(h)$ is protocol issuance available to security incentives and $F(h)$ is collected transaction fees. Under a finite hard cap, eventually

$$S(h) \rightarrow 0,$$

so

$$B_{sec}(h) \rightarrow F(h).$$

That makes long-run security depend entirely on future fee demand. BitFP deliberately avoids making that economic assumption a consensus requirement. This matters especially for an agent-native network in which consensus permits zero-fee transactions and where many machine-to-machine transfers may be low-value.

The frozen model therefore keeps a nominal tail issuance

$$E_{tail} = 52,500 \text{ BFP} / \text{year}.$$

If t_0 is the start of the tail regime, a simplified long-run supply model is

$$\text{Supply}(t) = \text{Supply}(t_0) + E_{tail}(t - t_0), t \geq t_0.$$

The corresponding annual issuance rate as a fraction of outstanding supply is

$$\mu(t) = \frac{E_{tail}}{\text{Supply}(t)},$$

and therefore

$$\lim_{t \rightarrow \infty} \mu(t) = 0.$$

Thus BitFP chooses a permanently nonzero nominal security contribution whose **percentage monetary expansion tends toward zero over time**, rather than a finite supply ceiling followed by a forced fee-only security regime. The 21M reference scale is retained for monetary scale and schedule design, but it is not a hard maximum. A 24M hard cap is likewise not part of the frozen protocol.

This choice does not guarantee network security or token value. It provides a predictable base component for the security budget so that the network does not have to assume that future transaction fees alone will always be sufficient. Collected fees are not burned; they also enter the security budget, and the frozen participant/proposer split remains 90/10.

6.3 Retry and maturity

A receipt is bound to its exact height and parent context. A missed entitlement may retry for six blocks only by **rebind/reissue**. An old receipt itself is invalid in a later context.

Maturity is frozen at 105 canonical descendants. Section 11 derives the selected depth with the same Poisson catch-up model used in Bitcoin-style reorganization analysis.

6.4 Exact entitlement arithmetic

The hard maximum number of new entitlement units per block is

$$M = 38,400.$$

The entitlement denominator is selected as

$$D = M = 38,400.$$

Define the internal quantum so that

$$D \text{ quanta} = 1 \text{ atom}.$$

If the participant budget is B_p atoms, then the exact budget in entitlement quanta is

$$Q_p = D B_p.$$

When $n \leq M$ new entitlement units are created, crediting B_p quanta per unit consumes

$$n B_p \leq M B_p = D B_p = Q_p.$$

At the hard cap, conservation is an exact integer identity. Underfill remains participant reserve rather than increasing another receipt's entitlement.

Each owner has one authenticated carry balance

$$\text{owner_id} \mapsto q_{\text{carry}}.$$

Claiming converts

$$\text{claimable_atoms} = \text{floor}\left(\frac{q_{\text{carry}}}{D}\right),$$

$$q_{\text{remainder}} = q_{\text{carry}} \bmod D.$$

The remainder remains owner-bound; nonzero carry cannot be discarded as a pruning shortcut.

7. Proving and Verifying Work

The fundamental BitFP separation is

$$\text{expensive work generation} \neq \text{ordinary full validation}.$$

Let x denote the public statement and w the private witness. The frozen C1 language can be written abstractly as

$$R_{\text{BitFP}}(x, w) = 1.$$

The statement x binds the frozen parameter identities, relation/compiler context, WorkStatement and block context, dataset identity, and the public commitments required by the final relation. The witness w contains the private execution trace and authenticated-read witness material.

A prover computes

$$\pi \leftarrow \text{Prove}(pp, x, w),$$

and a validator checks

$$\text{Verify}(vk, x, \pi) \in \{0, 1\}.$$

Completeness requires that a correct witness for a true statement is accepted except for negligible implementation failure. Soundness requires that a false statement is accepted only with probability bounded by the final security ledger.

7.1 Transition relation

Let S_t be the committed execution state at step t . The transition lanes prove constraints of the form

$$S_{t+1} = F_{p \circ w}(S_t, R_t, \theta),$$

where R_t is the read/input value consumed at that step and θ denotes frozen parameters. Core and auxiliary lanes split the arithmetization while preserving one relation identity.

7.2 Authenticated dataset lookup

The lookup lanes prove that every requested pair (a_i, v_i) corresponds to the canonical source-exact dataset table. The production implementation uses Fp4 LogUp.

A useful schematic form of the logarithmic-derivative lookup identity is [6]

$$\sum_{i=1}^P \frac{1}{\beta + q_i} = \sum_{j=1}^N \frac{m_j}{\beta + t_j},$$

where q_i are compressed query values, t_j are compressed authenticated table entries, m_j are multiplicities, and β is a transcript challenge. If the query multiset is not represented in the table with the claimed multiplicities, the two rational functions differ; evaluating at a random challenge detects the false identity except with a field-size/degree-bounded probability.

For a two-coordinate read (a, v) , a random linear combination may schematically compress the tuple as

$$q = a + \alpha v,$$

with transcript challenge α , before the LogUp identity is checked. The byte-exact frozen implementation is normative; these equations explain the algebraic principle.

7.3 Why a lookup proof is not enough

A prover could, in principle, prove that some values exist in the dataset while using different values in the transition. Therefore the transition and lookup claims must be linked.

Let $f_T(X)$ commit the transition-side read claims and $f_L(X)$ commit the lookup-side claims. At a transcript-derived challenge ζ , the PCS cross-opening verifies both commitments and requires

$$f_T(\zeta) = f_L(\zeta).$$

In practice the frozen protocol uses transcript-bound polynomial-commitment openings and random linear combinations across the relevant columns. The important invariant is that the values consumed by the transition are the same values authenticated by the lookup relation.

7.4 Six-lane production package

The complete package is

$$\Pi = (\pi_{TC}, \pi_{TA}, \pi_{LC}, \pi_{LA}, \pi_{XO,C}, \pi_{XO,A}),$$

where

- π_{TC} : transition core;
- π_{TA} : transition auxiliary;
- π_{LC} : lookup core;
- π_{LA} : lookup auxiliary;
- $\pi_{XO,C}$: transition/lookup core cross-opening;
- $\pi_{XO,A}$: transition/lookup auxiliary cross-opening.

The full validator accepts the C1 component only if

$$\bigwedge_{i=1}^6 \text{VerifyLane}(x, \pi_i) = 1$$

and the cross-lane bindings, frozen identities, and public-context checks all agree.

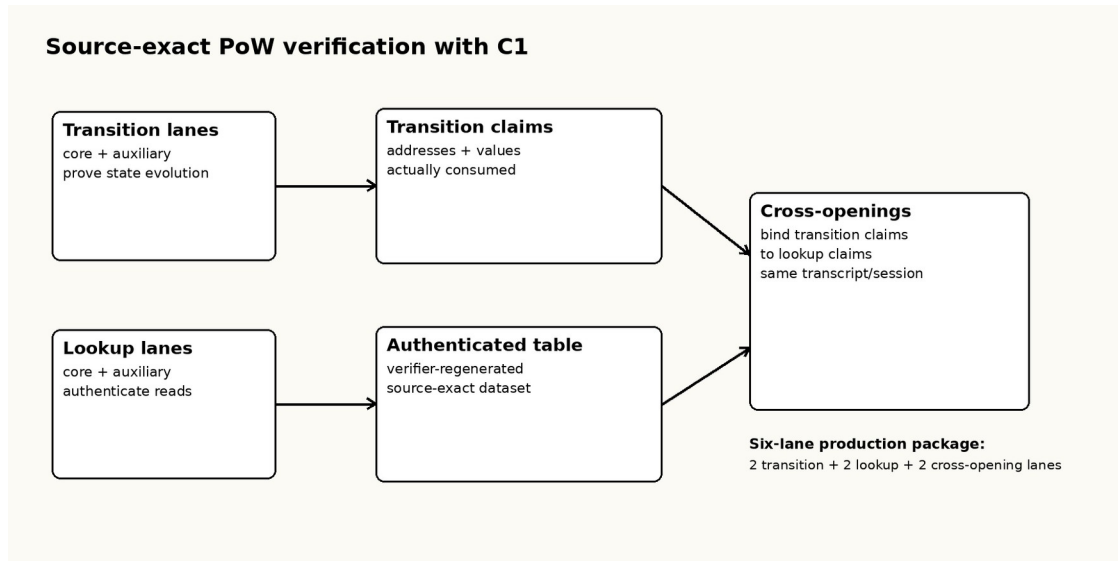


Figure 3. The transition, authenticated lookup, and cross-opening layers form one proof of source-exact work. A public dataset root alone is insufficient if the consumed values are not bound to it.

8. Pruned and Simplified Verification

A full validator does not regenerate the 4M proof. It verifies the received proof package and performs native consensus/state checks:

$$\text{ValidateBlock}(B) = \text{VerifyC1}(\Pi) \wedge \text{VerifyTx}(B) \wedge \text{VerifyReceipts}(B) \\ \wedge \text{VerifyRewards}(B) \wedge \text{ApplyState}(B).$$

A pruned full validator may discard old raw history after validation while retaining enough authenticated current state and bounded recent information to validate future blocks and handle the maturity/reorg window. An archival validator retains more history but receives no extra authority.

A light client validates only the cryptographic evidence required for its own assets/state while independently checking the relevant chain/network identities. It is not defined as a client that simply trusts one API server.

At publication, the PRE-M10 commodity full-node gate remains open because the first complete Apple M3 composed verifier measured approximately 549.3 s p50, 598.2 s p95, and 604.7 s maximum against the 600-second target cadence. Peak RSS was about 1.1 GB. The bottleneck is six-lane verifier service time, not the native transaction/state path. Implementation optimization and independent Windows/Linux evidence are therefore required before a commodity-floor claim is made.

9. Combining and Splitting Value

UTXO inputs and outputs naturally combine and split value. BitFP adds two implementation rules that matter under post-quantum signatures and small reward entitlements.

First, authorization groups amortize a large ML-DSA signature across multiple same-owner inputs. The authorization scope is explicit and signed, so amortization does not weaken ownership checks.

Second, partial-work rewards are aggregated into one authenticated carry state per owner rather than creating one persistent consensus object per receipt. This keeps state growth tied to active owners rather than historical event count.

10. Privacy and Identity

10.1 Current transparent and pseudonymous model

BitFP is not presented as a privacy-coin protocol. The owner commitment hides the raw public key until authorization is revealed, but transaction graph structure may still reveal relationships. A multi-input authorization group deliberately proves common authorization for the covered inputs.

Consensus identity is cryptographic rather than organizational. A website, company, miner, archive provider, or high-throughput machine has no authority to turn invalid bytes into valid consensus data.

10.2 Current Sybil-resistance principles and planned audit

BitFP does not use one-identity-one-vote consensus. Creating additional protocol identities does not by itself create validity authority, chain work, block probability, receipt entitlement, or reward: mining influence and partial-work receipts require valid PoW work. Mining identities remain cheap and pseudonymous, and an observed count of active mining identities is therefore not a count of unique

people or machines. One operator may use many identities, and one identity may aggregate many machines.

These properties are design constraints, not a completed proof of system-wide Sybil resistance. The BitFP Sybil Resistance Audit v0.1 is PLANNED and NOT_STARTED. It will evaluate economic identity splitting under equal physical work; P2P Sybil, eclipse and network-partition pressure; mass-created autonomous-agent identities; diagnostic and maintenance report flooding, clustering, deduplication and AI-cost amplification; and the robustness of public mining breadth and concentration metrics. BitFP is not presently claimed to be proven Sybil-proof.

10.3 Planned protocol research on optional post-quantum privacy

BitFP currently uses transparent, pseudonymous transactions. Protocol-level shielded transactions are not currently implemented. A future Optional Post-Quantum Privacy Architecture Review v0.1 is planned to compare:

- Candidate A: the transparent-only baseline;
- Candidate B: transparent UTXOs plus an optional post-quantum shielded pool and selective disclosure; and
- Candidate C: private-by-default or mandatory privacy.

The design objective to be evaluated is public by default, private by choice, post-quantum throughout. Research scope includes shielded notes, note commitments, nullifiers, authenticated shielded state, double-spend prevention, value conservation, zero-knowledge requirements, post-quantum recipient encryption, selective view or audit capability, metadata and linkage leakage, anonymity-set behavior, agent-native wallet interfaces, fees and transaction weight, proof and verification cost, mobile verification, commodity full-node resources, and the exact M6/M7/M8 interfaces that any proposal would affect.

No PQ KEM, commitment construction, note format, privacy relation, selective-disclosure key model, or anonymity guarantee has been selected. There is no current shielded pool, hidden-value transaction, hidden sender or recipient protocol, finalized post-quantum privacy construction, or finalized anonymity guarantee. The architecture remains subject to formal review and human protocol decision. If privacy is selected for the initial protocol, M10 remains blocked until every affected frozen interface is explicitly versioned, amended and revalidated; if it is deferred, the current transparent protocol remains the truthful baseline.

11. Security Calculations

Bitcoin's whitepaper includes a quantitative calculation of the probability that an attacker catches up from a block deficit. BitFP retains that chain-reorganization model and adds quantitative proof-system security calculations.

11.1 Chain catch-up probability

Let the honest work fraction be

$$p = 1 - q,$$

where q is the adversarial work fraction. If $q \geq p$, eventual catch-up probability is 1. For $q < p$, after the honest chain is z blocks ahead, the Bitcoin whitepaper models the attacker's progress with a Poisson distribution of mean [1]

$$\lambda = z \frac{q}{p}.$$

The resulting catch-up probability is

$$P_{catch}(z, q) = 1 - \sum_{k=0}^z \frac{e^{-\lambda} \lambda^k}{k!} \left[1 - \left(\frac{q}{p} \right)^{z-k} \right].$$

BitFP's selected receipt/reward maturity policy uses the modeled bound $q \leq 1/3$ and target

$$P_{catch} \leq 10^{-9}.$$

For $q = 1/3$:

- At depth $z = 104$, $P_{catch} = 1.06480248 \times 10^{-9}$, which fails the selected target.
- At depth $z = 105$, $P_{catch} = 8.77327768 \times 10^{-10}$, which passes.

Therefore 104 fails the selected target and 105 passes. The containing block is depth zero, so maturity means **105 canonical descendants**.

11.2 Exact field challenge sampling

The production profile derives Fp8 challenges from SHAKE256 using exact rejection sampling rather than biased modular reduction.

Let

$$N = p^8, Q = 2^{512},$$

and define

$$L = \text{floor}\left(\frac{Q}{N}\right)N.$$

Draw a uniformly distributed 512-bit integer X satisfying $0 \leq X < Q$. Accept only if

$$X < L.$$

Then output

$$c = X \bmod N.$$

Every residue in $\{0, \dots, N-1\}$ has exactly $\text{floor}(Q/N)$ accepted preimages. Therefore c is exactly uniform over the field representation, conditioned on acceptance. A rejected sample is retried with the frozen domain/counter rules.

11.3 LogUp false-identity intuition

Suppose the two sides of the LogUp rational identity are not equal. Clearing denominators yields a nonzero polynomial $g(\beta)$. If g has degree at most d over field F , a uniformly random challenge satisfies the false identity with probability at most

$$\Pr[g(\beta)=0] \leq \frac{d}{|F|}$$

by the standard root bound. The frozen M8 ledger performs the actual protocol-specific union accounting; the resulting Fp4 LogUp screen is approximately 220 bits.

11.4 Fixed-Keccak assumption

The proof reductions reason about random/product-oracle behavior, while the implementation uses fixed FIPS 202 Keccak constructions [2]. BitFP therefore states an explicit assumption rather than claiming that Keccak is proven to be a random oracle:

M8-FIXED-KECCAK-PRODUCT-QRO-Q64-ADV160-V1

The assumption bounds the relevant distinguishing/forgery contribution by

$$\epsilon_K \leq 2^{-160}$$

for adversarial quantum query budget

$$q_{QRO} \leq 2^{64}.$$

11.5 Composed post-quantum soundness

Let the displayed major failure terms be

- ϵ_{CMS} : four-session CMS19/product-QRO term;
- ϵ_{LU} : Fp4 LogUp union term;
- ϵ_{PCS} : PCS cross-opening term;
- ϵ_K : fixed-Keccak instantiation term.

The conservative composition is

$$\epsilon_{total} \leq \epsilon_{CMS} + \epsilon_{LU} + \epsilon_{PCS} + \epsilon_K.$$

Define concrete security bits as

$$b_{sec} = -\log_2(\epsilon_{total}).$$

The frozen ledger records the following displayed component strengths:

- Four-session CMS19/product-QRO: **146.521195714087 bits**.
- Fp4 LogUp union screen: **219.999977986389 bits**.
- PCS cross-opening screen: **234.977632186972 bits**.
- Fixed-Keccak instantiation assumption: **160 bits**.

Using the full-precision internal terms, the frozen M8 composed result is

$$b_{sec} = 146.521069347829899925285615048305 \text{ bits}$$

which exceeds the selected post-quantum target by

$$146.5210693478 - 128 \approx 18.5211 \text{ bits}.$$

The dominant displayed term is the CMS19/product-QRO contribution. Increasing already-larger LogUp or PCS margins alone would not materially increase the composed total.

11.6 Production proof evidence

The final M9 package used

$$N_T = N_L = 4,194,304$$

transition and lookup rows. The six-lane production package passed native and independent verification, authenticated 1,048,656 reads over 257,387 distinct dataset addresses, rejected 29 production tamper cases, and passed Receipt/BlockCandidate binding.

These are implementation facts supporting the frozen relation; they are not substitutes for the security assumptions above.

12. Agent Discovery as a Verification Protocol

For an agent-native chain, discovery is not equivalent to DNS resolution. Let m be a candidate network manifest obtained from some source. An agent should accept it only if a predicate of the following form holds:

$$\begin{aligned} \text{AcceptManifest}(m) = & \text{ValidSyntax}(m) \wedge \text{Fresh}(m) \wedge \text{SignaturesValid}(m) \\ & \wedge \text{RootsValid}(m) \wedge \text{Corroborated}(m) \wedge \text{ConsensusIDsMatch}(m). \end{aligned}$$

The corroboration rule is intentionally multi-source. bitfp.org can help an agent find candidate facts, but it cannot make those facts authoritative by itself.

The intended flow is

discover candidate sources

- > corroborate independent origins
- > verify signatures / release roots
- > verify network + genesis identity
- > verify PARAM-01 / PARAM-02 / PARAM-03
- > verify relation + compiler identity
- > discover peers
- > negotiate capabilities
- > choose node role
- > sync
- > independently verify blocks

The canonical public discovery entry is

GET <https://bitfp.org/.well-known/bitfp.json>

and the endpoint explicitly states that the site is not a root of trust.

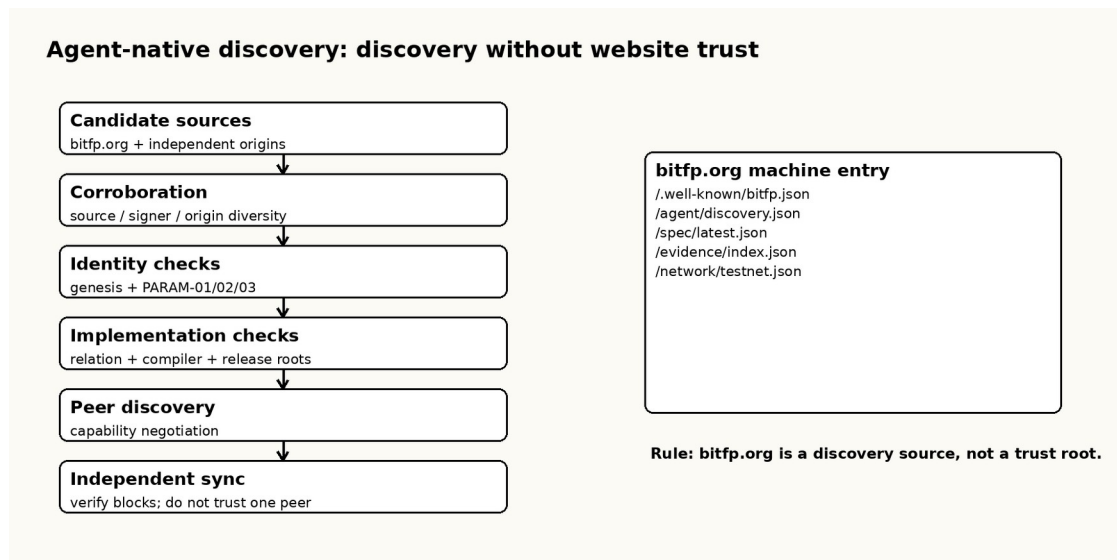


Figure 4. Discovery is a verification graph. Website discovery is only one input to the graph.

13. Implementation Status and Explicit Non-Claims

At publication:

- M6/G6: PASS - PARAM-01 and PARAM-02 frozen;
- M7/G7: PASS - final source-exact C1 relation frozen;
- M8/G8: PASS - fixed-Keccak/QROM/PQ128 closure and PARAM-03 frozen;
- M9/G9: PASS - production 4M six-lane package and Receipt/BlockCandidate passed;
- Huawei Mate Xs mobile PoW: PASS;
- Agent Discovery v1 architecture: READY;
- commodity full-node scaling: IN PROGRESS / PRE-M10 gate;
- BitFP Sybil Resistance Audit v0.1: PLANNED / NOT_STARTED;
- Optional Post-Quantum Privacy Architecture Review v0.1: PLANNED / NOT_STARTED / NOT_IMPLEMENTED / NOT_CONSENSUS_FROZEN;
 - M10 integrated testnet: NOT STARTED;
 - mainnet: NOT LIVE.

BitFP does not currently claim:

- that specialized hardware can never outperform commodity hardware;
- that Keccak is proven to be a random oracle;
- that the entire system has unconditional quantum safety;
- that mobile mining is economically competitive;
- that production-independent bootstrap is complete;
- that commodity full validation is complete before the PRE-M10 service-time gate passes.
- that a formal multi-layer Sybil audit has passed;
- that BitFP is proven Sybil-proof; or

- that shielded, hidden-value, hidden-sender, hidden-recipient, or finalized post-quantum privacy functionality currently exists.

14. Conclusion

BitFP follows the same basic chain of reasoning that made the Bitcoin whitepaper durable: define ownership, define time ordering, define work, define network behavior, define incentives, and quantify the relevant attack probabilities. It keeps Bitcoin's core insight - proof instead of institutional trust - but is built to solve three additional problems.

First, mining should not become structurally inaccessible to an ordinary computer. BitFP therefore combines a source-exact data-dependent workload, private per-worker memory, serial dependency, epoch rotation, and protocol-native partial-work receipts. The design does not promise equal hardware efficiency or prove ASIC impossibility; it preserves a direct path by which any machine capable of valid work can participate without being forced into a pool solely for payout recognition.

Second, a new chain should be designed for post-quantum threats from the beginning. Ownership uses ML-DSA-44. The C1 proof system separately freezes a fixed-Keccak/QROM security profile whose composed screen is 146.521 bits under stated assumptions. These two layers are deliberately not conflated.

Third, autonomous agents should be able to discover and verify the network on their own. BitFP makes consensus identities, manifests, node capabilities, release roots, and evidence machine-readable, while refusing to make bitfp.org or any other single origin a root of trust.

The economic model supports the same long-run objective. The 21M figure is a reference scale, not a terminal hard cap, and BitFP does not impose a later 24M hard ceiling. A nominal 52,500 BFP/year tail keeps a predictable issuance component in the security budget while its percentage expansion tends toward zero as supply grows. This is a deliberate trade: modest perpetual nominal issuance is preferred to forcing future security to depend entirely on transaction-fee demand.

Technically, the work function is not a single nonce hash. It is a frozen source-exact data-dependent computation with private worker state and a serial tail. The block and receipt thresholds remain ordinary 512-bit target tests, making their probabilities explicit. The expensive work is not repeated by validators; it is expressed as a relation $R_{\text{BitFP}}(x, w)$ and proven by six coupled lanes. LogUp authenticates the source-exact dataset reads, and PCS cross-openings bind those values to the transition that consumed them. Receipt maturity is derived with the Bitcoin-style Poisson catch-up calculation.

The remaining PRE-M10 challenge is practical rather than conceptual: the native full-node six-lane verifier must obtain meaningful service-time headroom on commodity hardware. BitFP will not call that gate complete until the measured evidence passes.

The pre-M10 roadmap now places two human-reviewed research gates after physical commodity-machine closure and external evidence: the formal Sybil Resistance Audit and the Optional Post-Quantum Privacy Architecture Review. Recording these reviews does not add privacy functionality, freeze privacy semantics, or reopen the frozen M6/M7/M8 protocol. It makes the unresolved questions visible before any M10 authorization decision.

References

- [1] S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2008.
<https://bitcoin.org/bitcoin.pdf>
 - [2] National Institute of Standards and Technology, *FIPS 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*, 2015. <https://doi.org/10.6028/NIST.FIPS.202>
 - [3] National Institute of Standards and Technology, *FIPS 204: Module-Lattice-Based Digital Signature Standard*, 2024. <https://doi.org/10.6028/NIST.FIPS.204>
 - [4] A. Chiesa, P. Manohar, N. Spooner, *Succinct Arguments in the Quantum Random Oracle Model*, TCC 2019 / Cryptology ePrint 2019/834.
 - [5] G. Arnon, A. Chiesa, G. Fenzi, E. Yogev, *WHIR: Reed-Solomon Proximity Testing with Super-Fast Verification*, Cryptology ePrint 2024/1586.
 - [6] U. Haboeck, *Multivariate Lookups Based on Logarithmic Derivatives*, Cryptology ePrint 2022/1530.
 - [7] BitFP Project, *BitFP M6 Consensus Parameter Specification v1* and M6.2 freeze audit.
 - [8] BitFP Project, *BitFP M7 Final C1 Relation v1* and G7 source-exact dataset-authentication evidence.
 - [9] BitFP Project, *BitFP M8 Fixed-Keccak / QROM Security v1*, PARAM-03 specification, and G8 security-closure audit.
 - [10] BitFP Project, *M9 G9 Production 4M Closure Audit*.
 - [11] BitFP Project, *BitFP Agent Discovery v1* and PRE-M10 readiness audits.
-

Document status: Pre-Testnet Draft v0.5.

Canonical public discovery surface: <https://bitfp.org/>

Mainnet: not live.

Public testnet: not live at publication of this draft.