Proof of Time

Complete Technical Specification

Version 1.1 — December 2025

This document provides complete technical specifications for implementing the Proof of Time consensus protocol. It covers cryptographic primitives, network protocol, consensus rules, state machine, economic parameters, wallet specifications, and operational procedures.

Table of Contents

Part I: Cryptographic Specifications

Part II: Network Protocol

Part III: Consensus Rules

Part IV: State Machine

Part V: Economic Parameters

Part VI: Operational

Part VII: Wallet

Part VIII: Genesis

Part I: Cryptographic Specifications

16. Elliptic Curve Parameters

16.1 Primary Curve: Ed25519

Purpose: Digital signatures, key agreement
Curve: $-x^2 + y^2 = 1 + dx^2y^2$ where $d = -121665/121666$
Group order (l): $2^{2■2}$ + 27742317777372353535851937790883648493
Security: ~128 bits

16.2 Ristretto255 Group

Purpose: Ring signatures, Pedersen commitments, Bulletproofs
Construction: Prime-order group from Curve25519 via Ristretto encoding
Generators: G (standard basepoint), H = hash_to_point("PoT_Pedersen_H_v1")

17. VDF Construction

17.1 Wesolowski VDF Parameters

Group: RSA group ■/N■, N = 2048-bit RSA modulus (RSA-2048 challenge)
Generator: g = 2
Security: Factorization of N unknown

```
VDF_COMPUTE(x, T): y = x^(2^T) mod N // T sequential squarings l = H_prime(x || y) // Challenge q = floor(2^T /
l) π = x^q mod N // Proof return (y, π) VDF_VERIFY(x, y, π, T): l = H_prime(x || y) r = 2^T mod l return π^l · x^r
≡ y (mod N)
```

Parameter	Value	Description
T_base	2^30	~10 min on reference hardware
T_min	2^28	Minimum difficulty
T_max	2^35	Maximum difficulty
Adjustment	2016 blocks	Recalculation period

18. Hash Functions

Function	Algorithm	Domain Tag	Usage
H_block	BLAKE3	"PoT_block_v1"	Block hashing
H_tx	BLAKE3	"PoT_tx_v1"	Transaction ID
H_prime	SHA-256	"PoT_vdf_v1"	VDF challenge
H_ring	Keccak-256	"PoT_ring_v1"	Ring signature
H_scalar	BLAKE2b	"PoT_scalar_v1"	Scalar derivation

Domain separation: H(domain_tag || len(input) || input)

19. Ring Signatures (LSAG)

Linkable Spontaneous Anonymous Group signatures provide sender anonymity with double-spend detection via key images.

19.1 Key Image

```
I = x · hash_to_point(P) Where: x = private key, P = x·G = public key Property: Same x always produces same I (linkability)
```

19.2 Ring Selection Parameters

Parameter	Value
Ring size	11 (1 real + 10 decoys)
Distribution	Gamma(19.28, 1/1.61)
Min age	10 blocks
Tier constraint	Same tier (T2/T3) only

20. Bulletproofs++

Range proofs for confidential amounts. Logarithmic proof size, no trusted setup.

Parameter	Value
Range	[0, 2^64)
Proof size	~512 bytes (single value)
Aggregation	Up to 16 proofs
Generators	G, H + 128 G_i, H_i

21. Key Derivation (HD Wallets)

```
Seed: 24-word BIP-39 mnemonic → PBKDF2 → 64 bytes Master: HMAC-SHA512("PoT_master", seed) Path: m/44'/770'/account'/change/index View key: H_scalar("PoT_view" || spend_key)
```

22. Address Encoding

Prefix	Type	Length	Encoding
pot1	Public (T0)	43 chars	Bech32m
pots1	Stealth (T1-T3)	95 chars	Bech32m
potv1	View-only	95 chars	Bech32m

Part II: Network Protocol

23. Message Format

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ ■ Magic (4B) ■ Command ■ Length ■
Checksum ■ Payload ■ ■ 0x706F7401 ■ (12B, pad) ■ (4B LE) ■ BLAKE3[:4]■ (var) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ Max payload: 32 MB
```

23.1 Message Types

Command	Description
version/verack	Handshake
ping/pong	Keepalive
getaddr/addr	Peer discovery
inv/getdata	Inventory announcement
tx/block	Data payload
headers/getheaders	Header sync
daginfo/getdagtips	DAG state

24. Handshake Protocol

```
Initiator Responder ■■■ version ■■■■■■■■■■→■ ■←■■■■■■■■■■ version ■■■ ■■■ verack ■■■■■■■■■■■→■
■←■■■■■■■■■■■ verack ■■■ [connection established]
```

24.1 Service Flags

Bit	Name	Description
0	NODE_NETWORK	Full node
1	NODE_BLOOM	Bloom filters
2	NODE_VDF	VDF production
3	NODE_ARCHIVE	Full history

25. Peer Discovery

Bootstrap methods (priority order):
1. DNS seeds: seed1.proofof.time, seed2.proofof.time, seed3.proofof.time
2. Hardcoded IPs: Fallback addresses in client
3. Peer exchange: getaddr/addr messages
4. Local discovery: UDP broadcast port 17770

Parameter	Default
max_outbound	8
max_inbound	117
max_total	125

26. Block/Transaction Propagation

26.1 Inventory Relay

```
A [new tx] ■■ inv(hash) ■■→ B A ←■■ getdata(hash) ■■■■■■■ B [not in mempool] A ■■ tx(full) ■■■■■■■■■■■■■→
B [validate, relay]
```

26.2 Compact Blocks

Short transaction IDs (6 bytes) allow receivers to reconstruct blocks from mempool. Reduces bandwidth ~90% for well-synchronized nodes.

27. DoS Protection

Message	Limit	Window	Action
tx	100	10s	Delay
inv	1000	10s	Ignore
block	50	10s	Delay
headers	2000	10s	Disconnect

27.1 Ban Scoring

```
+10: Invalid message format +20: Invalid transaction +50: Invalid block header +100: Invalid block / Equivocation
Ban threshold: 100 | Duration: 24h | Decay: -1/hour
```

Part III: Consensus Rules

28. Fork Choice Rule

```
block_weight(B) = vdf_difficulty(B) × node_weight(producer(B)) cumulative_weight(B) = Σ block_weight(ancestor)
for all ancestors canonical_tip = argmax(cumulative_weight(tip)) for all DAG tips
```

28.1 PHANTOM Ordering

```
PHANTOM_ORDER(DAG, k=8): 1. blue_set = {genesis} 2. For block B in topological order: anticone(B) = blocks
neither ancestor nor descendant if |anticone(B) ∩ blue_set| ≤ k: blue_set.add(B) 3. Order blue by
cumulative_vdf_weight desc 4. Insert red blocks between blue ancestors/descendants
```

29. Finality Conditions

Confirmations	Time	Reversal Prob	Use Case
1	~2 min	~10%	Low value
3	~10 min	~1%	Medium
6	~30 min	~0.1%	High value
12	~60 min	~0.001%	Very high
30	~3 hours	<10^-9	Exchange

29.1 Checkpoints

Every 10,000 blocks (~70 days): UTXO commitment + node state root + signatures from >67% weighted nodes. Blocks before checkpoint cannot be reorganized.

30. VDF Difficulty Adjustment

```
Every 2016 blocks: actual_time = timestamp[n] - timestamp[n-2016] target_time = 2016 × 600 = 1,209,600 sec ratio
= clamp(actual_time / target_time, 0.25, 4.0) new_difficulty = clamp(old × ratio, T_min, T_max)
```

31. Time Synchronization

Sources: System clock, NTP (pool.ntp.org), Peer median
Network time: median(system, ntp, peer_median)

Timestamp rules:
- timestamp > median(last 11 blocks)
- timestamp < network_time + 120s
- Drift >60s: warning | >300s: auto NTP resync

32. Slashing Mechanism

Offense	Detection	Penalty
Double signing	Two sigs same slot	Full slash
Equivocation	Conflicting blocks	Full slash
Invalid VDF	Verification fail	Rep reset

```
On slash: reputation=0, time_presence=0, 180-day quarantine Reporter receives 10% of offender's block rewards
```

Part IV: State Machine

33. UTXO Model Specification

```
transaction_output { version: uint8 privacy_tier: uint8 // 0=public, 1=stealth, 2=conf, 3=ring // Tier 0:
amount(u64) + pubkey_hash(32) // Tier 1: + ephemeral_pubkey(32) + encrypted_paymentid(8) // Tier 2-3: +
amount_commitment(32) + range_proof(~512) }
```

34. Transaction Validation

```
VALIDATE_TX(tx): // Structure 1. version ∈ {1,2}, inputs≥1, outputs≥1, size≤100KB // Inputs 2. Each output exists
in UTXO set 3. Scripts satisfy conditions 4. Tier 3: key_image not spent, ring valid 5. Signatures valid //
Outputs 6. Amounts valid (or commitments for T2-3) 7. Range proofs valid (T2-3) // Balance 8. T0-1: Σinputs ≥
Σoutputs + fee 9. T2-3: Σinput_commits = Σoutput_commits + fee_commit // Fee 10. fee ≥ MIN_FEE × weight
```

34.1 Transaction Weight

Tier	Input Weight	Output Weight
T0	50	30
T1	60	40
T2	150	100
T3	250	100

35. Script System

Restricted, non-Turing-complete scripts. Prevents unbounded computation.

Type	Pattern	Use
P2PKH	DUP HASH160 <h> EQUALVERIFY CHECKSIG	Single sig
P2SH	HASH160 <h> EQUAL	Script hash
MULTISIG	<M> <keys> <N> CHECKMULTISIG	M-of-N
TIMELOCK	<t> CLTV DROP <script>	Time-locked

Limit	Value
Max script size	10,000 bytes
Max stack	1,000 items
Max ops	201
Max multisig keys	20

36. State Transition Function

```
APPLY_BLOCK(state, block): 1. Verify header (parents, VDF, timestamp, producer weight) 2. For each tx: validate,
remove spent UTXOs, add new UTXOs Tier 3: add key_images to spent set 3. Add coinbase (reward + fees) 4. Update
producer: blocks_signed++, last_active=timestamp 5. Process slash evidence if any 6. Return new state
```

Part V: Economic Parameters

37. Fee Market Mechanism

Fee calculation: tx_fee = fee_rate × tx_weight
Minimum fee: 1 second (regardless of weight)
RBF: new_fee ≥ old_fee + MIN_RELAY_FEE, rate ≥ old_rate × 1.1

```
FEE_ESTIMATE(target_blocks): rates = [tx.fee/tx.weight for tx in last_100_blocks] if target == 1: return
percentile(rates, 90) if target ≤ 6: return percentile(rates, 70) if target ≤ 12: return percentile(rates, 50)
else: return percentile(rates, 25)
```

38. Block Size Limits

Parameter	Value
MAX_BLOCK_WEIGHT	4,000,000
MAX_TX_WEIGHT	400,000
COINBASE_WEIGHT	1,000

39. DAG Rewards

```
dag_width = count(blocks in same slot) adjusted_reward = base_reward / dag_width final_reward =
max(adjusted_reward, base_reward / 100) Red blocks (not in blue set): 50% of blue block reward
```

40. Long-term Sustainability

Post-emission (131.7 years): fees only.
Fee pressure: Block space scarcity, privacy premium (higher tiers pay more), L2 settlement, potential state rent.

Part VI: Operational Specifications

41. Node State Machine

```
INIT → CONNECTING → SYNCING → ACTIVE → PRODUCING (find peers) (download) (normal) (if weight≥θ_min)
```

42. Initial Block Download

```
1. Request headers from multiple peers 2. Validate header chain (VDF, timestamps, difficulty) 3. Identify best
chain by cumulative VDF weight 4. Download blocks in parallel 5. Validate blocks, build UTXO set 6. Verify UTXO
commitment matches checkpoint Optimization: Start from checkpoint (~70 days), download UTXO snapshot
```

43. Mempool Management

```
MAX_MEMPOOL = 300 MB Eviction (when full): 1. Sort by descendant_fee_rate 2. Evict lowest first (with
descendants) 3. Until size < 90% max dynamic_min_fee = base × (1 + mempool_size / MAX_SIZE)
```

44. Reorganization Handling

```
HANDLE_REORG(old_tip, new_tip): 1. Find common ancestor 2. Disconnect old blocks (return txs to mempool) 3.
Connect new blocks (remove txs from mempool) 4. Emit reorg notification Max reorg depth: 1000 blocks (hard limit
post-checkpoint)
```

45. Upgrade Mechanism

Soft forks: Version bit signaling, 95% threshold, 2016-block periods
Hard forks: 6+ months notice, specified activation height, >95% adoption recommended

Part VII: Wallet Specifications

46. Key Management

```
Wallet file: { "encrypted_seed": AES-256-GCM(seed, password_key), "kdf": {"alg": "argon2id", "mem": 65536,
"iter": 3}, "accounts": [...], "history": [...] } Backup: 24-word mnemonic + optional passphrase
```

47. Transaction Construction

```
COIN_SELECT(target, utxos, tier): candidates = filter(utxos, tier_compatible) // Try exact match (no change)
result = branch_and_bound(candidates, target, tolerance=0) if result: return result // Allow small excess result
= branch_and_bound(candidates, target, tolerance=1000) if result: return result // Fallback: largest-first
return knapsack(candidates, target) Change: internal path m/44'/770'/0'/1/n, same tier as inputs
```

48. Light Client Protocol

Compact block filters (BIP-157/158 style):
1. Download headers (verify VDF chain)
2. Download filters for each block
3. Test wallet scripts against filters
4. Download matching blocks only
5. Verify merkle proofs for relevant transactions

Part VIII: Genesis Configuration

49. Genesis Block

```
genesis_block { version: 1, timestamp: 2026-01-01T00:00:00Z, prev_hash: 0x0000...0000, vdf_proof:
INITIAL_VDF_PROOF, merkle_root: H(coinbase_tx), difficulty: T_base, nonce: 0, transactions: [ coinbase {
outputs: [], // No premine message: "The Times 01/Jan/2026 Time waits for no one" } ] }
```

50. Initial Network Bootstrap

```
Phase 1 (Testnet): - 3+ foundation nodes - Public testnet faucet - 6-month testing period Phase 2 (Mainnet
launch): - Genesis block broadcast - DNS seeds activated - No premine, fair launch - First 2016 blocks:
difficulty adjustment calibration Phase 3 (Decentralization): - Foundation nodes reduce to minority - Community
nodes reach majority weight - Governance transitions to protocol-based decisions
```

Appendix A: Constants Reference

Constant	Value	Description
COIN	1 second	Base unit
MAX_SUPPLY	21,000,000 minutes	1.26B seconds
BLOCK_TIME	600 seconds	10 minutes
HALVING_INTERVAL	210,000 blocks	~4 years
INITIAL_REWARD	50 minutes	3,000 seconds
MATURITY	100 blocks	Coinbase maturity
MAX_BLOCK_WEIGHT	4,000,000	Weight units
RING_SIZE	11	Ring members
SATURATION_TIME	180 days	Max time weight
SATURATION_REP	2,016 blocks	Max rep weight
MIN_NODES	3	Consensus minimum
CHECKPOINT_INTERVAL	10,000 blocks	~70 days

Appendix B: References

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008
[2] D. Boneh et al., "Verifiable Delay Functions," CRYPTO 2018
[3] B. Wesolowski, "Efficient Verifiable Delay Functions," EUROCRYPT 2019
[4] Y. Sompolinsky, A. Zohar, "PHANTOM: A Scalable BlockDAG Protocol," 2018
[5] B. Bünz et al., "Bulletproofs: Short Proofs for Confidential Transactions," IEEE S&P; 2018
[6] S. Noether, "Ring Signature Confidential Transactions for Monero," 2016
[7] N. van Saberhagen, "CryptoNote v2.0," 2013
[8] BIP-39, BIP-44, BIP-157, BIP-158 Bitcoin Improvement Proposals
[9] D. J. Bernstein et al., "Ed25519: High-speed high-security signatures," 2012
[10] H. de Valence et al., "The Ristretto Group," 2018