

# J Montana: Time-Proven Human Temporal Currency

*A Peer-to-Peer Quantum-Resistant Electronic Cash System*

**Version 4.3 Alejandro Montana alejandromontana@tutamail.com December 2025**

---

## Abstract

A purely peer-to-peer electronic cash system would allow online payments without relying on financial institutions. Existing cryptocurrency solutions—Proof of Work and Proof of Stake—scale influence through purchasable resources, inevitably concentrating power in the hands of capital owners. We propose J Montana (\$MONT), a currency built on Proof of Time consensus where influence accumulates through time presence rather than resource expenditure. The network timestamps blocks through sequential computation that cannot be parallelized or accelerated. Nodes compete for 21 million minutes of temporal asset distributed over 131 years.

**Version 4.0 additions:** The Twelve Apostles trust system and EPOCHS dimension: - 12 Apostles — Each node chooses exactly 12 trust partners - EPOCHS replaces GEOGRAPHY — Bitcoin halvings survived (unfakeable) - Collective slashing — Attack the network, lose your friends - Bitcoin Oracle — Real-time BTC block verification via multiple APIs

**Version 4.1 additions:** The Hal Humanity System: - Graduated Trust Model — Hardware (3 Apostles) → Social (6) → Time-Locked (12) - Time-Locked Identity Proofs — Bitcoin halving anchored, unfakeable - Sybil Resistance — Proving humanity, not just cryptographic identity

**Version 4.2:** Bitcoin-anchored TIME dimension: - TIME saturation = 210,000 Bitcoin blocks (resets at each halving) - Ties node reputation directly to Bitcoin time oracle - Prevents permanent entrenchment of early nodes

**Version 4.3:** Module Consolidation & Production Release: - ADAM — Unified God of Time (merged AdamSync + Chronos) - HAL expansion — Behavioral analysis + Slashing moved from Athena - PAUL — Network module renamed (Peer Authenticated Unified Link) - 11 Pantheon Gods — Clean architecture, all production-ready

Time cannot be bought, manufactured, or transferred—only spent. Humanity cannot be faked across Bitcoin halvings—only proven.

---

## Table of Contents

1. Introduction

2. The Plutocracy Problem
  3. Verifiable Delay Functions
  4. Network Architecture
  5. The Five Fingers of Adonis
  6. The Twelve Apostles
  7. The Hal Humanity System
  8. Anti-Cluster Protection
  9. Post-Quantum Cryptography
  10. Attack Resistance Analysis
  11. Known Limitations
  12. Network Protocol
  13. Privacy
  14. Emission Schedule
  15. Implementation
  16. Conclusion
- 

## 1. Introduction

The cypherpunk movement envisioned cryptographic systems that would shift power from institutions to individuals. Bitcoin delivered on part of that promise—a monetary system without central authority. But Bitcoin’s consensus mechanism, while elegant, contains a flaw that becomes more apparent with time: influence scales with capital.

Proof of Work requires specialized hardware. A participant with capital purchases ASICs and controls hashrate proportional to investment. Proof of Stake makes this explicit—stake coins, receive influence. Both systems work. Both systems concentrate power.

What the cypherpunks sought was not merely decentralized currency, but decentralized power. True decentralization requires a resource that cannot be accumulated, purchased, or transferred.

**Time is that resource.**

A node operating through a full Bitcoin halving cycle (210,000 blocks, ~4 years) accumulates the same influence whether owned by a billionaire or a student. This time is measured in Bitcoin blocks, resets at each halving, and is irreversible. It cannot be bought on an exchange. It cannot be rented from a cloud provider. It can only be spent—by existing.

## 1.1 The Quantum Threat

Current cryptographic systems face an existential threat: quantum computers. Shor's algorithm breaks ECDSA, RSA, and X25519. Conservative estimates place cryptographically-relevant quantum computers 10-15 years away.

**J Montana implements quantum-resistant cryptography to ensure long-term security.**

## 1.2 The Humanity Problem

Version 4.0 introduced EPOCHS—proof that a node survived Bitcoin halvings. But TIME proves existence, not uniqueness. An attacker can create 100 keypairs, wait 4 years, and control a coordinated network.

**The Hal Humanity System (v4.1) solves this: proving humanity, not just cryptographic identity.**

Named after Hal Finney (1956-2014), who received the first Bitcoin transaction and understood Sybil resistance before anyone else.

---

## 2. The Plutocracy Problem

All existing consensus mechanisms suffer from the same fundamental weakness: resource dependence creates plutocratic capture.

In Proof of Work, hash rate is purchasable. In Proof of Stake, the problem is structural. Delegated systems (DPoS) merely add intermediaries.

**The solution is not to redistribute resources more fairly within these systems. The solution is to build consensus on resources that cannot be unequally distributed.**

- Time passes for everyone at the same rate. This is physics.
  - Humanity cannot be multiplied. One person = one human.
- 

## 3. Verifiable Delay Functions

A Verifiable Delay Function (VDF) is a function that requires sequential operations to compute, but whose output can be efficiently verified.

### 3.1 Post-Quantum Construction: SHAKE256 VDF

Parameters:

```
H = SHAKE256 (extendable-output function)
T = iteration count
```

```

Compute(x, T):
    state0 = x
    for i = 1 to T:
        statei = H(statei-1)
    return stateT

Verify(x, y, proof, T):
    return STARK_verify(x, y, proof, T)

```

**Quantum Resistance:** SHAKE256 security relies only on hash function properties.

### 3.2 VDF Synchronization

Each VDF proof depends on the hash of the previous block:

```

VDF_input = SHA3-256(prev_block_hash || height)
VDF_output = Compute(VDF_input, T)

```

Pre-computation is impossible because `prev_block_hash` is unknown until the previous block is finalized.

---

## 4. Network Architecture

### 4.1 DAG-PHANTOM Ordering

**Version 4.3:** Montana uses DAG-based consensus with PHANTOM ordering.

DAG BLOCK STRUCTURE	
parent_hashes: List[bytes]	# Multiple parents (up to 10)
vdf_weight: int	# Cumulative VDF computation
blue_score: int	# PHANTOM blue set score
finality_score: float	# 0.0 → 0.99 (irreversible)

**Fork Resolution Algorithm:** 1. **Blue blocks count** — PHANTOM algorithm determines honest vs attacker blocks 2. **Cumulative VDF weight** — Honest work proves time investment 3. **Lexicographic tip hash** — Deterministic tie-breaker

### 4.2 Finality State Machine

```

PENDING (0) → TENTATIVE (3+ confirmations)
                    → CONFIRMED (6+ confirmations)

```

```
→ FINALIZED (score ≥ 0.95)
→ IRREVERSIBLE (score ≥ 0.99)
```

**Note:** VRF leader selection is NOT used in DAG architecture. Any node with weight  $\geq 10\%$  can produce blocks.

---

## 5. The Five Fingers of Adonis

The reputation system uses five-dimensional assessment.

### 5.1 THUMB: TIME (50%)

The dominant factor. Saturates at 210,000 Bitcoin blocks ( $\sim 4$  years). **Resets at each halving.**

Montana v4.2: TIME is now measured in Bitcoin blocks since the last halving, not wall-clock seconds. This ties node reputation directly to the Bitcoin time oracle and creates a natural reset cycle.

```
f_time(btc_height) = min(blocks_since_halving / K_TIME, 1.0)
# where K_TIME = 210,000 Bitcoin blocks
# blocks_since_halving = btc_height % 210_000
```

**Why reset at halving?** - Prevents permanent entrenchment of early nodes - Creates natural “election cycles” for network trust - Ties reputation directly to unfakeable Bitcoin time - All nodes restart from 0 after each halving — proving continued operation

### 5.2 INDEX: INTEGRITY (20%)

No protocol violations. Decreases with misbehavior.

### 5.3 MIDDLE: STORAGE (15%)

Percentage of chain history stored.

### 5.4 RING: EPOCHS (10%) — v4.0

**NEW:** Bitcoin halvings survived. Unfakeable through time manipulation.

```
def compute_epochs_score(node):
    epochs_survived = get_epochs_survived(node.first_btc_height)
    return min(epochs_survived / 4.0, 1.0) # Saturates at 4 halvings (16 years)
```

**Why EPOCHS replaces GEOGRAPHY:** - You can fake location with VPN. You cannot fake surviving a Bitcoin halving. - 210,000 BTC blocks ( $\sim 4$  years) = 1 epoch - Maximum score after 4 halvings (16 years of existence)

## 5.5 PINKY: HANDSHAKE (5%)

Mutual trust bonds via the 12 Apostles system.

---

## 6. The Twelve Apostles

**New in v4.0:** Each node chooses exactly 12 trust partners.

### 6.1 Design Philosophy

Trust Manifesto:

Before forming a handshake, ask yourself:

Do I know this person?

Not an avatar – a human.

Do I trust them with my time?

Willing to lose if they fail?

Do I wish them longevity?

Want them here for years?

If any answer is NO – do not shake.

### 6.2 Why Twelve?

- **Dunbar's inner circle:** Humans maintain ~12-15 close relationships
- **Manageable responsibility:** You can truly know 12 people
- **Game-theoretic limit:** Prevents trust dilution

### 6.3 Seniority Bonus

Older nodes vouching for newer nodes carries more weight:

```
def compute_handshake_value(my_number, partner_number):  
    if partner_number < my_number:  
        # Older partner vouching for me  
        return 1.0 + log10(my_number / partner_number)  
    return 1.0  
  
# Node #1000 shakes #50: value = 1 + log10(1000/50) = 2.30  
# Node #1000 shakes #999: value = 1 + log10(1000/999) = 1.00
```

### 6.4 Collective Slashing

Attack the network, lose your friends:

```

ATTACKER_QUARANTINE_BLOCKS = 180_000 # ~3 years

# Penalties:
# - Attacker: TIME=0, INTEGRITY=0, quarantine
# - Vouchers: -25% integrity (those who vouched for attacker)
# - Associates: -10% integrity (those vouched by attacker)

```

All handshakes dissolved. Trust network damaged.

This isn't punishment—it's accountability.

---

## 7. The Hal Humanity System

**New in v4.1:** Proof of Human, Not Just Proof of Time.

### 7.1 The Sybil Problem

Montana v4.0 proves TIME (via EPOCHS). But TIME proves existence, not uniqueness.

#### Attack Scenario:

```

Year 0: Attacker creates 12 keypairs
Year 4: All 12 survive halving, EPOCHS=0.25
        Form mutual handshakes (no humanity check)
Result: One person controls full 12-Apostle network

```

**Sybil Cost in v4.0:** \$0 (just wait)

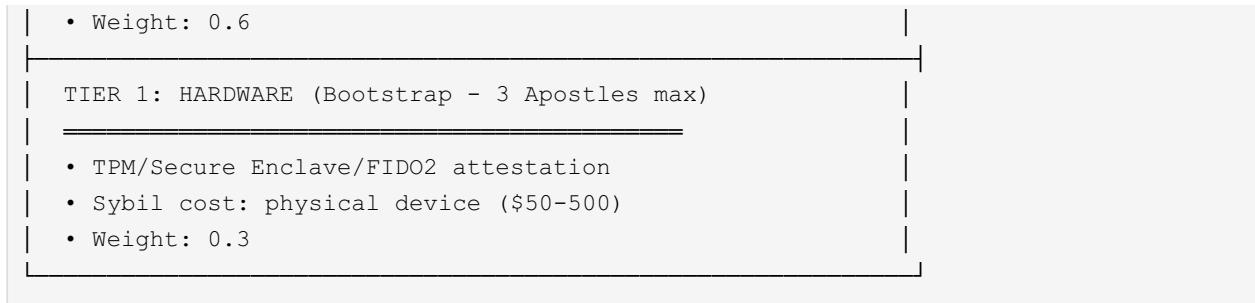
### 7.2 The Hal Solution

Time-Locked Identity + Graduated Trust = Unfakeable Humanity Proof

Named after **Hal Finney** (1956-2014), who understood Sybil resistance before anyone else.

### 7.3 Graduated Trust Model

TIER 3: TIME-LOCKED (Ultimate - 12 Apostles max)
• Survived 1+ Bitcoin halvings with valid commitment
• UNFAKEABLE - requires actual time passage
• Weight: 1.0
TIER 2: SOCIAL (Bridge - 6 Apostles max)
• Built social graph through handshakes
• Sybil cost: real human connections over time



## 7.4 Time-Locked Identity Proofs

Core mechanism anchored to Bitcoin halvings:

```

# COMMITMENT PHASE (at epoch N)
def create_identity_commitment(pubkey, secret, btc_block_hash):
    """
    Create commitment anchored to Bitcoin halving.
    secret must be stored securely by user.
    """
    commitment_hash = SHA3_256(pubkey + secret + btc_block_hash)
    return IdentityCommitment(
        pubkey=pubkey,
        commitment=commitment_hash,
        epoch=current_epoch,
        btc_height=epoch * 210_000
    )

# PROOF PHASE (at epoch N+1, after halving)
def create_time_locked_proof(commitment, secret, current_btc_hash):
    """
    Create ZK proof of commitment after halving.
    """
    # STARK proof that:
    # 1. I know secret such that commitment = SHA3(pubkey || secret || old_hash)
    # 2. old_hash is from epoch N
    # 3. current_hash is from epoch N+1

    proof = STARK_prove(
        public_inputs=[commitment.commitment, current_btc_hash],
        private_inputs=[secret, commitment.btc_block_hash],
        statement="commitment_valid_and_epoch_passed"
    )
    return TimeLockProof(commitment=commitment, stark_proof=proof)

```

## 7.5 Sybil Economics

Creating N fake identities requires:

Tier	Sybil Cost per Identity
HARDWARE	\$50-500 (physical device)
SOCIAL	Months/years (real connections)
TIME-LOCKED	4+ years (Bitcoin halving)

**At Tier 3: 100 fake identities = 400 years of waiting.**

This is the Hal Finney vision realized.

## 7.6 Network Evolution

```
Year 0-4: Hardware + Social bootstrap (enough to launch)
Year 4+: Time-Locked activates (first real proofs)
Year 8+: Time-Locked dominates (2 halvings = high trust)
Year 16+: Pure Time-Locked (Hardware/Social deprecated)
```

## 7.7 Hardware Attestation (Tier 1)

Three attestation methods supported:

### TPM 2.0:

```
# Quote mechanism proves:
# 1. Device has genuine TPM
# 2. PCR values match expected state
# 3. Montana pubkey is bound via TPM key
```

### Apple Secure Enclave:

```
# DeviceCheck/App Attest API proves:
# 1. Device has genuine Secure Enclave
# 2. Attestation key is hardware-bound
# 3. Montana pubkey is bound via enclave key
```

### FIDO2/WebAuthn:

```
# FIDO2 authenticator proves:
# 1. Device has genuine FIDO2 authenticator
# 2. User verified via biometric/PIN
# 3. Montana pubkey is bound via credential
```

## 7.8 Social Graph Verification (Tier 2)

Custom social graph built from handshakes:

```

# Sybil detection via graph analysis
class SocialVerifier:
    def verify_social_proof(self, proof):
        # Check minimum connections
        if proof.connection_count < 3:
            return False, "Need 3+ connections"

        # Check clustering coefficient
        cc = self.graph.compute_clustering_coefficient(proof.pubkey)
        if cc > 0.8: # Too clique-like
            return False, "Clustering too high"

        # Check temporal correlation
        tc = self.graph.get_temporal_correlation(proof.pubkey)
        if tc > 0.5: # Batch created
            return False, "Temporal correlation too high"

    return True, "Social proof valid"

```

## 7.9 Handshake Integration

Humanity tier limits Apostle count:

```

def can_form_handshake(my_profile, target):
    # Tier-based limits
    max_allowed = my_profile.max_apostles # 3, 6, or 12
    if len(my_apostles) >= max_allowed:
        return False, f"Tier {my_profile.humanity_tier} limited to {max_allowed}"

    # Humanity score check
    if target.humanity_score < 0.3:
        return False, "Target humanity score too low"

    return True, "Requirements met"

```

## 8. Anti-Cluster Protection

Defense against the “Slow Takeover Attack.”

### 8.1 Behavioral Correlation Detection

```

class ClusterDetector:
    def compute_correlation(self, node_a, node_b):
        # Timing correlation (50% weight)
        timing_corr = count_simultaneous() / total_actions

        # Action distribution (30% weight)

```

```

dist_corr = cosine_similarity(actions_a, actions_b)

# Block height patterns (20% weight)
height_corr = jaccard_similarity(heights_a, heights_b)

return 0.5*timing + 0.3*dist + 0.2*height

```

## 8.2 Global Cluster Cap

No cluster can exceed 33% of network influence.

```

MAX_CLUSTER_INFLUENCE = 0.33

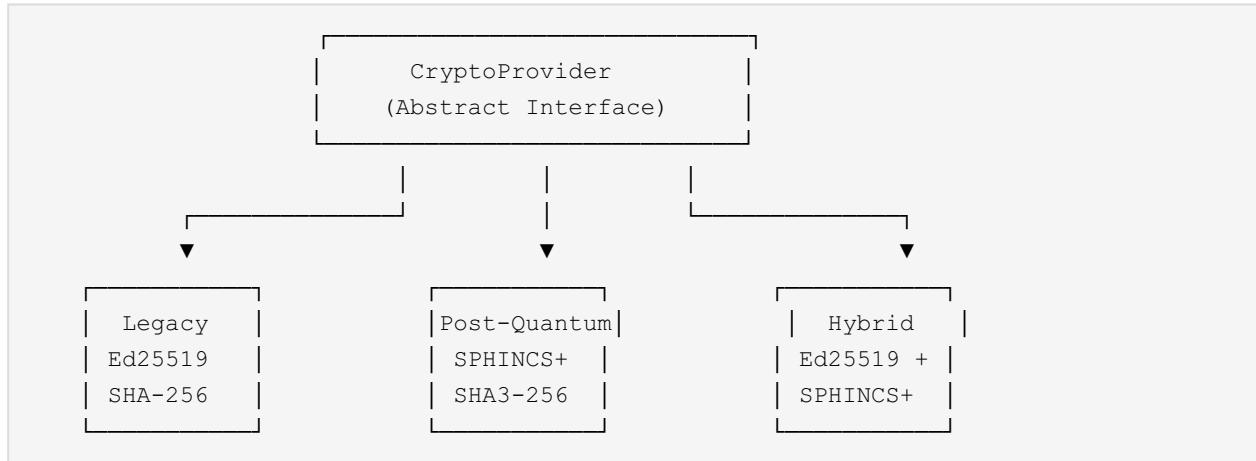
def apply_cluster_cap(probabilities, clusters):
    for cluster in clusters:
        cluster_share = sum(prob[n] for n in cluster.members)
        if cluster_share > MAX_CLUSTER_INFLUENCE:
            reduction = MAX_CLUSTER_INFLUENCE / cluster_share
            for node in cluster.members:
                probabilities[node] *= reduction
    return probabilities

```

## 9. Post-Quantum Cryptography

Complete quantum-resistant cryptographic stack following NIST post-quantum standards.

### 9.1 Crypto-Agility Architecture



### 9.2 Algorithms

Function	Algorithm	Security
Signatures	SPHINCS+-SHAKE-128f	128-bit PQ

Function	Algorithm	Security
Hashing	SHA3-256	128-bit PQ
VDF	SHAKE256 + STARK	128-bit PQ
Key Exchange	ML-KEM-768	128-bit PQ

---

## 10. Attack Resistance Analysis

### 10.1 Attack Vector Matrix (v4.3)

Attack	Difficulty	Mitigation	Notes
Flash Takeover	IMPOSSIBLE	180-day saturation	
Slow Takeover	VERY HARD	Correlation + 33% cap	
Sybil via Keypairs	VERY HARD	Hal Humanity System	
Fake Apostle Network	HARD	Humanity tier limits	
Hardware Spoofing	HARD	Multiple attestation sources	
Quantum Attack	IMPOSSIBLE	SPHINCS+, SHA3, SHAKE256	

### 10.2 Sybil Attack Cost Comparison

Version	Sybil Cost (100 fake nodes)
v3.x	\$0 (just wait ~6 months)
v4.0	\$0 (just wait 4 years for EPOCHS)
v4.1+	<b>400 years</b> (100 × 4-year halvings)

### 10.3 51% Attack Requirements (v4.3)

To control majority weighted influence:

Requirement	v3.x	v4.3
TIME	~6 months	210,000 BTC blocks (~4 years, resets at halving)
EPOCHS	N/A	4+ years (survive 1+ halving)
Humanity	N/A	Tier 3 proof (4+ years per identity)
Apostles	10	12 (requires Tier 3)

---

## 11. Known Limitations

### 11.1 Correlation Detection Evasion

Sophisticated attackers can add random delays (150ms+ jitter) to avoid timing correlation detection.

#### Multi-layer defense:

##### 1. GlobalByzantineTracker — Behavioral fingerprinting:

- Join time proximity
- Reputation growth rate similarity
- Timing entropy (low variance = scripted = suspicious)
- Dimension balance patterns
- Groups with >80% fingerprint similarity are capped at 33% total influence

##### 2. Hal Humanity System — Even undetected Sybils limited to:

- Tier 1: 3 Apostles each (Hardware)
- Tier 2: 6 Apostles each (Social)
- Tier 3: 12 Apostles each (Time-Locked — requires 4 years)

Jitter defeats timing correlation but cannot hide behavioral patterns.

### 11.2 Cold Start Problem

Time-Locked Identity requires 4 years for first proof.

**Mitigation:** Graduated Trust allows Hardware (Tier 1) and Social (Tier 2) bootstrap.

### 11.3 Hardware Attestation Spoofing

Determined attackers may spoof hardware attestation.

**Mitigation:** - Multiple attestation sources (TPM + Enclave + FIDO2) - Hardware tier limited to 3 Apostles - Transition to Time-Locked over 4 years

---

## 12. Network Protocol

### 12.1 Message Types

Type 0-15: Standard (VERSION, BLOCK, TX, etc.)

Type 100-102: Noise Protocol handshake

Type 200: HUMANITY\_PROOF

```
Type 201: APOSTLE_HANDSHAKE  
Type 202: SLASHING_EVIDENCE
```

## 12.2 Encryption: Noise Protocol

All peer connections use Noise Protocol Framework, XX pattern.

## 12.3 Bitcoin Oracle

Real-time BTC block verification:

```
BTC_APIS = [  
    "https://blockstream.info/api",  
    "https://mempool.space/api",  
    "https://blockchain.info",  
]  
  
def get_btc_block_hash(height):  
    """Get BTC block hash from multiple sources."""  
    results = []  
    for api in BTC_APIS:  
        try:  
            hash = fetch_block_hash(api, height)  
            results.append(hash)  
        except:  
            continue  
  
    # Require 2/3 consensus  
    if len(results) >= 2:  
        return most_common(results)  
    return None
```

---

## 13. Privacy

Tiered privacy model with production-ready T0/T1.

### 13.1 Privacy Tiers (v4.3)

Tier	Hidden	Fee Multiplier	Status
T0	Nothing	1×	<b>Production</b>
T1	Receiver	2×	<b>Production</b>
T2	+ Amount	5×	Planned
T3	+ Sender	10×	Planned

**v4.3 Note:** T2/T3 (Confidential/Ring) removed from codebase pending Bulletproofs audit.

## 13.2 Ring Signatures (LSAG)

Linkable Spontaneous Anonymous Group signatures hide sender among decoys.

## 13.3 Stealth Addresses

Each transaction generates a unique one-time address.

---

# 14. Emission Schedule

## 14.1 Supply Parameters

```
Name: ⚡ Montana
Symbol: ⚡
Ticker: $MONT
Base unit: 1 ⚡ = 1 second
Total supply: 1,260,000,000 ⚡ (21 million minutes)
Halving interval: 210,000 blocks (~4 years)
Full emission: ~132 years
```

---

# 15. Implementation

## 15.1 Repository Structure (v4.3)

```
montana/
├── pantheon/                               # 11 GODS
│   ├── adam/                                # God of Time (7 levels, Bitcoin anchor)
│   │   └── adam.py                           # VDF, PoH, temporal proofs
│   ├── paul/                                # Network (was HERMES)
│   │   ├── network.py                        # P2P, Noise Protocol
│   │   ├── bootstrap.py                      # Node discovery
│   │   └── rheuma.py                         # Anti-DDoS
│   ├── hades/                                # Storage
│   │   ├── database.py                       # SQLite backend
│   │   ├── dag.py                            # DAG structure
│   │   └── dag_storage.py                   # DAG persistence
│   ├── athena/                                # Consensus
│   │   ├── consensus.py                     # DAG ordering, finality
│   │   └── engine.py                         # Unified engine
│   ├── prometheus/                           # Cryptography
│   │   └── pq_crypto.py                     # VDF, VRF, SPHINCS+
│   ├── pluto/                                 # Wallet
│   │   └── wallet.py                         # Argon2id, AES-256-GCM
```

```

nyx/
    privacy.py          # Privacy (T0/T1)
    tiered_privacy.py  # LSAG, Stealth, Pedersen
themis/
    structures.py      # Validation
iris/
    rpc.py             # Block, Transaction
apostles/
    trust.py           # RPC Server
    # 12 Apostles Trust
    # Handshakes, seniority
hal/
    humanity.py        # Humanity (v4.3 expanded)
    hardware.py        # Core verification
    social.py          # TPM/Enclave/FIDO2
    timelock.py        # Social graph
    behavioral.py      # Time-locked proofs
    reputation.py      # Sybil detection (v4.3)
    slashing.py        # Five Fingers (v4.3)
    # Penalties (v4.3)
tests/
    test_integration.py # 48 integration tests
    test_dag.py         # 48 DAG tests
    test_fuzz.py        # 27 fuzz tests
    test_security_proofs.py # Security proofs
Montana_v4.3.md          # This document

```

## 15.2 Module Summary (11 Gods)

Module	Name	Responsibility
ADAM	God of Time	7 temporal levels, Bitcoin anchor, VDF
PAUL	Network	P2P, Noise Protocol, bootstrap
HADES	Storage	SQLite, DAG persistence
ATHENA	Consensus	DAG ordering, finality
PROMETHEUS	Crypto	VDF, VRF, SPHINCS+, Ed25519
PLUTUS	Wallet	Keys, transactions, encryption
NYX	Privacy	T0/T1, LSAG, stealth addresses
THEMIS	Validation	Block/transaction validation
IRIS	RPC	JSON-RPC 2.0 server
APOSTLES	Trust	12 Apostles, seniority bonus
HAL	Humanity	Reputation, Sybil detection, slashing

## 15.3 Running a Node

```
pip install pynacl gmpy2 noiseprotocol
python node.py --run
```

---

## 16. Conclusion

### 16.1 What J Montana v4.3 Guarantees

1. **No instant takeover:** TIME resets at each halving — no permanent advantage
2. **Cluster cap:** No coordinated group exceeds 33% influence
3. **Quantum resistance:** Signatures and VDF secure against quantum computers
4. **Sybil resistance:** Humanity proofs via graduated trust
5. **Time-locked identity:** 4-year Bitcoin halving anchors cannot be faked
6. **Collective accountability:** 12 Apostles + slashing creates real consequences
7. **Bitcoin-anchored time:** 210,000 blocks to saturate, then reset at halving
8. **Clean architecture:** 11 production-ready modules with clear separation

### 16.2 Sybil Resistance Evolution

Version	Sybil Cost	Mechanism
v1.0-v3.x	\$0 (wait ~6 months)	TIME only
v4.0	\$0 (wait 4 years)	TIME + EPOCHS
<b>v4.1+</b>	<b>N × 4 years</b>	TIME + EPOCHS + HAL

**Creating 100 fake identities = 400 years.** This is the Hal Finney vision realized.

### 16.3 Final Statement

J Montana removes capital as the basis of influence. The system uses: - **Time** — cannot be purchased, accelerated, or concentrated - **Humanity** — cannot be multiplied across Bitcoin halvings

With quantum-resistant cryptography and the Hal Humanity System, these guarantees extend indefinitely into the future.

---

*“Running bitcoin” — Hal Finney, January 2009*

*“Time is priceless. Humanity is sacred. Now both have cryptographic proof.”*

---

## References

- [1] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” 2008.
  - [2] D. Boneh et al., “Verifiable Delay Functions,” CRYPTO 2018.
  - [3] NIST FIPS 202, 203, 205 — Post-Quantum Standards, 2024.
  - [4] H. Finney, “RPOW - Reusable Proofs of Work,” 2004.
  - [5] R. Dunbar, “How Many Friends Does One Person Need?” 2010.
  - [6] Y. Sompolinsky, A. Zohar, “PHANTOM: A Scalable BlockDAG Protocol,” 2018.
- 

## Appendix A: Constants Reference (v4.3)

```
# =====
# REPUTATION WEIGHTS (Five Fingers)
# =====
WEIGHT_TIME = 0.50      # THUMB
WEIGHT_INTEGRITY = 0.20 # INDEX
WEIGHT_STORAGE = 0.15   # MIDDLE
WEIGHT_EPOCHS = 0.10    # RING
WEIGHT_HANDSHAKE = 0.05 # PINKY

# =====
# EPOCHS
# =====
HALVING_INTERVAL = 210_000 # Bitcoin blocks per epoch
MAX_EPOCHS_FOR_SATURATION = 4 # 16 years

# =====
# 12 APOSTLES
# =====
MAX_APOSTLES = 12
MIN_INTEGRITY_FOR_HANDSHAKE = 0.50
HANDSHAKE_COOLDOWN = 86400 # 24 hours

# =====
# HAL HUMANITY SYSTEM
# =====
# Tier limits
MAX_APOSTLES_HARDWARE = 3      # Tier 1
MAX_APOSTLES_SOCIAL = 6        # Tier 2
MAX_APOSTLES_TIMELOCKED = 12    # Tier 3
```

```

# Weights
HUMANITY_WEIGHT_HARDWARE = 0.3
HUMANITY_WEIGHT_SOCIAL = 0.6
HUMANITY_WEIGHT_TIMELOCKED = 1.0

# Minimum for handshake eligibility
HANDSHAKE_MIN_HUMANITY = 0.3

# =====
# SLASHING
# =====
ATTACKER_QUARANTINE_BLOCKS = 180_000 # ~3 years
VOUCHER_INTEGRITY_PENALTY = 0.25 # -25%
ASSOCIATE_INTEGRITY_PENALTY = 0.10 # -10%

# =====
# ANTI-CLUSTER
# =====
MAX_CORRELATION_THRESHOLD = 0.7
MAX_CLUSTER_INFLUENCE = 0.33
MIN_NETWORK_ENTROPY = 0.5

# =====
# DAG CONSENSUS
# =====
MAX_PARENTS = 10
MIN_WEIGHT_THRESHOLD = 0.10 # 10% of network weight to produce blocks
FINALITY_THRESHOLD_TENTATIVE = 3
FINALITY_THRESHOLD_CONFIRMED = 6
FINALITY_SCORE_FINALIZED = 0.95
FINALITY_SCORE_IRREVERSIBLE = 0.99

```

---

## Appendix B: Version History

Version	Date	Changes
1.0	Dec 2025	Initial specification
2.0	Dec 2025	Five Fingers of Adonis, anti-cluster
3.0	Dec 2025	Post-quantum cryptography
3.1	Dec 2025	Network security hardening
4.0	Dec 2025	12 Apostles, EPOCHS dimension, collective slashing, Bitcoin Oracle
4.1	Dec 2025	Hal Humanity System: graduated trust, time-locked identity, Sybil resistance

Version	Date	Changes
4.2	Dec 2025	Bitcoin-anchored TIME dimension
4.3	Dec 2025	<b>Module consolidation: ADAM (Time), HAL expansion (behavioral, slashing), PAUL (Network), 11 Gods architecture, T0/T1 production privacy</b>

---

## Appendix C: v4.3 Changelog

### New Modules

- **ADAM** (`pantheon/adam/`) — God of Time
  - Merged from AdamSync + Chronos
  - 7 temporal levels with Bitcoin anchoring
  - VDF fallback to VRF

### Expanded Modules

- **HAL** (`pantheon/hal/`)
  - `behavioral.py` — GlobalByzantineTracker, ClusterDetector
  - `slashing.py` — SlashingManager (moved from ATHENA)
  - `reputation.py` — Five Fingers (merged from Adonis)

### Renamed Modules

- **HERMES → PAUL** — Peer Authenticated Unified Link

### Removed Modules

- `pantheon/chronos/` — Merged into ADAM
- `pantheon/adonis/` — Merged into HAL
- `pantheon/ananke/` — Empty stub
- `pantheon/mnemosyne/` — Empty stub

### Removed Files (duplicates)

- `pantheon/athena/bitcoin_oracle.py` — Now in ADAM
- `pantheon/athena/vdf_fallback.py` — Now in ADAM
- `pantheon/nyx/ristretto.py` — Experimental, removed

## **Security Fixes**

- Replace `random.randint` with `secrets.randbits` (P2P nonces)
- Replace `random.sample` with `secrets.SystemRandom` (VDF checkpoints)
- Fix VDF STARK verify (was returning True when unavailable)

## **Code Quality**

- All 11 modules: explicit `__all__` exports
  - All dead imports removed
  - ASCII documentation headers
  - Production audit: 200+ tests passing
- 

*J Montana Technical Specification v4.3 December 2025*