Selendra: Blockchain for Real-World Applications

Version 4.0
Date: July 2025

Abstract

The blockchain trilemma forces networks to choose between security, decentralization, and scalability. Meanwhile, high transaction fees prevent mainstream adoption of useful applications like asset tokenization, loyalty programs, identity verification, and supply chain tracking.

We propose a solution that maintains security and decentralization while achieving practical scalability through parallel processing. Every user account is a smart contract supporting social recovery, removing the barrier of key management. Transaction fees are kept minimal through efficient consensus and optional sponsorship. The result is a blockchain suitable for real-world business applications without compromising on decentralization principles.

1. Introduction

Early blockchain networks solved the problem of digital money without central authorities. However, they created new problems: the blockchain trilemma and unusable interfaces that prevent real-world adoption.

The blockchain trilemma states that networks can only achieve two of three properties: security, decentralization, and scalability. Bitcoin chose security and decentralization but processes only 7 transactions per second. Ethereum improved programmability but suffers from high fees and network congestion. Newer networks achieve higher throughput by sacrificing decentralization.

Meanwhile, businesses want to use blockchain for practical applications:

- **Asset tokenization**: Making real estate, commodities, and securities tradeable as digital tokens
- **Loyalty programs**: Creating points that work across multiple businesses
- **Identity verification**: Issuing credentials that users control completely
- **Supply chain tracking**: Creating immutable records from production to delivery

These applications require three things current blockchains don't provide together: low fees, user-friendly accounts, and sufficient throughput. Most importantly, users shouldn't need to manage cryptographic keys or understand blockchain technology to benefit from these applications.

2. Smart Contract Accounts

The key insight is simple: make every user account a smart contract instead of just a cryptographic key pair. This solves the biggest barrier to blockchain adoption - the fear of losing access forever.

Traditional blockchain accounts work like this:

- User generates a private key
- If the key is lost, the account and all funds are gone forever
- No recovery options exist

Smart contract accounts work differently:

- User sets up recovery guardians (family, friends, or services)
- If access is lost, guardians can restore control
- Multiple authentication methods are supported
- Applications can be granted limited permissions

Consider Sophea setting up her account:

```
Sophea's Smart Contract Account:
├── Primary access: Her phone's biometric scanner
├── Guardians: Mom, Dad, and recovery service
├── Recovery rule: Any 2 of 3 guardians can restore access
└── App permissions: Gaming app can spend 100 tokens/day
```

When Sophea uses a loyalty program, the merchant can pay her transaction fees. She earns points without ever thinking about blockchain or gas costs. If she loses her phone, her guardians can help restore access to her account.

For businesses, this supports applications with better user experience.

3. Solving the Throughput Problem

Most blockchains process transactions one at a time, like a single-lane road. This creates bottlenecks. But consider that most transactions don't actually conflict with each other:

- Sophea sending tokens to Pisach
- Sreypov buying a coffee with loyalty points
- Dara updating his identity credentials
- Pich tracking a shipment

These can happen simultaneously because they touch different accounts and data.

Current Performance (v3 Network): Our live network processes 400-800 transactions per second with 1-second blocks and 2-3 second finality. This works for many applications but isn't enough for mainstream adoption.

Planned Improvement (v4): We're building a system that identifies which transactions can run in parallel:

```
Block Processing:
1. Analyze transactions for conflicts
2. Group non-conflicting transactions
3. Process groups simultaneously
4. Handle conflicts sequentially when needed
```

Since 70-80% of transactions typically don't conflict, this should achieve 3-5x improvement, targeting 1,200-4,000 transactions per second.

The key insight: you don't need complex sharding or multiple chains. You just need to process independent transactions at the same time.

4. Real-World Applications

Smart contract accounts and low fees enable applications that were previously impractical:

4.1 Asset Tokenization

Real estate, commodities, and company shares can be split into digital tokens that anyone can buy:

- A $1M building becomes 1,000 tokens at $1,000 each
- Rental income automatically flows to token holders
- Smart contracts enforce legal compliance
- Global 24/7 trading without intermediaries

Property owners get global investors, while small investors access assets previously limited to institutions.

4.2 Cross-Business Loyalty Programs

Each business issues their own loyalty tokens with defined values, creating a decentralized point exchange ecosystem:

Individual Business Tokens: A coffee shop issues tokens worth 1¢ each, while a fashion store issues tokens worth 10¢. Each business sets their own token value and redemption rules through smart contracts.

Decentralized Exchange: A dedicated application serves as a liquidity provider, holding reserves of various business tokens. Users can swap between different loyalty tokens based on current exchange rates and liquidity.

Cross-Business Value Creation: Sophea earns coffee tokens and fashion tokens but doesn't have enough of either for a purchase. She swaps 50 fashion tokens (worth $5) for 500 coffee tokens to buy her morning coffee. The exchange now holds more fashion tokens and fewer coffee tokens.

Liquidity Balancing: When Pisach later swaps coffee tokens for fashion tokens to buy socks, the exchange rebalances. The system maintains liquidity by allowing the exchange operator to purchase additional tokens directly from businesses.

Fee Structure: Each swap incurs a small fee (typically 1-2% in points), providing revenue for the exchange operator while keeping costs minimal for users.

Automatic Operations: Smart contract accounts with session keys let customers earn and spend points without apps or cards. The system recognizes purchases and processes transactions automatically.

No Blockchain Complexity: Merchants sponsor all transaction fees, so customers interact with a familiar point system while the underlying blockchain operations remain invisible.

The result: customers can use loyalty points across different businesses, individual businesses maintain control over their token economics, and the exchange creates additional value through improved liquidity and cross-business spending.

4.3 Self-Sovereign Identity

Instead of relying on centralized authorities, users control their credentials completely:

- Universities issue degree certificates as blockchain tokens
- Employers verify qualifications instantly without calling schools
- Users prove they have credentials without revealing private details
- No central database can be hacked or shut down

This reduces credential fraud while preserving privacy.

4.4 Supply Chain Transparency

Every step from production to delivery gets recorded immutably:

- Manufacturers record production details and quality tests
- Shippers update location and condition during transport
- Retailers verify authenticity before selling
- Consumers scan products to see complete history

This prevents counterfeiting and builds consumer trust.

4.5 Developer Experience

Applications integrate account abstraction through simple APIs supporting major frameworks and platforms. Developers choose between EVM and WebAssembly environments based on performance needs, with testing tools and automated pipelines for development and deployment.

5. Technical Architecture

The network uses a layered approach where each layer solves a specific problem:

```
Applications (Loyalty, RWA, Identity, Supply Chain)
                    ↓
Smart Contract Accounts (Recovery, Permissions, Fee Sponsorship)
                    ↓
Parallel Processing (3-5x faster transaction handling)
                    ↓
Consensus (1-second blocks, 2-3 second finality)
                    ↓
Network (Validators secure the system)
```

5.1 Account Abstraction Implementation

Every account is deployed as a smart contract with customizable validation logic. When Sophea creates her account, the system deploys a contract containing:

Authentication Module: Supports multiple signature schemes - ECDSA keys, biometric authentication via WebAuthn, or multi-party computation. Users choose their preferred method without being locked into one approach.

Guardian System: Recovery guardians are cryptographically linked to the account through threshold signatures. When 2-of-3 guardians agree to recover access, they provide proofs that are verified on-chain. A time delay (typically 24-48 hours) allows the original owner to cancel unauthorized recovery attempts.

Session Keys: Applications receive limited permissions through session keys that expire automatically. For example, a gaming app might get permission to spend up to 100 tokens per day for 30 days. These permissions are enforced at the smart contract level, not just the application level.

Fee Delegation: The account contract can be configured to accept fee payments from sponsors. When merchants pay transaction costs, users interact with blockchain applications without holding native tokens or understanding gas mechanics.

5.2 Consensus Mechanism

Block Production (Aura): Validators take turns producing blocks in predetermined time slots lasting exactly 1 second. This deterministic scheduling removes the randomness and competition found in proof-of-work systems, providing consistent block times suitable for real-time applications.

Finality (AlephBFT): While Aura handles block production, AlephBFT provides mathematical finality guarantees. This algorithm uses cryptographic signatures from validators to prove that a block cannot be reverted. Unlike probabilistic finality systems, AlephBFT delivers certainty: once finalized, transactions are permanent.

Validator Rotation: The system supports up to 100,000 validators with automatic rotation through stake-weighted elections. New validators join through existing validator approval, maintaining network quality while enabling decentralization.

5.3 Dual Runtime Environment

EVM Compatibility: Existing Ethereum applications run without modification. The system processes up to 36 million gas per block, supporting complex DeFi protocols and smart contract interactions.

WebAssembly Performance: New applications can use WebAssembly for higher performance. While EVM prioritizes compatibility, WASM supports computationally intensive applications like real-time gaming or complex financial calculations.

Cross-Runtime Communication: Contracts in both environments can interact directly. An EVM-based DeFi protocol can call a WASM identity verification service, creating hybrid applications that use both platforms.

6. Economics and Fees

Transaction fees are designed to be minimal while ensuring network sustainability:

Fee Structure:

- Current network: ~0.0005 SEL per transaction (~$0.000012 USD)
- Target for v4: 0.001 SEL per transaction (~$0.000025 USD)
- Fee sponsorship: Applications can pay user fees

Fee Distribution:

- 40% burned (reduces token supply)
- 35% to validators (security incentive)
- 15% to network treasury (development funding)
- 10% to paymaster pool (sponsorship incentives)

Validator Economics: Validators stake 31,416 SEL tokens minimum and earn rewards for honest participation. Smaller holders can delegate to validators. The system can tolerate up to 1/3 malicious validators while maintaining security.

6.1 Security Model

Economic Security

Network security relies on making attacks more expensive than potential gains. To compromise the network, an attacker would need to control more than 1/3 of staked tokens:

Attack Cost Analysis:

- Current network value: Assume 100M SEL staked across all validators
- Required attack stake: >33.3M SEL (1/3 + 1 token)
- Attack cost at $0.025/SEL: >$833,000
- Potential gains: Transaction fees and network disruption
- Economic result: Attack cost far exceeds any realistic gain

Slashing Penalties: Validators lose staked tokens for provably malicious behavior like signing conflicting blocks or being offline during assigned slots. This creates strong incentives for honest participation.

Delegation Security: Token holders who delegate maintain ownership and can withdraw delegation at any time. Validators cannot access delegated funds directly, only earn commission on rewards.

Guardian Security

The account recovery system addresses the most common guardian attack vectors:

Guardian Collusion Prevention:

- Time delays allow legitimate owners to cancel unauthorized recovery
- Social guardians (family/friends) are unlikely to collude maliciously
- Professional guardian services are reputation-based and auditable
- Multiple guardian models (2-of-3, 3-of-5) increase collusion difficulty

Recovery Process Security:

- Guardians must provide cryptographic proofs of identity
- Recovery requests are publicly visible on-chain
- Original owners receive notifications through multiple channels
- Emergency cancellation mechanisms override guardian decisions

Session Key Security:

- Keys have strict spending limits and expiration dates

- Permissions are granular (specific functions, not full account access)
- Users can revoke session keys instantly
- Applications cannot escalate permissions without user approval

Network Attack Resistance

51% Attack Prevention: Unlike proof-of-work systems where miners can rent hash power, validators must stake tokens long-term. Acquiring attack-level stake requires massive capital commitment and provides limited attack duration.

Nothing-at-Stake: AlephBFT consensus prevents validators from signing multiple competing chains without penalty. The protocol cryptographically proves which validators signed conflicting blocks, enabling automatic slashing.

Long-Range Attacks: New nodes joining the network use checkpoint mechanisms and social consensus to identify the canonical chain, preventing attackers from rewriting distant history.

Eclipse Attacks: The peer-to-peer network uses multiple connection strategies including DNS bootstrapping, known validator addresses, and peer reputation systems to prevent isolation attacks.

7. Performance and Comparative Analysis

7.1 Current Network Performance

Measured Performance:

- **Throughput**: 400-800 transactions per second (mixed workload)
- **Block time**: 1 second (deterministic)
- **Finality**: 2-3 seconds (mathematical guarantee)
- **Fee cost**: ~$0.000012 USD per transaction
- **Runtime support**: Both EVM and WebAssembly contracts

Technical Constraints:

- Block weight limit: 400ms computation per 1-second block
- EVM gas limit: 36 million gas per block
- Maximum block size: 5MB
- Validator set: Up to 100,000 validators supported

7.2 Competitive Analysis

Network	TPS	Finality	Fees	Account Model	Trade-offs
Bitcoin	7	60+ min	$1-50	Key pairs	Security over scalability
Ethereum	15	6-20 min	$1-100	Key pairs	Decentralization over speed
Solana	3,000	12.8s	$0.00025	Key pairs	Speed over decentralization
Avalanche	4,500	1-2s	$0.01-1	Key pairs	Performance over simplicity
Polygon	7,000	2.3s	$0.01	Key pairs	Scalability over sovereignty

Network	TPS	Finality	Fees	Account Model	Trade-offs
BSC	2,000	3s	$0.10	Key pairs	Speed over decentralization
Selendra v3	800	2-3s	$0.000012	Smart contracts	Balanced approach
Selendra v4	4,000	2-3s	$0.000025	Smart contracts	Trilemma balance

7.3 Technical Differentiation

Account Abstraction Features:

- Native smart contract accounts as default
- Built-in social recovery reduces key management risks
- Session keys support gasless user experiences
- Fee sponsorship enables broader adoption

Decentralization Approach:

- Higher validator count than many high-TPS networks
- Supports up to 100,000 validators vs Solana's ~1,500
- Geographic distribution reduces regulatory risks
- Maintains censorship resistance properties

Developer Experience:

- Full EVM compatibility for existing projects
- WebAssembly for high-performance new applications
- Account abstraction APIs simplify user onboarding
- No additional infrastructure required

7.4 Trade-off Analysis

What Selendra Optimizes For:

- User experience without sacrificing decentralization
- Real-world business applications over speculative trading
- Long-term sustainability over short-term performance metrics
- Proven technology over experimental approaches

Conscious Trade-offs Made:

- Lower theoretical TPS than centralized networks
- More complex account model than simple key pairs
- Longer development timeline for parallel processing
- Conservative approach to unproven technologies

v4 Target Position: With parallel processing, Selendra aims to achieve higher throughput while maintaining user experience and decentralization properties that other networks typically trade off for performance.

8. Development Roadmap

8.1 Current Status (v3 Network)

Operational Features:

- Smart contract accounts with guardian recovery
- 400-800 TPS with 1-second deterministic blocks
- Sub-cent transaction fees with merchant sponsorship
- Dual EVM and WebAssembly runtime environments
- AlephBFT mathematical finality in 2-3 seconds

Production Metrics:

- Network uptime: >99.9% since mainnet launch
- Validator set: Currently 50+ validators across 6 continents
- Transaction cost: Consistently under $0.000015 USD
- Developer adoption: 15+ applications in production

8.2 v4 Parallel Processing Implementation

Phase 1: Foundation (Months 1-3)

- **Dependency Analysis Engine**: Build transaction conflict detection algorithms
- **Runtime Modification**: Adapt Substrate's Executive for parallel execution
- **Testing Framework**: Create parallel execution test environments
- **Validation Criteria**: 100% correctness in conflict detection, zero state inconsistencies

Phase 2: Core Engine (Months 4-6)

- **Multi-threaded Executor**: Implement configurable thread pool architecture
- **State Partitioning**: Enable parallel access to different account ranges
- **Optimistic Concurrency**: Build rollback mechanisms for detected conflicts
- **Validation Criteria**: 2x throughput improvement, <5% rollback rate

Phase 3: Optimization (Months 7-9)

- **Dynamic Batching**: Real-time transaction grouping based on access patterns
- **Cache Management**: Optimize state access for parallel execution
- **Performance Tuning**: Achieve target 3-5x throughput improvement
- **Validation Criteria**: 1,200+ TPS sustained, <1ms additional latency

Phase 4: Production (Months 10-12)

- **Mainnet Integration**: Gradual rollout with fallback to sequential processing
- **Network Testing**: Stress testing with realistic transaction loads
- **Validator Upgrade**: Coordinate network-wide consensus upgrade
- **Validation Criteria**: 4,000+ TPS peak capacity, full backward compatibility

8.3 Technical Risks and Mitigations

High-Risk Areas:

1. **State Consistency**: Parallel execution could introduce race conditions

- *Mitigation*: Extensive formal verification and test coverage
- *Fallback*: Automatic reversion to sequential processing

2. **Consensus Compatibility**: AlephBFT may need modifications for parallel blocks
    - *Mitigation*: Early coordination with AlephBFT development team
    - *Fallback*: Optimize sequential processing instead
3. **Validator Coordination**: Network upgrade requires validator consensus
    - *Mitigation*: Extensive testnet validation and gradual rollout
    - *Fallback*: Optional parallel processing for willing validators

Medium-Risk Areas:

- Developer tooling compatibility with parallel execution
- Performance degradation in high-conflict scenarios
- Increased resource requirements for validator nodes

8.4 Success Metrics

Technical Validation:

- Minimum 3x throughput improvement over v3 baseline
- State consistency maintained across all parallel execution scenarios
- Less than 1% of transactions require sequential fallback processing
- Full compatibility with existing smart contracts and applications

Network Health:

- Validator participation rate remains above 95%
- Transaction confirmation times stay within 2-3 second range
- Fee levels remain competitive with current v3 pricing
- No security incidents during or after parallel processing deployment

Adoption Metrics:

- 50+ applications actively using parallel processing features
- 100,000+ daily active users across all applications
- Developer satisfaction scores above 4.5/5 for new tooling
- Successful migration of 90%+ existing applications to v4

8.5 Interoperability Strategy

Cross-Chain Connectivity: While Selendra provides a complete ecosystem for real-world applications, interoperability remains important for user choice and capital efficiency:

Bridge Infrastructure: Secure, audited bridges to Ethereum and other major networks enable asset transfers while maintaining decentralization. Users can bring existing tokens to Selendra for lower fees and better user experience.

Universal Account Access: Smart contract accounts can hold and manage assets from multiple chains through bridge contracts, creating unified user experiences across blockchain ecosystems.

Protocol Cooperation: Rather than competing directly with other networks, Selendra focuses on real-world business applications while connecting to DeFi and NFT ecosystems where appropriate.

9. Conclusion

Selendra solves the blockchain trilemma by combining three key innovations:

1. **Smart contract accounts** that reduce key management barriers through social recovery
2. **Low transaction fees** through efficient consensus and optional sponsorship
3. **Parallel processing** for higher throughput of non-conflicting transactions

The current network shows that blockchain systems can balance user experience with decentralization. With 400-800 TPS, 1-second blocks, and 2-3 second finality, it supports applications like asset tokenization, loyalty programs, identity verification, and supply chain tracking.

The v4 upgrade will add parallel processing to target 1,200-4,000 TPS, aiming to position Selendra for broader business adoption.

Rather than making the typical trade-offs between decentralization, security, and scalability, Selendra attempts to balance all three properties. The goal is a blockchain suitable for business applications with improved user experience compared to traditional blockchain interactions.

This approach aims to make blockchain technology more accessible beyond the current cryptocurrency user base.