BitLight: An Integrated Layer 1 and Layer 2 Architecture for Non-Custodial Bitcoin Operations

BitLight Labs
info@bitlightlabs.com

October 16, 2025, v1.0.5

Legal Disclaimer

Nothing in this whitepaper constitutes an offer to sell or a solicitation of an offer to purchase any tokens. BitLight Labs publishes this whitepaper solely to receive feedback and commentary from the public.

If BitLight Labs offers any sale of tokens (or Simple Agreement for Future Tokens) in the future, such offering will be made through definitive offering documents, including information disclosure documents and risk factors. Such definitive documents are expected to include an updated version of this whitepaper, which may differ materially from the current version. If BitLight Labs conducts such offering in the United States, the offering may be available only to accredited investors.

Nothing in this whitepaper should be considered or construed as a guarantee or promise regarding BitLight Labs' business, how the tokens will develop, or the utility or value of the tokens. This whitepaper outlines our current plans, which may be changed at our sole discretion at any time, and the success of which will depend on many factors beyond BitLight Labs' control, including market factors and other factors within the data and cryptocurrency industries. Any statements regarding future events are based solely on BitLight Labs' analysis of the issues described in this whitepaper. Such analysis may prove to be incorrect.

Abstract

The Bitcoin network's potential for supporting programmable assets and large-scale payment systems is constrained by its base layer's inherent limitations in throughput and scripting capabilities. Existing Layer 2 (L2) solutions, while addressing these constraints, introduce significant trade-offs between non-custodial asset security and operational usability, typically requiring users to manage continuously online nodes and associated private keys.

This paper presents BitLight, an integrated Layer 1 (L1) and Layer 2 (L2) architecture designed to address these challenges. The architecture integrates L1-level functionality (such as advanced Taproot scripts and transaction construction) with L2-level operations. Its core technical innovation is a separation of operation and signing model that functionally decouples the network interaction operations of an L2 node from the user's private key signing authorization. This separation mechanism aims to reduce the security risks and operational overhead associated with managing online "hot wallets" for L2 interactions.

The BitLight architecture integrates the RGB protocol for client-side validation asset management with the Lightning Network for instant payments. This integration provides an infrastructure designed for digital assets that balances transaction efficiency with user-controlled security, thereby offering a viable path for broader adoption of Bitcoin L2 technologies through reduced user-side complexity.

Contents

1 Introduction **4**
1.1 Bitcoin Network Scalability and Functional Limitations 4
1.2 Core Challenge: The Bitcoin Programmability Trilemma 4
1.3 Existing Solutions' Position in the Trilemma 5
1.4 Contribution: A Systematic Integration Architecture Addressing Dual Challenges 6

2 BitLight Architecture **6**
2.1 Overview 6
2.2 Layered Decoupling 7
2.2.1 Secure Settlement Layer (L_{settle}): Bitcoin L1 7
2.2.2 Contract State Layer (L_{state}): RGB Protocol 8
2.2.3 High-Speed Transport Layer ($L_{transport}$): Lightning Network 8
2.3 Functional Separation 8
2.3.1 Operational Component (C_{op}) 9
2.3.2 Signing Component (C_{sign}) 9
2.4 Component Interaction 10

3 Technical Foundation: RGB Protocol and Lightning Network Integration **10**
3.1 Lightning Network: Instant Payment Channel Layer 10
3.1.1 Operating Principles 10
3.1.2 Role in BitLight Architecture 11
3.2 RGB Protocol: Client-Side Validation Asset Layer 11
3.2.1 Key Technical Breakthroughs in RGB v0.12 11
3.2.2 Core Paradigm: Client-Side Validation 12
3.2.3 Comparison with Global Consensus Models 12
3.2.4 Role in BitLight Architecture 12
3.3 Technical Integration: Transferring RGB Assets in Lightning Network Channels . 13

4 Methodology: An Integrated Architecture for Bitcoin L2 **14**
4.1 Introduction 14
4.2 Design Challenges of Layered Decoupling 14
4.2.1 Consistency Challenges in Channel Operations 14
4.2.2 State Management in Asynchronous Environments 14
4.3 Security Analysis of Functional Separation 14
4.3.1 Design Principles of Permission Boundaries 14

4.3.2 Security Guarantee Mechanisms 15
4.3.3 Architectural Advantages for Availability 15
4.4 Core Method: HTLC Transactional Processing Model 15
4.4.1 Consistency Challenges for HTLC and RGB Transfers 15
4.4.2 Two-Phase Execution Process 15
4.4.3 Atomicity Guarantees 16

5 Architectural Support Analysis for Application Scenarios 16
5.1 Introduction 16
5.2 Infrastructure for Autonomous Agent Economic Networks 16
5.2.1 Technical Requirements Decomposition 16
5.2.2 BitLight Architecture Applicability Analysis 17
5.3 Large-Scale Stablecoin Payment Systems 17
5.3.1 Technical Requirements Decomposition 17
5.3.2 BitLight Architecture Applicability Analysis 17
5.4 Architecture Universality and Future Application Directions 18
5.4.1 Decentralized Finance (DeFi) Services 18
5.4.2 Digital Asset and Rights Management 19

6 Conclusion 19

1 Introduction

1.1 Bitcoin Network Scalability and Functional Limitations

The Bitcoin protocol [1] established a decentralized, proof-of-work-based electronic cash system whose security and resilience have been validated over an extended period. However, its base layer (Layer 1) design, optimized for decentralization and security, makes compromises in transaction throughput and scripting functionality. Specifically, the Bitcoin mainnet exhibits the following inherent limitations:

- **Limited Transaction Throughput:** The network is designed to produce blocks approximately every ten minutes with constrained block sizes, resulting in a global transaction processing capacity limited to approximately 3-7 transactions per second. This falls far short of meeting the demands of large-scale payment systems.
- **Non-Turing Complete Script System:** Bitcoin Script is an intentionally designed, functionally limited stack-based language that does not support loops and complex state computations. This makes it extremely difficult to directly implement complex smart contracts and state-dependent financial applications on L1.

To overcome these limitations, research and development efforts have turned toward building Layer 2 (L2) protocols on top of the Bitcoin mainchain. These protocols aim to extend Bitcoin's functionality without modifying the core consensus rules.

1.2 Core Challenge: The Bitcoin Programmability Trilemma

To extend Bitcoin's functionality without compromising its L1 core properties, various Layer 2 (L2) solutions have emerged. Through systematic analysis of these solutions, we identify a fundamental design trade-off that can be characterized as the **Bitcoin Programmability Trilemma**.

Unlike the broader "blockchain trilemma" (decentralization, security, scalability), this model focuses specifically on the challenges arising from Bitcoin L1 protocol's inherent characteristics, namely how to achieve rich programmability and massive scalability without compromising its L1-grade security. Specifically, Bitcoin Script's **non-Turing completeness**, **limited block space** (approximately 4MB), and **UTXO model state management constraints** create unique technical barriers for building complex smart contracts and high-throughput systems on Bitcoin.

This model posits that a Bitcoin L2 solution faces difficulty in simultaneously optimizing the following three core attributes:

1. **L1-Grade Security:** Whether the solution's asset security relies completely and solely on Bitcoin L1's consensus and UTXO model finality, without introducing any additional, less trustworthy consensus mechanisms or trust assumptions. In the Bitcoin context, this means security must be guaranteed through native mechanisms such as Bitcoin scripts and timelocks.

2. **Rich Programmability:** Whether the solution supports Turing-complete or near-Turing-complete smart contracts, allowing developers to build applications with complex state transition logic. Due to Bitcoin L1's script limitations, such programmability must be implemented at the L2 layer while addressing technical challenges of state verification and dispute resolution.

3. **Massive Scalability:** Whether the solution can handle high-frequency, low-cost transactions with throughput unconstrained by Bitcoin L1's block capacity and transaction fee volatility. Particularly during L1 congestion periods, maintaining high L2 performance becomes a key challenge.

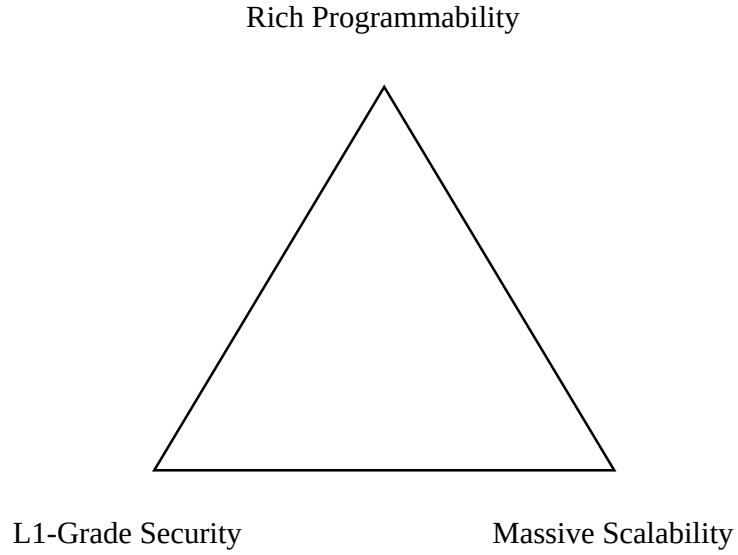

Figure 1: The Bitcoin Programmability Trilemma. L2 solutions often have to trade off one of these three properties.

1.3 Existing Solutions' Position in the Trilemma

Current mainstream L2 technology approaches can all be viewed as different compromises within this trilemma framework:

- **Sidechains:** Prioritize **programmability** and **scalability** (attributes 2 and 3). They achieve complex contracts and high throughput through independent blockchains, but sacrifice **L1-grade security** (attribute 1) because their asset security depends on their own consensus mechanisms, typically federated or PoS.

- **State Channels (exemplified by native Lightning Network):** Achieve **L1-grade security** and **massive scalability** (attributes 1 and 3). Channel final settlement is guaranteed by L1, with nearly unlimited off-chain transactions. However, they compromise on **programmability** (attribute 2), with native designs primarily for payments and no support for complex state machines.

- **Rollups (theoretical model on Bitcoin):** Aim to achieve **L1-grade security** and **rich programmability** (attributes 1 and 2) by placing computation off-chain and anchoring data or proofs to L1. However, under Bitcoin's current protocol, the lack of a dedicated data availability layer makes publishing Rollup data on L1 (to ensure verifiability) extremely expensive, severely limiting **scalability** (attribute 3). This represents a fundamental difference from the challenges faced in implementing Rollups on chains with stronger data availability, highlighting Bitcoin L1 architecture's unique constraints on advanced L2 solutions.

1.4 Contribution: A Systematic Integration Architecture Addressing Dual Challenges

This paper proposes and elaborates BitLight, a system architecture designed to simultaneously address the aforementioned dual challenges at both protocol and user layers.

Our core argument is that through novel, deep systematic integration of existing, optimized Bitcoin-native technologies—namely **Taproot protocol at the L1 layer, RGB protocol and Lightning Network at the L2 layer**—it is possible to construct an L2 paradigm that, in aggregate, surpasses existing single-point solutions.

The primary technical contribution of this paper is proposing and demonstrating such a **complete, three-layer integrated system architecture**. This architecture addresses the aforementioned dual challenges through two core design principles:

1. **Layered Decoupling:** The three dimensions of the "trilemma" are mapped to the most suitable technical layers (L1 anchoring, RGB contracts, Lightning Network channels) for handling each specific problem.

2. **Functional Separation:** A "separation of operation and signing" model is employed to decouple the user-layer problem of "node-wallet integration."

This paper will elaborate on the design principles of this integrated architecture, the interaction protocols between layers, and how to ensure state consistency across the entire complex system in asynchronous environments. Ultimately, this research aims to validate that such a systematic integration approach provides a viable and efficient path for building next-generation Bitcoin programmable financial infrastructure.

2 BitLight Architecture

2.1 Overview

Figure 2 presents the overall view of the BitLight architecture. The architecture consists of three logical layers and two functional components, each bearing specific technical responsibilities. The following sections detail the specific functions, interaction protocols, and overall collaborative mechanisms of each component.

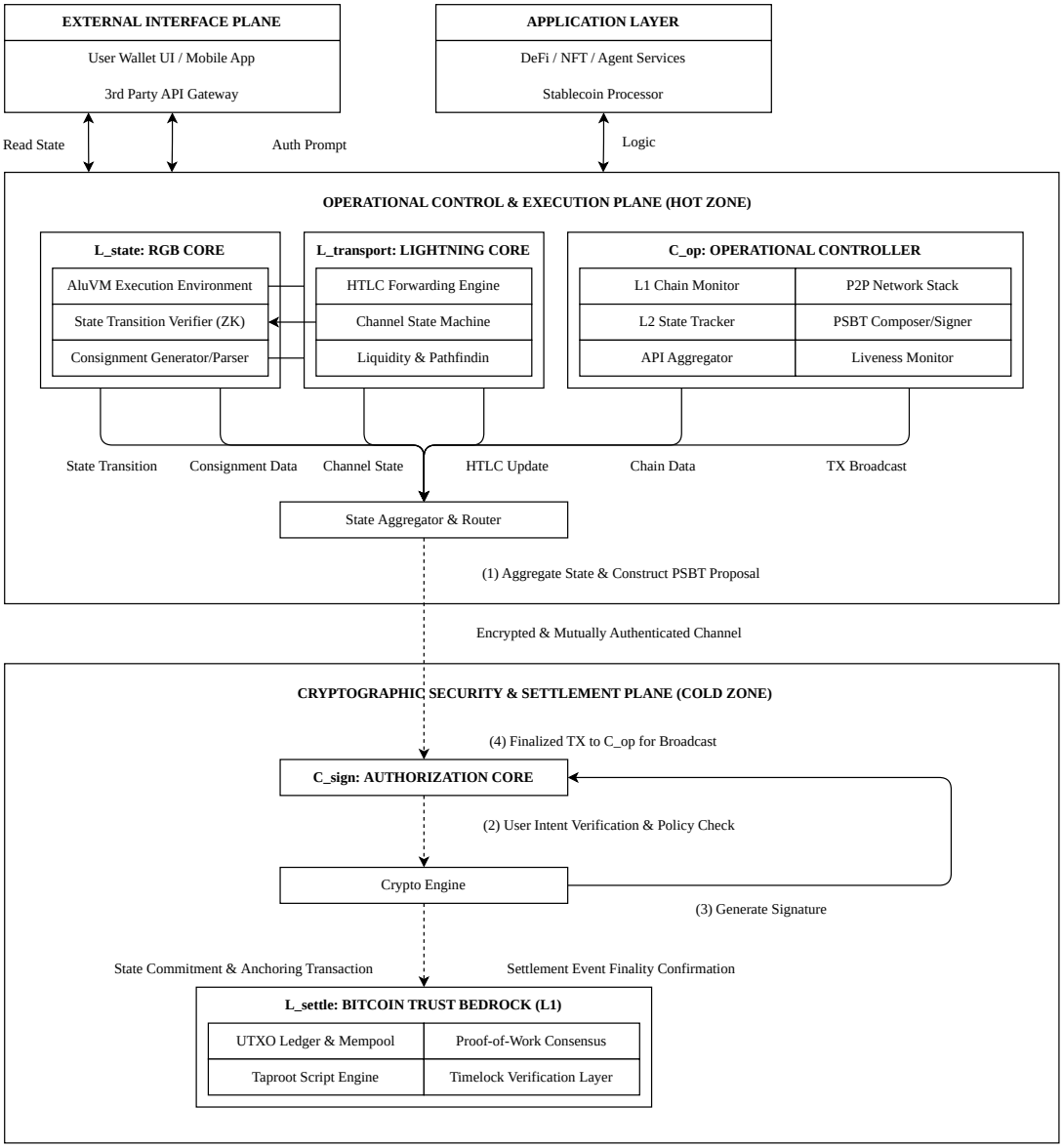

Figure 2: BitLight System Architecture

2.2 Layered Decoupling

This architecture decomposes the core functional requirements of L2 systems into three logically independent but mutually collaborative technical layers, with each layer responsible for a specific functional domain.

2.2.1 Secure Settlement Layer (L_{settle}): Bitcoin L1

- **Component:** Bitcoin mainnet (Layer 1)
- **Functions:**
  1. **Finality Anchor:** Serves as the ultimate trust anchor for all asset ownership and state transitions. The validity of all off-chain activities (L2) can ultimately be traced back to one or more UTXOs on L1.

2. **Dispute Arbitration:** Provides a final venue for executing dispute resolution logic (such as penalty transactions) for L2 protocols like Lightning Network.

- **Interface:** L1 interaction is achieved through constructing and broadcasting standard Bitcoin transactions, particularly utilizing Taproot (P2TR) outputs to optimize efficiency and data footprint.

2.2.2 Contract State Layer (L_{state}): RGB Protocol

- **Component:** RGB protocol stack, a client-side validation smart contract system
- **Functions:**

1. **Programmable Asset Definition:** Enables definition of digital assets (fungible and non-fungible tokens) with complex states and custom logic. Its AluVM-based virtual machine supports Turing-complete contracts.
2. **State Transition Verification:** Asset state transition logic is verified entirely on the client side. The verification process involves checking the complete cryptographic proof history (Consignments) from asset genesis to current state.
3. **State Commitment:** Binds asset state to an L1 UTXO through a cryptographic commitment, thereby anchoring asset ownership to the secure settlement layer.

- **Interface:** RGB asset transfers involve the sender generating Consignment data packages containing complete verification history for receiver validation.

2.2.3 High-Speed Transport Layer ($L_{transport}$): Lightning Network

- **Component:** Lightning Network protocol stack
- **Functions:**

1. **High-Frequency State Updates:** Provides a peer-to-peer payment channel network allowing participants high-frequency, low-cost state updates.
2. **Instant Payments:** Supports near-instant, trustless value transfers through Hash Time-Locked Contracts (HTLCs) routing.

- **Interface:** Interaction with other components occurs through standard Lightning Network P2P messages (BOLT specifications) and construction and signing of commitment transactions.

2.3 Functional Separation

To address the user-layer security-availability conflict, this architecture decouples operational entities into two functionally independent components.

2.3.1 Operational Component (C_{op})

- **Definition:** A software entity responsible for handling all network tasks requiring continuous online, high-frequency interaction. It can be deployed on cloud servers, home servers, or local computers.
- **Responsibilities:**
  1. **Network Monitoring and Communication:** Continuously monitors Bitcoin L1 and Lightning Network, conducting P2P communication with other nodes.
  2. **State Management:** Maintains the latest state of Lightning Network channels and handles RGB asset transfer Consignment verification processes.
  3. **Transaction Proposal Construction:** Based on user instructions or protocol events, constructs all transaction proposals requiring signatures. This includes L1 transactions (such as channel opening/closing) and L2 commitment transactions. Transaction proposals are formatted as Partially Signed Bitcoin Transactions (PSBTs).
- **Security Model:** This component operates in **watch-only** mode, **holding no private keys capable of authorizing fund transfers**. Even if this component's operating environment is completely compromised, attackers cannot steal funds.

2.3.2 Signing Component (C_{sign})

- **Definition:** A software or hardware entity responsible for all cryptographic signing operations, representing the sole technical embodiment of user asset sovereignty. It typically runs in a high-security, isolated environment such as a mobile device's Secure Enclave, hardware wallet, or dedicated desktop application.
- **Responsibilities:**
  1. **Key Management:** Securely generates, stores, and manages user L1 private keys. **Private keys never leave this component's security boundary**.
  2. **Transaction Verification and Authorization:** Receives PSBT proposals from the operational component, parses their content, and presents key information (such as transaction amounts, recipients, asset types, fees) to users in clear, human-readable format. **Critical Security Mechanism:** The signing component maintains independent state verification capabilities, ensuring accuracy of transaction information presented to users through cross-validation, preventing malicious behavior by the operational component.
  3. **Signature Execution:** After obtaining explicit user authorization (such as through biometric authentication, PIN codes, or physical key confirmation), executes cryptographic signing operations.
  4. **Signature Return:** Returns signed data to the operational component.

2.4 Component Interaction

- **Secure Communication Channel:** The operational component and signing component interact through an **end-to-end encrypted, mutually authenticated** communication channel. This channel ensures that transaction proposals in transit cannot be tampered with and that only legitimate, paired components can communicate with each other.
- **Workflow:** All value transfer operations in the system follow a standard **"propose-authorize-execute"** workflow. The operational component (C_{op}) handles "proposal," the signing component (C_{sign}) handles "authorization," and finally the operational component handles "execution" (broadcasting or sending to counterparties). This workflow ensures that no step can bypass explicit user permission.

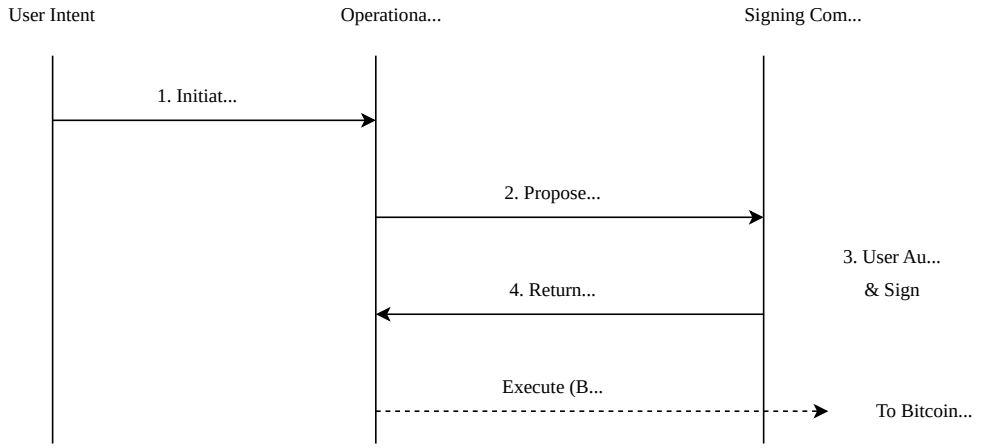

Figure 3: The "Propose-Authorize-Execute" Workflow. This illustrates the functional separation between the online operational component and the offline signing component.

3 Technical Foundation: RGB Protocol and Lightning Network Integration

The functional implementation of the BitLight architecture relies on two key Layer 2 protocols in the Bitcoin ecosystem: the **RGB protocol** for asset management and the **Lightning Network** for payment channels. The following sections elaborate on the core principles of these two technologies and explain how they work collaboratively within this architecture.

3.1 Lightning Network: Instant Payment Channel Layer

The Lightning Network [2] is a decentralized network built on Bitcoin's existing script capabilities, designed to enable instant, high-frequency, low-cost payments. Its core mechanism is **Payment Channels**.

3.1.1 Operating Principles

1. **Channel Opening:** Two participants lock a certain amount of bitcoin in a 2-of-2 multisig output through an L1 multisig transaction (funding transaction).

2. **Off-chain State Updates:** Within the channel, both parties can exchange signed **commitment transactions** unlimited times. Each new commitment transaction redistributes funds in the multisig output and invalidates the previous commitment transaction. These transactions are not broadcast to the Bitcoin mainnet.

3. **Channel Closing:** Either party can broadcast the last mutually agreed commitment transaction to the mainnet at any time, thereby settling the final fund distribution within the channel.

3.1.2 Role in BitLight Architecture

The Lightning Network constitutes the **payment layer** of BitLight. In our architecture, all high-frequency network communications related to channel state maintenance, payment routing, and HTLC (Hashed Time-Locked Contract) negotiation are handled by the **operational component**. All **commitment transaction signatures** required for creating, updating, and closing channels must be authorized by the **signing component**. This division of labor naturally aligns with Lightning Network protocol requirements.

3.2 RGB Protocol: Client-Side Validation Asset Layer

RGB is a smart contract system designed for Bitcoin and Lightning Network that implements programmable asset issuance and transfer. RGB's design paradigm differs fundamentally from smart contract platforms based on global consensus like Ethereum.

3.2.1 Key Technical Breakthroughs in RGB v0.12

The implementation of the BitLight architecture is built on **RGB v0.12**, which introduces several key technical innovations that are decisive for the feasibility and functional completeness of the BitLight architecture:

- **zk-AluVM Smart Contract Engine:** The core innovation of v0.12 is the introduction of the Zero Knowledge Algorithmic Logic Unit Virtual Machine (zk-AluVM) based on zero-knowledge proofs. Compared to earlier versions, zk-AluVM provides **Turing-complete computational capabilities with native support for finite fields**, allowing developers to write arbitrarily complex smart contract logic while significantly improving client-side verification efficiency and security. This functionality forms the technical foundation for BitLight's support of autonomous agent economic networks and complex DeFi applications.

- **Enhanced State Machine Model:** v0.12 extends RGB's state transition capabilities, supporting more complex conditional logic, timelocks, and multi-party coordination mechanisms. This enables BitLight to more easily implement advanced features such as atomic swaps, conditional payments, and automated contract execution.

- **Optimized Consignment Structure:** The new version significantly optimizes the structure of Consignment data packages, substantially reducing data transmission over-

head and improving verification performance, which is crucial for efficient data management in BitLight's "separation of operation and signing" model.

- **Improved Error Handling Mechanisms:** v0.12 introduces more robust error detection and recovery mechanisms, ensuring graceful handling of exceptional situations in complex multi-layer interaction environments (such as BitLight's L1-L2-LN integrated architecture).

Critical Significance for BitLight Architecture: The above v0.12 technical features directly support BitLight's two core design principles. The Turing completeness of zk-AluVM enables the contract state layer in "layered decoupling" to carry truly complex business logic; while the optimized Consignment structure provides the foundation for efficient data transmission in the "separation of operation and signing" model.

3.2.2 Core Paradigm: Client-Side Validation

The core idea of the RGB protocol is to transfer smart contract **state** and **verification** processes from the blockchain (global verifiable layer) to off-chain (client-side).

1. **State Commitments:** The existence and state of RGB assets are not directly recorded on the Bitcoin blockchain. Instead, a cryptographic commitment to the asset state is embedded in a Bitcoin UTXO's transaction output script. The owner of this UTXO is the owner of that RGB asset.

2. **Single-Use Seals:** Bitcoin's UTXO model naturally provides a "single-use seal" mechanism. When a UTXO containing an RGB commitment is spent, this "seal" is opened. For an RGB asset transfer to be valid, the transaction spending that UTXO must create a new "seal" (i.e., a new UTXO containing a state commitment) in its outputs and transfer asset ownership to it. This ensures that RGB asset transfers are synchronized with Bitcoin UTXO ownership changes, thereby preventing double-spending.

3. **Off-chain State Transition & Validation:** The core of an RGB asset transfer is a data package called a **Consignment**. This data package contains the complete state transition history from asset genesis to the current transfer, with cryptographic proofs. The sender transmits this Consignment directly to the receiver through peer-to-peer communication. After receiving it, the receiver's client independently verifies the validity of the entire historical record and accepts the transfer only after confirming accuracy.

3.2.3 Comparison with Global Consensus Models

3.2.4 Role in BitLight Architecture

RGB constitutes the **asset layer** of BitLight. Its client-side validation model is the key enabling technology for implementing our "separation of operation and signing" architecture. Since verification logic occurs on the user side, we can assign:

Table 1: Comparison of Global Consensus vs. Client-Side Validation Models

Feature	**Global Consensus (e.g., Ethereum)**	**Client-Side Validation (RGB)**
State Storage	Global state tree stored by all full nodes	Only commitments on-chain, state history held by asset owners
Transaction Verification	Executed and verified by all network nodes	Verified only by transaction receiver's client
Data Publicity	Transaction data (sender, receiver, amount) public	Transaction data not visible on-chain, transmitted only between participants
Scalability	Limited by global consensus throughput	Not limited by global consensus, theoretically higher limits

- **Consignment reception and verification** and other transfer processing tasks to the **operational component**.
- **Final verification and ownership changes (i.e., signing actions for spending old UTXOs and creating new UTXOs)** authorization is strictly reserved for the **signing component**.

This simplifies managing complex RGB asset states for end users to a simple signature confirmation action.

3.3 Technical Integration: Transferring RGB Assets in Lightning Network Channels

The BitLight architecture deeply integrates the RGB protocol with the Lightning Network to enable instant, low-cost transfer of RGB assets within payment channels (commonly referred to as RGB-LN).

1. Channel funding transactions not only lock bitcoin but also include **state commitments** for specific RGB assets in their output scripts.
2. Within channels, each state update not only redistributes BTC balances but also updates RGB asset allocation states. These updated state commitments are included in off-chain commitment transactions.
3. When an RGB asset is routed through the Lightning Network, in addition to HTLC forwarding, the sender must provide the final receiver with complete **Consignment data packages** for the receiver to verify the complete asset history.

Through this integration, BitLight can provide stablecoins and other non-BTC assets with speed and cost advantages equivalent to native Lightning Network payments, while all this complexity is encapsulated in the background and managed by our layered architecture.

4 Methodology: An Integrated Architecture for Bitcoin L2

4.1 Introduction

To address protocol-layer and user-layer challenges, this paper proposes a systematic integration architecture. The following sections elaborate on the two core design principles of this architecture: **Layered Decoupling** and **Functional Separation**, and define the **transactional state management model** designed to resolve the state consistency issues arising from this approach.

4.2 Design Challenges of Layered Decoupling

4.2.1 Consistency Challenges in Channel Operations

The aforementioned layered architecture faces a specific technical challenge when operating within Lightning Network channels: how to ensure consistency among multiple components involved in a single payment operation.

Specifically, when a user makes an RGB asset payment within a Lightning Network channel, the operation involves:

- Lightning Network layer HTLC state management
- RGB protocol layer asset state verification (based on Consignment history)
- Channel commitment transaction state updates

It is important to note that such channel payments do not immediately affect Bitcoin L1 UTXO states. L1 layer UTXO changes occur only when channels are opened or closed. Therefore, the consistency challenge here focuses primarily on coordination among multiple protocol components within channels.

4.2.2 State Management in Asynchronous Environments

Unlike traditional single-chain systems, BitLight architecture's state updates are asynchronous: Lightning Network state updates occur at millisecond scales, RGB client-side verification at second scales, while Bitcoin L1 final confirmation may require hours. This temporal difference requires the system to handle partial failures and rollbacks.

4.3 Security Analysis of Functional Separation

4.3.1 Design Principles of Permission Boundaries

The separation of operational and signing components in the BitLight architecture centers on establishing clear permission boundaries: the operational component handles all network activities requiring continuous online presence but is strictly prohibited from accessing any private keys; the signing component exclusively manages all private key operations and signing authorization functions but need not remain continuously online.

The key insight behind this design is that most L2 operation security risks stem from exposing signing permissions in continuously running, network-facing environments. By completely separating these two functions, the system achieves the high availability required for L2 applications without sacrificing user fund security.

4.3.2 Security Guarantee Mechanisms

Under the functional separation model, the system's overall security equals the signing component's security. Even if the operational component is completely compromised, attackers cannot:

- Directly access user private keys
- Bypass user authorization to execute fund transfers
- Tamper with transaction information presented to users (because the signing component has independent verification capabilities)

This "zero-trust" design ensures that user asset security remains protected even in worst-case scenarios.

4.3.3 Architectural Advantages for Availability

Functional separation simultaneously enhances system availability. The operational component can be deployed in high-availability cloud environments, handling network events 24/7, while users only need to activate the signing component when authorizing transactions. This avoids the limitations of traditional L2 solutions that require user nodes to remain continuously online.

4.4 Core Method: HTLC Transactional Processing Model

4.4.1 Consistency Challenges for HTLC and RGB Transfers

When conducting RGB asset transfers within Lightning Network channels, the system must ensure consistency between HTLC state changes and related RGB transfers. This is not a cross-layer state synchronization problem but rather an atomicity issue for handling multiple related components within the same operational context.

4.4.2 Two-Phase Execution Process

Computation Phase: The system first computes all necessary operations for HTLC state changes and related RGB transfers in an isolated environment without actually applying these changes. This phase generates a complete "change candidate set" containing all expected changes to HTLC states and RGB asset allocations. The key point is that this computation process does not affect the system's current state.

Commitment Phase: Only when all relevant parties (particularly Lightning Network counterparties) confirm acceptance of this change candidate set does the system apply HTLC and RGB transfer state changes atomically. If any problems occur during the confirmation

process (such as network interruptions or protocol errors), the system discards the entire change candidate set and maintains the original state unchanged.

4.4.3 Atomicity Guarantees

This design ensures atomicity of HTLC operations: an RGB asset Lightning Network payment either succeeds completely (both HTLC state and RGB asset allocation are updated) or fails completely (both maintain their original states). Through this approach, the system avoids inconsistent states between HTLCs and RGB transfers.

5 Architectural Support Analysis for Application Scenarios

5.1 Introduction

Through its layered design and operational model, the BitLight architecture provides systematic solutions for two types of application scenarios with specific technical requirements: **Autonomous Agent Economic Networks** and **Large-Scale Stablecoin Payment Systems**. The following sections decompose the core technical requirements of these two scenarios and analyze in detail how BitLight architecture's components and design principles systematically meet these requirements.

5.2 Infrastructure for Autonomous Agent Economic Networks

5.2.1 Technical Requirements Decomposition

Autonomous agent networks are distributed systems composed of automated software programs whose efficient operation depends on the underlying infrastructure meeting the following strict technical constraints:

- **State Transition Frequency & Cost:** The system must support extremely high frequency (theoretically unlimited) state transitions with marginal costs approaching zero for each transition. This corresponds to large-scale, automated service calls and micropayments between agents.
- **Settlement Latency:** State transition final confirmation time must reach millisecond levels to support complex service orchestration requiring instant feedback, such as scenarios where tasks are decomposed and completed sequentially by multiple agents in relay fashion.
- **Logical Programmability:** Value transfers must support complex, stateful conditional logic. For example, payment execution can be tied to external events (such as API calls successfully returning specific results) or timelocks, requiring the underlying system to have Turing-complete or near-Turing-complete contract capabilities.

5.2.2 BitLight Architecture Applicability Analysis

The BitLight architecture addresses these three mutually conflicting requirements by assigning them to different specialized layers through its **layered decoupling** design, thereby avoiding single-protocol bottlenecks.

- **High-Speed Transport Layer** meets high-frequency, low-cost state transition requirements
- **Contract State Layer** provides Turing-complete smart contract capabilities, supporting complex automated collaboration logic such as Service Level Agreements (SLAs), task decomposition, and dynamic revenue distribution mechanisms
- **Secure Settlement Layer** provides final settlement guarantees, ensuring all off-chain activities are anchored to the highest security ledger

Furthermore, the **functional separation** operational model is key to achieving agent **autonomy**. Agent business logic can be deployed on the **operational component**, enabling 24/7 autonomous handling of network interactions, state monitoring, and transaction proposal construction without exposing signing-authorized private keys to continuously running automated environments. This architecturally separates business logic from fund authorization, significantly enhancing overall system security.

5.3 Large-Scale Stablecoin Payment Systems

5.3.1 Technical Requirements Decomposition

A robust, large-scale adoptable stablecoin payment system must meet strict standards across the following four dimensions:

- **Performance:** The system's transaction throughput (TPS) and confirmation latency must compete with existing centralized payment networks (such as VisaNet).
- **Settlement Assurance:** Once confirmed, transactions must possess legal and technical finality and be irreversible.
- **Asset Security:** With potential total values reaching hundreds of billions of dollars, the underlying security must meet the highest standards and resist nation-state-level attacks.
- **Data Processing Model & Auditing:** The system must provide complete, verifiable transaction histories to participants and authorized third parties (such as regulatory agencies and auditors) without publicly broadcasting all transaction details.

5.3.2 BitLight Architecture Applicability Analysis

The BitLight architecture provides a comprehensive solution for building stablecoin payment systems that meet all the above requirements.

- **Performance:** High-speed transport layer provides high-TPS, low-latency payment channel networks
- **Settlement Assurance & Asset Security:** Secure settlement layer anchors stablecoin ownership to Bitcoin UTXOs, inheriting the highest security of proof-of-work consensus
- **Data Processing Model & Auditing:** Contract state layer's client-side validation model implements "selective disclosure," protecting transaction privacy while supporting regulatory audits. Transaction details are not publicly disclosed on-chain, but each transaction generates complete cryptographic proof history that can be provided to authorized third parties as needed.

5.4 Architecture Universality and Future Application Directions

Beyond providing specialized infrastructure for autonomous agent networks and stablecoin payment systems, BitLight architecture's layered design and programmability enable it to support broader application scenarios, providing a foundation for building an open Bitcoin-native application ecosystem. The following briefly explores several representative future application directions.

5.4.1 Decentralized Finance (DeFi) Services

The BitLight architecture provides necessary technical components for building more complex, non-custodial DeFi applications on Bitcoin.

- **Technical Mapping:**
  - **Contract State Layer (RGB Protocol)** Turing completeness allows developers to define and implement complex financial instruments such as decentralized exchange (DEX) order book logic, lending protocol liquidation engines, or financial derivative contract terms.
  - **High-Speed Transport Layer (Lightning Network)** provides high-performance trading environments for these DeFi applications, supporting high-frequency trading and instant settlement.
  - **Secure Settlement Layer (Bitcoin L1)** ensures that all assets involved in DeFi activities have their final ownership secured by Bitcoin network security guarantees.
- **Application Examples:**
  - **Trustless Asset Exchange:** Users can conduct instant, peer-to-peer trading of different RGB assets (such as stablecoins and digital tokens) within Lightning Network channels through atomic swap protocols without entrusting assets to any centralized exchanges.
  - **Collateralized Lending:** Protocols can be built where users lock one type of RGB asset (such as synthetic BTC assets) in contracts as collateral to borrow another type of RGB asset (such as stablecoins), with all liquidation logic automatically executed by client-side validated RGB contracts.

5.4.2 Digital Asset and Rights Management

RGB protocol is not limited to fungible tokens; its state machine model equally applies to defining and managing unique digital assets, namely Non-Fungible Tokens (NFTs).

- **Technical Mapping:**
  - **Contract State Layer (RGB Protocol)** can define unique states and specific state transition rules for each NFT. These rules can represent ownership of digital artwork, software license usage rights, or even digital identity credential (DID) attributes.
- **Application Examples:**
  - **Digital Content Monetization:** Content creators can tokenize their works (such as articles, music, videos) as RGB NFTs and receive instant micropayments from global users through Lightning Network. Each payment can be designed as one-time, time-limited access authorization managed by RGB contracts.
  - **Decentralized Identity & Credentials:** User education, professional qualifications, or membership status can be issued as non-transferable RGB NFTs held by users in their personal wallets. Users can achieve autonomous control over their personal data by selectively disclosing proofs of these credentials to third parties.

6 Conclusion

This paper addresses the dual challenges faced when building complex, scalable, and non-custodial Layer 2 (L2) applications on the Bitcoin network: the protocol-level "L2 design trilemma trade-offs" and the user-level "security-availability conflicts in non-custodial operations."

To address these challenges, this paper proposes and elaborates the BitLight system architecture. The core contribution of this architecture lies in its **systematic integration approach** rather than single protocol innovation. Through deep integration of existing Bitcoin-native technologies—namely the Taproot protocol at the L1 layer, and the RGB protocol and Lightning Network at the L2 layer—this architecture demonstrates a viable path for addressing the aforementioned dual challenges.

This paper demonstrates the architecture's two core design principles:

1. **Layered Decoupling:** Assigns the dimensions of the Bitcoin programmability trilemma to specialized technical layers for processing
2. **Functional Separation:** Decouples network operations from private key management, addressing user-layer security-availability conflicts

To address the state consistency challenges introduced by the layered architecture, this paper proposes a transactional state management model as a technical solution.

In conclusion, the BitLight architecture demonstrates a viable path for building L2 systems that balance security, programmability, scalability, and user-friendliness through systematic integration of existing Bitcoin-native technologies.

References

[1] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[2] Joseph Poon and Thaddeus Dryja. The bitcoin lightning network: Scalable off-chain instant payments. 2016.