Surge: A Composable MetaLayer on Bitcoin

tech@surge.build

December 2024

Abstract. We propose Surge, a **Composable Metalayer on Bitcoin** built to provide a secure network for launching Rollups, and dApps to scale Bitcoin innovation. Combining Bitcoin's robust security model with advanced cryptographic techniques like **Zero-Knowledge Proofs**. At the core of the architecture are zk-aggregation and decentralized verification mechanisms, which ensure scalability without compromising security. Surge serves as the interstitial metalayer for Rollups and dApps, inheriting Bitcoin's security and extending it to everything built on the network.

Surge's vision is establishing a metalayer that extends **Bitcoin's security** model to decentralized applications, protocols, and rollups. Surge offers a composable environment where developers can deploy rollups that interoperate seamlessly while unlocking **Bitcoin liquidity** and **scaling** capabilities.

1. Introduction

The Surge network adopts a modular approach, leveraging the best practices and lessons from other ecosystems. Inspired by the Cosmos modular design, Surge enables decentralized proof validation through CometBFT for Byzantine Fault Tolerant (BFT) consensus across the network, offering high performance and security. The Surge Integration Kit empowers developers to integrate **Zero-Knowledge Proofs** (ZKPs) and deploy secure rollups and dApps, inheriting Bitcoin's robust security guarantees. This approach provides a reliable foundation for key network components:

1. **Verifiers**: Dedicated nodes for zk-proof verification, Surge verifiers are integral in aggregating and settling validated proofs on Bitcoin.

2. **Integrated Signers**: These handle transaction signing and manage distributed cryptographic keys, governing BTC vault operations with precision and reliability.

3. **UTXO Indexers**: Open to anyone, Indexers ensure transparent and seamless data retrieval, critical for enabling real-time audits across the network.

2. Bitcoin-Native Metalayer

2.1 Why a Native Metalayer is Needed for Bitcoin

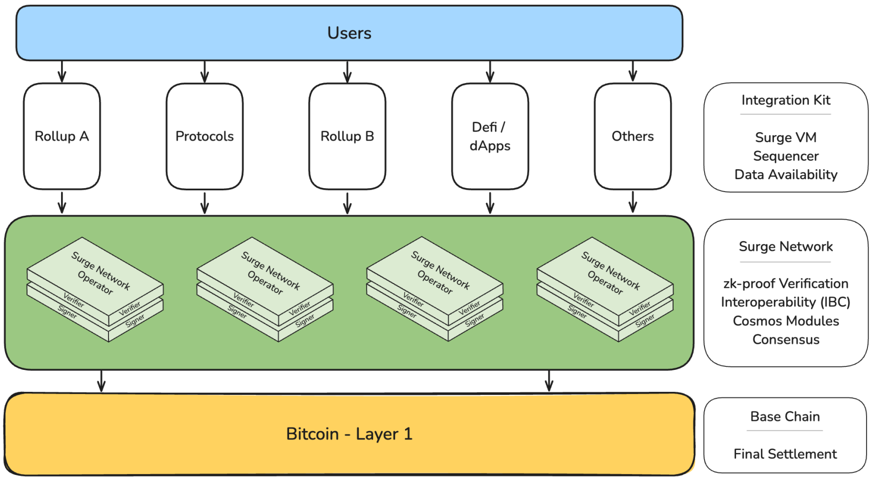
Bitcoin's security and decentralization have established it as the foundation of the crypto economy. However, its limited scalability and programmability have sparked a race for optimal scaling solutions. Many existing approaches, such as protocol-level modifications (e.g., OP_CAT) and inscription-based solutions, risk bloating Bitcoin's state.

The Surge Metalayer overcomes these challenges without requiring a soft fork to Bitcoin. By leveraging off-chain aggregation with recursive zk-proofs, it enhances throughput while minimizing on-chain data requirements. This approach preserves Bitcoin's core attributes and creates an interoperable execution layer.

2.2 Composability and Interoperability

As multiple solutions attempt to scale Bitcoin, fragmentation across systems becomes a growing challenge. Surge offers composable infrastructure through Inter-Blockchain Communication (IBC) and facilitates seamless interaction with other blockchain ecosystems, enabling the transfer of assets, data, and zk-proofs. This composability empowers developers to integrate functionalities from different blockchains into unified applications, significantly enhancing both flexibility and interoperability.

The Surge architecture allows developers to leverage Bitcoin's robust security while building rollups that interact with other chains. This creates a multi-layered, interoperable environment where decentralized solutions can be easily composed, addressing fragmentation and fostering a cohesive ecosystem.



Surge Network Overview

3. Verifiers

Verifiers are essential to securing the Surge network by validating zk-proofs that represent transaction histories. Using off-chain aggregation, they consolidate multiple proofs into recursive succinct zk-proofs, reducing on-chain costs and enhancing scalability. By posting only minimal data, such as state commitments and aggregated signatures, Surge maintains high throughput and data availability. Periodic commitments to Bitcoin anchor these state transitions within its Proof-of-Work, ensuring finality. Verifiers are rewarded for their contributions, making them integral to Surge's decentralized consensus

3.1 ZK Aggregation and Verification Layer

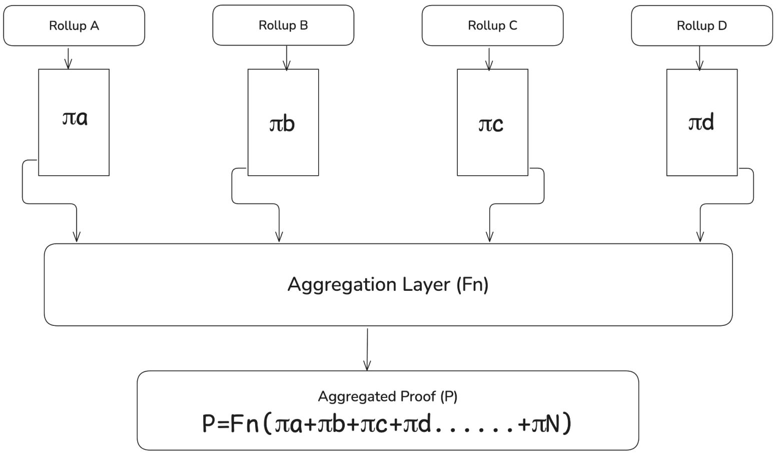
To transcend the limitations of conventional rollups, which requires each transaction to be submitted to the base layer, Surge employs zk-aggregation. This process compresses multiple proofs (n proofs) into a single concise proof (P_{agg}), significantly reducing computational overhead and on-chain fees. This aggregation also enhances throughput by enabling the verification of entire rollup histories using a single cryptographic proof.

3.1.1 zk-Aggregation Technique

Surge's zk-aggregation compresses multiple zk-proofs into a single proof that can be efficiently verified on-chain.

$$P_{agg} = Agg(\pi_1, \pi_2, ... \pi_n)$$

Here, π_1, $\pi_2 ... \pi_n$ are the individual zk-SNARK or zk-STARK proofs generated for each transaction batch, and Agg is the aggregation function that compresses these into a single proof P_{agg} that can be succinctly verified on-chain.



Proof Aggregation

By minimizing the verification load, Surge reduces the gas fees and enhances the finality speed of zk-proofs submitted to the network.

3.1.2 Hashing for State Commitments

To ensure integrity and immutability, Surge produces a state commitment that reflects the new state of the system. This is done using a Merkle tree, where each leaf represents the hash of a transaction with zk-proofs submitted to the chain, and the root represents the commitment to the entire set of transactions.

Let $hx(T_i)$ be the cryptographic hash of transaction T_i. The Merkle root M_{root} for n transactions is computed as:

$$M_{root} = MerkleRoot(hx(T_1), hx(T_2), ..., hx(T_n))$$

This root is then inscribed onto Bitcoin, providing a secure and verifiable state commitment.

3.2 Anchoring to Bitcoin Security

Surge leverages Bitcoin's security by anchoring state commitments and zk-proof aggregations to Bitcoin. This is achieved using a Commit-Reveal Framework, embedding the Merkle root of Surge's state and aggregated proofs in a Bitcoin UTXO where it can be validated using basic Bitcoin Script operations. Once confirmed, this provides verifiable proof that Surge's state existed and is backed by Bitcoin's consensus.

Key Security Components:

- **State Commitment & Proof Aggregation:** A Merkle root (Mroot) and aggregated zk-proof (Pagg) are embedded into Bitcoin as $UTXO_{Surge}$.

- **On Chain Validation**: Submitted proofs are inscribed onto Bitcoin as $UTXO_{Surge}$, and it can be verified on-chain using Bitcoin Script validations.

- **Bitcoin Inscription:** Each Surge state is linked to a specific Bitcoin block, inheriting Bitcoin's Proof-of-Work integrity. This results in probabilistic finality, where the likelihood of reversal diminishes exponentially with each confirmation.

- **Timestamp Integrity:** Bitcoin's native timestamping provides verifiable time references for each block. Every Bitcoin block includes a timestamp t_{block}, which is used to record the time of the Surge state commitment. This timestamp can be verified by referencing the block height h_{block} where the transaction $UTXO_{Surge}$ was included.

$$t_{commit} = t_{block}(h_{block})$$

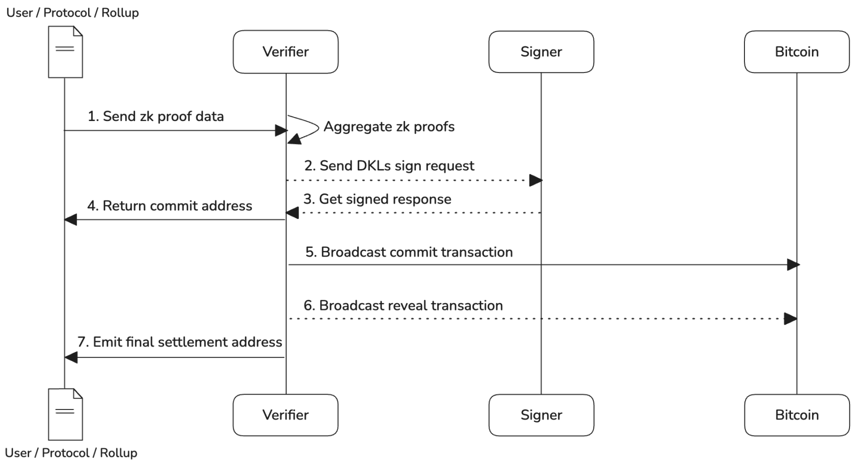
Participants can verify that the Surge state existed at or before t_{commit} by verifying the inclusion of $UTXO_{Surge}$ Bitcoin's blockchain t_{commit}.

Security Inheritance:

- **Probabilistic Finality:** The probability of a block reversal decreases exponentially with each subsequent block confirmation:
  $P(reversal) \approx (p/(1 - p))^\wedge n$
  Where p is the proportion of hostile mining power, and n is the number of confirmations.

- **Computational Security:** The cost of altering a block rises exponentially with the length of the chain since that block, further strengthening the integrity of the anchored data

By anchoring Surge's state commitments and aggregated proofs to Bitcoin, Surge inherits Bitcoin's finality and timestamp integrity. This security is based on the cryptographic robustness of zk-proofs and Bitcoin's Proof-of-Work (PoW) consensus, providing a verifiable solution for ensuring finality and precise time references.



Commit-Reveal framework journey

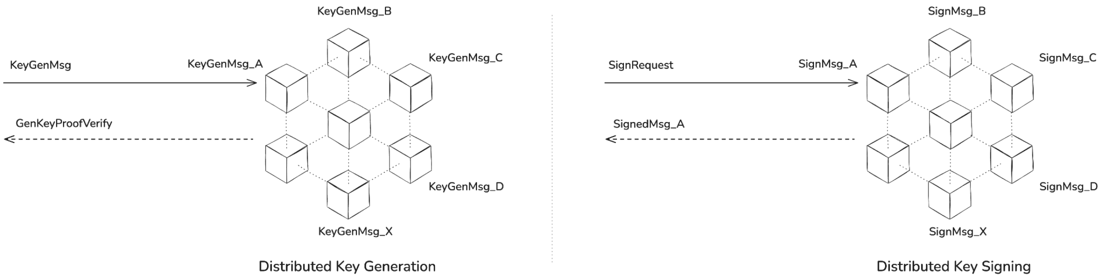
4. Signers

The **Signers** manage decentralized signing operations using the **DKLs23 EdDSA/ECDSA** scheme, ensuring secure, decentralized management of cryptographic keys for inscribing zk-proofs into Bitcoin.

DKLs23 is preferred for its use of **Oblivious Transfer-based Multiplication (OT Mul)**, which significantly reduces computational overhead and latency compared to traditional **Paillier-based** methods. This efficiency enables faster key generation and signing, making DKLs23 highly scalable for large decentralized networks.

4.1 Decentralized Signing and Inscription

- **Decentralized Key Management**: A **quorum-based signing process** of nodes is required to sign a transaction, ensuring a decentralized approach to key management.



DKLs Key Management Flow

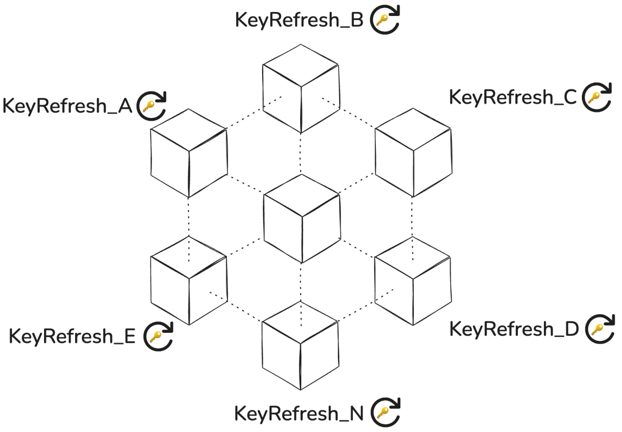
- **BTC Vault & Distributed Signers**: Signers manage BTC vault and Bridge operations, enabling secure cross-chain transfers between Bitcoin and other rollups. The bridge utilizes peg-in and peg-out processes, where users lock their BTC in the vault to access its value on another blockchain. The core of this system is the **DKLs23 signature scheme** combined with a **TSS decentralized network**, where cryptographic key management and signing are distributed across multiple **Signers**. This ensures that no single entity controls the vault, enhancing security and decentralization. The DKLs23 scheme's use of advanced cryptography reduces latency and computational overhead, allowing efficient and secure authorization of BTC releases from the vault.

4.2 Security and Resilience via TSS Network

The Signer Node operates within a standalone **TSS (Threshold Signature Scheme) network**, ensuring that no single node controls the signing process and utilizes **M of N consensus** to securely manage key shares and signatures.

- **Distributed Key Generation (DKG)**: The key generation is distributed across the TSS network, ensuring that no single party has access to the entire private key.

- **Proactive Key Refresh**: The system includes regular key rotation and refresh processes, ensuring that keys are never permanently exposed and remain secure even in the face of node compromise.



Proactive Key Refresh

- **Weighted Nodes:** Nodes are assigned weights based on factors like stake, reputation, performance, or governance roles

    - **Prioritized Signing:** Higher-weighted nodes are prioritized in the signing process, allowing for faster consensus and sig generation.

    - **Reduced Latency:** Reliable and fast nodes with higher weights can reduce overall latency in the TSS signing process

    - **Security and Fault Tolerance:** High-weighted nodes improve security by being more accountable and enhance fault tolerance by ensuring progress even if some nodes fail.

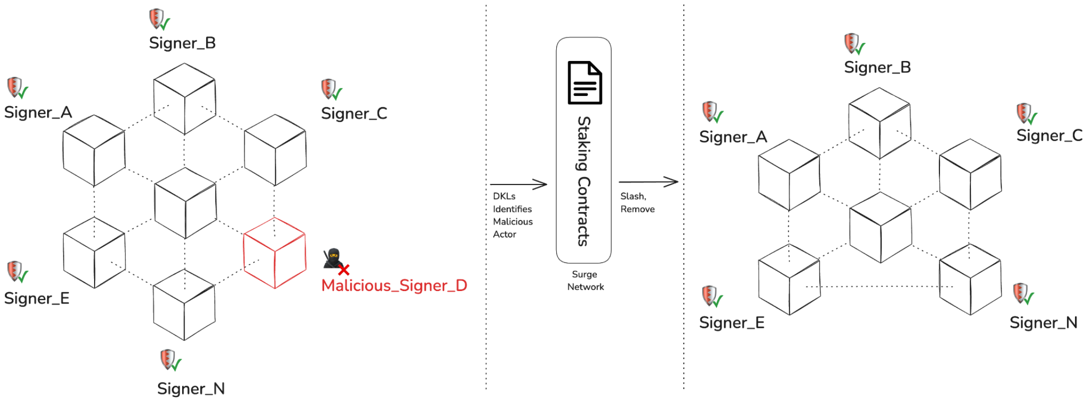
- **Malicious Node Detection:** The DKLs23 protocol includes mechanisms to detect and mitigate malicious nodes during keygen and threshold signing

    - **Consistency Checks:** The protocol checks message consistency, allowing nodes to detect and flag deviations from expected behavior.

    - **Audit Trails:** Actions and messages are logged, enabling the identification of malicious nodes through post-event analysis.

    - **Complaint Mechanism:** Nodes can file complaints against suspicious behavior, leading to the exclusion of confirmed malicious nodes.

- **Stake-Based Security and Slashing:** In the event of identifying a malicious actor, the network employs economic penalties through slashing. Validators are required to stake cryptocurrency as a security bond. If they engage in malicious or faulty behavior, a portion of their stake is slashed, serving as a deterrent and maintaining the network's security and integrity.



Economic Guarantee of TSS Network

5. UTXO Indexer

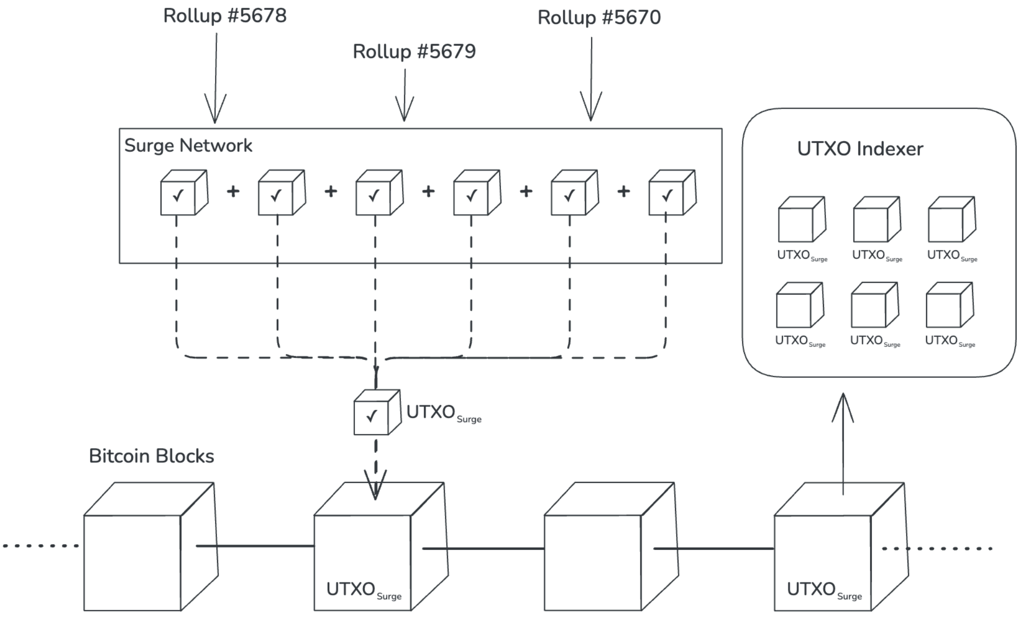
To enhance transparency and enable auditing of the Surge Network's data achieving Bitcoin finality, we have a specialized Unspent Transaction Output (UTXO) Indexer. This Indexer systematically scans the Bitcoin blocks, focusing exclusively on transactions associated with the Surge Network—specifically, UTXOs containing unique identifiers.

The Indexer integrates with a Bitcoin node to access real-time blockchain data. It monitors newly mined blocks and parses each transaction to detect Surge identifiers embedded within transaction data, such as encoded within **scriptPubKey**. By filtering out irrelevant transactions, the Indexer optimizes performance and resource utilization.

When a transaction with a Surge identifier is detected, the Indexer extracts key information, including the sender's public key P_{sender}, signature S, and the associated data blob B. The authenticity and integrity of the data are verified by checking the signature:

$$Verify(P_{Sender}, S, H(B)) = True$$

Where, $H(B)$ is the hash of the data blob. The Indexer updates the set of Surge-related UTXOs, continuously tracking both new and spent outputs. This UTXO set $UTXO_{Surge}$ is maintained in an optimized database for efficient management and quick retrieval.



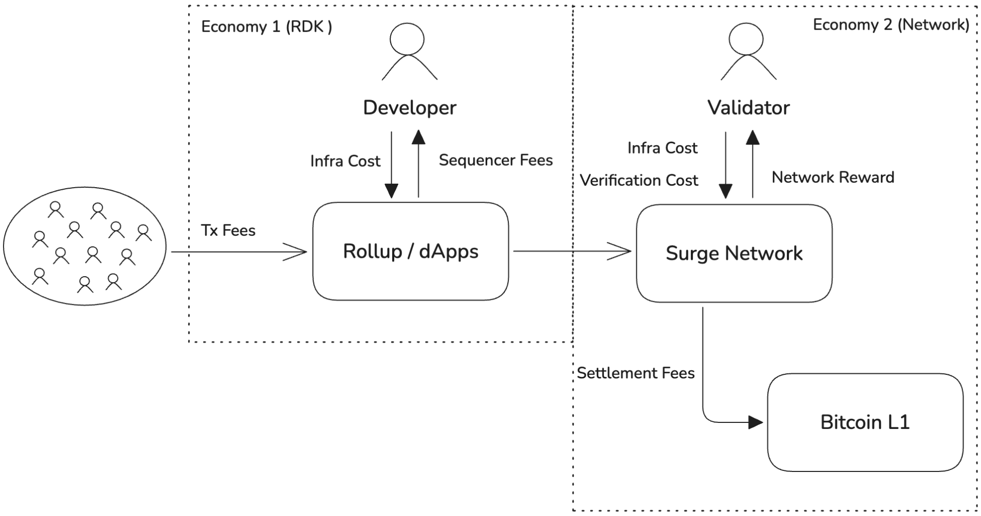
UTXO Indexer Flow

5.1 OP_CAT Proposal and On-Chain Validation

The proposed reintroduction of OP_CAT enables on-chain proof validation directly within Bitcoin. With OP_CAT, critical parts of transaction and proof data can be concatenated using opcodes within Bitcoin Script itself, allowing participants to reconstruct and verify aggregated proofs against the committed data stored in UTXOs.

6. Network Economics

The Surge Network employs a Delegated Proof-of-Stake (dPoS) consensus mechanism with the SURGE token as its native utility asset. This economic model aligns validator incentives with network security and performance.



Network Economics

6.1 Validator Economics

The dPoS system operates with a fixed maximum validator set N_{max}. Each validator must stake a minimum of SURGE tokens S_{min} to participate. Validator rewards R are a function of transaction fees and block rewards. A slashing condition Slash(v) is applied if a validator v exhibits malicious behavior or experiences downtime exceeding a threshold $T_{threshold}$.

6.2 SURGE Token Utility

SURGE serves as the gas token for all network transactions. To enhance user experience, a native swapping mechanism allows fee payment in BTC, automatically converting to SURGE at a rate determined by current prices and liquidity pool states:

$$swap\_rate = f(BTC_{price}, SURGE_{price}, liquidity\_pool\_state)$$

6.3 Fee Distribution

Transaction fees are collected and distributed among validators proportional to their staked amount. This distribution model incentivizes higher stake amounts and consistent performance, promoting network stability and security.

This economic structure encourages active participation, maintains network integrity, and supports the sustainable growth of the Surge ecosystem.

7. Security & Trust Assumptions

7.1 Zero-Knowledge Proof Security

Surge relies on zk-proofs to ensure the correctness of off-chain computations while preserving privacy. Each rollup submits zk-proofs summarizing batches of transactions, which the network verifies before inscribing them on Bitcoin. This approach guarantees secure state transitions without compromising scalability or revealing sensitive transaction details.

7.2 Threshold Signature Scheme (TSS) for Decentralized Signing

The Threshold Signature Scheme (TSS) distributes signing authority across multiple nodes, ensuring no single node controls the entire key. This decentralized approach prevents key compromise by requiring a quorum of nodes to generate valid signatures for rollup transactions and BTC vault operations. The TSS enhances security by eliminating single points of failure in the signing process

7.3 Byzantine Fault Tolerance (BFT) in Verifier Nodes

Byzantine Fault Tolerant (BFT) consensus ensures that the network remains operational even if up to one-third of nodes act maliciously. Surge implements a BFT algorithm that allows honest nodes to reach consensus despite the presence of Byzantine (malicious or faulty) nodes, maintaining network integrity and reliability.

7.4 DKLs Key Refresh

Regular key refresh in the **TSS network** prevents long-term exposure of cryptographic materials. Even if a subset of nodes is compromised, new key shares are frequently generated, maintaining security across the network.

7.5 Slashing for Malicious Actors

Malicious behavior, such as submitting incorrect zk-proofs or failing to participate in consensus, results in slashing of staked tokens. This economic penalty deters dishonesty, ensuring verifiers act in the best interests of the network.

7.6 Trust Minimization with Zero-Trust Architecture

Surge follows a **zero-trust architecture**, where no single participant is trusted by default. Security is maintained through cryptographic proofs for all state transitions, decentralized consensus mechanisms, and continuous verification of all network operations. This approach significantly reduces reliance on central authorities and minimizes trust assumptions.

8. Conclusion

Surge Network establishes itself as the **Bitcoin-native composable metalayer**, designed to scale Bitcoin by enabling rollups with zk-aggregation, decentralized verification, and robust signing and indexing modules. Surge provides developers with a high-performance framework, utilizing Bitcoin's unparalleled security for building scalable decentralized applications. By combining the scalability and privacy of zk-rollups with Bitcoin's security, Surge positions itself as the preferred platform for developers seeking to build advanced, scalable solutions in the Bitcoin ecosystem.

9. Future Work and Research required

As Surge continues to evolve as the Bitcoin native metalayer, several areas of future work and research have been identified to further enhance the capabilities and reach of the ecosystem. These areas represent both the immediate next steps and the long-term vision for Surge, aiming to address current limitations and unlock new possibilities for decentralized applications.

1. **Cross-Rollup Communication:** Develop enhanced Inter-Rollup Communication Protocols to enable seamless data and asset transfers between rollups on the Surge network, improving interoperability and liquidity sharing.

2. **Advanced ZK-Proof Aggregation Techniques:** Continue enhancing zk-proof aggregation to reduce on-chain costs, exploring recursive proof

techniques that further optimize scalability for high-throughput applications.

3. **Decentralized Sequencing:** Explore decentralized sequencing methods to optimize transaction ordering and throughput, reducing reliance on centralized entities and improving network resilience against censorship.

4. **Bitcoin Security Integration:** Leverage Bitcoin's robust proof-of-work security to further enhance Surge's settlement layer, ensuring that all rollup activities inherit Bitcoin's immutability and trust model.

5. **Minimizing Trust Assumptions:** Develop mechanisms to minimize trust assumptions, focusing on cryptographic guarantees and zero-trust architecture to eliminate reliance on centralized entities while ensuring decentralized validation and security.

References

1. DKLS23. *DKLS Signature Scheme Documentation*. Available at: https://dkls.info, 2023.
2. Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. *Threshold Signature Schemes*. Cryptology ePrint Archive, 2023. Available at: https://eprint.iacr.org/2023/765.pdf.
3. Bitcoin Optech. *OP_CAT: Bitcoin Opcodes Overview*. Available at: https://bitcoinops.org/en/topics/op_cat/, and Bedlam Research, 2024. Available at: https://www.bedlamresear.ch/posts/script/.
4. Jae Kwon, Ethan Buchman. *Cosmos: A Network of Distributed Ledgers*. Cosmos Whitepaper, 2020. Available at: https://github.com/cosmos/cosmos/blob/master/WHITEPAPER.md.
5. CometBFT. *CometBFT Documentation*. Available at: https://docs.cometbft.com/, 2024.
6. Polyhedra. *How to Verify ZK Proofs on Bitcoin?* Available at: https://hackmd.io/@polyhedra/bitcoin, 2024.
7. BitVM.org. *SNARK Verifier in Bitcoin Script*, 2024.
8. Protocol Labs. *The Future of ZK Proofs*. Available at: https://protocol.ai/protocol-labs-the-future-of-zk-proofs.pdf, 2023.