MultiversX

Bitcoin

Powered by
MultiversX Sovereign Chains

Bitcoin^X - A Bitcoin Layer 2 Proposal Using MultiversX Sovereign Chain Technology

Abstract

Bitcoin Layer 2 solutions, such as the Lightning Network, Stacks, and Rootstock, aim to enhance Bitcoin's scalability and functionality by processing transactions off-chain. These solutions address Bitcoin's limitations, including slow transaction speeds, high fees, and limited smart contract capabilities.

However, common issues persist, such as reliance on centralized entities, complex interoperability, and security vulnerabilities.

We propose building a Bitcoin L2 solution using MultiversX Sovereign Chains. Such a Bitcoin L2 can achieve unprecedented scalability and programmability without compromising its core principles of decentralization and security.

Sovereign Chains enable native cross-chain operations, fast finality, and complex interoperability with other blockchain networks, including Ethereum. This architecture addresses the limitations of existing Bitcoin L2 solutions and provides a future-proof platform for developers to build advanced decentralized applications and financial systems on Bitcoin.

A look at the current Bitcoin Layer 2 space

Current Bitcoin L2 solutions scale Bitcoin by moving computation off-chain and posting some data on the mainchain. However, Bitcoin nodes cannot verify the correctness of this data due to the lack of a VM. Bitcoin scripts and verifiable computation algorithms can enable inheriting Bitcoin security for BTC_SovereignChains.

Bitcoin L2s create an off-chain execution environment that computes transactions and executions, then submits the resulting data/root hash to the Bitcoin network's consensus. The Bitcoin network does not validate these off-chain processes, only recording the final settlement information, which can be as small as a root hash. Some projects use watchtowers to monitor execution correctness and detect suspicious activity. BTC spending can be scripted via Bitcoin Scripts, and OP_CAT can reduce computational costs for operations returning to the Bitcoin network.

Citing from **BIP-420**:

> ***"OP_CAT expands the toolbox of the tapscript developer with a simple, modular, and useful opcode. It enables:***
>
> - **Bitstream for atomic swaps of bitcoins for decryption keys, making decentralized file hosting systems paid in Bitcoin more practical.**
> - **Tree signatures for multisignature scripts, enabling complex spend conditions with smaller transaction sizes.**
> - **Non-equivocation contracts to punish double spending in Bitcoin payment channels.**
> - **Replicating CheckSigFromStack for creating simple covenants and advanced contracts without presigning spending transactions."**

Even without OP_CAT, the first two features can be used, albeit at higher costs. With OP_CAT, efficiency improves significantly.

Bitcoin L2 Models

1. **State Channels:**
   Like the Lightning Network, limited to transfers between defined users.
2. **Blockchain Rollup:**
   Separate execution, compress data for each block, and settle on Bitcoin.
3. **Sidechains:**
   Run a separate chain connected via a third-party bridge controlled by a multiSig.

These solutions run separate chains and periodically settle data on the Bitcoin network. SovereignChains can adopt these models, enhancing interoperability and security by inheriting Bitcoin's network security.

Additional notes on other solutions:

BitVM proposes general computations verifiable on Bitcoin, but this is currently inefficient.
MAP protocol offers modules for receiving and sending BRC-20 tokens.

We propose a Bitcoin Layer 2 solution based on MultiversX SovereignChains, which would achieve the following:

- **High-performance Secure Proof of Stake chain**
- **Integrate multiple interoperability/composability modules and VMs.**
- **Process over 50,000 transactions per second on inexpensive machines.**
- **Native cross-chain operation module towards MultiversX mainnet.**
- **Integrate a native bridge/rollup solution towards Bitcoin in the SovereignChain stack.**

Two-Way Peg Bridge

1. Build a bridge enabling Bitcoin to be staked on SovereignChain alongside EGLD from the Gravity ReStaking Layer.

2. Bridged BTC is wrapped into wrappedBTC, pegged 1-to-1 with BTC locked in the 2WP MPC wallet.

3. Assets coming to the MultiSignature wallet are cached by Bitcoin light clients run by all SovereignChain validators. Validators include Bitcoin block headers in Sovereign blocks. If a Bitcoin header contains a UTXO moving BTC/BRC-20 assets to the MultiSig address, validators create an IncomingTX and add it to the SovereignChain block.

Process

- If the leader does not add the IncomingTXs, the block is not signed by validators

- The next honest leader adds the IncomingTX. Each validator generates the IncomingTX separately, ensuring the same data and order

- Considering Bitcoin's Proof of Work and finality concept, validators push IncomingTXs only for finalized blocks (6-10 confirmations)

Settlement

- Create settlement of processing and data to both MultiversX and Bitcoin

- Use an optimistic rollup with fraud proofs. In case of fraud, first slash staked Bitcoin, then staked EGLD.

- The 2WP MPC wallet is controlled by SovereignChain validators, similar to the native cross-chain operation module towards MultiversX mainchain.

Outgoing Operations

- Deploy a Bitcoin Outgoing bridge SC: Create an outgoing miniblock UTXO data, signed by each node with their multiSig share.

- Use a multiSig generator algorithm to create a new multiSig at the end of each epoch for the new set of validators.

- Post the resulting full TX on the Bitcoin chain.

Integrated MPC

- SovereignChain creates the root hash and OP rollup data on the Bitcoin chain.
- Integrated MPC at the validator set level eliminates the need for third-party bridges, achieving high economic security through Gravity Layer ReStaking and Bitcoin Staking.

Additional topics are addressed: validator set change for the SovereignChains, economic security of the funds coming from the Bitcoin network and inheriting the Bitcoin networks security for all the user interactions.

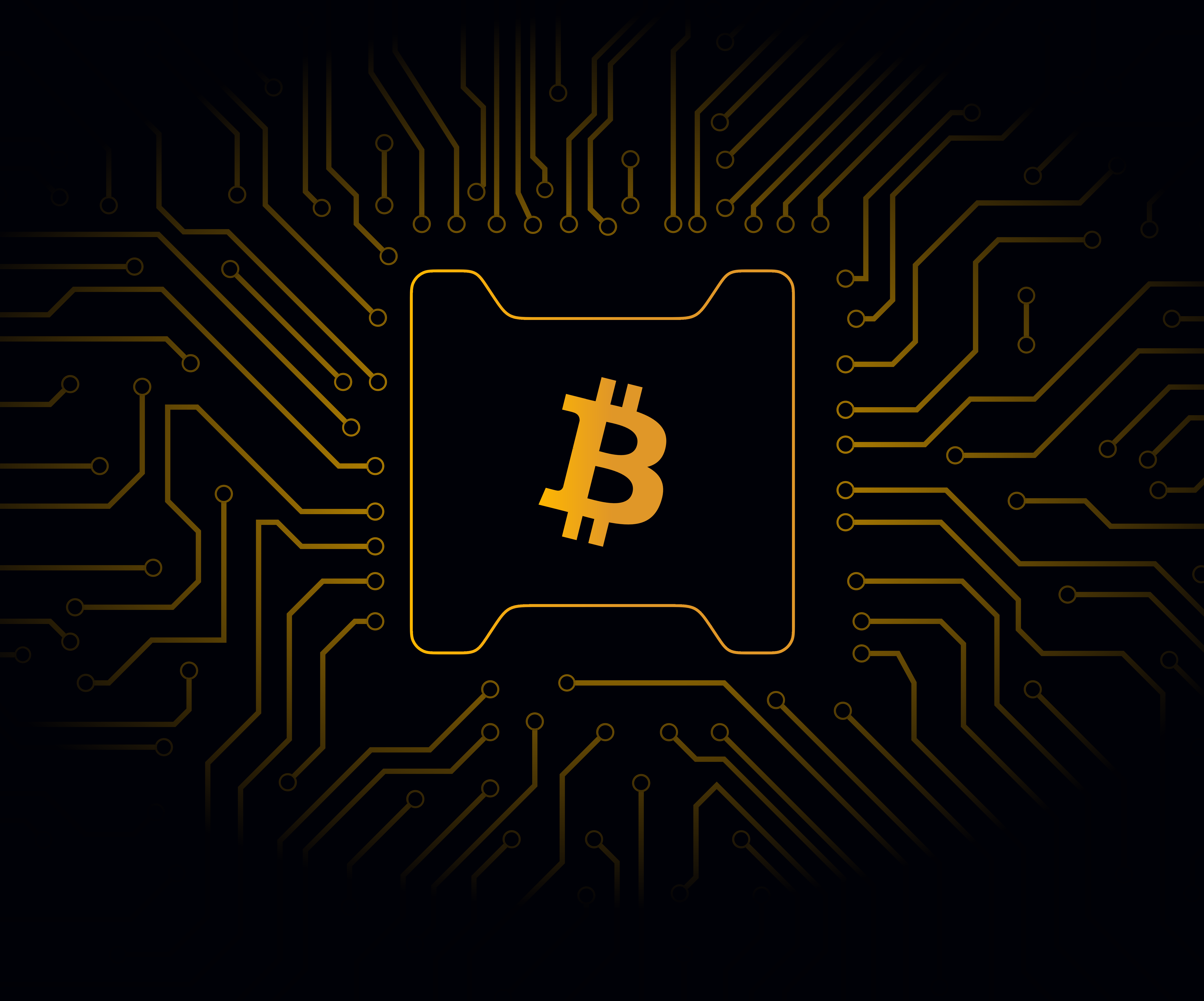
- The basic idea is to add rules of how the funds can be spent from the MultiSignatureWallet with the integration of spending routes through the Bitcoin scripts and how to make the bridging out procedure more safe for every participant.
- In case of Ethereum L2s, have the possibilities of exit roots through providing Merkle proofs of funds and getting back their funds towards the L1 without the need of specific signatures from the MultiSig of L2 contracts. But that is hard to do on Bitcoin as there is no VM.
- So a few ideas: only those nodes which have staked enough Bitcoin will be the participant on the MPC, and but they can post only data which is signed by all the validators from the SovereignChains, otherwise they are slashed in the Sovereign chains.
- Using trie signatures from all/selected set of validators spend conditions can be written for every outgoing operation, which is actually all the user needs to claim his tokens back on the Bitcoin network.
- Rotation of MultiSignature keys is an open problem when the SovereignChange validator set is changing. One way of doing it is to move all the UTXOs from the current MultiSig address into the new MultiSignature address, by having the current validators sign the transactions to move the UTXOs after a big set of validator set changes.

Spending Conditions

- Use Bitcoin scripting capabilities to define how SATs/BTCs can be spent, even inside a state channel.
- Change outgoing UTXO into a spending condition with trie signatures after enabling OP_CAT.

Raising Bitcoin to the power of MultiversX

Bitcoin builders and lovers - you long have championed the principles of decentralization, security, and trust. We invite you to explore MultiversX Sovereign Chains. This tech can expand and scale a programmable internet value system based on Bitcoin, without compromising its foundational values.



We call on like minded developers, builders and visionaries to pick up these tools and create something amazing.

It’s time to build!