Article | Received 31 October 2024; Accepted 10 January 2025; Published 11 February 2025
https://doi.org/10.55092/blockchain20250003

Zunesha: enhancing throughput of blockchains through relayer-free multi-chain architecture

Pengze Li[1], Jieyi Long[2], Qiuyu Ding[1], Zhen Xiao[1,*] Zhenxing Hu[1], and Shengjie Guan[1]

[1] School of Computer Science, Peking University, Beijing, China
[2] Theta Labs. Inc, San Jose, USA

* Correspondence author; E-mail: xiaozhen@pku.edu.cn.

Highlights:

- Portable and relayer-free multi-chain architecture – Zunesha enables seamless multi-chain scalability upgrades for existing blockchains using a smart-contract-based toolkit, eliminating relayers to enhance security and efficiency.
- Dynasty-Based Consensus Node Set Verification (DB-CNSV) Protocol – Zunesha introduces the DB-CNSV protocol to mitigate timing issues and ensure consensus consistency across dynamically changing node sets in inter-chain transactions.
- Scalability and performance optimization – Zunesha achieves near-linear throughput scalability with the number of subchains, reducing inter-chain transaction latency and outperforming cosmos in efficiency.

Abstract: Web3 is the next-generation internet, utilizing blockchain technology to power decentralized applications and give users greater control. However, the scalability limitations of blockchain create performance bottlenecks that hinder Web3's overall processing capabilities. Among current scalability solutions, multi-chain architecture has been considered a promising approach with high flexibility. However, current multi-chain architecture lacks *portability* to existing blockchains and relies on relayers to solve *timing issues* in the interoperability process. The lack of portability makes it challenging for existing blockchains to adopt the current multi-chain architecture, significantly impeding multi-chain promotion. Moreover, relying on relayers to address timing issues leads to low efficiency and potential reliability risks. This paper introduces *Zunesha*, a multi-chain architecture that designs a smart-contract-based multi-chain toolkit (STACK) to provide a portable multi-chain architecture. Additionally, it introduces the Dynasty-Based Consensus Node Set Verification (DB-CNSV) protocol as a foundational safety mechanism to eliminate relayers in the interoperability process and address timing issues. Our evaluation shows that Zunesha significantly enhances the overall performance of the blockchain. As the number of subchains increases, the throughput grows almost linearly. Furthermore, the performance of inter-chain transactions surpasses that of the current mainstream multi-chain architecture, Cosmos.

Keywords: Distributed systems; performance and reliability; network communications; blockchain; scalability; multi-chain; interoperability.

1. Introduction

In recent years, blockchain technology has seen remarkable growth, driven by emerging trends like the Metaverse [1] and smart contract platforms [2]. As the foundational infrastructure of Web3 [3], blockchain

plays a critical role in enabling decentralized, trustless applications and services. However, permissionless blockchains like Ethereum [2] often suffer from congestion due to limited transaction throughput, creating a significant bottleneck. This scalability challenge is particularly problematic for Web3, which demands high performance to support decentralized applications on a global scale. Despite years of development, overcoming blockchain's scalability issues remains challenging due to the scalability trilemma [4], which imposes fundamental constraints on the design of scalable solutions.

Various blockchain scaling solutions have been proposed, including blockchain storage layer optimization [5–7], sharding [8,9], Rollup [10], sidechains [11], and multi-chain architecture [12–16]. Among these techniques, the multi-chain architecture offers the advantage of high flexibility. In this design, one blockchain, referred to as the *mainchain*, serves as the primary network to be scaled, while its throughput is enhanced by incorporating multiple *subchains* to form a multi-chain system, offloading transactions from the mainchain to subchains. Furthermore, subchains can be tailored to meet specific requirements, allowing customization of their consensus models, block generation times, and the integration of additional scalability solutions like Rollup. Meanwhile, multi-chain interoperability also supports inter-chain transactions, further enhancing the functionality and diversity of blockchain applications.

Multi-chain architecture presents a promising approach to enhancing blockchain scalability and flexibility. However, it introduces several research challenges in developing a portable, relayer-free multi-chain architecture. The first challenge is to address the need for portability when designing a multi-chain architecture. Since Web3 is expected to remain continuously online, integrating a multi-chain architecture into existing blockchains must be done without disrupting their ongoing operations—comparable to the challenge of "changing an engine while the plane is flying." This objective necessitates the multi-chain architecture to be compatible with the existing blockchain without interrupting their ongoing process, whereas deploying this solution should only involve a "chain upgrade". However, portability is a key aspect often overlooked by prominent multi-chain architecture such as Cosmos [13], Polkadot [12], and Avalanche Subnet [14]. These architectures assume that the multi-chain system is built from the ground up. Consequently, it would be difficult for an existing blockchain to integrate with their frameworks without substantially overhauling its original structure. This lack of portability hinders the widespread adoption of multi-chain architecture.

The **second** challenge is addressing timing issues in a multi-subchain system in a decentralized manner, thereby enabling a relayer-free inter-chain transaction mechanism. In an inter-chain transaction involving two blockchains, the consensus nodes (**CNodes**) of each blockchain must be capable of independently verifying the latest consensus nodes of the other blockchain. This is essential to ensure the integrity of the inter-chain transaction and prevent malicious activity or fraudulent transactions across the blockchains. Note that the CNode set of a blockchain may change over time, which makes the problem even more complex. Therefore, it is essential for CNodes to periodically update their views of the latest legal CNode set of the blockchains that have transactions with them. However, due to the nature of distributed systems, two CNodes on the same blockchain may obtain different views of the latest CNode set. This inconsistency potentially leads to a series of problems. These problems are called *timing issues*, which are present in all multi-chain systems and can risk their safety and liveness.

Current multi-chain architecture relies on relayers like Polkadot's Relaychain [12] and Cosmos' Hub [13] to enable interoperability and address timing issues. While practical, relayers introduce performance overhead and safety challenges. A relayer is an off-chain process that facilitates the transfer of inter-chain transactions from one blockchain to another. However, this additional step can negatively affect inter-chain transactions' security, throughput, and perceived latency. Eliminating relayers from the interoperability process is non-trivial, as it requires ensuring that inter-chain transactions are completed in a decentralized manner without compromising the integrity and security of the multi-chain architecture.

To address these challenges, we propose *Zunesha* to provide a portable and relayer-free multi-chain architecture with improved interoperability performance. Moreover, Zunesha is open-sourced, offering users a convenient and decentralized method to build subchains.

The first challenge of "on-flight" multi-chain upgrade necessitates the mainchain's seamless support for interoperability and subchain meta-information management. To address this, we designed a **S**mart-

con**Tr**A**c**t-based multi-**C**hain tool**K**it (STACK). Enabling interactions with subchains simply requires deploying STACK onto the original blockchain. As STACK is smart-contract-based, our solution can be integrated with existing blockchains without affecting ongoing operations. The multi-chain architecture based on smart contracts thus offers excellent portability as it seamlessly integrates with existing blockchains. Any blockchain that supports smart contracts can enable multi-chain using Zunesha. Experimental results show that the overhead of smart contracts is reasonable.

To address the challenges of timing issues and the elimination of third-party relayers in inter-chain transactions, this paper proposes the **Dynasty-Based Consensus Node Set Verification** (DB-CNSV) protocol. The core concept of DB-CNSV is to introduce a time unit called a *dynasty*, which manages the inconsistent views of the CNode set among distributed CNodes. This protocol establishes trust between blockchains in a multi-chain system and ensures the safety and liveness of subchains, even in the presence of timing-related inconsistencies. Furthermore, as demonstrated in the evaluation, eliminating relayers enhances interoperability performance in Zunesha, while maintaining safety guarantees.

Below, we summarize the main contributions of the paper:

- We propose Zunesha, a portable multi-chain architecture designed to upgrade any blockchain that supports smart contracts to a multi-chain system for better scalability. The upgrade only requires deploying STACK on the blockchain, which can be done without disrupting the blockchain's normal operation.
- We design the DB-CNSV protocol to address timing issues and eliminate relayers in the inter-chain transaction process. To the best of our knowledge, this paper is the first to formally define and analyze timing issues in multi-chain systems.
- We have implemented Zunesha on an existing blockchain and conducted extensive experiments on it. Experimental results demonstrate that enhanced blockchain achieves linear throughput improvement as the number of subchains increases proportionally, and our relayer-free inter-chain workflow outperforms Cosmos's interoperability performance and resilience against high latency.

2. Background

The blockchain aims to provide transaction management between entities that do not trust each other. Different users can complete transactions with anyone without the endorsement of a centralized third party. There are various techniques to scale up the blockchain [17–20], and a common classification is based on the layer where the technique is applied, namely Layer1, and Layer2. Layer1 is on-chain scaling, which includes improvements on block data [6, 7, 21, 22]. Layer2 seeks to extend the blockchain through an off-chain approach [10–13, 23–25]. Representative technologies are Rollups [10], off-chain channel [23, 24], sidechain [11, 25], and multi-chain architecture [12, 13]. The following subsections are dedicated to introducing the background in detail.

2.1. Multi-chain architecture

Multi-chain architecture [12–14, 16, 20, 26–28] as a scalability solution has emerged as a key approach to improve blockchain's scalability. These solutions distribute workload across multiple parallel blockchains, enabling the network to process more transactions simultaneously. Currently, multi-subchain technology is primarily applied in industry, with many solutions already being widely used and validated in real-world products. Therefore, this section focuses on the existing multi-subchain technologies in the industry to provide a clearer overview of the current state of research in this field.

Cosmos [13]: Cosmos is a multi-chain network that helps developers build blockchains and enables cross-chain connectivity via the relayers on the Cosmos Hub chain. The interoperability in Cosmos is based on the Inter Blockchain Communication protocol (IBC). However, it relies on additional relayers, such as a Hub chain, to facilitate inter-chain transactions, which adds complexity and introduces potential security and reliability risks.

Polkadot [12]: Polkadot is a multi-chain network based on Substrate [29]. It consists of a Relay Chain and multiple Para Chains, with the former only performing consensus on transactions, leaving their

execution to the latter. All Para Chains communicate with each other using Cross-Chain Message Passing (XCMP). Nevertheless, the number of Para Chains is limited to about 100 since all Para Chains depend on the Relay Chain to reach a consensus. In comparison, there is no limit on the number of subchains in Zunesha.

Avalanche [14]: The Avalanche Subnet is responsible for maintaining a blockchain and is made up of a dynamic set of CNodes. In Avalanche, all members of any subnet must also be part of the mainchain, which adds to the load on the mainchain. However, the mainchain's role is limited to maintaining built-in blockchains, making it less versatile than Ethereum and other base chains. Additionally, the Avalanche network and its subnets currently cannot interact with each other.

Hierarchical Consensus [15]: Hierarchical Consensus allows users to deploy subnets to enhance scalability. However, the proposed tree-structured subnet topology can cascade security issues and performance bottlenecks. For instance, if one subnet is compromised, all its child subnets cannot complete cross-subnet transactions. Additionally, the topology requires cross-subnet messages to be relayed from one subnet to the next, which can be slow and inefficient. In some cases, the message may even need to be routed back to the root before reaching its destination. In contrast, Zunesha addresses these challenges, offering a more efficient and secure solution for inter-chain transactions.

2.2. Sharding

Sharding [8, 9, 30–33] is a prominent blockchain scaling technique that partitions a blockchain network into multiple "shards" to enable parallel processing and increase overall throughput. Both sharding and multi-chain architectures achieve scalability by utilizing multiple blockchains. However, comparing the two approaches directly is challenging due to their differing priorities and design principles.

In the sharding blockchain, the chains are tightly integrated, functioning as a cohesive unit. This tight coupling leads to more efficient inter-chain communication mechanisms since the shards are designed to operate in synchrony. However, this close integration also introduces complexities, such as the need for shard reconfiguration to maintain balance across the network. Adding a new shard can impact the performance and structure of all existing shards, as the system must redistribute resources and workloads to maintain consistency.

In contrast, multi-chain architecture involves loosely coupled chains, where subchains and the mainchain operate independently with greater autonomy. This self-sovereignty allows for more flexibility and customization, as each subchain can be designed to meet specific use cases without affecting the overall network.

Overall, while sharding prioritizes efficiency through tight integration, multi-chain architectures focus on flexibility and modularity, making each approach suited to different scalability and application needs.

2.3. Inter-chain bridges

Inter-chain bridges, such as Ronin [34], xDai [35], and Gravity [36], act as relayers, facilitating transactions between two distinct blockchain networks. These relayers are crucial intermediaries, responsible for passing data and proofs between chains to enable the transfer of assets. In most cases, relayers are either implemented via smart contracts deployed on the blockchain or operated as independent third-party nodes, depending on the specific design of the cross-chain system. The typical process employed by these bridges is known as the lock-mint [37] mechanism. In this process, the assets are first locked on the source chain, and the relayer communicates proof of this action to the target chain. Once the proof is verified, an equivalent amount of assets is minted on the target chain.

While inter-chain bridges provide a practical solution for cross-chain transactions, some of them introduce a significant limitation: users must place trust in the relayer nodes. Whether managed by smart contracts or third parties, the reliability, security, and transparency of these nodes become central concerns. Any vulnerability or mismanagement in the relayer network can compromise the integrity of the transaction, exposing users to potential risks.

In contrast, Zunesha aims to harness the advantages of the lock-mint mechanism while mitigating the

drawbacks associated with third-party relayers. By designing a more secure and decentralized method for managing inter-chain transactions, Zunesha offers a more robust solution that eliminates the need for users to trust external relayers. This innovation will enhance both the security and efficiency of cross-chain asset transfers.

2.4. Rollups

Rollups are a Layer 2 scaling solution designed to increase blockchain transaction throughput while maintaining the security and decentralization of the Layer 1 chain. They process transactions off-chain and periodically submit compressed data (Rollups) back to the main chain, reducing congestion and lowering fees. There are two main types of Rollups:

(1) **Optimistic Rollups**: These assume transactions are valid by default but allow challenges through fraud proofs when disputes arise, ensuring system integrity without processing all data on-chain.
(2) **ZK-Rollups (Zero-Knowledge Rollups)**: ZK-Rollups use cryptographic proofs to validate transactions, submitting a compact proof of the entire transaction state to the main chain, which is more efficient than submitting individual transactions.

However, sequencers, responsible for submitting proofs to the main chain, are typically managed by third parties, introducing risks of centralization and censorship [38].

3. System design

3.1. System models

Before introducing the design of Zunesha's architecture, we first present the related assumptions and system models:

- **Security model**. As Zunesha is based on the PoS, we assume that the malicious nodes possess less than $1/3$ of the total stakes in the mainchain or subchain. We will further discuss the security model of subchains in Subsection 5.3.
- **Threat model**. Malicious nodes' behavior is assumed to be arbitrary, such as sending transactions with incorrect content and deliberately rejecting valid transactions [39]. However, malicious nodes cannot forge signatures.
- **Network model**. A peer-to-peer network connects nodes in the blockchain. It is assumed that the network is partially synchronous [40], where the network partition will heal after an unknown amount of time.

3.2. Zunesha architecture

Zunesha is a multi-chain architecture for blockchains that support smart contracts. It can enable multi-chain upgrades in existing blockchains without affecting their operation. Executing these on-the-fly upgrades necessitates only deploying the **S**mart-con**T**r**A**ct-based multi-**C**hain tool**K**it (STACK) on the existing blockchain, achieving a seamless upgrade process and good portability.

The left part of Figure.1 presents a schematic diagram of Zunesha. The mainchain is in the center since it is the blockchain to be scaled by Zunesha. Furthermore, each subchain functions independently and is deployed in a decentralized manner. Like any public blockchain, each subchain possesses its unique set of CNodes responsible for block production. Besides, the mainchain is not on the critical path of normal subchain-to-subchain transactions, which will be introduced in Section 3.4..

The right side of Figure.1 depicts the hierarchical architecture of a subchain CNode. The storage, network, and consensus layers are standard components of a blockchain node that provide basic functionalities. To meet interoperability requirements, we introduce the inter-chain layer, consisting of three functionalities:

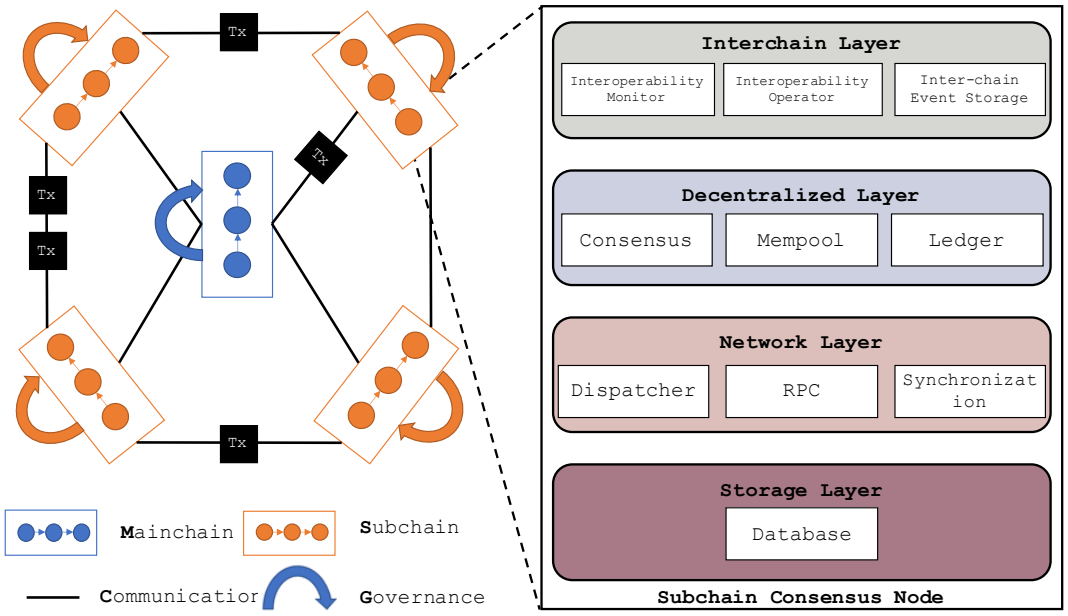


Figure 1. Overview of Zunesha's architecture.

(1) **Interoperability Monitor.** It monitors inter-chain-related messages, periodically collecting CNode set information from the mainchain and events related to inter-chain transactions from the mainchain and the local subchain. These events are stored in the local inter-chain event storage and will be introduced at the end of Subsection 3.3.
(2) **Interoperability Operator.** It accomplishes interoperability operations by issuing smart contract invocations in STACK based on the type of inter-chain events.
(3) **Inter-chain Event Storage.** It has durable storage for inter-chain events and the latest sequence number of processed events. With the help of this storage, CNodes can continue processing events even after restart.

In the rest of the sections, the **inter-chain layer** refers to the inter-chain layer of a CNode.

3.3. Smart-contract-based multi-chain toolkit (STACK)

With their great convenience and rich functionality, smart contracts are well-suited to provide decentralized inter-chain services. In Zunesha, multi-chain-related functionalities like subchain registration, CNode registration, and inter-chain transactions are accomplished via our designed **S**mart-con**Tr**A**c**t-based multi-**C**hain tool**K**it (STACK). The advantage of such design is portability and non-interfering: any blockchain that supports EVM-style smart contracts can use STACK to support multi-chain on the fly. STACK contains the following smart contracts, which are also shown in Table 1.

Smart Contract	Function	Event
Mainchain Registrar	(de)registerSubchain	Subchain(De)Registered
	deposit/withdrawCollateral	CollateralDeposited/Withdrawn
	deposit/withdrawStake	StakeDeposited/Withdrawn
Inter-Subchain Registrar	enableInterSubchainTx	SubchainParameterSubmitted
	verifySubchainLegality	SubchainLegalityVerified
Token Bank	(un)lockToken	Token(Un)Locked
	mint/burnVoucher	VoucherMinted/Burnt

Table 1. Interface of smart contracts and corresponding events.

Mainchain Registrar. The *Mainchain Registrar* is responsible for subchain bootstrap, security, and subchain CNode registration, which is deployed on the mainchain. Recording registration in the mainchain's smart contract saves and protects subchain and CNode set information. Besides, we have developed a token economic mechanism integral to the process of subchain CNode registration within this contract. This mechanism is essential for bolstering the autonomy and security of the subchain. The *Mainchain Registrar* ensures subchain safety through the following functionalities:

(1) Subchain registration: It records the subchain's meta-information like ChainID, genesis block hash, etc.

(2) Subchain CNode registration: To safeguard against corruption and subversion, we designed a **two-phase deposit** mechanism for ordinary nodes to become subchain CNodes. Initially, node owners must deposit collateral using the mainchain's native token, which transforms the node into a CNode candidate. This collateral deposit ensures that any malicious behavior from the CNode candidate would result in a slash of its mainchain's native token, incurring significant costs.

The second phase involves a stake deposit using the subchain's governance token. Each subchain has its own governance token deployed on the mainchain by the entity responsible for registering the subchain. The governance token is typically deployed as an ERC-20 token [41]. As security is not just concentrated on the mainchain, this mechanism promotes a balanced and decentralized security posture across the entire ecosystem. Moreover, it increases the cost of corruption for potential attackers, thereby enhancing the reliability and autonomy of the subchain. Finally, it prevents subchains within Zunesha from being vulnerable to long-range attacks [42, 43], as all deposit histories and CNode sets are recorded in the smart contract.

Inter-Subchain Registrar. This contract is mainly for interoperability between subchains and is deployed on the subchain. The trust level between subchains is lower than that between the mainchain and subchains due to the inherent independence of subchains. Each subchain operates autonomously with its own governance, consensus mechanisms, and security assumptions, which can differ significantly from one another. Consequently, interactions between subchains are more vulnerable to risks. In contrast, the mainchain typically offers stronger security guarantees and a more established trust model, resulting in higher trust for its interactions with subchains. Since the trust level between two subchains is lower than that between the mainchain and subchains, a mechanism is needed to increase the security of subchain-to-subchain transactions. This contract helps enable inter-chain transactions between subchains. The workflow of this contract will be introduced in Section 3.4..

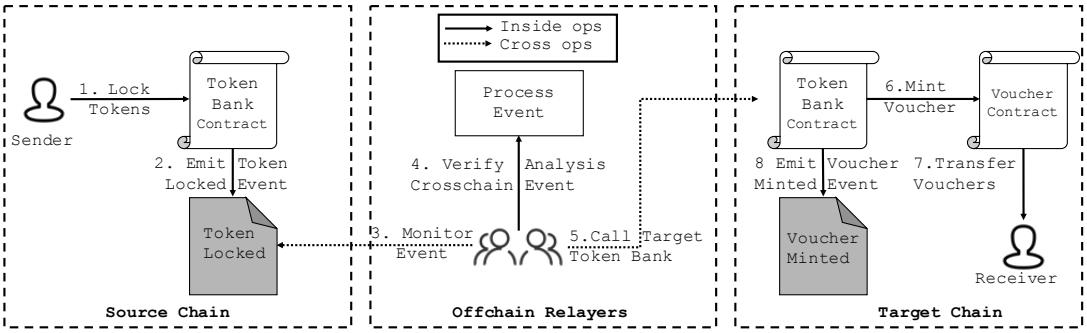
Token Bank. A series of *Token Bank* smart contracts based on the lock-mint mechanism [37] is used for inter-chain token transfers. These contracts are deployed on both the mainchain and the subchain, and each token type (such as ERC20, ERC721, ERC1155, and the mainchain native token) has a separate *Token Bank* contract with similar interfaces. New token types can be added easily with this method.

In the lock-mint mechanism, assets are not transferred directly to another chain but are first **locked** in the source chain. Subsequently, a corresponding amount of *vouchers* is **minted** on the target blockchain. The voucher in the target chain represents the asset in the source chain, implemented through a smart contract with the same type of original asset. When the owner wants to **unlock** the assets on the source chain, the voucher needs to be **burnt** on the target chain first. Each operation issues an event when it is successfully completed on the blockchain. Each event has a unique *nonce* ID to prevent double-spending attacks. A *denomination* is designed to provide a unique identifier for each inter-chain transaction, encompassing information like the ChainID of the source chain. This identifier guides the inter-chain layer throughout the inter-chain transaction process.

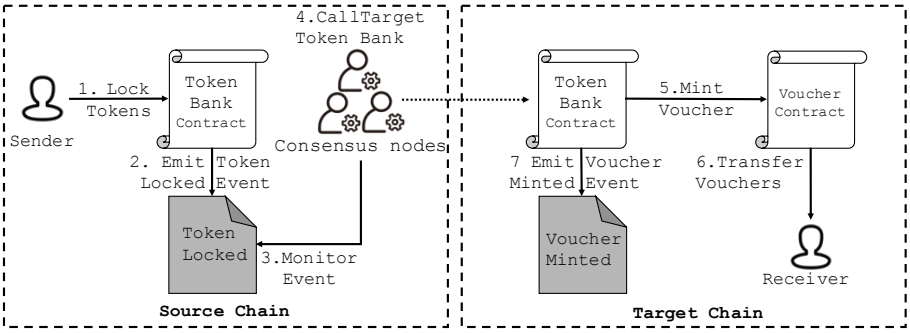
3.4. Subchain bootstrap

Several mechanisms have been designed to enhance subchain safety and increase the cost of malicious behavior during subchain bootstrap. The subchain bootstrap needs both off-chain preparations and interactions with STACK to ensure the subchain's safety and decentralization, which includes the following steps:

(1) The deployer must obtain a chainID [44], which necessitates the submission of enough publicly available meta-information about the subchain.



(a) Classic inter-chain transaction workflow with relayers



(b) Optimized inter-chain transaction workflow in Zunesha

Figure 2. Inter-chain transaction workflow.

(2) The subchain deployer must attract enough CNodes to decentralize the subchain, as no users will join a centralized blockchain. The subchain deployer can invite mainchain CNodes to become CNodes on the subchain. Owners of mainchain CNodes have the capability to operate a distinct subchain CNode using the same private key as their mainchain CNode, enabling them to join a subchain. Given that mainchain nodes hold substantial deposits on the mainchain, combining these deposits with the two-phase deposit and slashing mechanisms described in Subsection 3.3. enhances the security of the subchain.

(3) The deployer must generate a genesis block containing the initial subchain CNodes and ChainID information. Since CNodes verify a CNode's address and balance, a mismatched CNode set will not be accepted by honest CNodes.

(4) The deployer must register the subchain with the STACK using the genesis block hash and other metadata to double-check the information generated in previous steps.

4. Interoperability in Zunesha

4.1. Inter-Chain transaction workflow

Traditional relayer-based interoperability workflow is shown in Figure.2(a). Relayers are intermediaries between different blockchains for profit-making purposes [45]. However, relying on relayers introduces additional trust assumptions and security risks, potentially compromising reliability and leading to significant financial losses [46]. Simply designating the relayer as the subchain CNode could address some safety concerns caused by intermediaries, but it does not address the interoperability performance and reliability issues since three roles (sender, relayer, and receiver) are still involved.

As shown in step 4 in Figure.2(b), subchain CNodes in Zunesha complete inter-chain transactions related to its local subchain, thereby reducing the number of roles required from three to two. The inter-chain layer initially captures an inter-chain event such as *TokenLocked*/*VoucherBurnt* associated with the local subchain and, after verification, places it in the inter-chain event cache. Subsequently, the

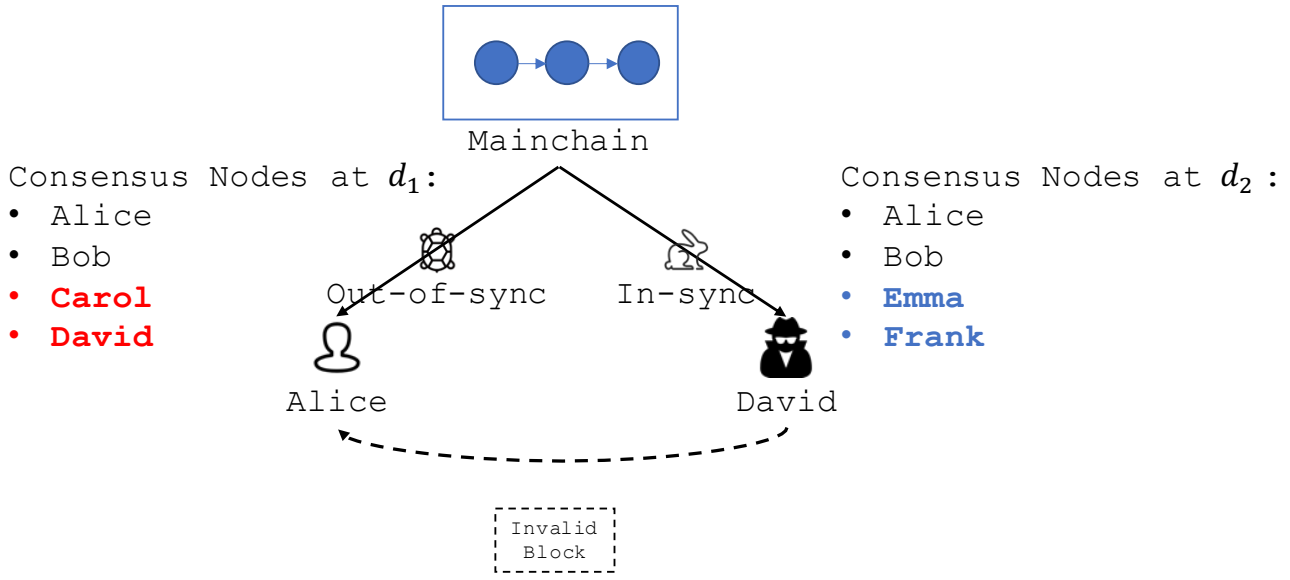


Figure 3. An example of timing issues.

inter-chain layer processes the valid inter-chain transaction by invoking the *mintVoucher*/*unlockToken* on the target blockchain, as shown in step 5. Finally, if the inter-chain transaction is successfully completed, *VoucherMinted*/*TokenUnlocked* will be issued on the target blockchain.

As the success of *mintVoucher*/*unlockToken* only requires the agreement of more than 2/3 of the voting power, CNodes who have not invoked those functions but have witnessed the event can observe that the inter-chain transaction has been processed, thereby avoiding redundant invocations. Moreover, the inter-chain layer can batch up inter-chain transactions and invoke them in the target chain to save the invocation time. Besides, the aggregated signature [47] can be applied to improve inter-chain transactions' efficiency further. To the best of our knowledge, Zunesha is the first multi-chain system architecture among existing solutions to offer interoperability without needing relayers.

To incentivize CNodes to relay transactions and to ensure that honest CNodes have adequate funds to invoke smart contracts on the target chain, the client initiating the inter-chain transaction will pay additional fees to the subchain CNodes. Besides, the extra cost can also prevent Distributed Denial-of-Service attacks [48] on multi-chain systems.

4.2. Timing issues

Attaining interoperability without relayers poses significant challenges in multi-chain, primarily due to the dynamic CNode set and timing issues. The dynamic CNode set means that subchain CNodes in Zunesha can join or leave without requiring approval from any central authority. As mentioned in Section 3., the node uses *Mainchain Registrar* contract to de/register as a CNode via deposit/withdraw stakes and collaterals. This decentralized nature introduces complexities in achieving consensus among subchain CNodes regarding the legitimate participants in inter-chain transactions. Given the distributed system's inherent nature and the low levels of trust among subchains, reaching consensus in a decentralized manner becomes challenging. Furthermore, CNodes with better network conditions can promptly access the updated CNode set from the *Mainchain Registrar*. In comparison, those with poorer network conditions may experience delays in obtaining the latest information. The problems caused by such inconsistencies are referred to as **timing issues** (TI).

Figure 3 illustrates an example of timing issues in a blockchain system. Initially, at time period d_1, the system has four consensus nodes: Alice, Bob, Carol, and David. In the subsequent time period d_2, Emma and Frank replace Carol and David as consensus nodes.

Among the consensus nodes, Alice is a valid node with $\frac{1}{3}$ of the total stake. However, due to a slow network environment, Alice becomes out-of-sync with the latest time period and still believes she is in

d_1. On the other hand, David, a malicious node with less than $\frac{1}{3}$ of the total stake but a better network connection, realizes he is no longer a consensus node. Despite this, David continues to propose a block. Alice, still believing she is in d_1 and trusting David as a valid consensus node, receives and accepts David's block proposal, casting her vote for it.

With Alice holding $\frac{1}{3}$ of the tokens and David holding some, the honest nodes' total governance tokens fall below $\frac{2}{3}$, disrupting consensus and potentially stalling block production. If Alice waits to sync with the latest consensus set, she cannot participate in consensus during the delay, which might still cause the blockchain to stall due to her significant governance token share.

Overall, TI primarily affects reaching a consensus on the dynamically changing CNode set, which is the foundation for the security of all inter-chain transactions. Inconsistent views of CNodes could hurt the safety and liveness of the subchain. Consequently, a mechanism is necessary to ensure that all CNodes participating in inter-chain transactions on subchains have the same view of the valid CNodes.

4.3. Dynasty-based consensus node set validation protocol

To facilitate CNode set updates and to ensure a consistent view of CNode set in inter-chain transactions under TI, we introduce a time unit called **dynasty**. The dynasty is defined as the duration of a fixed number of mainchain blocks. It is consistent across all subchains. Similar concepts, such as view and epoch [49] [50], do not specifically tackle TI. Within a single blockchain, epochs and views typically represent the tenure of consensus nodes or specific periods in the blockchain's lifecycle. For instance, in a multi-chain system, a newly added subchain might start its epoch/view at 0, while another subchain could already be at epoch/view 1000. To ensure synchronization and a shared understanding of the current term across all subchains, a unified time unit is required. To address this, we propose the term dynasty, which provides a unified concept while avoiding confusion with the traditional meanings of epoch and view in individual blockchains.

Before presenting the new algorithm, we will first outline the process for determining the appropriate CNode set for a subchain and verifying inter-chain transactions. CNode set updates occur exclusively when the dynasty changes. Initially, when an ordinary node finishes the two-phase deposit in the *Mainchain Registrar*, the contract updates the CNode set, using the current mainchain block height as its version. To query the CNode set for a subchain, one must specify the mainchain block height of the CNode set. The *Mainchain Registrar* will then return the CNode set corresponding to the closest height that is less than the specified height.

Assume the main chain block height is M_h, the dynasty length is D_l, and the calculation of the dynasty is $D = \lfloor M_h/D_l \rfloor$. After determining the dynasty of each consensus node, the block-producing node will determine which dynasty is the most recent one visible to more than $\frac{2}{3}$ of the consensus nodes. Finally, the consensus nodes corresponding to that dynasty will form the current valid set of consensus nodes. For instance, suppose the mainchain block height is M_h, dynasty length is $D_l = 100$, and the highest heights visible to the four CNodes are 230, 340, 350, and 460, respectively. The dynasty seen by each CNode is calculated as $D = \lfloor M_h/D_l \rfloor$, which are 2, 3, 3, and 4, respectively. As the most advanced dynasty seen by the majority of CNodes is 3, all CNodes except the straggler only need to query the CNode set at the mainchain's height of $M_h^D = D \times D_l = 300$. Since we assume a partially synchronous network model, the lagging CNode will eventually catch up and access the latest dynasty.

For inter-chain transactions, the operations in the *Token Bank* contract necessitate the dynasty d as a parameter. Thus, CNodes on the target blockchain independently verify whether the CNode set that executed the transaction on the source blockchain corresponds to d and whether d is consistent with its local view of the dynasty. This method ensures that, despite variations in mainchain block heights observed by CNodes due to TI, most CNodes can still authenticate the legality of transactions using the consistent CNode set.

To mitigate the impact of TI and safely complete the change of CNodes in the subchain, we propose the **Dynasty-Based Consensus Node Set Verification** protocol (DB-CNSV). The workflow of DB-CNSV in the proposer's and other CNodes' views is described in Algorithm 1 and Algorithm 2. In each subchain,

the change in its CNode set is accomplished through a special transaction called the **consensus node set change transaction** (CNSCX). As depicted in Algorithm 1, the CNSCX is included in a block only when *more than 2/3 of CNodes have entered the new dynasty*. During block voting, CNodes attach the mainchain block height they observe, and the proposer determines whether to include the CNSCX in the block based on the votes. Due to TI, if a proposer proposes the CNSCX immediately after witnessing a new CNode set, it may get rejected as other CNodes might not have seen it yet. Hence, retired CNodes need to remain on duty until the CNode set change is finalized on the blockchain. Furthermore, when proposing blocks, the proposer must include the mainchain block height, and the digest of the CNode set information in the block header as they determine which CNode set will be utilized in the CNSCX.

Subchain CNodes need to protect themselves from potential deception by a malicious proposer. As demonstrated in lines 11-13 of Algorithm 2, when a CNode receives a block containing a CNSCX, it must verify the authenticity of the CNSCX. If the block is valid, the CNode casts a vote in favor of the block. After the block is finalized through the subchain consensus protocol, the CNode set can be updated as the CNode executes the CNSCX. The safety and liveness analysis of DB-CNSV is presented in Section 4.4.

Algorithm 1 DB-CNSV at proposer

```
1:  Input: current CNode set C, dynasty d, next block B
2:  Update mainchain block height h
3:  Calculate the dynasty d' with h
4:  if d' ≥ d + 1 then
5:      Query CNode set C' at d' from mainchain
6:      if C ≠ C' then
7:          if At least 2/3 of CNodes are in d' then
8:              Add CNSCX into B and propose B
9:          else
10:             Propose B without CNSCX
11:         end if
12:     end if
13: end if
```

Algorithm 2 DB-CNSV at Consensus Node

```
1:  Input: current CNode set C, dynasty d, received block B
2:  if B.d < d then
3:      Reply to remind the proposer is out-of-date
4:  else if B.d = d then
5:      if B.C = C then
6:          Continue the consensus process
7:      else
8:          Reject the block
9:      end if
10: else
11:     if B includes a CNSCX tx then
12:         Query CNode set C' at d' = tx.d from mainchain
13:         if tx.C = C' & B is finalized then
14:             Start CNode set changing
15:         else if tx.C ≠ C' then
16:             Reject the block
17:         end if
18:         Continue the consensus process
19:     end if
20: end if
```

4.4. Subchain-to-subchain transaction workflow

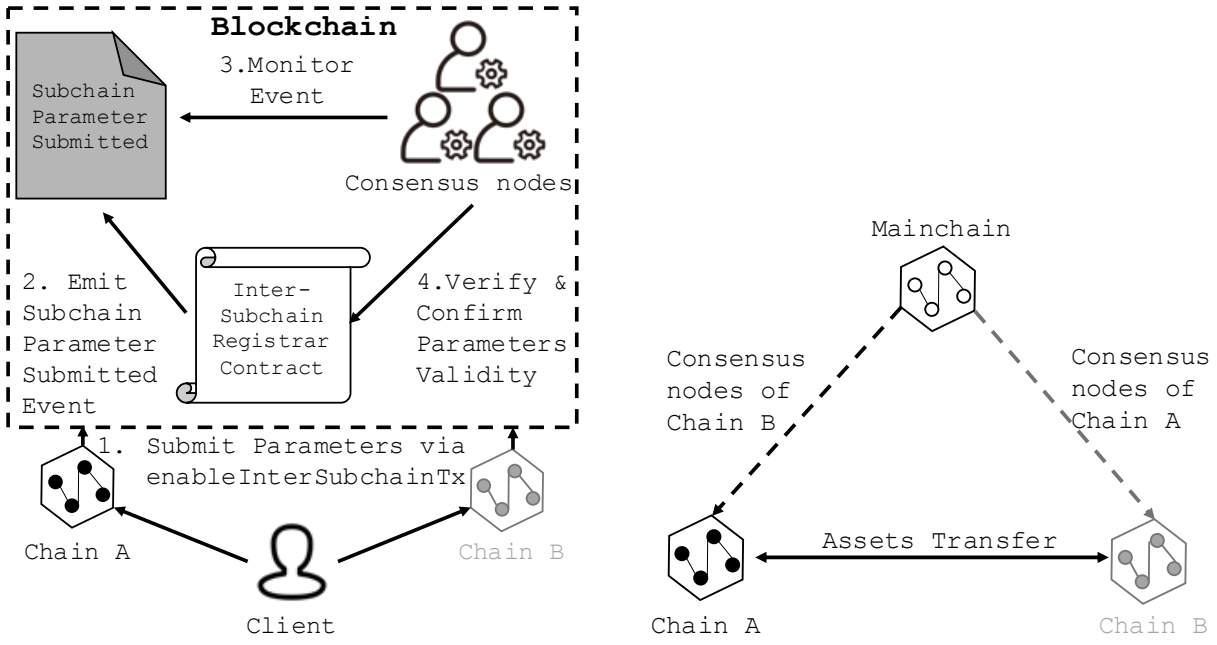
As discussed in Section 3., if all subchain-to-subchain (S-S) transactions need to go through the mainchain, the mainchain will be overloaded, which is the drawback of many multi-chain projects such as Cosmos

and Polkadot. The DB-CNSV protocol is used as a trust foundation between CNodes on the two subchains involved in S-S transactions to avoid this.

S-S transactions should only occur when there is a demand from a user who intends to transfer assets between these two subchains. Since each subchain operates independently, asset transfers become meaningful only with a clear objective. Consequently, the transaction issuer needs to provide essential information about the counterpart subchain to initiate the transaction. As illustrated in Figure 4(a), the issuer first invokes the *enableInterSubchainTx* function in the *Inter-Subchain Registrar* contract on both subchains to supply relevant parameters, such as seed IP, ChainID, and so on, about the counterpart chain. A *SubchainParameterSubmitted* event is then issued. When CNodes observe the event, they will independently invoke the *verifySubchainLegality* to check the legality of the submitted parameters. Given that the subchain possesses a lower trust level compared to the mainchain, it is imperative for the owner of the subchain CNode to thoroughly assess the risks associated with conducting inter-chain transactions with another subchain prior to initiating the *verifySubchainLegality* process. If most CNodes vote that the parameters are legitimate, then the inter-chain layer within the CNode can process S-S transactions between these two subchains. The validation step includes the following:

(1) The subchain with the ChainID in the parameter is registered on the *Mainchain Registrar* smart contract.
(2) An RPC client like Geth can be established with the given seedIP. The Geth is an open-sourced original implementation of the Ethereum protocol [51] that interacts with the EVM-compatible blockchain.
(3) The ChainID can be queried through the RPC and whether the returned ChainID is the registered value.

All conditions listed above must be satisfied to complete S-S transactions through the inter-chain layer. The chainID [44] is stored in a decentralized file system (IPFS) [52], and the SeedIP is publicly available. These features make it difficult for an adversary to forge a blockchain.



(a) Subchain parameter submission

(b) Asset transfer

Figure 4. Subchain-subchain transaction workflow.

Figure.4(b) shows that, after the legality of the counterpart subchain is verified, the lock-mint mechanism mentioned in Section 3. is used for the remaining steps of the S-S transaction. The only difference is that the inter-chain layer of CNode on the target blockchain will not actively poll the *TokenLocked*/*VoucherBurnt* event on the source blockchain. Since no one knows when an S-S transaction

will happen, active polling will be a waste when there are many S-S transaction sources. Therefore, the inter-chain layer of CNode on the source blockchain will directly call the *Token Bank*'s function on the target blockchain to complete the S-S transaction.

5. Security and liveness in Zunesha

5.1. DB-CNSV protocol

In this subsection, we present the safety and liveness analysis of the DB-CNSV protocol.

Theorem 1. *DB-CNSV achieves **safety** if less than 1/3 of the total stakes are controlled by malicious nodes.*

Proof. Assuming that CNode c_1 initiates an incorrect CNSCX tx^* with a wrong dynasty or the wrong CNode set, and a non-faulty CNode c_2. The current dynasty corresponding to the working CNode set is d and the latest height of the mainchain block c_2 sees is h'. Height h' corresponds to the dynasty of $d'(d' \geq d)$. Malicious CNode c_1 attempts to change dynasty to d^* and CNode set to C^*. Two scenarios of the malicious behavior of the CNode c_1 are discussed here.

- Suppose that $d^* < d+1$, then according to line 2 in Algorithm 2, c_2 will not accept tx^*.
- Suppose that $d^* \geq d+1$ but $C^* \neq C_{d'}$, then according to lines 11-17 in Algorithm 2, c_2 will query the correct CNode set $C_{d'}$ from the mainchain using h'. Since $C^* \neq C_{d'}$, c_2 will not accept tx^*.

Given that less than 1/3 of the total stakes are controlled by the malicious nodes, and whether a CNode accepts the CNSCX is based on the result of its independent query from the mainchain, the non-faulty CNode on a subchain will not accept the wrong CNSCX, i.e., DB-CNSV guarantees safety.

Theorem 2. *DB-CNSV achieves **liveness** if less than 1/3 of the total stakes are controlled by malicious nodes.*

Proof. Assume that the current dynasty corresponding to the working CNode set is d_1, and that CNode c_1 has seen the mainchain block at height h_2, whose corresponding dynasty is $d_2 \geq d_1 + 1$. In the partially synchronous network model, messages will finally reach their destination after an unknown finite time δ [40]. As CNode c_1 has seen h_2, according to the network model, CNodes in the subchain can see the block at h_2 on the mainchain after at most δ and realize that the dynasty needs to be changed to d_2. Since less than 1/3 of the total stakes are controlled by malicious CNodes, the block containing CNSCX will be approved by more than 2/3 of the CNodes with DB-CNSV protocol. After that, the dynasty is updated to d_2, and the corresponding CNode is also changed. Consequently, even if the δ is longer than the length of the dynasty, the honest CNodes will make the right change of the dynasty and its corresponding CNode set, *i.e.*, DB-CNSV guarantees liveness.

In practice, Zunesha has been implemented into an existing public blockchain, and the duration of a dynasty is set to a large value, like several days, to account for network latency. Zunesha has been operating steadily without any TI-related issues, which demonstrates its practical viability. Moreover, as described in Algorithm 2, only when more than 1/3 of the CNodes' latency exceeds the duration of the dynasty the current CNode set will not be able to switch to the next CNode set in time. However, DB-CNSV requires that old CNodes still need to perform their duties before the switchover is completed. Therefore, DB-CNSV can still guarantee the safety and liveness of the system in the case that the latency exceeds the duration of the dynasty.

5.2. Inter-chain transaction workflow

This subsection discusses the liveness and safety properties of inter-chain transactions. Suppose that less than 1/3 of the total stakes in both the source blockchain and target blockchain are controlled by the malicious nodes. As described in Section 3., the inter-chain layer scans *every* block from the mainchain and the local subchain. Based on the assumptions outlined in Section 3., the inter-chain layer can retrieve the inter-chain transaction event within a maximum time frame of δ. Once retrieved and verified, the inter-chain layer can invoke the corresponding smart contract on the target chain, ensuring the **liveness** of inter-chain transactions.

When the inter-chain layer invokes the *mint*/*unlock* functions in the *Token Bank* smart contracts, it

signifies a vote in favor of those operations, confirming its agreement with the client's *lock*/*burn* requests. To complete the *mint*/*unlock* operations, votes from CNodes holding over 2/3 of the total stakes are required. The hash of the parameters, such as the denomination, nonce of each inter-chain transaction, and token quantities, distinguishes each request and prevents replay attack [44]. If a malicious CNode attempts a double-spending attack by modifying the parameters, they cannot obtain votes from honest CNodes as the hash will differ. Hence, the **safety** of inter-chain transactions is guaranteed.

5.3. Security model of subchains

In Subsection 3.1., we suppose that less than 1/3 of the total stakes of a subchain are controlled by malicious CNodes, which is the same assumption as current multi-chain solutions like Cosmos and Polkadot. As outlined in Subsection 3., each subchain within Zunesha functions autonomously, with its governance token signifying its voting authority. Accordingly, the security of each subchain relies on the vigilance of its CNodes and can be enhanced by existing techniques [53], and ensuring less than 1/3 of the CNodes in the subchain are malicious falls outside our primary focus.

Current multi-chain solutions rely on relayers to transfer inter-chain transactions, which means if the relayers are compromised, interoperability in all blockchains will be affected. While in Zunesha, the influence of a compromised subchain is limited to the blockchain that has approved inter-chain transactions with the compromised subchain. Besides, Zunesha still ensures that even if a subchain becomes compromised, its negative impacts can be effectively contained. This containment is possible because CNodes from the compromised subchain cannot affect the operation of other chains. Moreover, digital assets deployed in other chains which have not been transferred to the compromised subchain remain secure. For tokens already transferred to the compromised subchain, the smart contract on well-behaved chains stores the number of locked tokens, preventing malicious CNodes from unlocking an unlimited amount on other chains. Besides, economic deterrents, such as mainchain deposit slashing and gas fees, make engaging in malicious activities economically irrational.

6. Evaluation

We have implemented Zunesha with 4.7K lines of Solidity smart contract code. Besides, the subchain is built by adding 5K lines of Golang code to the Theta Network. The mainchain is the vanilla version of the Theta Network. Theta Network [54] is an open-sourced PoS blockchain — Theta Network [55], which is referred to as **basechain** in this paper. Basechain's consensus protocol is a variant of HotStuff [56].

Our experimental evaluation is based on up to 260 servers provided by Alibaba Cloud. Each server is an ecs.c7.2xlarge machine with eight vCPUs of Intel(R) Xeon(R) Platinum 8369B @ 2.70GHz, 16 GB memory, and 10Mbps bandwidth. Due to cluster size limitations, in the multi-chain system, including the mainchain and all subchains, each blockchain has four CNodes, and each server runs only one CNode.

6.1. Performance overhead

We first evaluate the cost of adopted smart contracts used for inter-chain transactions. Transaction costs are defined differently in Zunesha, Cosmos, and Polkadot, and the monetary cost is closely tied to fluctuating token prices. Therefore, we use gas as the metric and establish ERC20 transfer as the baseline, considering it a well-known fundamental smart contract function. Each function shown in Table 1 is evaluated with the EIP-2387-based [57] EVM, and we use *ERC20 Token Bank* as a representative of *Token Bank*.

Figure 5 displays the results of gas usage. "D" and "W" in the legend refer to "deposit" and "withdrawal", respectively, and "*" denotes calling the specific function for the first time. The leftmost bar represents the cost of an ERC20 *transfer*.

It is noteworthy that *Mint**, *WCollateral**, and *WStake** consume much more gas than others. This is because, as mentioned in Section 3., the voucher is implemented via the smart contract. When the voucher is created for the first time, the *Token Bank* deploys a Voucher smart contract based on the token type from the denomination. Therefore, the gas used in these functions includes deploying a new smart

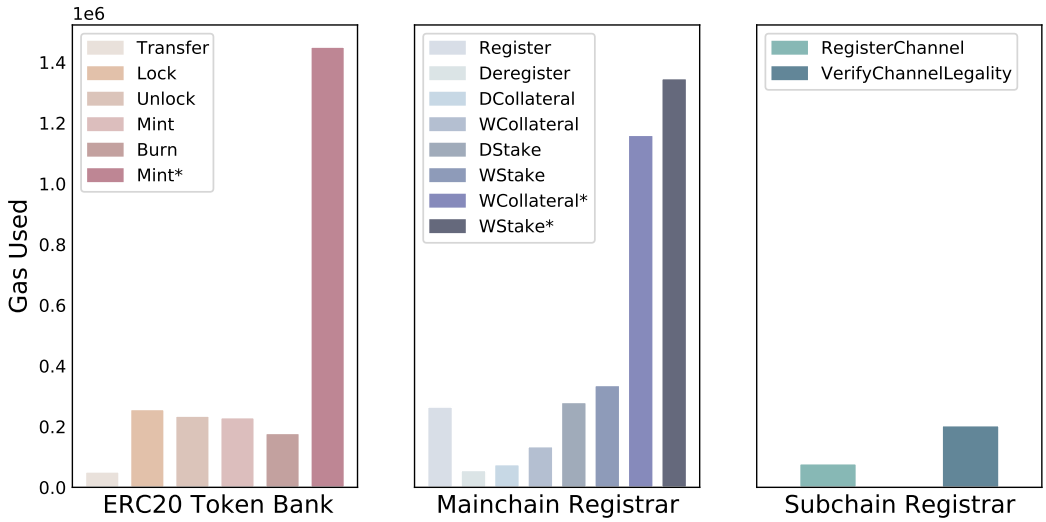


Figure 5. Gas cost of smart contract functions in Zunesha.

contract. This is also true for *WCollateral** and *WStake**. When a CNode issues a withdrawal request, the request is added to a withdrawal queue first and then enters the pending period. If the CNode misbehaves during the pending period, the *Mainchain Registrar* will slash the fund. Therefore, a *withdrawal queue*, which is also a smart contract, is established upon the first *WCollateral** and *WStake** call, with each subchain having its own withdrawal queue. Overall, the overhead is acceptable since the costly operations for a specific subchain only need to be performed once.

6.2. Throughput scalability experiment

This experiment is set up to test how throughput improves with the help of subchains. In this paper, we measure the throughput by transaction per second (**TPS**). It is important to note that Zunesha enhances the scalability of the mainchain by offloading transactions to subchains rather than improving the speed of consensus or data transmission efficiency. Therefore, the original blockchain TPS in Zunesha is not improved. The TPS in this experiment refers to the overall TPS, which is the sum of the basechain TPS and all subchains' TPS.

The experiment is conducted with 2, 4, 8, 16, 32 and 64 subchains with extensive inter-chain and intra-chain transactions. As inter-chain transactions are carried out via smart contracts, we use ERC20 token transfer as the workload within blockchains. Besides, *lock-mint* represents inter-chain transactions since it completes transferring the token from one blockchain to another, and all inter-chain transactions are subchain-to-mainchain transactions. In each subchain, the client sends transactions intensively to saturate TPS. The mainchain only processes the inter-chain transactions. The proportion of inter-chain transactions in the overall workload is α. There is no universal agreement on a reasonable value of α. In this experiment, α is set to 2.5%, 5%, and 10%, meaning the inter-chain transaction is issued every 40, 20, and 10 ERC20 transfers, respectively.

Figure 6 shows an almost exponential climb of TPS as the number of subchains increases exponentially. Note that the x-axis is in the log scale. If we redraw the figure with the x-axis in linear scale, then we can see a linear increase of TPS with the number of subchains. When the number of subchains is less than and equal to 4, the difference between different α settings is negligible. As the number of subchains exceeds 4, the gap among the curves becomes obvious. Moreover, the gap almost remains constant when there are more than or equal to 16 subchains because the basechain reaches its inter-chain processing capacity.

As inter-chain transactions are normal smart contract transactions, changing the value of α will not influence the linear scalability trend. Moreover, if the basechain's processing capacity improves, the overall TPS will also be better off, which is orthogonal to the contribution of Zunesha. Overall, Zunesha has demonstrated good scalability in overall throughput and can support numerous subchains.

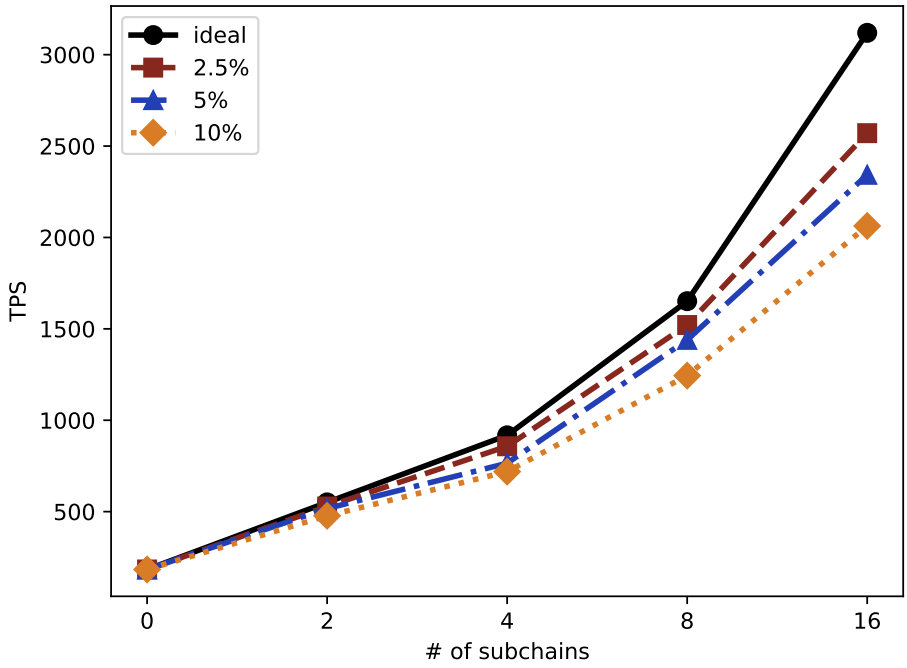


Figure 6. Basechain TPS improvement after scaled by Zunesha. The legend (2.5%, 5%, 10%) refers to the ratio of inter-chain transactions.

6.3. Latency and throughput comparison

This experiment compares the implementation of inter-chain transaction interoperability in terms of throughput and latency. Since Zunesha is a portable multi-chain architecture, and it can be combined with different blockchains, the value of TPS and latency depend more on the native performance of the basechain. Moreover, the advantage of Zunesha mainly lies in optimizing the inter-chain transaction process since Zunesha does not need a relayer to relay the inter-chain transaction. Consequently, this experiment mainly shows the benefit of removing relayers in the inter-chain transaction.

Some current multi-chain projects cannot be directly included in this comparison as they either lack support for inter-chain transactions or do not implement the claimed inter-chain transaction mechanism. Consequently, we choose Cosmos [13] for comparison. Cosmos is a state-of-the-art multi-chain project with a significant user base, and its IBC protocol has been successfully deployed in real industrial scenarios.

However, an apple-to-apple comparison between Zunesha and Cosmos would be difficult due to the differences in their consensus protocols and architectures. Cosmos utilizes Tendermint [58] as its consensus protocol, while the basechain employs HotStuff [56]. In HotStuff, two consecutive blocks are required to finalize a previous block, whereas Tendermint has no such requirement. Moreover, Cosmos carries out inter-chain transactions through IBC, written in Go, while basechain employs Solidity-based smart contracts. Despite the difference, Cosmos remains the project closest to Zunesha concerning subchains' independence and security. For comparison, we utilize the official Cosmos SDK [59] to implement Cosmos blockchains and relayers with default settings if not specified.

Moreover, to clearly show the improvement by our optimization in the inter-chain transaction process and represent other blockchains that adopt relayer-based inter-chain transaction mechanism, we implement Zunesha-Relayer for comparison. In Zunesha-Relayer, a relayer is deployed on the separated node to relay the transactions from the source blockchain to the target blockchain, rather than through the inter-chain layer of the CNode.

In this experiment, two blockchains are involved, i.e., source and target blockchain. As shown in Figure 2(a), the inter-chain transaction in Cosmos and Zunesha-Relayer need two hops from the source blockchain to the target blockchain, while Zunesha only needs one. We use Linux traffic control (TC) [60]

to inject latency in the inter-chain transaction transmission to demonstrate the improvement in resilience of interoperability performance brought by this optimization. Injecting the latency allows us to simulate the real latency in different geographical location settings. In Cosmos and Zunesha-Relayer, we add the latency to the relayer to simulate its distance from the source and target blockchain. Meanwhile, in Zunesha, latency is added to the CNodes in the source blockchain to simulate its distance from the target blockchain. The injected latency ranges from 0 ms to 250 ms.

To mitigate the impact of the consensus protocol difference, we set the block interval in the Zunesha and Zunesha-Relayer to 0.5 seconds, which is half the interval used in Cosmos, such that the block finalization time of Zunesha, Zunesha-Relayer, and Cosmos all equals 1.0 second. Moreover, the comparison experiments are conducted in a local area network to eliminate the impact of the fluctuating wide area network.

Latency. The latency is defined as the interval between the finalization of an inter-chain transaction on the source chain and its counterpart on the target chain. To eliminate the influence of transaction queuing, which adds extra latency, we throttle the sending rate to one transaction every two seconds and measure the average latency after a 120-second runtime.

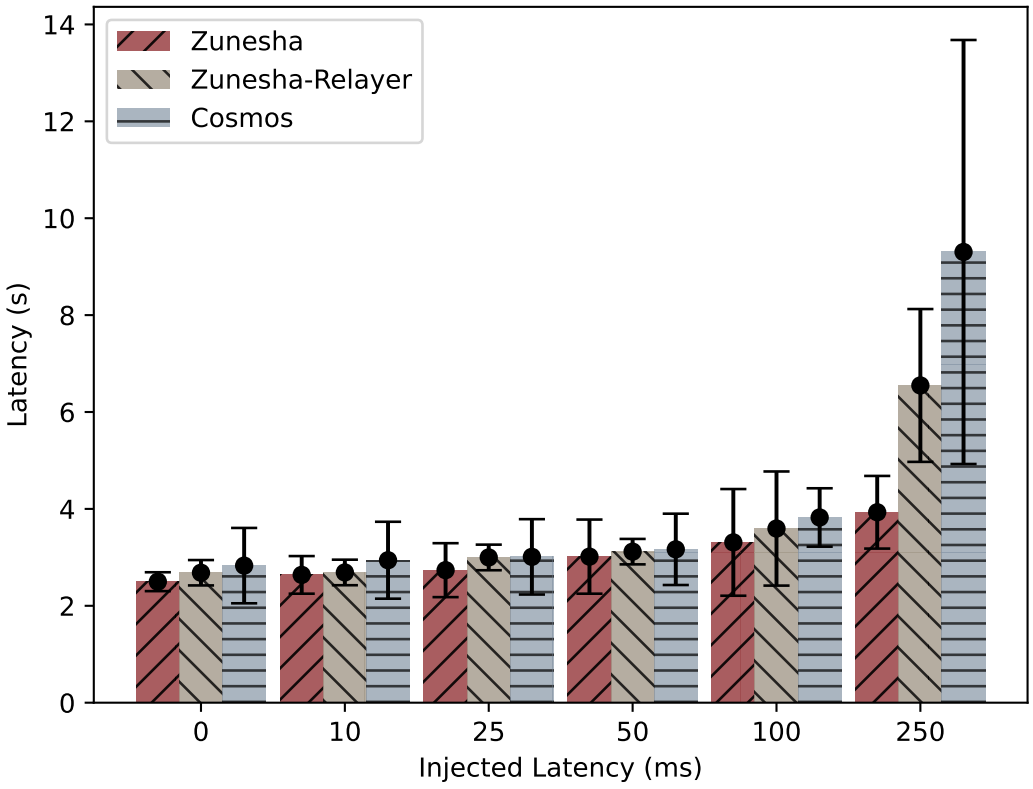


Figure 7. Inter-chain transaction latency in Zunesha, Zunesha-Relayer and Cosmos with different amount of injected latencies.

Figure 7 demonstrates that a clear divide is formed at 100 ms. When the injected latency is less than or equal to 100 ms, performances of Zunesha, Zunesha-Relayer and Cosmos are close, with the first two slightly faster than the Cosmos. This difference is likely due to the difference between the basechain and Cosmos implementation details. Moreover, end-to-end latency grows slowly in all blockchains at this stage. The injected latency accounts for a small fraction of the operation, as the latency is around 2 seconds when there is no extra latency. In the latter stage, when the injected latency rises to 250 ms, the relayer is impacted more, causing the end-to-end latency in Zunesha-Relayer and Cosmos to surge and fluctuate. Since relayers need one more hop to relay the transaction, they are less resilient when faced with rising latency.

Throughput. The throughput is measured as the TPS. To demonstrate the TPS of inter-chain transactions, the client on the source chain extensively sends transactions, with all transactions classified

as inter-chain transactions, i.e., $\alpha = 100\%$. As Figure 8 highlights, TPS decreases as the injected latency increases. More specifically, the TPS decreases by 26%, 46%, and 67% when moving from no injected latency to 250 ms for Zunesha, Zunesha-Relayer, and Cosmos, respectively. The distinct difference in the decreased amount of TPS also indicates that the relayer is particularly susceptible to high latency. It is noteworthy that the TPS of Zunesha and Zunesha-Relayer exceeds that of Cosmos, highlighting the efficiency of Zunesha as a Multi-chain architecture. Furthermore, Zunesha has the potential to achieve higher inter-chain TPS if the basechain has a higher TPS capacity.

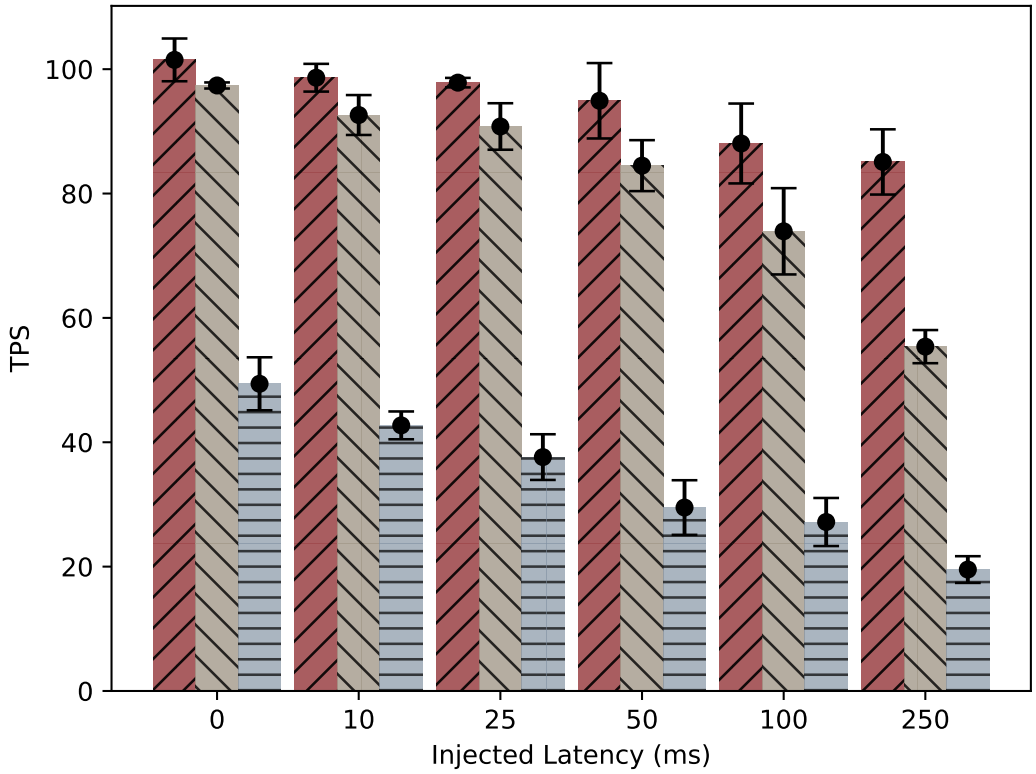


Figure 8. Inter-chain transaction throughput in Zunesha, Zunesha-Relayer and Cosmos with different amount of injected latencies.

Compared to Zunesha-Relayer, Zunesha achieves at most 40% latency reduction and 53% throughput improvement. In real scenarios where the distance between relayers and the blockchains is uncertain, it is more reasonable to have CNodes directly forward inter-chain transactions with proper incentives and safety guarantees in the multi-chain network. Zunesha significantly improves interoperability performance and is more resilient to high latency.

6.4. Reliability and robustness

To show Zunesha's reliability and robustness, we test whether inter-chain transactions can be appropriately processed under hostile attacks and TI. Both mainchain-to-subchain and subchain-to-subchain transactions undergo the robustness test in the inter-chain transaction experiment. In this experiment, one CNode is set as the attacker, and we inject 500 ms latency into another CNode to simulate the TI. It will become a straggler as it gradually falls behind other CNodes due to the injected latency. Initially, all nodes demonstrate correct behaviors. After 60 seconds, the malicious node begins sending invalid transactions and refuses to vote for the blocks proposed by others. After another 60 seconds, the straggler falls behind and starts to catch up after the following 60 seconds. A client is established to send one transaction every second and collect the end-to-end latency during the experiment, which is the interval from the finalization of an inter-chain transaction on the source blockchain to that on the target chain. Each experiment lasts for 300 seconds, and both experiment types are run ten times.

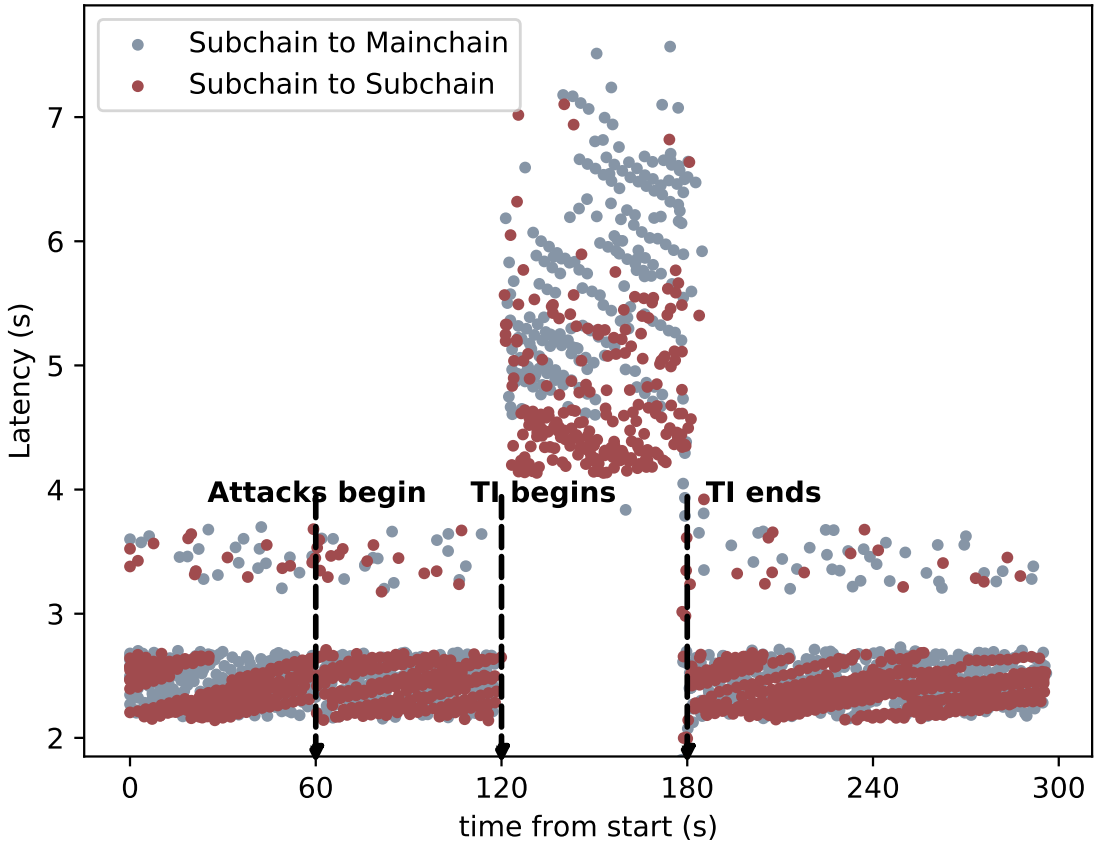


Figure 9. Inter-chain latency under malicious attacks and timing issues.

Figure 9 demonstrates that in the first 60 seconds, as nothing goes wrong, the transaction latency is approximate to the end-to-end latency in the leftmost bar of Figure 7. Since the experiment is conducted ten times and the operation latency in the distributed system is not constant and is affected by many factors such as system busyness, the latency data in Figure 9 are sometimes scattered. When the adversary node begins to send malicious transactions and refuses to vote, the latency remains almost unchanged because a transaction only needs a majority of CNodes to vote. Although the adversary could initiate arbitrary attacks, they will not get confirmed by honest CNodes. Therefore, the malicious node cannot sabotage the blockchains' reliability, safety, and liveness in Zunesha. However, the latency increases when the straggler falls behind since it offers the last vote for the transaction. After it catches up, the inter-chain transactions are completed at a normal speed. In both mainchain-subchain and subchain-subchain transactions, no blockchain forking happens, and no malicious transaction gets finalized. This experiment also shows the correctness and feasibility of subchain-to-subchain transactions.

Overall, in an extreme situation where the block needs the straggler for the last vote, the inter-chain transactions will still be correctly completed, albeit requiring more time than usual to wait for the last vote.

7. Conclusion

This paper introduces Zunesha, a portable and relayer-free multi-chain architecture significantly enhancing inter-chain transaction performance. Zunesha facilitates the seamless integration of multi-chain scalability solutions into existing blockchains with the smart-contract-based multi-chain toolkit. With Zunesha, the scalability of the target blockchain grows linearly alongside the number of subchains. Moreover, Zunesha streamlines interoperability by optimizing the inter-chain transaction process from three to two roles, improving efficiency and outperforming Cosmos. To address timing issues that can cause inconsistencies in multi-chain networks, we have devised the innovative dynasty-based consensus node set verification protocol to mitigate timing issues. Notably, Zunesha has been successfully implemented in an existing

public blockchain, offering excellent scalability in terms of throughput.

Acknowledgment

The authors would like to thank the anonymous reviewers for their comments. This work was supported by the National Key R&D Program of China under Grant 2023YFB2703800 and the Beijing Natural Science Foundation under Funding No. IS23055. The contact author is Zhen Xiao.

Conflicts of interests

The authors declared that they have no conflicts of interests.

Authors'contribution

Conceptualization, P.L., Q.D., Z.H., S.G.; Methodology, P.L., J.L., Q.D., Z.H., S.G.; Formal analysis, P.L.; Software, P.L., J.L.; Visualization, Investigation, Q.D.; Data curation, P.L.; Writing-Original draft preparation, P.L.; Writing-Reviewing and Editing, J.L., Z.X., Z.H., S.G.; Supervision, Z.X.; Project Administration, J.L., Z.X.; All authors have read and agreed to the published version of the manuscript.

References

[1] Lee LH, Braud T, Zhou P, Wang L, Xu D, *et al.* All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda. *arXiv preprint arXiv* 2021, 2110.05352.

[2] Wood G. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper* 2014, 151(2014):1–32.

[3] Weyl EG, Ohlhaver P, Buterin V. Decentralized Society: Finding Web3's Soul. *Available at SSRN* 2022, 4105763.

[4] Ethereum Foundation. Blockchain Scalability Trilemma, 2022. Available: https://ethereum.org/en/upgrades/vision/ (accessed on 17 January 2023).

[5] Xu J, Xie Q, Peng S, Wang C, Jia X. AdaptChain: Adaptive Scaling Blockchain With Transaction Deduplication. *IEEE Trans. Parallel Distrib. Syst.* 2023, 34(6):1909–1922.

[6] Hu Z, Xiao Z. Dino: A Block Transmission Protocol with Low Bandwidth Consumption and Propagation Latency. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, London, United Kingdom, May 02–05, 2022, pp. 1319–1328.

[7] Hu Z, Guan S, Xu W, Xiao Z, Shi J, *et al.* A Data Flow Framework with High Throughput and Low Latency for Permissioned Blockchains. In *2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)*, Hong Kong, China, July 18-21, 2023, pp. 1–12.

[8] Zamani M, Movahedi M, Raykova M. Rapidchain: Scaling blockchain via full sharding. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, Toronto, Canada, October 15–19, 2018, pp. 931–948.

[9] Huang H, Peng X, Zhan J, Zhang S, Lin Y, *et al.* BrokerChain: A Cross-Shard Blockchain Protocol for Account/Balance-based State Sharding. In *IEEE INFOCOM*, London, United Kingdom, May 02–05, 2022, pp. 1968–1977.

[10] Buterin V. On-chain scaling to potentially 500 tx/sec through mass tx validation. *Ethereum Blog* 2018 .

[11] Poon J, Buterin V. Plasma: Scalable autonomous smart contracts. *White paper* 2017, 1–47.

[12] Wood G. Polkadot: Vision for a heterogeneous multi-chain framework. *White Paper* 2016, 21:2327–4662.

[13] Kwon J, Buchman E. Cosmos whitepaper. *A Netw. Distrib. Ledgers* 2019 .

[14] Rocket T. Snowflake to avalanche: A novel metastable consensus protocol family for cryptocurrencies Available [online].[Accessed: 4-12-2018], 2018 .

[15] De la Rocha A, Kokoris-Kogias L, Soares JM, Vukolić M. Hierarchical consensus: A horizontal

scaling framework for blockchains. In *2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, Bologna, Italy, July 10, 2022, pp. 45–52.
[16] Liu W, Cao B, Peng M, Li B. Distributed and Parallel Blockchain: Towards A Multi-Chain System with Enhanced Security. *IEEE Trans. Dependable Secure Comput.* 2024, (1):723–739.
[17] Belchior R, Vasconcelos A, Guerreiro S, Correia M. A survey on blockchain interoperability: Past, present, and future trends. *ACM Comput. Surv.* 2021, 54(8):1–41.
[18] Wang G. Sok: Exploring blockchains interoperability. *Cryptology ePrint Archive* 2021 .
[19] Zhou Q, Huang H, Zheng Z, Bian J. Solutions to scalability of blockchain: A survey. *Ieee Access* 2020, 8:16440–16455.
[20] Umran SM, Lu S, Abduljabbar ZA, Nyangaresi VO. Multi-chain blockchain based secure data-sharing framework for industrial IoTs smart devices in petroleum industry. *Internet of Things* 2023, 24:100969.
[21] Gilad Y, Hemo R, Micali S, Vlachos G, Zeldovich N. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th symposium on operating systems principles*, Shanghai, China, October 28, 2017, pp. 51–68.
[22] Li C, Li P, Zhou D, Xu W, Long F, *et al.* Scaling nakamoto consensus to thousands of transactions per second. *arXiv preprint arXiv* 2018, 1805.03870.
[23] Poon J, Dryja T. The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments, 2016. Available: https://www.bitcoinlightning.com/wp-content/uploads/2018/03/lightning-network-paper.pdf (accessed on 17 January 2023).
[24] Raiden Network Team. Raiden Network, 2022. Available: https://raiden.network/ (accessed on 17 January 2023).
[25] Liquidity Network Team. Liquidity Network, 2022. Available: https://liquidity.network/ (accessed on 17 January 2023).
[26] Sion SI, Zhang K, April A, Lutete TM, Bouchard C. A comprehensive review of multi-chain architecture for blockchain integration in organizations. In *International Conference on Business Process Management*, Springer, Cham, September 01, 2024, pp. 5–24.
[27] Qu L, Wen F, Huang H, Wang Z. Aggregation-chain: a consortium blockchain based multi-chain data sharing framework with efficient query. *Cluster Comput.* 2025, 28(1):1–16.
[28] Darshan M, Amet M, Srivastava G, Crichigno J. An architecture that enables cross-chain interoperability for next-gen blockchain systems. *IEEE Internet Things J.* 2023, 10(20):18282–18291.
[29] Parity Technologies. Substrate, 2022. Available: https://substrate.io/ (accessed on 17 January 2023).
[30] Kokoris-Kogias E, Jovanovic P, Gasser L, Gailly N, Syta E, *et al.* Omniledger: A secure, scale-out, decentralized ledger via sharding. In *2018 IEEE Symposium on Security and Privacy (SP)*, San Francisco, CA, USA, May 20-24, 2018 pp. 583–598.
[31] Ethereum Foundation. Ethereum the Merge, 2022. Available: https://ethereum.org/en/upgrades/merge/ (accessed on 17 January 2023).
[32] Li P, Song M, Xing M, Xiao Z, Ding Q, *et al.* SPRING: Improving the Throughput of Sharding Blockchain via Deep Reinforcement Learning Based State Placement. In *Proceedings of the ACM on Web Conference 2024*, Singapore, May 13–17, 2024, pp. 2836–2846.
[33] Cheng F, Xiao J, Liu C, Zhang S, Zhou Y, *et al.* Shardag: Scaling dag-based blockchains via adaptive sharding. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, Utrecht, Netherlands, May 13–16, 2024, pp. 2068–2081.
[34] Ronin Bridge Team. Ronin Bridge, 2022. Available: https://bridge.roninchain.com/ (accessed on 17 January 2023).
[35] xDai Bridge Team. xDai Bridge, 2022. Available: https://bridge.gnosischain.com/ (accessed on 17 January 2023).
[36] Gravity Bridge Team. Gravity Bridge, 2022. Available: https://github.com/Gravity-Bridge/Gravity-Bridge (accessed on 17 January 2023).
[37] Singh A, Click K, Parizi RM, Zhang Q, Dehghantanha A, *et al.* Sidechain technologies in blockchain networks: An examination and state-of-the-art review. *J. Netw. Comput. Appl.* 2020, 149:102471.

[38] Motepalli S, Freitas L, Livshits B. Sok: Decentralized sequencers for rollups. *arXiv preprint arXiv* 2023, 2310.03616.
[39] Duan L, Sun Y, Ni W, Ding W, Liu J, *et al.* Attacks against cross-chain systems and defense approaches: A contemporary survey. *IEEE/CAA J. Autom. Sin.* 2023, 10(8):1647–1667.
[40] Dwork C, Lynch N, Stockmeyer L. Consensus in the presence of partial synchrony. *J. ACM* 1988, 35(2):288–323.
[41] Ethereum Foundation. EIP-20, 2022. Available: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md (accessed on 17 January 2023).
[42] Deirmentzoglou E, Papakyriakopoulos G, Patsakis C. A survey on long-range attacks for proof of stake protocols. *IEEE Access* 2019, 7:28712–28725.
[43] Li C, Palanisamy B, Xu R, Duan L, Liu J, *et al.* How hard is takeover in dpos blockchains? understanding the security of coin-based voting governance. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, Copenhagen, Denmark, November 26 – 30, 2023, pp. 150–164.
[44] Ethereum Foundation. EIP-155, 2022. Available: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-155.md (accessed on 17 January 2023).
[45] Geng Y, Qin B, Wang Q, Shi W, Wu Q. Subsidy Bridge: Rewarding Cross-Blockchain Relayers with Subsidy. In *International Conference on Information and Communications Security*, Springer, Singapore, October 20, 2023, pp. 571–589.
[46] Zhang J, Gao J, Li Y, Chen Z, Guan Z, *et al.* Xscope: Hunting for Cross-Chain Bridge Attacks. *arXiv preprint arXiv* 2022, 2208.07119.
[47] Boneh D, Gentry C, Lynn B, Shacham H. Aggregate and verifiably encrypted signatures from bilinear maps. In *Advances in Cryptology—EUROCRYPT 2003: International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4–8, 2003 Proceedings 22*, Springer, Berlin, Heidelberg May 13 2003 pp. 416–432.
[48] Mirkovic J, Reiher P. A taxonomy of DDoS attack and DDoS defense mechanisms. *ACM SIGCOMM Comput. Commun. Rev.* 2004, 34(2):39–53.
[49] Buterin V, Griffith V. Casper the friendly finality gadget. *arXiv preprint arXiv* 2017, 1710.09437.
[50] Buterin V, Hernandez D, Kamphefner T, Pham K, Qiao Z, *et al.* Combining GHOST and casper. *arXiv preprint arXiv* 2020, 2003.03052.
[51] Ethereum Foundation. Go Ethereum, 2022. Available: https://geth.ethereum.org/ (accessed on 17 January 2023).
[52] Benet J. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv* 2014, 1407.3561.
[53] Sheng P, Wang X, Kannan S, Nayak K, Viswanath P. TrustBoost: Boosting Trust among Interoperable Blockchains. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, Copenhagen, Denmark, November 26–30, 2023, pp. 1571–1584.
[54] Theta Labs. Theta Blockchain Network, 2022. Available: https://github.com/thetatoken/theta-protocol-ledger (accessed on 17 January 2023).
[55] Long J, Wei R. Scalable BFT consensus mechanism through aggregated signature gossip. In *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, Seoul, Korea (South), May 14-17, 2019, pp. 360–367.
[56] Yin M, Malkhi D, Reiter MK, Gueta GG, Abraham I. Hotstuff: Bft consensus with linearity and responsiveness. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, Toronto ON, Canada, July 29 2019 – August 2 2019, pp. 347–356.
[57] Ethereum Foundation. EIP-2387, 2022. Available: https://eips.ethereum.org/EIPS/eip-2387 (accessed on 17 January 2023).
[58] Cason D, Fynn E, Milosevic N, Milosevic Z, Buchman E, *et al.* The design, architecture and performance of the Tendermint Blockchain Network. In *2021 40th International Symposium on Reliable Distributed Systems (SRDS)*, Chicago, IL, USA, September 20-23, 2021, pp. 23–33.
[59] Cosmos Network. Cosmos SDK, 2022. Available: https://v1.cosmos.network/sdk (accessed on 17 January 2023).

[60] Linux Foundation. TC, the Traffic Control in the Linux Kernel, 2022. Available: https://man7.org/linux/man-pages/man8/tc.8.html (accessed on 17 January 2023).