Journal of Network and Computer Applications 217 (2023) 103686

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Review

A survey of state-of-the-art sharding blockchains: Models, components, and attack surfaces

Yi Li, Jinsong Wang *, Hongwei Zhang

School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, 300384, China
Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology, Tianjin, 300384, China
National Engineering Laboratory for Computer Virus Prevention and Control Technology, Tianjin, 300457, China

ARTICLE INFO

Keywords:
Blockchain
Sharding
Attack surface
Countermeasures

ABSTRACT

Blockchain has been widely used in various fields, such as health management, finance, the Internet of Things (IoT), identity management, and supply chains. However, the performance of blockchain, including its throughput and transaction processing latency, has hindered its large-scale practical application, which is known as the scalability issue of blockchain. Many on-chain and off-chain solutions have been proposed to solve this problem, such as directed acyclic graph (DAG) blockchain and Rollup schemes. Despite these schemes' ability to enhance the scalability of the blockchain, they compromise on other essential properties. The consensus is that the DAG blockchain can never provide deterministic security for transactions, and Rollup is disputed because it makes a trade-off of decentralization for scalability. Sharding technology is a proposed solution for these limitations, promising to tackle the scalability issue while maintaining security and decentralization at the same time. To date, no survey has been done on sharding design models and their key components and corresponding attack surfaces. To fill this gap, in this survey, we first provide a detailed analysis and comparison of both permissioned and permissionless sharding blockchains. Then, we describe the common building blocks chosen by the above sharding schemes as key components. Finally, we investigate how these components may be attacked and suggest corresponding countermeasures. Several open challenges are also given as future research directions. The survey aims to help researchers, engineers, and educators quickly grasp a consolidated body of knowledge about current sharding blockchain development.

1. Introduction

Blockchain, as the underlying technology of Bitcoin (Nakamoto, 2008), has attracted significant attention in academia and industry. Blockchain has several main features, including decentralization, transparency, anonymity, tamper resistance, auditability, and security. It is therefore used in multiple fields, such as healthcare (Qu, 2022; Liu et al., 2022), finance (Tschorsch and Scheuermann, 2016; Yao, 2018), the Internet of Things (IoT) (Liu et al., 2023, 2022c), e-voting (Lee et al., 2016; Kshetri and Voas, 2018), supply chains (Perboli et al., 2018; Gao et al., 2022), and identity management (Liu et al., 2020a; Liao et al., 2022). Despite some achievements of blockchain in these areas, mainstream blockchain systems have not achieved the performance of traditional non-blockchain systems in terms of throughput and processing latency. The scalability issue of blockchain has become one of the major factors hindering its large-scale practical application (Singh et al., 2020). For example, the current throughput of the widely used Bitcoin and Ethereum is only 3–5 and 15 transactions per second (TPS), respectively. In comparison, popular payment systems such as Visa have a peak throughput of 24,000 TPS. If more resources are added to non-blockchain systems like this, throughput could be further improved. Recently, many researchers have proposed methods to enhance the scalability of blockchain. Some of these methods scale blockchain by improving its core components, such as by adopting a better block structure (Lombrozo et al., 2015; Cao et al., 2020) or reducing the communication overhead of the consensus algorithm (Eyal et al., 2016; Kogias et al., 2016) and the peer-to-peer (P2P) network (Zhou et al., 2023; Qiu et al., 2022). As all these methods run on-chain, they are called layer-1 scaling, in which layer 1 refers to the scaled blockchain. In contrast, layer-2 scaling separates the computation from layer 1. The blockchain is only treated as a secure data layer. For example, a payment channel (Ge et al., 2022; Sivaraman et al., 2018) and Rollup (Thibault et al., 2022; Xu and Chen, 2022) process batch transactions off-chain and then settle the status on the blockchain. The above two kinds of solutions are hosted on layer 1 for security and data

* Corresponding author at: School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, 300384, China.
E-mail addresses: 1024liyi@stud.tjut.edu.cn (Y. Li), jswang@tjut.edu.cn (J. Wang), hwzhang@email.tjut.edu.cn (H. Zhang).

https://doi.org/10.1016/j.jnca.2023.103686
Received 8 November 2022; Received in revised form 2 May 2023; Accepted 13 June 2023
Available online 22 June 2023

Table 1
Comprehensive Comparison of surveys of sharding Blockchain.

Surveys	Covered topics					
	Blockchain type	Scalability definition	Protocol review	Key components	Attacks	Countermeasures
Avarikioti et al. (2019)	Permissionless and Permissioned	✓	✓	✓	×	×
Yu et al. (2020)	Permissionless	×	✓	✓	×	×
Wang et al. (2019)	Permissionless	×	✓	✓	×	×
Han et al. (2021)	Permissionless	×	×	✓	✓	×
Liu et al. (2022a)	Permissionless and Permissioned	×	×	✓	✓	✓
This work	Permissionless and Permissioned	✓	✓	✓	✓	✓

accessibility. In addition, some layer-2 schemes are entirely separated from layer 1 by creating independent parallel blockchains, such as sidechains (Singh et al., 2020; Gaži et al., 2019) and cross-chains (Tian et al., 2021). These parallel chains are in independent consensus and communicate with layer-1 through bridges.

As described by the CAP theory in a distributed database (Gilbert and Lynch, 2002), a robust distributed system can only simultaneously satisfy two of the three properties of consistency, availability, and partition tolerance. Similarly, in the blockchain context, this problem is known as a scalability trilemma; that is, blockchain cannot satisfy the three properties of scalability, decentralization, and security at the same time (Altarawneh et al., 2020). Therefore, the above scalability schemes always make a trade-off among the three properties of scalability, decentralization, and security when improving scalability. The sharding technique, a common technique used to expand performance horizontally in the traditional database domain, was first introduced into blockchain by Luu et al. (2016). By splitting nodes into smaller groups, called shards, and processing transactions in parallel, sharding can significantly improve the performance of the blockchain. The founder of Ethereum even called it the most promising solution to break the impossible trilemma and solve the blockchain scalability issue (Avarikioti et al., 2019; Buterin, 2021). Sharding blockchain has been applied to the IoT (Yang et al., 2022), medicine (Ramanan et al., 2022), wireless networks (Cai et al., 2022a), and many other fields. Ethereum, the most popular blockchain platform, has also adopted sharding as the direction for future upgrades.

There have been some surveys on sharding blockchains. For example, the authors of Yu et al. (2020) reviewed the advantages of existing sharding schemes and compared the performance of each scheme. However, they have only compared limited sharding schemes in permissionless blockchains, which overlooks the fact that permissioned blockchains are more often used in industries that require high scalability (Dabbagh et al., 2021). Other authors (Avarikioti et al., 2019) tried to formalize components from existing sharding schemes to establish a standard framework for sharding schemes. The authors of Wang et al. (2019) proposed a workflow for a sharding blockchain and evaluated some existing sharding schemes. These studies focused either on comparing limited sharding schemes or only on the building blocks of sharding schemes. More importantly, the above surveys ignored the potential attack surfaces caused by the extra complexity in sharding compared with the mature traditional blockchain Avarikioti et al. (2019). Han et al. (2021) discusses potential attacks in permissionless sharding but only analyzes the security risks in cross-shard transaction processing without providing corresponding countermeasures. Similar issues are also present in Liu et al. (2022a). Table 1 compares this work and the above-related surveys in terms of comprehensiveness.

Therefore, this paper provides a comprehensive review of the permissioned and permissionless sharding blockchains, including an analysis of design models and performance comparisons. Then, we outline common building blocks in these schemes and present key components. Finally, we explore the possible way to attack key components and take corresponding countermeasures. In summary, our contributions include the following:

1. We give an overview of the blockchain scalability issue and provide a new definition of it in the sharding blockchain context.
2. We comprehensively review the well-known sharding schemes and then classify them by sharding type and blockchain type.
3. We propose the key components of a sharding blockchain and discuss the difference between the permissioned and permissionless sharding blockchains.
4. We explore the key components' attack surfaces and give the corresponding defense methods.

The rest of the paper is organized as follows. Section 2 introduces the scalability issue in blockchain as background knowledge for the following study. Section 3 provides a comprehensive review of sharding blockchain schemes, including permissioned and permissionless blockchains. Section 4 presents the key components of a sharding blockchain and compares the differences between the two types of blockchains. Section 5 explores ways to attack these key components and gives the corresponding countermeasures. Section 6 discusses insights and future research directions. Section 7 concludes this survey.

2. Blockchain scalability

Before discussing sharding, we need to briefly introduce blockchain scalability. Although there have been many studies and reviews on the blockchain scalability issue (Sanka and Cheung, 2021; Zhou et al., 2020; Croman et al., 2016; Nasir et al., 2022; Pandey et al., 2022; Kuzlu et al., 2019; Sohrabi and Tari, 2020; Xie et al., 2019), scalability remains poorly defined. Currently, the definition of blockchain scalability can be classified into two categories. Some researchers believe scalability is a composite term including throughput, latency, storage, and other system indexes. Such as Sanka and Cheung (2021), Zhou et al. (2020) and Xie et al. (2019), equate scalability with system metrics like throughput, storage overhead, and read/write delays. In Zheng et al. (2018), poor scalability is characterized by low throughput, high storage overhead, and transaction and network delays. The second category defines scalability as a fundamental blockchain property akin to decentralization and immutability. The definition in Nasir et al. (2022) uses the established concept of scalability in the database field. However, it only considers aspects such as throughput and storage overhead as vertical scalability, omitting that these metrics also measure horizontal scalability. Vitalik, the founder of Ethereum, defined it as the ability of the blockchain to process transactions beyond the processing capacity of a single node (Buterin, 2021). Pandey et al. (2022), Kuzlu et al. (2019) and Sohrabi and Tari (2020) considered that a blockchain has scalability when its processing ability increases with the number of participating nodes without affecting the performance. The comparison of the scalability definitions is shown in Table 2.

We consider scalability to be a single systemic property of blockchain. Specifically, when we say that a particular blockchain has scalability, we mean that its system metrics will exhibit beneficial changes as the system resources (including computing power, storage capacity, and network bandwidth) increase. These beneficial changes may encompass increases in transaction throughput, reductions in storage overheads, improvements in network bandwidth utilization, and decreases in transaction latency. These four essential metrics significantly impact the user's quality of experience (QoE) (Bano et al., 2017)

Table 2
scalability definitions comparison.

Definition	Reference
Scalability is a composite term including throughput,latency,storage,and other system indexes	Sanka and Cheung (2021), Zhou et al. (2020), Croman et al. (2016), Zheng et al. (2018), Xie et al. (2019)
Scalability is a single-system property. Like decentralization and immutability,scalability can be observed on system indexes like throughput,latency,and storage costs	Nasir et al. (2022), Pandey et al. (2022), Kuzlu et al. (2019), Sohrabi and Tari (2020)

Throughput Blockchain throughput is the rate at which transactions are added to the ledger, usually measured by transaction confirmations per second (TPS). It is an essential indicator of the blockchain transaction processing capacity. The throughput of a blockchain is related to the number of transactions contained in one block, and the time interval between the raising of two blocks. Bitcoin, for example, has a block interval of approximately 10 min, and each block contains approximately 2000 transactions. Therefore, the throughput is 3–5 TPS.

Latency Latency, also known as confirmation time, is the time it takes for a transaction sent from a client to be included in blocks. Typically, as the number of transactions sent by clients increases, the network load increases, and the time required for transaction confirmation becomes longer. In the case of Bitcoin, for example, the block interval is 10 min, which means that the average transaction latency is approximately 10 min. Because Bitcoin uses proof-of-work (POW) consensus, which does not have final confirmation, it takes up to 50–60 min to fully confirm one transaction.

Storage The full node in the blockchain stores all historical block records, from the genesis block to the most current block, which causes the continuous growth cost of storage resources. As of September 2022, when this paper was written, the historical records of Bitcoin and Ethereum were 625 GB and 1.2 TB, respectively. Bitcoin and Ethereum are growing at 10 GB per month and 40 GB per month, respectively, which undoubtedly makes running a full node extremely expensive in terms of hardware. Moreover, the bootstrap time for new nodes is also increase, which may lead to the mainstreaming of light nodes that only store block headers, making the network less decentralized.

Network Blockchain broadcasts new transactions and blocks through the P2P network. Each node must download all transactions and blocks to verify and forward them. Therefore, as the number of transactions increases, the bandwidth requirements on the nodes become higher, which significantly limits the blockchain's ability to handle more transactions.

The final goal of improving scalability is by increasing system resources to improving the transaction confirmation speed and throughput while maintaining or reducing the node overhead in the storage and network without compromising decentralization and security. In addition, based on the insights gleaned from the database field, we further classify scalability into two types (Henning and Hasselbring, 2022; Nasir et al., 2022). A blockchain is horizontally scalable if its system metrics improve as the number of participating nodes increase. Instead, suppose the system metrics of the blockchain can be improved by enhancing the computational power of the nodes and storage space or by adjusting parameters such as the block size or block interval; in that case, the blockchain has vertical scalability.

3. Overview of sharding blockchain

Nodes are responsible for three activities to keep the blockchain running properly: downloading and verifying transactions, propagating valid transactions or blocks, and maintaining local storage. Specifically, the node downloads every transaction received from the network and checks the validity. Afterward, it adds the transaction to the local pool of transactions while propagating it to its peer nodes. This way, each node in the blockchain repeatedly verifies all transactions. This mechanism is one reason that blockchain can build trustless trust. In addition, each node maintains its local ledger, and the consensus algorithm ensures that every node catches the same state. A multi-copy gives redundancy protection to the historical ledger and allows nodes to propose new blocks independently, which is another critical reason to give blockchain the decentralization feature. When there are more pending transactions in the network, the nodes need more network resources to pass all transactions across the network. More computing power is needed to verify transactions and maintain the quality of service. Furthermore, extra space is also needed to store the new transactions. Regardless, the performance of each node is limited (Monte et al., 2020). It cannot simply be strengthened by using better machines, which leads to super nodes and centralization. Therefore, in the current blockchain mechanism, each node is a scalability bottleneck.

Sharding was initially proposed for traditional distributed databases and is a well-researched scalability technique. Popular databases like MongoDB, MySQL, and Bigtable (Croman et al., 2016) use sharding to reduce the pressure on the central server. They divide data into disjointed segments and store them separately in different servers, thus improving performance. However, the technology cannot be directly used in the blockchain owing to the different adversary models of blockchain. The malicious node in the database is weak and can only withhold messages or go offline. In contrast, malicious nodes in the blockchain can perform arbitrary attacks. Therefore, additional mechanisms must be added to sharding to ensure its robustness. Specifically, the sharding technique divides the nodes into multiple shards. Then, the associated activities and objects are divided into multiple independent partitions. Objects include transactions, blocks, and ledgers. Activities include transaction verification, data broadcasts, and storage updates. Nodes in different shards only forward, process, and store transactions related to their shards, thus reducing the computational, network, and storage resources required. Multiple shards process the assigned activities in parallel, ultimately increasing the scalability of the blockchain.

The core idea behind sharding is to divide and conquer, improving scalability from computation, networking, and storage (Avarikioti et al., 2019; Xie et al., 2019). Therefore, the sharding blockchain can be divided into three types: network sharding, transaction sharding, and state sharding. Network sharding involves dividing the entire blockchain network into smaller shards, also called committees. Instead of downloading and verifying all transactions in the network, nodes receive only the portion of assigned transactions, which reduces network bandwidth requirements, especially in public blockchains that are sensitive to network overhead, such as Bitcoin and Ethereum. Therefore, network sharding is often considered the basis for the latter two kinds of sharding in public blockchains (Huang et al., 2022a). However, this is not necessarily the case in permissioned blockchains, where the nodes are stable and resourceful. Transaction sharding divides valid transactions into multiple disjointed sets and assigns them to different committees. Each committee only reaches a consensus within the current shard. The sharding reduces the computational load of individual nodes and lowers the threshold of node computational resources. Blockchain is essentially a state transition machine, and the current state of a blockchain can be calculated from the genesis state and historical blocks. State sharding, also known as storage sharding, means that instead of all nodes storing the entire historical ledger, each

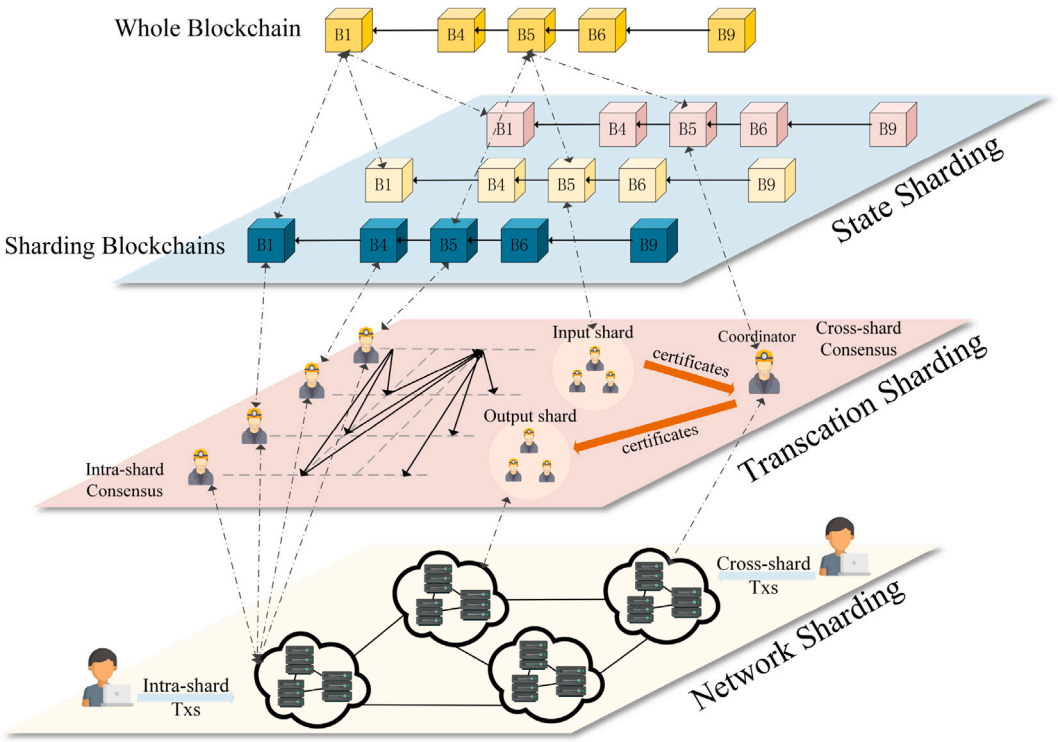

Fig. 1. Relationship between the three types of sharding.

committee stores the ledger associated with this shard separately. The overview of the relationship between the three types of sharding is shown in Fig. 1.

The three sharding types are related to each other but are not necessarily adopted simultaneously. The early version of Ethereum sharding with shards computation enabled requires nodes to join two kinds of networks simultaneously: one is a horizontal network consisting only of nodes within the shard, which is used to propagate consensus-related information; the other is a vertical network involving all nodes, which is used to transmit transactions and blocks. So, it performs transaction and state sharding. Now Project Danksharding in Ethereum (Jon, 2022) only performs network and storage sharding. The blockchain acts as a highly available and secure data layer for Rollup. Project Zilliqa, the first industry-sharding public blockchain, conducts only transaction and network sharding. Each node still needs to keep all historical data (Zhou et al., 2020). Therefore, by combining the above techniques, we can propose four sharding patterns: (1) transaction sharding and state sharding,(2) network sharding and transaction sharding,(3) network sharding and state sharding, and (4) adopting three sharding types at the same time. As shown in Fig. 2.

These are not all theoretical combinations, but after reviewing the existing sharding schemes, we find that these four combinations cover all practical sharding patterns. We discuss the details in Sections 3.1 and 3.2. We refer to sharding patterns 1, 2, and 3 partial sharding. As they improve only some factors that limit scalability, their upper bound of performance can be detected. Sharding pattern 4 can be called full sharding.

Blockchains can be classified as permissionless or permissioned blockchains. Permissionless blockchains, such as Bitcoin and Ethereum, allow anyone to run nodes to join or leave the network anytime. All nodes can easily access historical blocks or current transactions in the network. In contrast, permissioned blockchains, such as Hyperledger Fabric, can only enter the network after approval by an authority node. While the two types of blockchain have scalability issues, their dilemmas are not the same. We separately analyze major sharding schemes in the two blockchain types and describe the design models of each in the below section.

3.1. Sharding preliminaries

In order to provide a more concise description of the following sharding protocol, we first establish a general abstract model for blockchain. Assuming there are n participating nodes, and f of them are Byzantine. If the Byzantine nodes can quickly select corrupt targets based on the current system situation, and after a period, the target node is entirely controlled by the Byzantine node. We consider it to be slowly adaptive (Buterin, 2021). Conversely, if a Byzantine node can only select an attack target far in advance, it can only implement static corruption. Transactions, blocks, and other shard-related messages in the network are propagated through Synchronous, Partially Synchronous, or Asynchronous Networks (Dwork et al., 1988). Suppose message m is sent at time t, the recipient in the Synchronous network will receive m at $t + \Delta$, where Δ is a finite and known network delay. While the network delay in Partially Synchronous is unknown and variable, the message will be delivered at $t + \Delta t$. In the Asynchronous network, m will eventually be received with an unknown time-bound. Considering that messages in sharding blockchains may propagate within and between shards, the following analysis is based on the overall network model of the system. Saving all received messages will consume at least x storage resources for each node.

3.2. Permissioned sharding blockchain

In permissioned blockchains, nodes do not necessarily have a trust relationship with each other. However, the identities of participating nodes have been verified by authority nodes. In addition, authority nodes are also responsible for constructing blocks. When the network size increases, the workload of authority nodes increases and becomes the scalability bottleneck.

3.2.1. Partial sharding

Meepo (Zheng et al., 2021) is a protocol that focuses on consortium sharding blockchain. Unlike the private blockchain, which is another type of permissioned blockchain, Meepo has participants that are no

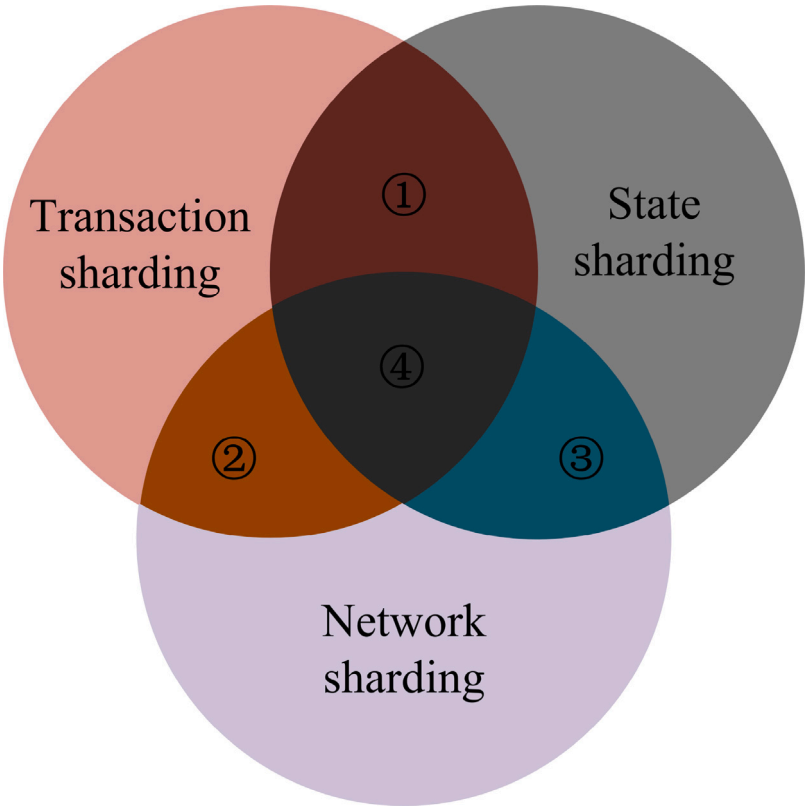

Fig. 2. Practical sharding type.

longer individual nodes but a cluster of servers. Sharding happens not among Meepo participants but within each member cluster. The nodes in each cluster only process the assignment transactions. Thus, transaction sharding is carried out. Transactions in Meepo are forwarded to all participant members, meaning that each member holds the complete blockchain state. However, every node in the cluster only stores a partial ledger. Therefore, Meepo can also be considered adopting state sharding. The cluster nodes of each member receive transactions in the same P2P network, and network sharding is not adopted. Therefore, this paper classifies Meepo as a partial sharding protocol rather than a self-claimed full sharding protocol. Meepo relies on multiple execution environments per member node to increase the throughput, which brings two significant enhancements to cross-shard transaction processing. First, the shards in the same cluster trust each other, reducing the overhead of signature generation and verification. Second, most servers in one cluster are on the same local area network (LAN), which allows the network model to be fully synchronized. The communication overhead incurred across the shards can also be ignored.

In addition, Meepo reserves two unique time slots before the next block is raised. One is the cross-epoch, where all cross-calls raised by cross-shard transactions are pooled into this time for batch processing. The other is replay-epoch, which removes erroneous transactions from blocks and re-executes them, thus ensuring the atomicity of transactions.

Like Meepo, Aeolus (Zheng et al., 2022) expands the single node in the traditional blockchain into a master–slave structure computing cluster, where the master server acts as the shard leader, responsible for transaction partitioning and delivering cross-shard transactions. Slave servers execute and store transactions. To facilitate cross-shard transactions, researchers add two new fields to the transaction. The shard flag is given by the transaction initiator, and the stage flag is formulated by the contract developer. Each transaction is divided into multiple stages. Every stage is processed in different shards, thus converting cross-shard transactions into intra-shard transactions. This approach is also known as distributed transactions. A stage is executed in only one shard, and different stages of multiple transactions can be executed in parallel in multiple shards, thus reducing transaction latency. Finally, each member takes the state roots of all the shards and computes them to obtain a new state.

3.2.2. *Full sharding*

AHL (Dang et al., 2019) is a full sharding protocol. It first introduces the trusted execution environment (TEE) to optimize sharding. By executing the sgx-read-rand function in SGX, the protocol generates an unbiased random number as the basis for dividing nodes into different shards and periodic shard reconfiguration. Randomly allocating nodes is the key to keeping adversaries from exceeding the consensus protocol's tolerance. In addition, Byzantine nodes, which can perform arbitrary malicious behaviors, are weakened by TEE into general nodes, which can only omit information without choice. The weakened adversary makes the consensus fault tolerance reach $(n-1)/2$, where n is the number of total nodes in the network. Moreover, this additional security guarantee dramatically reduces the communication complexity of the PBFT consensus to O(n), significantly improving the blockchain's performance to over 3000 TPS. In addition to the above innovations, AHL uses the 2PL and 2PC protocols to process the cross-shard transaction to achieve atomicity and isolation of transactions. It also ensures the liveness of cross-shard transactions, even under the assumption of malicious coordinators.

The above random node assignment approach can only provide probabilistic security to the shards. Therefore, Sharper (Amiri et al., 2021) proposed the geographical-based assigning method to provide deterministic security with pre-determined fixed shards. In addition, communication latency between geographically close nodes is low, facilitating the propagation of transactions and blocks. Despite some benefits, such a node assigning method cannot balance the number of nodes in each shard and is easily attacked by adversaries. Therefore,

Table 3
Comparison of permissioned sharding blockchain protocols.

	Dang et al. (2019)	Amiri et al. (2021)	Zheng et al. (2021)	Zheng et al. (2022)
Account Model	Account-base	Account-base	Account-base	Account-base
Contract Enable	Yes	!	Yes	Yes
Network Model*	Part Sync	Async	Sync	Sync
Byzantine Tolerance**	(N-1)/2	N/3	N/3	N/3
Storage	x/K	x/K	x/k	x/k
Throughput	3000 tx/s[1]	27000 tx/s[2]	120000 tx/s[3]	95696 tx/s[3]
Transaction Latency	100 ms	240 ms	526 ms	684 ms
Communication	O(n)	O(n2)	O(n2)	O(n2)
Adversary	Slowly adaptive	Slowly adaptive	!	!

* The Network Model refers to the whole system;
**The Byzantine Tolerance and Communication refer to intra-shard consensus;
1 27 nodes/shard and 36 shards in total;
2 13 nodes/shard and 5 shards in total;
3 4 nodes/shard and 32 shards in total;
! means that the value is missing in the literature;
- means that the property does not apply to the given category.

the Sharper method can only secure shards if the number of nodes is large enough. Sharper is a full sharding protocol consisting of a set of clusters. Each cluster maintains part of the blockchain state, including multiple intra-shard transactions and cross-shard transactions. Note that cross-shard transactions are stored in every involved cluster. Transactions in Sharper are organized in the DAG, with each block only consisting of a single transaction. When the primary receives a new transaction, it assigns a sequence number and signed message with a timestamp to the transaction to ensure that different nodes have the same order of transactions. In contrast with AHL, Sharper processes cross-shard transactions in a distributed manner based on PBFT, but the client accepts multi-replay from all involved clusters. In addition, non-overlapping cross-shard transactions can be processed in parallel, and TEE can also be used to reduce further the communication overhead caused by the flattening PBFT protocol. As a result, Sharper achieves a TPS of up to 27,000.

Thus, the characteristics of the permissioned chain have led to various design models. For example, the above schemes are all based on the account model for sharding. Because most of the applications on the permissioned blockchain are based on smart contracts (Zheng et al., 2022), adopting the account model can manage contracts more conveniently. The nodes in the permissioned blockchain are relatively fixed, and the nodes' identities are endorsed by authority, so the chances of collusion among nodes are low. Therefore, the above schemes ignore performing shard reconfiguration to save much time and bandwidth waste. In addition, the nodes in the permissioned chain have good network conditions, so most of the schemes focus on transaction sharding and state sharding to achieve higher performance. The detailed performance comparison of permissioned sharding blockchain protocols can be found in Table 3.

3.3. Permissionless sharding blockchain

Compared to permissioned blockchains, permissionless blockchains make it more difficult to implement sharding. This is because the identity of participating nodes in the permissionless blockchain is unknown, and nodes can join or leave the network anytime. Each node trusts only itself, so the node itself needs to verify every transaction and every block in the network. This mechanism causes the transaction processing capacity of the entire network to be limited by single nodes. In addition, to maintain the permissionless blockchain's decentralized feature, an individual node's network resource requirements cannot be too high. Thus, network bandwidth is an essential factor limiting scalability, as transactions and blocks must be delivered to each node.

3.3.1. Partial sharding

Elastico (Luu et al., 2016) is the first permissionless sharding blockchain protocol. In this protocol, time is divided into different epochs. At the beginning of each epoch, nodes first run the POW algorithm to generate verifiable legal identities. Then nodes are then assigned to different committees according to the last s-bits of the identity. The innovations of Elastico that other sharding schemes have widely referenced are the directory committee and the final committee. The directory committee is responsible for coordinating the formation of the other subsequent committees. After the committee reaches a consensus on the current set of transactions, it passes the transaction set to the final committee. The final committee combines all legitimate transaction sets and broadcasts them to each node for storage. Elastico is the first attempt to apply sharding technology to blockchains. Despite achieving some scalability, it has a series of problems. Elastico cannot handle cross-shard transactions and may risk locking up funds. In addition, each node needs to store the global ledger, which may hinder its performance further.

Repchain (Huang et al., 2021) proposes that most current sharding schemes assume that all validators are homogeneous, but, each node is different in terms of hardware and network. Some nodes are actively involved in consensus, while others are inactive. Shards with more inactive nodes may be more vulnerable to takeover attackers. Therefore, Repchain proposes a reputation value scheme to encourage more contributions from honest, capable nodes. The reputation value is calculated based on how much a node contributes to the consensus. At the beginning of every epoch, nodes are sorted by a reputation value and then assigned to the shard with minimal size according to this rank. This method can ensure that the number and quality of nodes in the shard are relatively even. The nodes with the highest reputation in shards are chosen as leaders, responsible for intra-shard consensus and handling cross-shard transactions. Therefore, they accumulate reputation scores faster than other nodes, which give them a positive incentive. Repchain's intra-shard consensus is innovative. It uses a double-chain structure, including a transaction and reputation chains. Intra-shard consensus consists of Raft and CSBFT, so a shard stores both a transaction and reputation chain blockchains. The transaction chain uses Raft, which has high performance, and therefore generates the block TB at a higher speed. However, the Raft consensus cannot resist Byzantine nodes, so the TB block at this time can only be seen as a candidate block. The reputation chain uses the CSBFT consensus to generate the block RB at a medium speed, and RB contains the hash of multiple candidate TB blocks. In this way, the included TBs can finally be confirmed after the RB consensus is reached. The transaction chain can achieve high security while maintaining high performance. Repchain assumes that the adversary is static because of its emphasis on the variability of participating nodes. Some nodes are much more easily

corrupted than others. Therefore, the best strategy for an adversary is to control a fixed number of nodes, keep pretending to be normal nodes until one of them is elected leader, and then launch an attack. Another reason is that a reputation-based system is difficult to defend against a slowly adaptive adversary.

CycLedger (Zhang et al., 2020) also uses reputation scores as an incentive mechanism, similarly to Repchain. Specifically, a node's vote on a transaction is considered a contribution to the network. The closer a node's vote is to the final consensus result, the greater the contribution made by the node. Nodes with more computational resources can process more transactions and therefore gain more rewards, thus incentivizing all nodes with abundant computational resources to work honestly. However, CycLedger posits that Repchain's reputation-based node assignment method trades security for better incentives because it sacrifices some randomness. Therefore, it proposes a VRF-based node assignment scheme. In CycLedger, a referee committee is responsible for managing node identity and packing blocks. The node with the highest reputation in each shard is selected as the leader and acts as the representative of the shard to establish a network connection with the referee committee to receive transactions and other information. Interestingly, CycLedger creates an additional partial set in each shard. Nodes in this partial set monitor the behavior of the current shard leader, and when a leader is found to violate the protocol, one of the nodes in this set replaces the leader. Therefore, this set can also be considered the candidate leader. There are four types of nodes in the CycLedger: the referee committee node, the leader of the shard, the partial set members, and the remaining non-key nodes. Although the non-key node only needs to store the related transactions, the nodes in the referee committee still need to store all other committee transaction sets for verification and forwarding. Therefore, CycLedger can only be considered a partial sharding scheme.

SSchain (Chen and Wang, 2019) adopts a two-layer network structure, with the first layer being the root blockchain network. This structure is similar to the original bitcoin network, where the nodes store the complete ledger. The second layer is the sharding network, consisting of multiple shards, and transactions within each shard are processed by POW consensus. The root blockchain performs a second verification of the blocks generated by shards to prevent double-spend attacks. Specifically, the root block contains transactions in all newly generated blocks by shards. As the two layers do not run synchronously, a root block may contain multiple shard blocks, making the two-layer network form a DAG structure. The root chain provides security to the sharding network. The sharding network enhances the throughput of the whole system. Owing to the two-layer network architecture, SSchain no longer requires random node allocation to ensure the shard's security. Instead, nodes can freely choose to join multiple shards or the root chain. As the root chain provides security, SSchain proposes an incentive mechanism to ensure that most nodes' computing power focuses on the root chain to maintain consensus security and give each shard enough nodes to process transactions simultaneously. In SSchain, users are encouraged to use intra-shard transactions as much as possible to obtain faster confirmation and lower fees. If a transaction has multiple input addresses and does not need to maintain atomicity, the user's wallet automatically splits it into multiple intra-shard transactions. The remaining cross-shard transactions that cannot be split are forwarded to the root chain, which keeps the complete ledger and can directly validate cross-shard transactions.

Pyramid (Hong et al., 2022) is the first to propose the concept of layered sharding based on the existing two-layer structure. In Pyramid, nodes establish legitimate identities through POW and are divided into different types of shards based on these identities. One type is the i-shard, which is the same as the existing shards, and its nodes process and store intra-shard transactions using PBFT consensus. The other type is the bridge shard (b-shard), which connects multiple i-shards. Nodes in the b-shard can be considered joining multiple shards simultaneously and storing all the relevant states of the i-shards. Therefore, they can independently verify the validity of cross-shard transactions. For cross-shard transactions that no b-shards can process, the authors introduce the transaction split processing method of Omnileger. These transactions are first decomposed to meet the b-shard's connection range and then processed by the b-shard. Although overlapping shards can effectively improve the efficiency of cross-shard transaction processing, it also brings two main issues. Firstly, nodes in the b-shard must have more extensive computing, network, and storage resources to maintain better service quality. However, Pyramid divides nodes into shards based on unexpected identities, which cannot guarantee that nodes in the b-shard are resourceful enough. Secondly, the configurations of i-shards and b-shards must be written into the genesis block before launch. This static sharding configuration method significantly reduces system security. As shown in experiments, Pyramid can only tolerate 16% of malicious nodes, much lower than other schemes that use dynamic sharding and reconfiguration.

Benzene (Cai et al., 2022b) is the first double-chain sharding scheme, in which each shard maintains the proposer chain and vote chain simultaneously. The double-chain architecture of Benzene separates the transaction recording from the consensus execution, enabling cross-shard cooperation verification without affecting independent transaction recording processes in each shard. Every node will check every proposer block in each shard. To mitigate the cross-shard overhead, an SGX-enabled node placed in every shard will first prove the candidate blocks' validity and produce validity proofs. Only the validity proofs need to be propagated to all shards to request votes, and the candidate block with the most votes will be submitted. Compared with non-cooperative designs, Benzene realizes much higher fault tolerance resilience. Malicious nodes must control over half of the shards to change vote results. Benzene uses the traditional 2PL protocol to process cross-shard transactions, and every shard has to wait for at least k subsequent blocks to ensure that transactions are fully committed, resulting in a confirmation delay of more than 300 s for cross-shard transactions. Additionally, Benzene failed to explain how SGX ensures the validity of the loaded state when verifying candidate blocks. If the whole state is loaded into SGX, it may exceed the memory limit for large states. If Merkle proof for every accessed state is provided, it may affect transaction verification speed due to frequent context switching and encryption/decryption.

3.3.2. Full sharding

OmniLedger (Kokoris-Kogias et al., 2018) was proposed as an improvement to Elastico. Unlike Elastico, OmniLedger establishes a blockchain dedicated to managing identities. The sliding window mechanism of ByzCoin is used to determine the qualified nodes in the next epoch. The most significant innovation of OmniLedger is Atomix. This client-driven two-phase cross-shard transaction processing protocol can effectively guarantee the atomicity of the transaction, allowing cross-shard transactions to be executed correctly. Rapidchain (Zamani et al., 2018) is the first sharding protocol that does not require a trusted initial setup. In Rapidchain, a one-time root group is elected during the initialization phase. This group generates the first random number and establishes the reference committee to start the protocol correctly. To mitigate the extra burden caused by client-driven cross-shard transaction process protocol, Rapidchain enforces that the destination shard of cross-shard transaction breaks up the transaction and generates new relay transactions to turn cross-shard transactions into multiple intra-shard transactions.

The above scheme divides time into epochs of the same interval, and each shard runs synchronously according to the epoch. In contrast, in Monoxide (Wang and Wang, 2019), time is continuous, and each shard runs asynchronously. In Monoxide, the cross-shard transaction decomposition is performed by the source shard, and the Merkle inclusion proof is provided to guarantee the transaction's validity. Monoxide has two significant innovations, and the first is to use POW as the

intra-shard consensus mechanism to avoid BFT consensus high communication cost. However, this consensus mechanism can only provide probability finality (Zhang and Lee, 2020). Therefore, longer waiting times must be given to cross-shard transactions to reduce the likelihood of failure. The second one is that Monoxide does not periodically reconfigure shards but instead takes a fixed shard configuration according to node addresses. To protect the system from takeover attacks, Monoxide introduces the Chu-ko-nu mining algorithm, which allows nodes to mine new blocks in multiple shards simultaneously, spreading the computing power of malicious nodes in terms of economic incentives.

Maintaining a large shard size is crucial in preventing malicious nodes from controlling it and compromising its liveliness and security (David et al., 2022; Ozdayi et al., 2022). However, when the number of nodes in the network is relatively fixed, a larger shard size may result in fewer shards, ultimately impacting the system's throughput. Cochain (Li et al., 2023) has relaxed the requirement for shard liveliness and allows each shard to tolerate no more than 2/3 of malicious nodes. Using shard supervision and shard replacement, Cochain has achieved a six times smaller shard size than Monoxide. Specifically, the blocks within each shard produced through leaderless BFT consensus (Crain et al., 2021) are further verified by a CoC group composed of multiple shards. This cross-shard verification is called CoC consensus. Each shard in the system is continuously monitored by its corresponding CoC group. Suppose a shard loses liveliness or behaves maliciously. In that case, the CoC group will replace it and take over its existing state and subsequent consensus, thus restoring the system from an unsafe state. The supervised shard transmits the block's header to the CoC group during CoC consensus without transferring any related state information. This stateless verification assumes that when there are fewer than 2/3 malicious nodes in the shard, the consensus is valid if and only if it is reached. Otherwise, honest nodes will never be able to collect sufficient signatures to reach a consensus. Thus, Cochain has strictly calculated the size of each shard to ensure that it is highly likely to have fewer than 2/3 of malicious nodes. Ultimately, the experiment demonstrated that Cochain achieved a 35-fold improvement in throughput compared to Harmony (Harmony, 2023), with the same number of nodes.

The Brokerchain (Huang et al., 2022a) introduces the issue of "hot shards" in random state sharding for the first time. When an account stored in a shard initiates a large number of transactions, the workload of the shard will exceed its maximum processing capacity, resulting in temporary blockages of transactions within and across shards. Therefore, the Brokerchain mitigates this problem through load balancing of shard allocation and reducing cross-shard transactions. Brokerchain includes a p-shard responsible for transaction partitioning. It continuously collects transactions generated by each shard in the current epoch and uses Metis to partition them into different shards to ensure load balancing among shards. To reduce cross-shard transactions, Brokerchain allows a single account to exist in multiple shards, and the total sum of state fragments in each shard is the current state of the account. Brokerchain expands Ethereum's MPT tree to MSST tree. The leaf nodes still represent the state of each account, but a new storage map field is added to indicate which shards the account exists. Specifically, the storage map is represented by a vector, where each element indicates whether the account has a state fragment in the corresponding shard. Accounts in multiple shards can act as brokers to assist in processing cross-shard transactions. Cross-shard transaction processing in Brokerchain is a 2-phase locking (2PL) protocol. However, the brokers exist in both the source and destination shards, turning cross-shard transactions into two intra-shard transactions. Brokerchain is a fully sharded solution that only stores allocated transactions and related account states. However, to maintain consistency of account states across shards, each shard must store the storage map of all accounts, but balances, code, and other states do not need to be stored.

The above analysis shows that permissionless blockchains have a common design model. Most of them choose sharding based on the UTXO model, as it is much easier for cross-shard transaction processing. Every participant in permissionless blockchains must have an identity generated by the node selection mechanism, which combines the distributed random number generation algorithm and an anti-Sybil mechanism, such as the POW used in Bitcoin. In addition, a sharding reconfiguration is usually performed to ensure that adversaries do not take over individual shards. To ensure the network's decentralization, the above schemes are mostly full sharding, that is, sharding in the transaction, state, and network. The aim is to reduce the threshold for nodes to join the network and attract more nodes. The detailed performance comparison of permissionless sharding blockchain protocols can be found in Table 4.

4. Components

Although many sharding schemes have been proposed, the complexity of sharding blockchains makes it challenging to propose a standard architecture (Avarikioti et al., 2019; Garay et al., 2015). Therefore, we define the common building blocks in the above sharding schemes as key components (Wang et al., 2019; Liu et al., 2022a). Not all the Components are adopted simultaneously. Instead, they can be selectively studied according to different scenarios and goals. In the following, we classify these components into common components and permissionless blockchain exclusive components. All these components are discussed under Byzantine consumption. The relationship between the components in the full sharding scheme is shown in Fig. 3.

4.1. Common components

4.1.1. Node allocation

Node allocation is the key to reducing communication overhead or realizing parallel transaction processing. Node allocation should disallow nodes from being able to choose which shard to join freely to keep malicious nodes from gathering in one shard. In addition, the number of nodes in each shard should be balanced so that each shard can process transactions efficiently and have sufficient nodes to maintain consensus security. Node allocation should generally follow the following three properties (Monte et al., 2020): (1) liveness, whereby honest nodes must be assigned to the shard in time; (2) randomness, where nodes are randomly assigned to different shards; and (3) unbiasedness, where the node allocation process is immune to any external disturbance. In the permissioned blockchain, the CA owns the information of the whole network nodes and implements centralized membership management. Therefore, node allocation can be performed randomly by CAs according to a predefined protocol. It is also easy to adjust and control the number of nodes in each shard. In contrast, in permissionless blockchains, the most common method is random node allocation, such as using some bytes in the node identity as a random source for allocation.

4.1.2. Transaction partition

A transaction partition divides the main workloads in the blockchain and assigns them to different shards for parallel processing. Transaction partitioning follows two principles. The first is to minimize the generation of cross-shard transactions. The overhead of cross-shard transactions may eventually offset the benefits of partitioning (Avarikioti et al., 2019). Second, the transaction load of each shard should be as balanced as possible (Nguyen et al., 2019). Otherwise, the phenomenon known as hot-shard occurs, leading to a single-shard blockage and reducing the throughput of the whole system (Huang et al., 2022a). Most early research adopted the principle of random allocation according to transaction hash or participant address, which caused most of the transactions to become cross-shard transactions (Kokoris-Kogias et al., 2018). Recent research has focused on context-aware methods, that is, knowing the current state of shards before the transactions are partitioned, thus reducing cross-shard transactions at the source.

Table 4
Comparison of permissionless sharding blockchain protocols.

	Luu et al. (2016)	Kokoris-Kogias et al. (2018)	Zamani et al. (2018)	Wang and Wang (2019)	Huang et al. (2021)	Zhang et al. (2020)	Chen and Wang (2019)	Hong et al. (2022)	Cai et al. (2022b)	Li et al. (2023)	Huang et al. (2022a)
Account Model	UTXO	UTXO	UTXO	Account	UTXO	UTXO	UTXO	UTXO/Account	Account	Account	Account
Contract Enable	No	No	No	Yes	No	No	No	Yes	No	No	Yes
Network Model*	Sync	Part Sync	Async	Sync	Part Sync	Part Sync	Part Sync	Part Sync	Part Sync	Part Sync	Part Sync
Byzantine Tolerance**	N/4	N/4	N/3	N/2	N/2	N/3	N/2	N/3	N/3	N/4	N/3
Storage	x	x/K	x/k	x/k	x/K	x/K	x/K	x/K - x	x/K	x/K	x/K
Throughput	16 blocks[1]	12000 tx/s[2]	7300 tx/s[3]	11694 tx/s[4]	5682 tx/s[5]	!	6500 tx/s[6]	3821 tx/s[7]	12000 tx/s[8]	65066 tx/s[9]	352 tx/s[10]
Transaction Latency	110s	1.3 s	8.7 s	21 s	58 s	!	300 ms	3 s	30 s	40 s	275 s
Communication	O(n2)	O(n)	O(n)	O(n)	O(n2)	O(n)	O(n)	O(n2)	O(n2)	O(n2)	O(n2)
Adversary	Slowly adaptive	Slowly adaptive	!	Slowly adaptive	Static	Slowly adaptive	!	Slowly adaptive	!	Slowly adaptive	!

* The Network Model refers to the whole system;
** The Byzantine Tolerance and Communication refer to intra-shard consensus;
1 100 nodes/shard and 16 shards in total. The original article does not specify how many transactions a single block contains;
2 72 nodes/shard and 25 shards in total;
3 250 nodes/shard and 16 shards in total;
4 24 nodes/shard and 2048 shards in total;
5 225 nodes/shard and 8 shards in total;
6 30 nodes/shard and 60 shards in total;
7 100 nodes/shard and 20 shards in total;
8 5 nodes/shard and 20 shards in total;
9 100 nodes/shard and 61 shards in total;
10 7 nodes/shard and 16 shards in total;
! means that the value is missing in the literature;
- means that the property does not apply to the given category.

A common approach is to represent transactions as points and edges. Then, heuristic graph partitioning is used to partition high-correlation transactions into the same shard (Nguyen et al., 2019; Huang et al., 2022b).

4.1.3. Intra-shard consensus

The intra-shard consensus algorithm is responsible for agreeing on the state of the ledger within the shard. There are two common types of intra-shard consensus. One is a Satoshi-Nakamoto-type consensus that provides probabilistic finality, such as POW and GHOST. The other is the BFT-type consensus that provides ultimate finality, such as PBFT and Hotstuff. Compared with traditional blockchains, sharding blockchains have two characteristics, which result in most sharding blockchains choosing to use BFT-type consensus. First, the number of nodes in a single shard is small, and there is a dedicated P2P network connection within shards. Therefore, the BFT-type consensus can achieve better performance while maintaining a low communication overhead. In contrast, in traditional blockchains, the communication overhead of BFT-type consensus increases steeply when the number of nodes is large. Of course, some sharding blockchains choose the Satoshi-Nakamoto-type consensus, such as the blockchain used in one study (Wang and Wang, 2019). Second, cross-shard transactions involve consensus results from multiple shards. Already confirmed transactions in the Satoshi-Nakamoto-type consensus may become invalid owing to the blockchain fork, which may threaten the whole cross-shard transactions. Therefore, a BFT-type consensus that provides ultimate finality is a better choice. In permissioned blockchains, the network condition of the nodes is generally acceptable. Therefore, most shards can maintain a large node size by directly adopting the original PBFT protocol. However, in permissionless blockchains, the state of nodes varies. Therefore, the communication complexity of PBFT is optimized mainly by employing aggregated signatures.

4.1.4. Cross-shard process

Cross-shard transactions are unique transactions that occur only in state sharding blockchains. Every transaction at least contains one input–output pair or from and to addresses, whether in UXTO-based or account-based blockchain. The input and from addresses of a transaction may be stored in a different shard than the output and to addresses, so multiple shards need to cooperate to verify the validity of cross-shard transactions. As the number of shards increases, the performance of the sharding blockchain improves and the number of cross-shard transactions rises dramatically. For example, in Rapidchain, when the number of shards reaches 16, 99.98

4.1.5. Randomness generation

Randomness plays an essential role in three sharding components. In node selection, nodes are usually required to include random numbers in the identity acquisition process, thus preventing the adversary's selective pre-computing. Most permissionless blockchains (Luu et al., 2016; Kokoris-Kogias et al., 2018; Zamani et al., 2018) use POW as an identity generation mechanism. Random numbers are added to the hash computation process to prevent adversaries from performing Sybil attacks. In the node allocation component, the random identities obtained in the previous stage are the basis for node assignment. Therefore, the random number is an essential guarantee of achieving random node allocation. The same reason applies to shard reconfiguration. The random number needs to satisfy three properties: verifiability, unpredictability, and availability. Verifiability means that each node can verify the correctness of the random number (Liu et al., 2022a). Unpredictable means that no one can obtain the random number in advance. Availability means that the random number can be generated on time. In the permissioned blockchain, random numbers can be generated by central trusted sources, for example, directly by CA. In contrast, permissionless blockchains usually need to generate random numbers interactively by multiple participants. Therefore, it is also known as distributed random number generation protocol (DRG).

4.2. Exclusive components

4.2.1. Node selection

As nodes are divided into different shards, the computing resources and voting rights of the whole network are also divided. In addition, the intra-shard consensus is usually chosen as the BFT-type consensus, which can only guarantee consensus security when the number of malicious nodes is less than one-third of the total number of nodes. It is much easier to attack a single shard than the whole network. An attacker can increase the proportion of malicious nodes in shards by creating multiple Sybil identities (Bissias et al., 2014) and thus break the tolerance of intra-shard consensus. Therefore, each participating node in the sharding protocol must acquire a Sybil-resistance Identity. In a permissioned blockchain, a trusted third party usually manages the node identity, so nodes can request an endorsed identity from a CA. In contrast, in permissionless blockchains, Sybil attacks are usually prevented by increasing the cost of obtaining an identity for the node. Examples include the commonly used POW and POS, which can be combined with other mechanisms to select eligible nodes from all participating nodes.

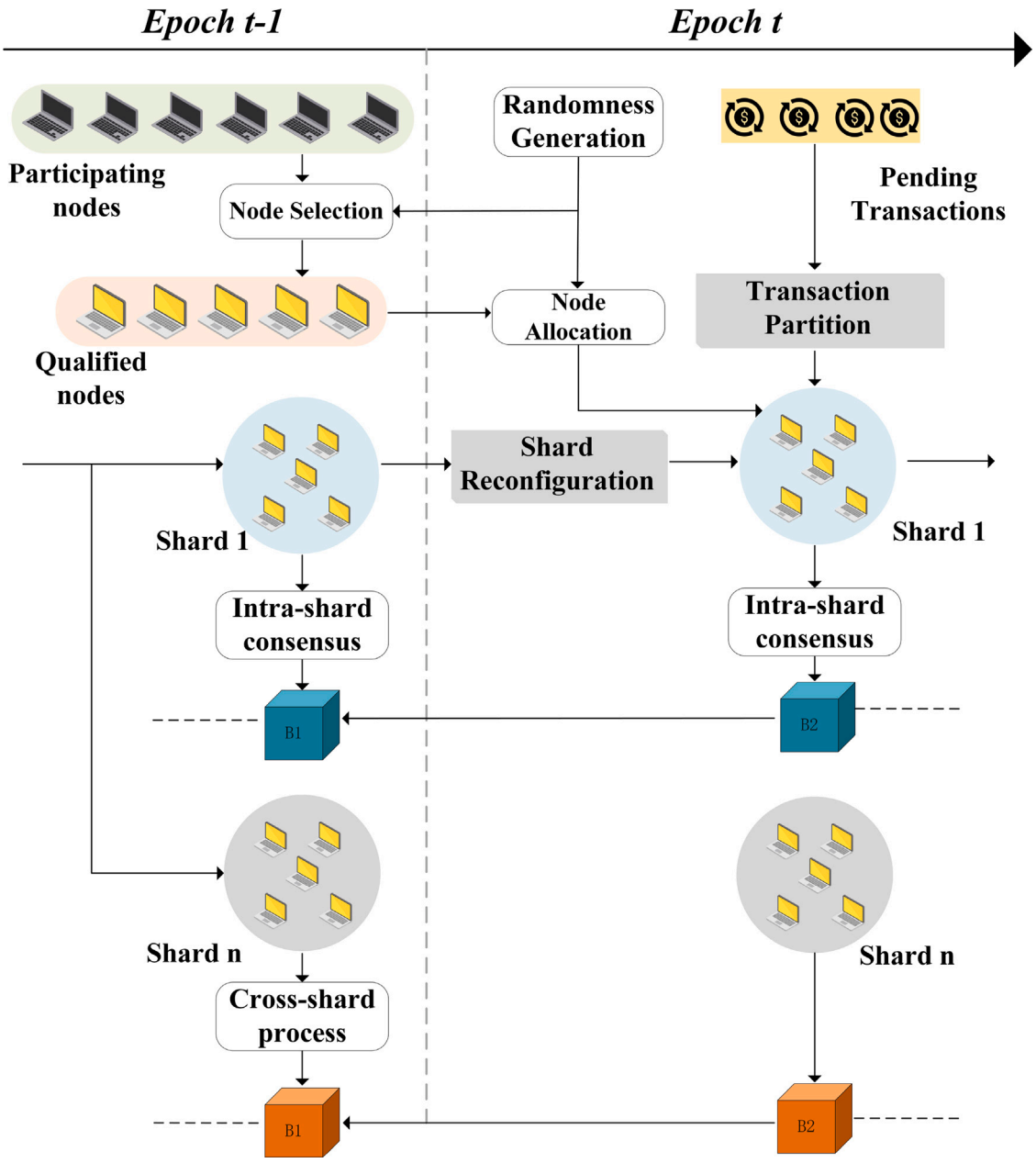

Fig. 3. Relationship between the components in the full sharding scheme.

4.2.2. Shard reconfiguration

Existing sharding schemes usually assume that the adversary is slowly adaptive. This is because, in the Byzantine environment, no sharding scheme can guarantee sharding security if the adversary is adaptive (Avarikioti et al., 2019). As we said above,Slowly adaptive refers to the fact that malicious nodes can corrupt nodes periodically, but the adversary cannot change the corruption target during the protocol's operation. Thus, to prevent the adversary from corrupting enough nodes and threatening the consensus security, nodes need to be periodically reallocated. Considering the downtime and communication overhead caused by node reconfiguration, only partial nodes are usually replaced. In addition, reconfiguration is the way to add new nodes to shards and maintain the node balance in shards. Sharding reconfiguration and node allocation typically use the same method but happen at different times. This component is rarely used in permissioned blockchains because the nodes are relatively fixed and stable online. However, in permissionless blockchains, it is a necessary component to maintain security. The details of the components used by the abovementioned sharding schemes in Section 3 can be found in Table 5.

5. Attack surfaces

Only a few studies have discussed the limited security risks of sharding without comprehensive analysis. In this paper, we review and analyze how sharding can be attacked and if some components are adopted. Note that sharding research is still in its early stages. The components not talked about below may have unexplored attack surfaces. The relationship between the attacks and sharding components can be found in Table 6.

5.1. Transaction injection

Attack surface Blockchain denial-of-service (DOS) attacks against transaction pools have been extensively studied (Chaganti et al., 2022; Raikwar and Gligoroski, 2022). Specifically, the adversary blocks a

Table 5
Comparison and analysis of components in the abovementioned sharding schemes.

Reference	Chain type	Sharding type	Node selection	Node allocation	Intra-shard consensus	Cross-shard transaction	Shard reconfiguration	Transaction partition	Epoch randomness
Luu et al. (2016)	Permissionless	Network and transaction	POW	Based on ID	PBFT	–	Based on ID	Based on transaction hash	Hash function
Kokoris-Kogias et al. (2018)	Permissionless	Full sharding	POW	Based on Randomness	ByzCoinX	Atomix	Based on Randomness	Based on transaction hash	RandHound and VRF
Zamani et al. (2018)	Permissionless	Full sharding	POW	Based on ID	BFT-like	Relay-tx	Based on ID	Based on transaction hash	VSS and Lagrange interpolation
Wang and Wang (2019)	Permissionless	Full sharding	–	Based on ID	GHOST	Relay-tx	–	transaction input address	–
Huang et al. (2021)	Permissionless	Transaction and state	POW	Based on reputation	Raft and CSBFT	Atomix	–	Based on transaction hash	RandHound and VRF
Zhang et al. (2020)	Permissionless	Network and transaction	POW	Based on Randomness	Raft	Semi-Commitment	!	!	SCRAPE
Chen and Wang (2019)	Permissionless	Transaction and state	–	Freely join	POW	Relay-Tx + rootchain	–	transaction input address	–
Dang et al. (2019)	Permissioned	Full sharding	SGX	Based on Randomness	AHL+	2PL and 2PC	Based on Randomness	!	SGX
Amiri et al. (2021)	Permissioned	Full sharding	CA	Geographical distribution	PBFT	PBFT	!	!	!
Zheng et al. (2021)	Permissioned	Transaction and state	CA	–	PBFT	Cross-epoch and Cross-call	!	!	!
Zheng et al. (2022)	Permissioned	Transaction and state	CA	–	PBFT	Distributed blockchain transaction	!	!	!
Hong et al. (2022)	Permissionless	Transaction and state	POW	Based on Randomness and ID	PBFT	Overlapping shards	!	!	VRF/VDF/SGX
Cai et al. (2022b)	Permissionless	Transaction and state	!	Based on ID	POW + shards cooperation	2PL	!	transaction input address	!
Li et al. (2023)	Permissionless	Full sharding	!	Based on Randomness and ID	BFT + shards cooperation	Relay-tx	Based on Randomness and ID	transaction input address	VRF+VDF
Huang et al. (2022a)	Permissionless	Full sharding	POW	Based on ID	PBFT	2PL+ broker	Based on ID	transaction input address	!

! means that this article uses the component but does not explicitly explain the related technology ;
- means that this article did not use the component.

Table 6
Relevance between attacks and sharding components.

Attack surface	Target component	Follow-up attacks	Countermeasures
Transaction injection	Transaction partition	DOS attack	Nguyen et al. (2019), Huang et al. (2022b), Nguyen and Thai (2022), Tao et al. (2022)
Sybil attack	Node selection	Double-spend attack Mixing protocol attack	Rajab et al. (2020), Eisenbarth et al. (2022), Otte et al. (2020), Lao et al. (2020), Biryukov and Feher (2020), Hafid et al. (2019, 2020b,a)
Shard takeover	Intra-shard consensus	BCP attack GFT attack	Dang et al. (2019), Chen and Wang (2019), Kokoris-Kogias et al. (2018), Han et al. (2020), Abraham et al. (2016), Chauhan et al. (2018)
Replay attack	Cross-shard consensus	Double-spend attack	Sonnino et al. (2019), Liu et al. (2022b, 2020b)
Cross-shard transaction censorship	Cross-shard process	Censorship attack	Liu et al. (2021), Kim et al. (2022)

blockchain node's transaction pool by generating many dusty transactions or low-value transactions (Saad et al., 2019; Baqer et al., 2016), resulting in normal transactions not being processed. Sharding blockchain can reduce the impact of such attacks by increasing the number of shards. However, individual nodes in the shard are still not scalable. Therefore, blocking a single shard is more straightforward than blocking the whole blockchain. The key to improving blockchain scalability by sharding is allocating transactions to different shards to be processed in parallel, also called transaction sharding. Most permissionless blockchains use hash-based transaction partitioning, meaning that the transaction hash value determines which shard is assigned to the process. Generally, the transaction hash value satisfies a uniform random distribution, allowing transactions to be randomly and evenly allocated to different shards. The transaction hash is usually obtained by doing a hash calculation on the transaction body, including the from and to addresses, transaction value, and other information. Therefore, an attacker can manipulate the transaction hash by selectively generating these two addresses and can achieve the purpose of injecting the transaction into a specific shard. After many maliciously constructed transactions are injected into the target shard, the transaction pool of nodes in this shard is blocked, achieving the effect of the DOS attack, and eventually affecting the whole system's performance. Such an attack is known as a transaction injection attack (Huang et al., 2022b) or a single-shard flooding attack (Nguyen and Thai, 2022). Recent studies have shown that this problem exists in permissioned blockchains (Huang et al., 2022b). When the transaction injection attack blocks the input shard, the output shard cannot confirm the validity of the input, and the validation of cross-shard transactions is stopped. The cost of this attack is not high, requiring only a consumer-grade laptop and a few hundred dollars to execute. Furthermore, the attacker can cause continuous blocking at a much lower cost by paying no transaction fee. The process of the attack is shown in Fig. 4.

Counter measure The main idea in dealing with transaction injection attacks is to divide transactions based on unpredictable randomness to prevent adversaries from disturbing the transaction division process. Therefore, many improved transaction partitioning methods have been proposed. Optchain (Nguyen et al., 2019) is a novel transaction partitioning method to improve the throughput of a sharding blockchain. It proposes a lightweight PageRank-based algorithm for calculating the relevance between a transaction and shards, called the T2S score. Transactions are partitioned to the shard with a high T2S score to reduce the number of cross-shard transactions. In addition, the client periodically obtains network congestion and uses it to calculate a feedback score called the L2S, which measures the real-time state of the shards. Optchain eventually considers both the T2S and L2S scores and places the transaction in the shard with the highest relevance and the lowest transaction latency. The adversary can still construct malicious transactions with a high correlation with the target shard. However, the L2S score is not known in advance because it changes in real time, so it is almost impossible to execute transaction injection attacks in Optchain. Despite being promising, Optchain still has some shortcomings. For example, one study (Tao et al., 2022) observed that Optchain could not maintain load balancing when there were many aggregated transactions. Therefore, this study (Tao et al., 2022) performs normalization operations before calculating the L2S score, which effectively enhances the load-balancing capability of Optchain. Besides that, it reduced the transaction partitioning problem to the colored packing problem (Liu and Layland, 2002) and proposed the ATCF algorithm to group transactions. Through this scheme, the transactions are assigned according to the shard congestion status. Another research (Nguyen and Thai, 2022) introduces nodes with the TEE into the network, which is responsible for transaction partitioning. The specific transaction partitioning method can be adapted from existing studies, such as Optchain, mentioned above. Furthermore, when executing the transaction partition process, the nodes must attach the execution correctness proof generated by the TEE, ensuring that all transactions are correctly and randomly partitioned. One study (Huang et al., 2022b) proposed a drift plus penalty (DPP)-based algorithm that can dynamically allocate limited resources in the permissioned blockchain, such as CPU and network bandwidth. It enables shards to maintain stable performance even with a transaction injection attack underway. Most of the above schemes were not designed to prevent injection attacks on purpose but to seek lower cross-shard transaction rates and better load-balancing performance. However, the transaction partitioning of these schemes carries unpredictable randomness, making it impossible for the transaction sender to determine in advance the shard to which the transaction will be assigned, thus avoiding the attacks.

5.2. Sybil attack

Attack surface The Sybil attack is typical in P2P systems (Mohaisen and Kim, 2013) and is especially serious in permissionless blockchains (Liu and Layland, 2002). The Sybil attack, which was first proposed by Douceur (2002), involves a scenario where the attacker deceives the system by generating or forging many legitimate identities. Thus, one entity can have multiple identities to gain influence in the network, which further disrupts the network redundancy mechanism. After executing the Sybil attack, the adversary can further execute a double-spend attack (Zhang and Lee, 2019), a majority attack, and a

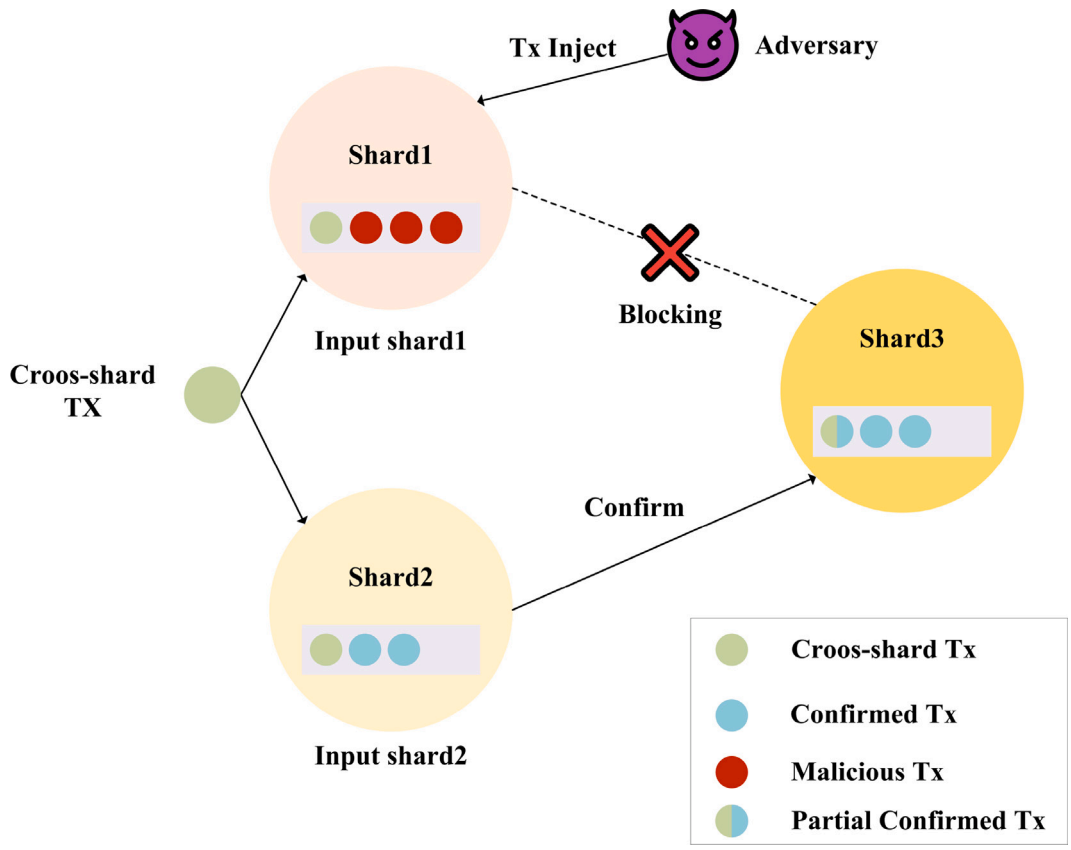

Fig. 4. Transaction injection attack process.

mixing protocol attack (Bissias et al., 2014). In a sharding blockchain, nodes are divided into smaller shards. The number of nodes in one single shard is much lower than that in the whole network, so a small number of Sybil nodes may produce considerable damage. Therefore, nodes can only participate in the sharding protocol after obtaining anti-Sybil identities first. There is a centralized identity management mechanism in the permissioned sharding blockchain, where the authority node is usually responsible for node auditing and registration to prevent Sybil attacks. In contrast, nodes are free to join in permissionless sharding blockchains. Mechanisms such as POW are usually used to prevent Sybil attacks. These schemes often require nodes to solve the POW puzzle, with periodically random numbers to obtain a legitimate identity, which limits the malicious pre-computing and the frequency of calculations. Existing sharding schemes such as Elastico, Rapidchain, and Monoxide use this kind of anti-Sybil mechanism, as shown in Fig. 5. However, these anti-Sybil methods are only valid when the attacker has limited computational power, which is often difficult to fulfill in practice (Douceur, 2002). Therefore, adversaries with sufficient computational power may generate many Sybil identities to increase their probability of being selected for sharding (Rajab et al., 2020). Other mechanisms, such as POS, may also be vulnerable to such attacks (Zhang and Lee, 2019). Sybil identities challenge the decentralization of the sharding. The anti-Sybil mechanism generates multiple legitimate identities. However, these identities are the same entity being selected multiple times.

Counter measure Existing research on anti-Sybil attacks mainly focuses on traditional P2P networks or non-sharding blockchains. One study (Eisenbarth et al., 2022) examined the state of the Ethernet P2P network through web crawlers and calculated thresholds to detect suspicious Sybil nodes. When some nodes found the Sybil node, they would record it in the smart contract via transaction. After other peer nodes had verified it to be true, they refused to connect with the Sybil node and created a blacklist to inform the whole network. Trustchain (Otte et al., 2020) proposes a contribution-based anti-Sybil algorithm called NetFlow, which calculates the reputation value of each node according to its contribution to the consensus. Sybil nodes cannot handle transactions and therefore cannot contribute to the consensus. Therefore, the reputation value of the Sybil node gradually decreases until it cannot receive any transaction in the network. Another study (Lao et al., 2020) established a mapping between nodes and identities by using the geographic location information of IoT devices. Different IoT nodes in the network cannot report the exact geographic location at the same time, thus limiting the number of Sybil nodes in the IoT-blockchain system. Biryukov and Feher (2020) proposed a credit-based consensus protocol, which ranks nodes according to their reputation values. Only the top nodes of the list can participate in the consensus. Although the studies mentioned above were not explicitly focused on sharding blockchains, their anti-Sybil approach can be adopted by the node selection component of sharding blockchains. Many studies have quantified the probability of Sybil attacks in sharding blockchain and have given better network parameters that can reduce the probability of Sybil attacks. For example, Rajab et al. (2020) modeled the sharding blockchain, taking Elastico as the study object, and studied the probability that the attacker can generate multiple valid IDs within an Epoch. Hafid modeled a sharding blockchain based on various generating functions such as PGFA and JHDA. Then, he computed the probability of Sybil attacks occurring in two different cases, the first shard and at least one shard (Hafid et al., 2019, 2020b,a). However, research on anti-Sybil mechanisms in sharding blockchains is still in its infancy.

5.3. Shard takeover

Attack surface In the traditional blockchain, the 51% attack is one of the most famous attack methods (Karame et al., 2012; Aponte-Novoa et al., 2021). The 51% attack is also called a majority attack (Nakamoto, 2008), which means that when an adversary has more than 50% of

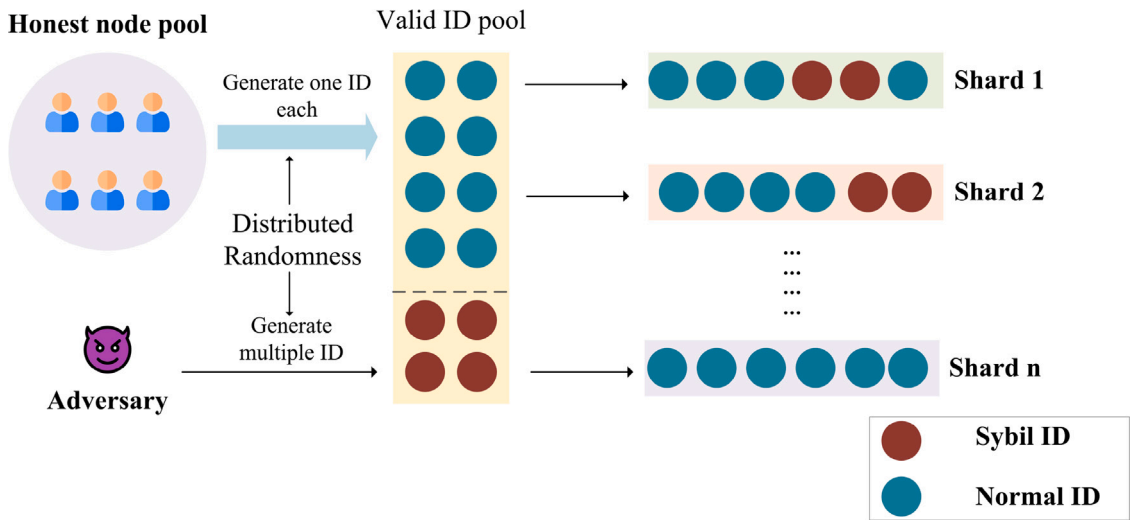

Fig. 5. Process of node selection and node allocation.

all resources, including computing power and stakes, it can control the consensus protocol. As a result, other attacks can be implemented, such as the double-spend attack (Karame et al., 2012) and the short-selling attack (Lee and Kim, 2020). Practically, it is difficult to achieve the above attacking requirements. However, the number of nodes in each shard is much less than that in the whole blockchain. Therefore, it is possible to control most nodes in one shard. Even though the adversary owns only a few resources of the entire network, it can launch a 51% attack in one shard by aggregating all the resources in that shard. Such an attack is called the shard takeover attack (Tennakoon and Gramoli, 2022). Owing to cross-shard transactions, the whole network is cascaded as soon as one shard is compromised (Hafid et al., 2023). More specifically, when the corrupted nodes in one shard exceed the Byzantine tolerance of the consensus, the adversary can further raise the break consensus protocol (BCP) attack or generate fake transaction (GFT) attack (Rajab et al., 2020). The purpose of the BCP attack is to prevent the shard from reaching a consensus, which affects the liveness of the whole blockchain. In comparison, the GFT attack aims to add invalid or false transactions in the block, thus affecting the integrity of the blockchain. The shard takeover attack may occur at any time during every epoch. Before the shard formation, in the initial stage of an epoch, attackers can first generate multiple IDs through the Sybil attack to increase their probability of being selected into committees. One study (Rajab et al., 2020) demonstrated that the Sybil attack would increase the probability of a takeover attack. The adversary can then influence the node allocation through the Node injection attack. Similarly, to the above hash-based transaction partitioning, Elastico, Rapidchain, and Monoxide use ID-based node partitioning protocols. An attacker can generate multiple desired IDs and inject them into the target shard. There has been no research yet to prove the viability of this attack. However, as described above, this is still a potential attack surface when the adversary has sufficient computing power. In addition, the adversary can launch a shard takeover attack by corrupting enough nodes in the same shard.

Counter measure The key idea to coping with the shard takeover attack is to randomly allocate nodes to different shards. For example, OmniLedger (Kokoris-Kogias et al., 2018) uses random numbers generated by RandHound as the basis for node allocation instead of node IDs. Another study (Dang et al., 2019) used trusted and unbiased random numbers generated by SGX, a TEE, as the basis for node allocation. WormHole (Han et al., 2020) proposes the concept of memory dependence, using the world state of the previous epoch as a seed to generate random numbers. TBSD (Abraham et al., 2016) uses the genetic algorithm to minimize the probability of malicious nodes allocated in the same shard. The genetic algorithm (GA) generates a set of chromosomes for nodes, and each chromosome value represents the node's relevance to shards. The consensus result of every epoch filters the node's chromosomes and ultimately determines the node allocation. In addition, most schemes assume that the adversary can corrupt all nodes in shards. However, this process is gradual, and it cannot corrupt all nodes instantaneously. Therefore, periodically reconfiguring shards can prevent the attacker from taking over shards. One author proposed a method (Abraham et al., 2016) to select the nodes that are replaced based on the sliding time window rule. The nodes are sorted according to the join time, and the newly joined nodes replace the old nodes in the window. Rapidchain uses the bounded-Cuckoo rule to replace the nodes in shards. The node activity is calculated according to the total verification transaction numbers in the previous epoch. Newly joined nodes replace those inactive nodes. Another research group (Chauhan et al., 2018) proposed the concept of the inspector who acts as a supervisor. Inspector nodes are responsible for monitoring and eliminating the potential risks in shards. When there are risks, such as one node holding too much computing power, the inspector immediately reorganizes this shard. The more risks the inspector finds, the more revenue they gain. In this way, the inspector is motivated to secure the shard proactively. In the method proposed in another study (Chen and Wang, 2019), the nodes in SSchain can freely choose to join any shards, which leads to a high probability of a takeover attack. Therefore, the method introduces a double-chain structure, where any block committed by shards is verified by the root chain again. Even if the shard has been taken over, the attacker cannot add any invalid transactions or blocks to the main chain.

5.4. Replay attack

Attack surface In the replay attack, the adversary maliciously repeats or delays the propagation of messages in different contexts to achieve the attack purpose (El Abbadi and Jamouli, 2021). In sharding blockchain, the replay attack mainly affects cross-shard transaction processing. Most cross-shard transaction schemes are designed based on the two-phase commit (2pc) protocol. This protocol consists of two phases, prepare and commit, with a third party coordinating the two phases. In the prepare phase, the coordinator collects the availability certificate of inputs from input shards. Meanwhile, the input shard locks relevant assets to prevent them from being used by other transactions. In the commit phase, the coordinator submits all certificates to the output shard for validation. There are two types of 2pc-based protocols. One is the client-driven type, such as the Atomix protocol

used by OmniLedger. The other is the shard-driven type, such as the S-BAC protocol used by Chainspace (Sonnino, 2018). 2pc-based protocols are vulnerable to replay attacks, which can compromise the security and liveness of the system. Further, an attacker can implement a double-spend attack and create fake coins (Sonnino et al., 2019). Specifically, the attack exploits the vulnerability of the insufficient binding of shards to transaction instances. The attacker induces the response from target shards by sending them maliciously fake transactions. Then, the attacker replays these responses during other transaction instances. Finally, the attacker tricks the coordinator into achieving the attacking goal. To better understand the attack process, we use Chainspace as an example. As shown in Fig. 6, the attacker initiates a cross-shard transaction T1 (x1, x2) -> (y3) at time t1, where the subscript indicates the shard in which it is stored. Shard 1 receives T1 and verifies the validity of x1 by the intra-shard consensus. If x1 is spendable, it broadcasts a pre-accept message, and if x1 is not spendable, a pre-abort is issued. Shard 2 also verifies the validity of x2. Suppose that x1, located in shard 1, has already been spent. The attacker broadcasts the pre-recorded pre-accepted message to trick shard 3 into viewing x1 as still valid. When the actual shard 1 response arrives at shard 3 in time t3, shard 3 rejects the pre-abort message, as it has already received the pre-accept from the attacker at time t2. Finally, shard 3 accepts the transaction T1, and the attacker completes the replay attack. **Counter measure** The core idea of countering replay attacks is adding links between shards and transactions. With limited computing power, an attacker cannot forge transaction signatures. Therefore, it is only necessary to add identifiers in transactions to prevent replay attacks. For example, the method proposed by Liu et al. (2022b) adds the ID of each transaction and the input address of the transaction in the availability certificate so that the shards quickly identify any attempt to replay the transaction multiple times. Zhang and Lee (2019) proposed the concept of state commitments to coordinate cross-shard transactions by sorting all transactions in the system to form a global transaction view of all shards, thus preventing the replay attack. Fleetchain Liu et al. (2020b) proposes the FBFT consensus protocol, which concatenates the transaction initiator address with the transaction hash as the identifier so that every transaction can be unique. Sonnino et al. (2019) proposed the Byzcuit cross-shard protocol, which uses the address of the transaction initiator and the transaction hash as the transaction's identifier, thus avoiding the replay of transactions by attackers with different addresses. The Byzcuit cross-shard consensus combines the advantages of Atomix and S-BAC. It introduces the Tx manager to manage the two-phase commit protocol. After the preparation phase, the dummy transactions with sequence numbers are generated by the output shard to prevent replay attacks.

5.5. Cross-shard transaction censorship

Attack surface Most sharding schemes that utilize BFT-like consensus as intra-shard consensus has adopted 2pl protocols for handling cross-shard transactions. The advantage of this approach is that existing BFT leaders can directly coordinate the intermediate process of two-phase commitment and transmit critical information with no extra election process. In addition, the view change protocol of BFT-like consensus enables nodes can replace malicious leaders and maintain the liveness of shards. However, when the leader acts honestly within the shard but maliciously across shards, shard nodes will know nothing because ordinary intra-shard nodes cannot observe their leader's behavior. For instance, when receiving cross-shard transactions, the leader executes the two-phase protocol typically and generates valid proof of the transaction but does not send it to the relevant shard. This attack surface is called cross-shard transaction censorship. All sharding schemes employing two-phase commit and BFT-like intra-shard consensus may be affected by this attack surface, which may result in blocking cross-shard transactions, permanently locking related assets, and ultimately the destruction of the system's liveness

Counter measure The most natural way to resist censorship is to adopt a multi-coordinator approach. Multiple nodes are randomly selected from different shards to participate in cross-shard transaction coordination instead of relying solely on the current leader. By selecting enough nodes, at least one honest node will eventually propagate the validity proof to the target shard. However, this method may increase network overhead, so the number of nodes selected must be carefully considered. Liu et al. (2021) proposed the Secure Cross-Shard View-Change Protocol based on external observation. Its core idea is to allow the relevant shards involved in cross-shard transactions to supervise each other. When a node detects malicious behavior by other leaders, co-signed evidence will be sent to the malicious leader's shard. After verifying the validity of the evidence, shard members will initiate a view-change to replace the malicious leader. In addition to the above active defense methods, passive measures such as introducing behavior-based credit evaluation mechanisms can also be helpful, such as assigning a reputation score to each leader based on transaction processing speed and participation in cross-shard transactions. Leaders with low reputation scores will no longer be eligible for the next election. Staking or periodically reorganizing can also mitigate the impact of censorship (Kim et al., 2022).

6. Discussion and future direction

As mentioned above, sharding is one of the most promising technologies for solving the blockchain scalability issue. Compared to existing blockchain solutions that are mature but lack scalability, sharding technology is still in its infancy. Based on the comprehensive review above, we will discuss several unresolved issues and provide a list of potential future directions to advance research efforts in this area further.

6.1. Performance improvement

While sharding has the potential to achieve near-linear scalability improvements in theory, its complexity has posed significant obstacles to implementation, specifically regarding performance loss caused by cross-shard transaction processing and shard reorganization.

6.1.1. Reducing the number of cross-shard transactions

A natural approach to reducing cross-shard transactions is carefully designing the transaction partitioning method. Huang et al. (2022a) proposed a transaction partitioning proxy shard, where all client transactions are analyzed for their correlation in real-time. Transactions with high correlation are assigned to the same shard to reduce the occurrence of cross-shard transactions. However, such a centralized architecture would make the proxy shard a scalability bottleneck for the entire system. Mizrahi and Rottenstreich (2020) proposed to dynamically adjust the transaction partitioning rules at the end of each epoch, mapping closely related addresses to the same shard in the next epoch and achieving consensus among all nodes. Although it maintains decentralization, this method is significantly less effective than real-time partitioning. Therefore, there is currently no transaction partitioning method that can achieve both decentralization and effectiveness. One possible research direction is to increase the adjustment frequency and perform multiple transaction partitioning rule adjustments within an epoch based on the count of cross-shard transactions. This method can improve the effectiveness of partitioning while retaining decentralization. Another possible direction is to introduce deep learning to explore more features related to transaction correlation, thereby improving the effectiveness of a single adjustment.

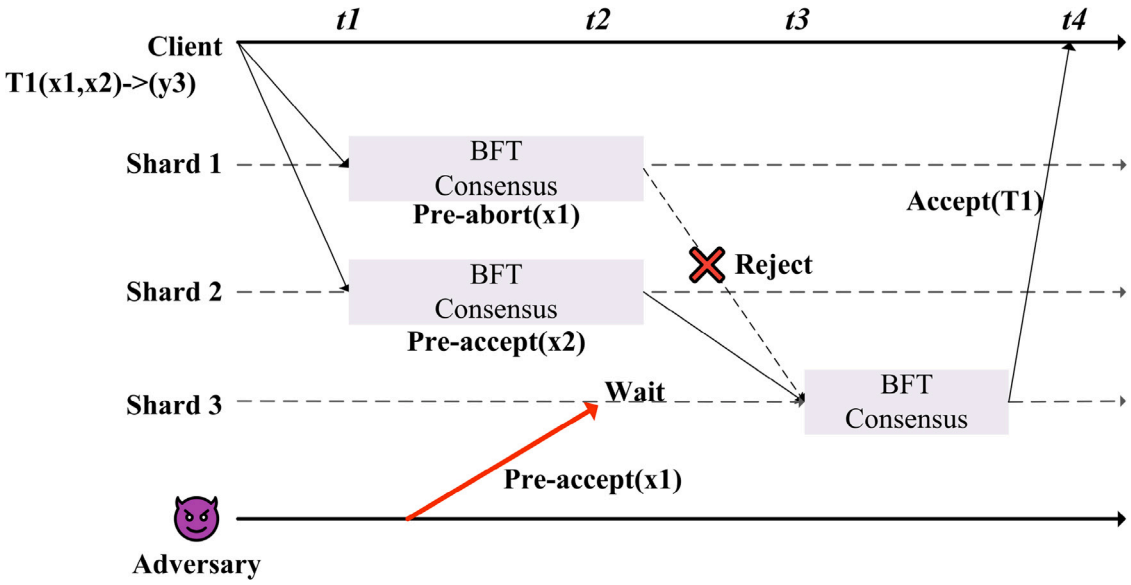

Fig. 6. Process of replay attack in Chainspace.

6.1.2. Reducing the processing overhead of cross-shard transactions

In addition, to designing practical transaction partition algorithms, another orthogonal approach to reducing the performance overhead of cross-shard transactions is to minimize the validation, consensus, and network costs associated with processing cross-shard transactions. Current cross-shard transaction processing methods, such as the 2PL protocol, transaction splitting, and relay transactions, propagate the transaction across all involved shards and make consensus within each shard multiple times. A transaction involving m shards will cause at least O(mk) network overhead, where k is the size of each shard and at least m rounds of consensus. The network and consensus delays reduce the number of confirmed transactions per second, lowering the system's throughput. Therefore, reducing the processing costs of cross-shard transactions and achieving single consensus round processing is one of the future research directions.

6.1.3. Reducing shard reconfiguration delay

The delay caused by reconfiguring sharding includes the node selection, node allocation, and new node bootstrapping time. The system cannot process transactions during reconfiguration, which leads to performance degradation. Some studies, such as sschain (Chen and Wang, 2019) and monoxide (Wang and Wang, 2019), allow nodes to freely choose shards and protect the system's security through incentive mechanisms. However, the security of incentive mechanisms is based on the assumption of rational economic actors and needs more systematic security proof. Other studies (Zamani et al., 2018) avoid system downtime by replacing only some old nodes in the shard rather than a complete reassembly, but they also lack strict security proof. Therefore, how to shorten the delay caused by sharding reconfiguring is still an open question. One possible solution to reduce node selection time is to utilize trusted execution hardware to lower the time cost of obtaining node identities. Adopting a stateless client design pattern is an effective way to shorten bootstrapping time. However, the current stateless sharding scheme based on cryptographic accumulators or error codes only supports the UTXO account model. Hence, smart contract-enabled stateless sharding is also one of the future research directions.

6.2. Security issues

6.2.1. Epoch duration and corruption ability

Many current sharding solutions adopt the slowly-adaptive security model. A too-long epoch will give malicious nodes enough time to launch a sharding takeover attack, which can undermine the system's security. Assuming the epoch lasts for T, the shard size is k, the intra-shard consensus Byzantine tolerance is f, the proportion of malicious nodes when the epoch start is d, and t represents the time it takes to corrupt a target node. When the malicious nodes can only learn about their attack targets after the current epoch ends, the shard is safe when $T \leq tk(f\text{-}d)$. However, T must be shorter if the malicious nodes can launch their attacks before the epoch ends. Most existing solutions set the epoch length without explaining the reasoning, as seen in Jon (2022), Zamani et al. (2018) and Hong et al. (2021), where the epoch length is set to 1 day. There is currently no quantitative analysis of the relationship between epoch duration and corruption ability, which remains an open question.

6.2.2. Unbiased and liveness randomness generation

Multiple sharding components, such as node selection, node allocation, transaction partition, and shard reconfiguration, rely on random numbers. Some of the existing schemes cannot guarantee the unbiasedness and liveness of randomness generation, and some of them rely on trusted initialization to bootstrap, which is challenging to implement in practice. Therefore, random number generation methods still need to be studied.

6.2.3. Components alignment

There still needs to be more research on the security of sharding. The existing research has only focused on the shortcomings of specific mechanisms. However, sharding blockchains are complex systems with multiple components, which may rely on different system assumptions. For example, while intra-shard consensus may depend on the synchronous network, cross-shard transaction processing may only work under partial synchronous network assumptions. Such differences may cause serious security issues. There is still a lack of a comprehensive security evaluation framework to analyze whether the components align.

6.3. Incentive machine and applications

6.3.1. Transaction fee distribution

Transaction fees are usually used to incentivize miners and prevent DoS attacks. However, cross-shard transactions may render the functions above ineffective in sharding blockchains. Cross-shard transactions typically need to be included in all involved shards. For example,

a 2PL protocol requires the transaction packed in the source shards and consensus in destination shards. Just like in the current non-sharding blockchains, the miner in the source shard collects the transaction fees by including the transaction in a block and updating the balance state of the transaction initiator. However, the miner in the destination shard cannot directly access or modify this state and hence cannot receive any benefit. Thus miners will be more willing to include intra-shard transactions to obtain more transaction fees. If the target shard processes all cross-shard transactions for free, it will be vulnerable to DoS attacks. Therefore, cross-shard transaction fee payments are one of the future research directions.

6.3.2. Complex applications-enabled sharding

Most current research on permissionless sharding blockchain is based on the UTXO model, which does not support smart contracts. Smart contracts are essential for blockchain to be widely used in areas other than finance. Even the research on account-based permissioned sharding blockchain that supports smart contracts does not show other application scenarios other than asset transfer. This is far from the original purpose of the proposed sharding technology, which is to improve blockchain scalability for broader applications. Therefore, researching sharding blockchain that supports complex applications such as NFTs and contract calls is one of the future research directions.

6.4. Research tools

6.4.1. Experimental platforms for sharding

For most researchers in sharding, there is a challenge: the lack of a powerful simulation/experimental platform to validate their new ideas or protocols quickly. Current research mainly relies on modifications of non-sharding blockchain software, such as Bitcoin or Ethereum, where the components are often highly coupled. Therefore, validating each new idea or protocol requires significant effort. Therefore, a modular, easy-to-deploy, sharding blockchain simulation/experimental platform would benefit the research community.

6.4.2. Powerful benchmark tools

Most sharding protocols do not have open-source code, so researchers can only reproduce based on their understanding of existing research and then compare it with their new proposals. However, the reproduced solutions may deviate from the original ones. Therefore, a unified, general benchmarking framework would benefit the development of sharding research and is one of the future research directions.

7. Conclusion

This survey first illustrates the scalability issue, which is the main reason preventing blockchain from being used in some areas. We define scalability issues based on previous research in terms of throughput, storage, and network. Then, we study the sharding blockchain, the most promising technology to solve the scalability issue. We systematically analyze currently well-known permissioned and permissionless sharding blockchain protocols and propose the key components of sharding protocols. Furthermore, for the first time, we explore how some components may be attacked and how their defense strategies work. Finally, by outlining the state-of-the-art sharding and corresponding attacks, we highlight future research directions to develop more efficient and secure sharding blockchain protocols.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

The work was supported by the National Natural Science Foundation of China [No. 62072336]; the New Generation Artificial Intelligence Major Project of Tianjin, China [No. 19ZXZNGX00080]; the Natural Science Foundation Youth Project of Tianjin [No. 22JCQNJC00310]; the Tianjin Research Innovation Project for Postgraduate Students, China [No. 2022BKYZ043].

References

Abraham, I., Malkhi, D., Nayak, K., Ren, L., Spiegelman, A., 2016. Solida: A blockchain protocol based on reconfigurable byzantine consensus. arXiv preprint arXiv:1612.02916.

Altarawneh, A., Herschberg, T., Medury, S., Kandah, F., Skjellum, A., 2020. Buterin's scalability trilemma viewed through a state-change-based classification for common consensus algorithms. In: 2020 10th Annual Computing and Communication Workshop and Conference. CCWC, IEEE, pp. 0727–0736.

Amiri, M.J., Agrawal, D., El Abbadi, A., 2021. Sharper: Sharding permissioned blockchains over network clusters. In: Proceedings of the 2021 International Conference on Management of Data. pp. 76–88.

Aponte-Novoa, F.A., Orozco, A.L.S., Villanueva-Polanco, R., Wightman, P., 2021. The 51% attack on blockchains: A mining behavior study. IEEE Access 9, 140549–140564, (Online; Accessed 30 April 2023).

Avarikioti, G., Kokoris-Kogias, E., Wattenhofer, R., 2019. Divide and scale: Formalization of distributed ledger sharding protocols. arXiv preprint arXiv:1910.10434.

Bano, S., Al-Bassam, M., Danezis, G., 2017. The road to scalable blockchain designs. USENIX; Login: Mag. 42 (4), 31–36.

Baqer, K., Huang, D.Y., McCoy, D., Weaver, N.C., 2016. Stressing out: Bitcoin "stress testing". In: Financial Cryptography Workshops.

Biryukov, A., Feher, D., 2020. ReCon: Sybil-resistant consensus from reputation. Pervasive Mob. Comput. 61, 101109, (Online; Accessed 30 April 2023).

Bissias, G., Ozisik, A.P., Levine, B.N., Liberatore, M., 2014. Sybil-resistant mixing for bitcoin. In: Proceedings of the 13th Workshop on Privacy in the Electronic Society. pp. 149–158.

Buterin, V., 2021. Why sharding is great: Demystifying the technical properties. https://vitalik.ca/general/2021/04/07/sharding.html. (Online; Accessed 30 April 2023).

Cai, T., Chen, W., Psannis, K.E., Goudos, S.K., Yu, Y., Zheng, Z., Wan, S., 2022a. Scalable on-chain and off-chain blockchain for sharing economy in large-scale wireless networks. IEEE Wirel. Commun. 29 (3), 32–38, (Online; Accessed 30 April 2023).

Cai, Z., Liang, J., Chen, W., Hong, Z., Dai, H.-N., Zhang, J., Zheng, Z., 2022b. Benzene: Scaling blockchain with cooperation-based sharding. IEEE Trans. Parallel Distrib. Syst. 34 (2), 639–654.

Cao, B., Zhang, Z., Feng, D., Zhang, S., Zhang, L., Peng, M., Li, Y., 2020. Performance analysis and comparison of PoW, PoS and DAG based blockchains. Digit. Commun. Netw. 6 (4), 480–485, (Online; Accessed 30 April 2023).

Chaganti, R., Boppana, R.V., Ravi, V., Munir, K., Almutairi, M., Rustam, F., Lee, E., Ashraf, I., 2022. A comprehensive review of denial of service attacks in blockchain ecosystem and open challenges. IEEE Access 10, 96538–96555, (Online; Accessed 30 April 2023).

Chauhan, A., Malviya, O.P., Verma, M., Mor, T.S., 2018. Blockchain and scalability. In: 2018 IEEE International Conference on Software Quality, Reliability and Security Companion. QRS-C, IEEE, pp. 122–128.

Chen, H., Wang, Y., 2019. SSChain: A full sharding protocol for public blockchain without data migration overhead. Pervasive Mob. Comput. 59, 101055, (Online; Accessed 30 April 2023).

Crain, T., Natoli, C., Gramoli, V., 2021. Red belly: A secure, fair and scalable open blockchain. In: 2021 IEEE Symposium on Security and Privacy. SP, pp. 466–483.

Croman, K., Decker, C., Eyal, I., Gencer, A.E., Juels, A., Kosba, A., Miller, A., Saxena, P., Shi, E., Gün Sirer, E., et al., 2016. On scaling decentralized blockchains: (a position paper). In: Financial Cryptography and Data Security: FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers 20. Springer, pp. 106–125.

Dabbagh, M., Choo, K.-K.R., Beheshti, A., Tahir, M., Safa, N.S., 2021. A survey of empirical performance evaluation of permissioned blockchain platforms: Challenges and opportunities. Comput. &Amp; Secur. 100, 102078, (Online; Accessed 30 April 2023).

Dang, H., Dinh, T.T.A., Loghin, D., Chang, E.-C., Lin, Q., Ooi, B.C., 2019. Towards scaling blockchain systems via sharding. In: Proceedings of the 2019 International Conference on Management of Data. pp. 123–140.

David, B., Magri, B., Matt, C., Nielsen, J.B., Tschudi, D., 2022. GearBox: Optimal-size shard committees by leveraging the safety-liveness dichotomy. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. pp. 683–696.

Douceur, J.R., 2002. The sybil attack. In: Peer-to-Peer Systems: First International-Workshop, IPTPS 2002 Cambridge, MA, USA, March 7–8, 2002 Revised Papers 1. Springer, pp. 251–260.

Dwork, C., Lynch, N., Stockmeyer, L., 1988. Consensus in the presence of partial synchrony. J. ACM 35 (2), 288–323, (Online; Accessed 30 April 2023).

Eisenbarth, J.-P., Cholez, T., Perrin, O., 2022. Ethereum's peer-to-peer network monitoring and sybil attack prevention. J. Netw. Syst. Manage. 30 (4), (Online; Accessed 30 April 2023).

El Abbadi, R., Jamouli, H., 2021. Takagi–Sugeno fuzzy control for a nonlinear networked system exposed to a replay attack. Math. Probl. Eng. 2021, 1–13, (Online; Accessed 30 April 2023).

Eyal, I., Gencer, A.E., Sirer, E.G., Van Renesse, R., 2016. Bitcoin-ng: A scalable blockchain protocol. In: 13th {USENIX} Symposium on Networked Systems Design and Implementation. {NSDI} 16, pp. 45–59.

Gao, J., Adjei-Arthur, B., Sifah, E.B., Xia, H., Xia, Q., 2022. Supply chain equilibrium on a game theory-incentivized blockchain network. J. Ind. Inf. Integr. 26, 100288, (Online; Accessed 30 April 2023).

Garay, J.A., Kiayias, A., Leonardos, N., 2015. The bitcoin backbone protocol: Analysis and applications. In: International Conference on the Theory and Application of Cryptographic Techniques.

Gaži, P., Kiayias, A., Zindros, D., 2019. Proof-of-stake sidechains. In: 2019 IEEE Symposium on Security and Privacy. SP, IEEE, pp. 139–156.

Ge, Z.-L., Zhang, Y., Long, Y., Gu, D., 2022. Shaduf++: Non-cycle and privacy-preserving payment channel rebalancing. IACR Cryptol. EPrint Arch. 2022, 388.

Gilbert, S., Lynch, N., 2002. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. Acm Sigact News 33 (2), 51–59.

Hafid, A., Hafid, A.S., Samih, M., 2019. New mathematical model to analyze security of sharding-based blockchain protocols. IEEE Access 7, 185447–185457, (Online; Accessed 30 April 2023).

Hafid, A., Hafid, A.S., Samih, M., 2020a. A methodology for a probabilistic security analysis of sharding-based blockchain protocols. In: Blockchain and Applications: International Congress. Springer, pp. 101–109.

Hafid, A., Hafid, A.S., Samih, M., 2020b. A novel methodology-based joint hypergeometric distribution to analyze the security of sharded blockchains. IEEE Access 8, 179389–179399, (Online; Accessed 30 April 2023).

Hafid, A., Hafid, A.S., Samih, M., 2023. A tractable probabilistic approach to analyze sybil attacks in sharding-based blockchain protocols. IEEE Trans. Emerg. Top. Comput. 11 (1), 126–136, (Online; Accessed 30 April 2023).

Han, R., Yu, J., Lin, H., Chen, S., Paulo Esteves-Veríssimo, P., 2021. On the security and performance of blockchain sharding. Cryptology EPrint Archive.

Han, R., Yu, J., Zhang, R., 2020. Analysing and improving shard allocation protocols for sharded blockchains. IACR Cryptol. EPrint Arch. 2020, 943.

Harmony, T., 2023. Harmony ONE whitepapers. https://whitepaper.io/document/512/harmony-whitepaper. (Online; Accessed 30 April 2023).

Henning, S., Hasselbring, W., 2022. A configurable method for benchmarking scalability of cloud-native applications. Empir. Softw. Eng. 27 (6), (Online; Accessed 30 April 2023).

Hong, Z., Guo, S., Li, P., 2022. Scaling blockchain via layered sharding. IEEE J. Sel. Areas Commun. 40, 3575–3588.

Hong, Z., Guo, S., Li, P., Chen, W., 2021. Pyramid: A layered sharding blockchain system. In: IEEE INFOCOM 2021-IEEE Conference on Computer Communications. IEEE, pp. 1–10.

Huang, H., Peng, X., Zhan, J., Zhang, S., Lin, Y., Zheng, Z., Guo, S., 2022a. BrokerChain: A cross-shard blockchain protocol for account/balance-based state sharding. In: IEEE INFOCOM 2022 - IEEE Conference on Computer Communications. pp. 1968–1977.

Huang, C., Wang, Z., Chen, H., Hu, Q., Zhang, Q., Wang, W., Guan, X., 2021. RepChain: A reputation-based secure, fast, and high incentive blockchain system via sharding. IEEE Internet Things J. 8 (6), 4291–4304, (Online; Accessed 30 April 2023).

Huang, H., Yue, Z., Peng, X., He, L., Chen, W., Dai, H.-N., Zheng, Z., Guo, S., 2022b. Elastic resource allocation against imbalanced transaction assignments in sharding-based permissioned blockchains. IEEE Trans. Parallel Distrib. Syst. 33 (10), 2372–2385, (Online; Accessed 30 April 2023).

Jon, C., 2022. The Hitchhiker's guide to ethereum. (Online; Accessed 30 April 2023). https://members.delphidigital.io/reports/the-hitchhikers-guide-to-ethereum/.

Karame, G.O., Androulaki, E., Capkun, S., 2012. Double-spending fast payments in bitcoin. In: Proceedings of the 2012 ACM Conference on Computer and Communications Security. pp. 906–917.

Kim, J.-S., Shin, J.-M., Choi, S.-H., Choi, Y.-H., 2022. A study on prevention and automatic recovery of blockchain networks against persistent censorship attacks. IEEE Access 10, 110770–110784.

Kogias, E.K., Jovanovic, P., Gailly, N., Khoffi, I., Gasser, L., Ford, B., 2016. Enhancing bitcoin security and performance with strong consistency via collective signing. In: 25th USENIX Security Symposium. USENIX Security 16, pp. 279–296.

Kokoris-Kogias, E., Jovanovic, P., Gasser, L., Gailly, N., Syta, E., Ford, B., 2018. OmniLedger: A secure, scale-out, decentralized ledger via sharding. In: 2018 IEEE Symposium on Security and Privacy. SP, pp. 583–598.

Kshetri, N., Voas, J., 2018. Blockchain-Enabled e-voting. IEEE Softw. 35 (4), 95–99, (Online; Accessed 30 April 2023).

Kuzlu, M., Pipattanasomporn, M., Gurses, L., Rahman, S., 2019. Performance analysis of a hyperledger fabric blockchain framework: throughput, latency and scalability. In: 2019 IEEE International Conference on Blockchain. Blockchain, IEEE, pp. 536–540.

Lao, L., Dai, X., Xiao, B., Guo, S., 2020. G-PBFT: A location-based and scalable consensus protocol for IOT-blockchain applications. In: 2020 IEEE International Parallel and Distributed Processing Symposium. IPDPS, IEEE, pp. 664–673.

Lee, K., James, J., Ejeta, T., Kim, H., 2016. Electronic voting service using block-chain. J. Digit. Forens. Secur. Law (Online; Accessed 30 April 2023).

Lee, S., Kim, S., 2020. Short selling attack: A self-destructive but profitable 51% attack on pos blockchains. Cryptology EPrint Archive.

Li, M., Lin, Y., Zhang, J., Wang, W., 2023. Cochain: High concurrency blockchain sharding via consensus on consensus. In: IEEE INFOCOM 2023.

Liao, C.-H., Guan, X.-Q., Cheng, J.-H., Yuan, S.-M., 2022. Blockchain-based identity management and access control framework for open banking ecosystem. Future Gener. Comput. Syst. 135, 450–466, (Online; Accessed 30 April 2023).

Liu, Y., He, D., Obaidat, M.S., Kumar, N., Khan, M.K., Raymond Choo, K.-K., 2020a. Blockchain-based identity management systems: A review. J. Netw. Comput. Appl. 166, 102731, (Online; Accessed 30 April 2023).

Liu, C., Layland, J.W., 2002. Scheduling algorithms for multiprogramming in a hard-real-time environment. In: Readings in Hardware/Software Co-Design. Elsevier, pp. 179–194, (Online; Accessed 30 April 2023).

Liu, Y., Liu, J., Hei, Y., Xia, Y., Wu, Q., 2021. A secure cross-shard view-change protocol for sharding blockchains. In: Information Security and Privacy: 26th Australasian Conference, ACISP 2021, Virtual Event, December 1–3, 2021, Proceedings 26. Springer, pp. 372–390.

Liu, Y., Liu, J., Li, D., Yu, H., Wu, Q., 2020b. Fleetchain: A secure scalable and responsive blockchain achieving optimal sharding. In: Algorithms and Architectures for Parallel Processing: 20th International Conference, ICA3PP 2020, New York City, NY, USA, October 2–4, 2020, Proceedings, Part III. Springer, pp. 409–425.

Liu, Y., Liu, J., Vaz Salles, M.A., Zhang, Z., Li, T., Hu, B., Henglein, F., Lu, R., 2022a. Building blocks of sharding blockchain systems: Concepts, approaches, and open problems. Comp. Sci. Rev. 46, 100513, (Online; Accessed 30 April 2023).

Liu, Y., Liu, J., Wu, Q., Yu, H., Hei, Y., Zhou, Z., 2022b. SSHC: A secure and scalable hybrid consensus protocol for sharding blockchains with a formal security framework. IEEE Trans. Dependable Secure Comput. 19 (3), 2070–2088, (Online; Accessed 30 April 2023).

Liu, Y., Xiong, Z., Hu, Q., Niyato, D., Zhang, J., Miao, C., Leung, C., Tian, Z., 2022c. VRepChain: A decentralized and privacy-preserving reputation system for social Internet of Vehicles based on blockchain. IEEE Trans. Veh. Technol. 71 (12), 13242–13253, (Online; Accessed 30 April 2023).

Liu, Y., Yu, W., Ai, Z., Xu, G., Zhao, L., Tian, Z., 2022. A blockchain-empowered federated learning in healthcare-based cyber physical systems. IEEE Trans. Netw. Sci. Eng. 1, (Online; Accessed 30 April 2023).

Liu, Y., Zhang, C., Yan, Y., Zhou, X., Tian, Z., Zhang, J., 2023. A semi-centralized trust management model based on blockchain for data exchange in IoT system. IEEE Trans. Serv. Comput. 16 (2), 858–871, (Online; Accessed 30 April 2023).

Lombrozo, E., Lau, J., Wuille, P., 2015. Segregated Witness (Consensus Layer). Tech. Rep. BIP 141, Bitcoin Core Develop. Team.

Luu, L., Narayanan, V., Zheng, C., Baweja, K., Gilbert, S., Saxena, P., 2016. A secure sharding protocol for open blockchains. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. pp. 17–30.

Mizrahi, A., Rottenstreich, O., 2020. State sharding with space-aware representations. In: 2020 IEEE International Conference on Blockchain and Cryptocurrency. ICBC, IEEE, pp. 1–9.

Mohaisen, A., Kim, J., 2013. The sybil attacks and defenses: a survey. arXiv preprint arXiv:1312.6349.

Monte, G.D., Pennino, D., Pizzonia, M., 2020. Scaling blockchains without giving up decentralization and security. In: Proceedings of the 3rd Workshop on Cryptocurrencies and Blockchains for Distributed Systems. ACM, New York, NY, USA, (Online; Accessed 30 April 2023).

Nakamoto, S., 2008. Bitcoin: A peer-to-peer electronic cash system. Decentralized Business Rev. 21260.

Nasir, M.H., Arshad, J., Khan, M.M., Fatima, M., Salah, K., Jayaraman, R., 2022. Scalable blockchains — A systematic review. Future Gener. Comput. Syst. 126, 136–162, (Online; Accessed 30 April 2023).

Nguyen, L.N., Nguyen, T.D.T., Dinh, T.N., Thai, M.T., 2019. OptChain: Optimal transactions placement for scalable blockchain sharding. In: 2019 IEEE 39th International Conference on Distributed Computing Systems. ICDCS, pp. 525–535.

Nguyen, T., Thai, M.T., 2022. Denial-of-Service vulnerability of hash-based transaction sharding: Attack and countermeasure. IEEE Trans. Comput. 1, (Online; Accessed 30 April 2023).

Otte, P., de Vos, M., Pouwelse, J., 2020. TrustChain: A Sybil-resistant scalable blockchain. Future Gener. Comput. Syst. 107, 770–780, (Online; Accessed 30 April 2023).

Ozdayi, M.S., Guo, Y., Zamani, M., 2022. Instachain: Breaking the sharding limits via adjustable quorums. Cryptology EPrint Archive.
Pandey, A.A., Fernandez, T.F., Bansal, R., Tyagi, A.K., 2022. Maintaining scalability in blockchain. In: Intelligent Systems Design and Applications: 21st International Conference on Intelligent Systems Design and Applications. ISDA 2021 Held During December 13–15, 2021, Springer, pp. 34–45.
Perboli, G., Musso, S., Rosano, M., 2018. Blockchain in logistics and supply chain: A lean approach for designing real-world use cases. IEEE Access 6, 62018–62028, (Online; Accessed 30 April 2023).
Qiu, H., Ji, T., Zhao, S., Chen, X., Qi, J., Cui, H., Wang, S., 2022. A geography-based P2P overlay network for fast and robust blockchain systems. IEEE Trans. Serv. Comput. 1–14, (Online; Accessed 30 April 2023).
Qu, J., 2022. Blockchain in medical informatics. J. Ind. Inf. Integr. 25, 100258, (Online; Accessed 30 April 2023).
Raikwar, M., Gligoroski, D., 2022. Dos attacks on blockchain ecosystem. In: Euro-Par 2021: Parallel Processing Workshops: Euro-Par 2021 International Workshops, Lisbon, Portugal, August 30-31, 2021, Revised Selected Papers. Springer, pp. 230–242.
Rajab, T., Manshaei, M.H., Dakhilalian, M., Jadliwala, M., Rahman, M.A., 2020. On the feasibility of sybil attacks in shard-based permissionless blockchains. arXiv, arXiv:2002.06531.
Ramanan, M., Singh, L., Kumar, A.S., Suresh, A., Sampathkumar, A., Jain, V., Bacanin, N., 2022. Secure blockchain enabled Cyber- Physical health systems using ensemble convolution neural network classification. Comput. Electr. Eng. 101, 108058, (Online; Accessed 30 April 2023).
Saad, M., Njilla, L.L., Kamhoua, C.A., Kim, J., Nyang, D., Mohaisen, A., 2019. Mempool optimization for defending against ddos attacks in pow-based blockchain systems. In: 2019 IEEE International Conference on Blockchain and Cryptocurrency. ICBC, pp. 285–292.
Sanka, A.I., Cheung, R.C., 2021. A systematic review of blockchain scalability: Issues, solutions, analysis and future research. J. Netw. Comput. Appl. 195, 103232, (Online; Accessed 30 April 2023).
Singh, A., Click, K., Parizi, R.M., Zhang, Q., Dehghantanha, A., Choo, K.-K.R., 2020. Sidechain technologies in blockchain networks: An examination and state-of-the-art review. J. Netw. Comput. Appl. 149, 102471, (Online; Accessed 30 April 2023).
Sivaraman, V., Venkatakrishnan, S.B., Ruan, K., Negi, P., Yang, L., Mittal, R., Fanti, G.C., Alizadeh, M., 2018. High throughput cryptocurrency routing in payment channel networks. In: Symposium on Networked Systems Design and Implementation.
Sohrabi, N., Tari, Z., 2020. On the scalability of blockchain systems. In: 2020 IEEE International Conference on Cloud Engineering. IC2E, pp. 124–133.
Sonnino, A., 2018. Chainspace: A sharded smart contract platform. In: Network and Distributed System Security Symposium 2018. NDSS 2018.
Sonnino, A., Bano, S., Al-Bassam, M., Danezis, G., 2019. Replay attacks and defenses against cross-shard consensus in sharded distributed ledgers. In: 2020 IEEE European Symposium on Security and Privacy. EuroS&P, pp. 294–308.
Tao, L., Lu, Y., Ding, X., Fan, Y., Kim, J.Y., 2022. Throughput-oriented associated transaction assignment in sharding blockchains for IoT social data storage. Digit. Commun. Netw. 8 (6), 885–899, (Online; Accessed 30 April 2023).
Tennakoon, D., Gramoli, V., 2022. Dynamic blockchain sharding. In: 5th International Symposium on Foundations and Applications of Blockchain 2022. FAB 2022, Schloss Dagstuhl-Leibniz-Zentrum für Informatik.
Thibault, L.T., Sarry, T., Hafid, A.S., 2022. Blockchain scaling using rollups: A comprehensive survey. IEEE Access 10, 93039–93054, (Online; Accessed 30 April 2023).
Tian, H., Xue, K., Luo, X., Li, S., Xu, J., Liu, J., Zhao, J., Wei, D.S.L., 2021. Enabling cross-chain transactions: A decentralized cryptocurrency exchange protocol. IEEE Trans. Inf. Forensics Secur. 16, 3928–3941, (Online; Accessed 30 April 2023).
Tschorsch, F., Scheuermann, B., 2016. Bitcoin and Beyond: A technical survey on decentralized digital currencies. IEEE Commun. Surv. &Amp; Tutor. 18 (3), 2084–2123, (Online; Accessed 30 April 2023).
Wang, G., Shi, Z.J., Nixon, M., Han, S., 2019. Sok: Sharding on blockchain. In: Proceedings of the 1st ACM Conference on Advances in Financial Technologies. pp. 41–61.
Wang, J., Wang, H., 2019. Monoxide: Scale out blockchains with asynchronous consensus zones. In: Symposium on Networked Systems Design and Implementation.
Xie, J., Yu, F.R., Huang, T., Xie, R., Liu, J., Liu, Y., 2019. A survey on the scalability of blockchain systems. IEEE Netw. 33 (5), 166–173, (Online; Accessed 30 April 2023).
Xu, Z., Chen, L., 2022. L2chain. Proc. VLDB Endow. 16 (4), 986–999, (Online; Accessed 30 April 2023).
Yang, Z., Yang, R., Yu, F.R., Li, M., Zhang, Y., Teng, Y., 2022. Sharded blockchain for collaborative computing in the Internet of Things: Combined of dynamic clustering and deep reinforcement learning approach. IEEE Internet Things J. 9 (17), 16494–16509, (Online; Accessed 30 April 2023).
Yao, Q., 2018. A systematic framework to understand central bank digital currency. Sci. China Inf. Sci. 61 (3), (Online; Accessed 30 April 2023).
Yu, G., Wang, X., Yu, K., Ni, W., Zhang, J.A., Liu, R.P., 2020. Survey: Sharding in blockchains. IEEE Access 8, 14155–14181, (Online; Accessed 30 April 2023).
Zamani, M., Movahedi, M., Raykova, M., 2018. Rapidchain: Scaling blockchain via full sharding. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 931–948.
Zhang, S., Lee, J.-H., 2019. Double-Spending with a sybil attack in the bitcoin decentralized network. IEEE Trans. Ind. Inform. 15 (10), 5715–5722, (Online; Accessed 30 April 2023).
Zhang, S., Lee, J.-H., 2020. Analysis of the main consensus protocols of blockchain. ICT Express 6, 93–97.
Zhang, M., Li, J., Chen, Z., Chen, H., Deng, X., 2020. Cycledger: A scalable and secure parallel protocol for distributed ledger via sharding. In: 2020 IEEE International Parallel and Distributed Processing Symposium. IPDPS, IEEE, pp. 358–367.
Zheng, Z., Xie, S., Dai, H.-N., Chen, X., Wang, H., 2018. Blockchain challenges and opportunities: A survey. Int. J. Web Grid Serv. 14 (4), 352–375.
Zheng, P., Xu, Q., Luo, X., Zheng, Z., Zheng, W., Chen, X., Zhou, Z., Yan, Y., Zhang, H., 2022. Aeolus: Distributed execution of permissioned blockchain transactions via state sharding. IEEE Trans. Ind. Inform. 18 (12), 9227–9238.
Zheng, P., Xu, Q., Zheng, Z., Zhou, Z., Yan, Y., Zhang, H., 2021. Meepo: Sharded consortium blockchain. In: 2021 IEEE 37th International Conference on Data Engineering. ICDE, pp. 1847–1852.
Zhou, Q., Huang, H., Zheng, Z., Bian, J., 2020. Solutions to scalability of blockchain: A survey. IEEE Access 8, 16440–16455, (Online; Accessed 30 April 2023).
Zhou, M., Zeng, L., Han, Y., Li, P., Long, F., Zhou, D., Beschastnikh, I., Wu, M., 2023. Mercury: Fast Transaction Broadcast in High Performance Blockchain System. IEEE.

Yi Li received his B.E. degree in computer science and technology from the Tianjin University of Technology, Tianjin, China, in 2019. Now he is a Ph.D. candidate in Tianjin University of Technology. His main research interests include blockchain, computer networks, and deep learning.

Jinsong Wang received the B.Sc. degree from the Department of Computer Science and Engineering, Tianjin University of Technology, Tianjin, China, and the M.Sc. and Ph.D. degrees from Nankai University, Tianjin. He is currently a Professor with the School of Computer Science and Engineering, Tianjin University of Technology. He has published several papers on journals and international conferences, such as IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS (TITS), and IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT. He has invented 10 patents. His main research interests include computer networks, blockchain, and data intelligence.

Hongwei Zhang received the Ph.D. degree s from the Department of Computer Science and Engineering, Tianjin University of Technology, Tianjin, China. He is currently a lecturer with the School of Computer Science and Engineering, Tianjin University of Technology. His main research interests include blockchain, privacy protection and data security sharing.