Sidechains With Optimally Succinct Proof

Yin, Lingyuan ; Xu, Jing; Liang, Kaitai ; Zhang, Zhenfeng

DOI
10.1109/TDSC.2023.3328430

Publication date
2023

Document Version
Final published version

Published in
IEEE Transactions on Dependable and Secure Computing

Sidechains With Optimally Succinct Proof

Lingyuan Yin, Jing Xu, Kaitai Liang, *Member, IEEE*, and Zhenfeng Zhang

***Abstract*—Sidechains have been widely used to improve the interoperability and scalability of blockchain systems. Despite several interesting sidechain constructions have been proposed in the literature, they suffer from the following downsides: 1) their designs do not easily support pluggable consensus mechanisms, and 2) their communication and storage costs for cross-chain operations are not yet optimized. In this work, we first propose Ge-Co, a generic sidechain construction to realize secure asset transfers between blockchains, supporting different consensus algorithms, such as Proof-of-Stake (PoS) and Proof-of-Work (PoW). Our design is built on top of the proposed voting committee selection approach and threshold signature schemes (TSS) and meanwhile, it achieves optimally succinct and constant proof size, only yielding lightweight communication and storage costs. Ge-Co works in the semi-adaptive corruption model. To provide stronger security, we further propose PoS-Co, a PoS-based sidechain construction in the fully-adaptive corruption model. PoS-Co is based on the proposed anonymous committee selection approach, and preserves optimally succinct proof. We also formally prove that Ge-Co can achieve the security properties of atomicity and timeliness. Finally, we develop a proof-of-concept (PoC) implementation for Ge-Co, and the results demonstrate that the design is efficient and practical.**

***Index Terms*—Blockchains, distributed systems, efficiency, interoperability, security, sidechains.**

I. Introduction

After the Blockchain 1.0 and 2.0 eras pushed by Bitcoin [1] and Ethereum [2], blockchain technologies have been fast developed and widely deployed in real-world applications, such as decentralized finance (DeFi). To provide massive data processing and collaborations, blockchain systems are moving forward to an interactive and cross-chain design. This type of design heavily depends on the interoperability of blockchains. Sidechains [3] are introduced as a key mechanism to improve the interoperability in blockchain systems.

Sidechains enable communication and interaction for different blockchains, and allow data to flow between blockchain systems, such that assets can be transferred and synchronized. A data flow is called one-way peg, if it may move from a local chain to a remote one; and further if the data can be transferred back to the local chain, that is called two-way peg. Sidechains can be divided into two forms: the parent-child relationship and the equal relationship. In the former, a sidechain can be seen as a "child" of the mainchain, and the child must rely on the parent, at least during the initialization period. As for the latter, any two blockchains can be seen as a sidechain for each other, and they are treated equally. In this works, we mainly deal with the design for two-way peg and parent-child relationship. We note that our design can also be transformed to support equal relationship with minor changes.

Sidechains also provide a solution to increase the scalability of blockchain systems. Rather than a single blockchain, sidechains use multiple blockchains to handle all the transactions, which can increase transaction throughput. More practically, they provide a new method to upgrade a blockchain system without hard forks or soft forks [4][5]. Specifically, one may generate a new sidechain to execute the upgraded blockchain protocol. If the operation of sidechain meets the desired requirements, then all assets in the mainchain could be transferred to the sidechain. After that, the previous mainchain could be abandoned while the sidechain becomes dominant, i.e. the new mainchain.

Despite of the advantages listed above, there still exist two long-lasting open problems in the context of sidechains: proof size and compatibility. As all the chains created in a sidechain construction work independently, to support cross-chain asset transfers, e.g., the asset transfers from a chain X to another chain Y, X needs to submit a proof (also called certificate) to Y, such that Y can validate and react to the state of X. After validating the proof successfully, Y will store it locally and further transfer assets to the receivers. The proof size here directly determines the communication and storage costs. Nowadays many financial service providers deliver mobile digital payments on low-capacity mobile clients, such as mobile phones and smartwatches, and enable consumers to conveniently and efficiently pay for goods and services. Unfortunately most current sidechain solutions do not suit for such clients due to their limited access to storage and bandwidth. For instance, for current PoW-based sidechain constructions, such as [6] [7], the proof size depends on the chain length logarithmically or even linearly. When implementing Drivechains [7] in Ethereum of Sep 2022, the proof size

Manuscript received 28 May 2023; revised 14 October 2023; accepted 24 October 2023. Date of publication 30 October 2023; date of current version 11 July 2024. This work was supported in part by "A New Technology Architecture for Web 3.0 Based on Blockchain" under Grant 2023YFB2704200, in part by the Industrial Internet Innovation and Development Project-Industrial Internet Identification Resolution System: National Top-Level Node Construction Project (Phase I), and in part by the National Natural Science Foundation of China under Grant 62172396. *(Corresponding authors: Jing Xu.)*

Lingyuan Yin is with the Institute of Industrial Internet and Internet of Things, China Academy of Information and Communications Technology, Beijing 100191, China (e-mail: lingyuan2018@iscas.ac.cn).

Jing Xu is with TCA Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China, also with Zhongguancun Laboratory, Beijing 102629, China, and also with the School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: xujing@iscas.ac.cn).

Kaitai Liang is with Cybersecurity Group, Delft University of Technology, 2628 Delft, CD, The Netherlands (e-mail: kaitai.liang@tudelft.nl).

Zhenfeng Zhang is with TCA Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: zhenfeng@iscas.ac.cn).

Digital Object Identifier 10.1109/TDSC.2023.3328430

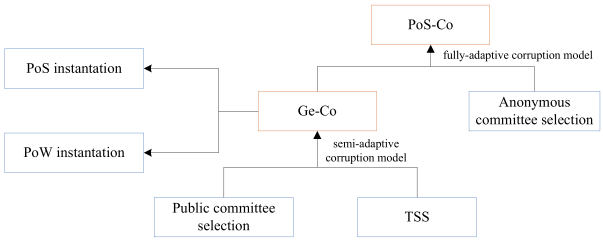


Fig. 1. Technical roadmap of our design.

of a cross-chain asset transfer is about 7.1 GB, resulting in expensive communication and storage costs on mobile clients, and the costs will become large with the chain grows. To reduce the costs of cross-chain asset transfers, a PoS-based sidechain construction [8] has been proposed. But the public keys of voters and their Merkle proofs included in the proof makes the proof size still linear in the size of the committee. If the sidechain construction [8] is deployed to the PoS-based blockchains in the fully-adaptive corruption model (e.g., Ouroboros Praos [9]), the proof size will be even larger due to the fact that the VRF-proofs [10] of voters are also included in the proof.

Beyond the proof size, existing sidechain designs are quite limited in terms of the compatibility. Except for some centralized sidehchain constructions that can suit for various consensus algorithms, almost all existing decentralized sidechain constructions only support cross-chain asset transfers between blockchains with specifical consensus algorithms. For example, PoW-based sidechain constructions [6] [7] cannot be deployed in PoS blockchains.

Following the above discussion, to support a wider range of applications, there is a pressing need to design a sidechain construction that supports various consensus mechanisms, and meanwhile achieves more succinct proof size.

A. Our Contributions

We propose two sidechain constructions to achieve asset transfers between blockchains, where the technical roadmap of our design is shown in Fig. 1. Both of the two constructions can work in the permissionless setting without requiring any trusted third party and furthermore, the proof size for cross-chain transfer is optimally succinct which avoids heavy communication and storage costs. More specifically, our technical contributions are summarized as follows.

• *Generic sidechain construction with optimally succinct proof.* We first present Ge-Co, a generic sidechain construction with optimally succinct proof. In our design, a small subset of sidechain users are randomly selected as the public voting committee per epoch. To resist the Sybil attack [11], the committee members are selected based on user's computational power or stake. The committee members vote for cross-chain transactions of sidechain by generating a threshold signature [12]. Specifically, the committee members generate their votes by using their key shares, which are obtained by jointly invoking a DKG protocol [13][14], and sufficient signature votes from the committee members are aggregated into a single threshold signature, then a valid certificate including the single threshold signature can be generated. The committee selection method of Ge-Co ensures that a sufficient fraction of committee members are honest, thus the adversary cannot generate a valid certificate on incorrect messages. By this method, we bring down the proof (or certificate) size to be optimally succinct. As shown in Table I, compared with the previous constructions, our proof size remains constant and takes only 0.1 KB, which is extremely succinct. In addition, Ge-Co uses the voting committee rather than all users to run DKG protocols and TSS, significantly reducing the running overheads.

Due to the fact that the processes of committee selection and certificate generation do not rely on a specific consensus mechanism, Ge-Co is naturally generic and it can be applied to different blockchain systems, such as PoS-based and PoW-based blockchains. Based on Ge-Co, we thus propose two concrete instantiations for PoW-based and PoS blockchains. Especially for the PoW instantiation, we develop a solution to resist the possible selfish mining attack [15], ensuring a sufficient fraction of committee members are honest. And our PoW instantiation is able to provide various block difficulties, which matches the realistic assumption that the block difficulty is changed periodically.

• *PoS-based construction in fully-adaptive corruption model.* In Ge-Co, we use a public committee to invoke DKG protocols and TSS. This is because in the existing DKG protocols, in order to secretly share its own secret key among participants, each participant needs to know the identities (or the public keys) of others. But this makes an indication that Ge-Co is only applicable to the semi-adaptive corruption model. To provide stronger security, we further propose PoS-Co, a PoS-based sidechain construction in the fully-adaptive corruption model, where the adversary can corrupt nodes without any delay. Specifically, we select an additional anonymous committee to invoke a DKG protocol and generate votes, such that the identities or the long term keys of committee members are hidden. The adversary is not able to adaptively corrupt the committee members, and the security of PoS-Co can be guaranteed. Different from the anonymous committee method introduced in [16], PoS-Co does not require any trusted party to secretly share the secret key, and uses a new public committee selection method, thus it is more secure and efficient. Also unlike other PoS-based sidechains, such as [8], PoS-Co maintains constant proof size as the VRF-proofs of voters are not included in the certificate.

• *Formal security analysis and performance evaluation.* We formally prove that Ge-Co satisfies the atomicity and timeliness properties. We also develop a PoC implementation to evaluate the costs of Ge-Co. The results demonstrate that Ge-Co not only has small time cost but also achieves much smaller proof size than existing sidechains [6][8].

B. Related Work

Inspired by the concept of sidechains [3], the first sidechain protocol called Drivechains was proposed in [7]. But the proof size for cross-chain transfers is linear in the chain length. For example, when implementing Drivechains [7] in Ethereum of Sep 2022, the proof size is about 7.1 GB. Moreover the computational cost for generating proofs also depends on the chain

TABLE I
COMPARISON OF OUR SIDECHAINS WITH EXISTING WORKS

Schemes	Generic[1]	Security analysis[2]	Proof size[3]	Proof size/KB[4]	Computational cost[5]	Variable difficulties[6]
PoS-based sidechains for Ouroboros [8]	×	√	$O(k)$	565	$O(k)$	/
PoS-based sidechains for Ouroboros Praos [8]	×	×	$O(k)$	1927	$O(k)$	/
PoW-based sidechains [6]	×	×	$O(\log n)$	1439	$O(\log n)$	×
Drivechains [7]	×	×	$O(n)$	7475482	$O(n)$	×
BTCRelay [19]	×	×	$O(n)$	7475482	$O(n)$	×
Ge-Co for Ethereum	√	√	$O(1)$	0.1	$O(L)$	√
Ge-Co for Ouroboros	√	√	$O(1)$	0.1	$O(L)$	/
PoS-Co for Ouroboros Praos	×	×	$O(1)$	0.1	$O(L)$	/

[1] Whether the sidechain constrcution is generic without relying on any specific blockchain platforms.
[2] Whether the construction provides a formal security analysis.
[3] The proof (certificate) size for cross-chain asset transfers, where k is the common prefix parameter, and n is the chain length.
[4] As for PoS-based sidechains [8] and Ge-Co for Ouroboros [25]: we calculate proof sizes in the implementations of Cardano [26], where 30% of participants fail to vote and $k = 2160$; for PoS-based sidechains [8] and PoS-Co for Ouroboros Praos [9], we calculate proof sizes in the implementations of Cardano SL [27], where 70% of participants generate the votes and $k = 2160$; for all these PoW-based sidechain constructions, we calculate proof sizes in the implementations of Ethereum [2], where $n \approx 1.5 \times 10^7$ (as of Sep 2022 [28]). Our constructions adopt the BLS threshold signature scheme [12].
[5] The computation cost for generating proofs, where L is the committee size of our constructions and $L = 2000$ in the PoC implementation of section VI.
[6] Whether the sidechain construction can be applied to PoW-based blockchains with various block difficulties, where / means not applicable.

length linearly. To reduce the proof size, a PoW-based Sidechain construction [6] was proposed to achieve logarithmic proof size. However, the proof size still depends on the chain length, and the storage and communication costs are expanded as the chain grows. The cost for generating proofs also depends on the chain length logarithmically. Such sidechains can only be applied to PoW blockchains with fixed block difficulty, which violates the practical and dynamic requirements of block generation.

In 2019, a PoS-based sidechain construction [8] was proposed, where a set of users are elected as the committee and sufficient number of votes from the committee can be used to generate the proof for cross-chain asset transfers. The construction is based on ad-hoc threshold multisignatures (ATMS) so that it achieves much lightweight proof size. But an ATMS does not have a unique main public key to verify the aggregated signature, then both the public keys of members who vote (or members who fail to vote) and their Merkle-proofs are included in the certificate, such that the mainchain users can compute the public key to verify the final signature. This makes the proof size linear with the committee size and so is the computational cost for generating proofs. Further, if the PoS-based sidechain construction [8] is adapted to the PoS-based blockchains in the fully-adaptive corruption model, such as Ouroboros Praos [9], an extra overhead - the VRF-proof associated with the committee membership will be taken into the certificate, which results in an even larger proof size.

Some SNARK-based sidechain constructions, such as Coda [17] and Zendoo [18], provide constant proof size, but like many other zero-knowledge proof schemes, the computational costs for generating the cross-chain proofs in these constructions are extremely expensive. Other existing works, e.g., BTCRelay [19], Cosmos [20], RSK [21], Polkadot [22], and Plasma [23], lack a a formal security analysis.

It is also interesting to see that almost all existing decentralized sidechains are only designed for specific consensus algorithms. It is unknown whether there is a high-compatible sidechain that is suitable for various consensus algorithms.

C. Comparison With Related Sidechains

The comparison between our constructions and some classic sidechain constructions are given in Table I. Different from existing sidechain approaches, our Ge-Co can be applicable to various blockchains, such as PoW and PoS-based blockchains. Consider that most existing blockchains use ECDSA [24] for digital signatures, while Ge-Co and PoS-Co use the BLS signature scheme [12], we can deploy our sidechain protocols in existing real world blockchains through a soft fork [4]. We also give a formal security analysis for Ge-Co, while other related works except for [8] lack a formal security analysis.

Our constructions achieve optimally succinct and constant proof size, where the certificate only contains the epoch index, the Merkle-tree commitment to pending transactions, the latest main public key and a TSS signature. In the context of PoW-based sidechains, our Ge-Co achieves much smaller proof size than [6][7] [19]. This is because the proof size of Ge-Co does not depend on the chain length and it will not be affected by the chain growth. Furthermore, Ge-Co can be applied to PoW-based blockchains with variable mining difficulties, while other sidechains adopt a fix mining difficulty assumption. Compared to PoS-based sidechains [8], our Ge-Co and PoS-Co also achieve much smaller proof size. Specifically, Ge-Co and PoS-Co are based on TSS and a TSS has a unique main public key to verify the final signature, thus the certificate does not include the public keys of voters and their Merkle proofs. In particular, PoS-Co enables the verification of the TSS signature to indicate the committee membership of voters, and thus it does not require any VRF proof in the certificate.

Different from [6][7] [19], our computational cost for generating proofs is not expanded as the increase of chain length. This is because in our constructions, the certificate is generated by aggregating sufficient votes from the committee, such that the computation cost for generating proofs is linear with scale of the committee.

II. PRELIMINARIES

A. Our Model

We use the same model as in the prior works [9] [25] [29].

Protocol execution: We divide time into discrete units, called slots (or rounds). An epoch includes multiple sequential slots.

Corruption model: Our constructions work in the fully-adaptive or semi-adaptive corruption model. In the fully-adaptive corruption model, the adversary can dynamically corrupt parties immediately. In the D-semi-adaptive corruption model, the adversary can dynamically corrupt parties only after D slots (or rounds). No matter which corruption model is adapted, it is assumed that the ratio of total computational power or stake of honest parties is $h > 1/2$.

Network assumptions: Our schemes work in synchronous or semi-synchronous network. In synchronous network, messages broadcast by a honest party will be received by others within a known maximum delay Δ slots. On the contrary, the semi-synchronous network enables the messages to be received by those within an unknown maximum delay Δ slots.

B. Blockchains and Ledgers

A blockchain (or a chain) is a sequence of blocks, and each block contains the hash value of the previous block. At each slot, the leader proposes a new block. The leader election can be achieved by specific consensus mechanisms, such as PoW [1], PoS [9] [25] and others [30][31]. A secure blockchain must satisfy the chain growth, chain quality and common prefix properties [29].

Definition 1: (Chain Growth). For the chains C_1 and C_2 held by any two honest parties in round r_1 and r_2 respectively, where $r_2 = r_1 + t$, then it must hold that $length(\mathrm{C}_2) - length(\mathrm{C}_1) \geq \tau \cdot t$, where τ is the speed coefficient.

Definition 2: (Chain Quality). For any chain C held by an honest party and any $\ell > \kappa$ consecutive blocks of C, at least μ fraction of these ℓ blocks are mined by honest nodes except for a negligible probability, where $0 < \mu \leq 1$ is the chain quality coefficient, and the ideal chain quality coefficient is equal to h.

Definition 3: (Common Prefix). For the chains C_1 and C_2 held by any two honest parties in round r_1 and r_2 respectively, where $r_1 < r_2$, then it must hold that $\mathrm{C}_1{}^{\lceil k} \preceq \mathrm{C}_2$, where $\mathrm{C}_1{}^{\lceil k}$ denotes the chain obtained by removing the last k blocks from C_1, $\mathrm{C} \preceq \mathrm{C}'$ means that C is a prefix of C', and $k \in \mathbb{N}$ is the common prefix parameter.

We say a transaction is stable in a chain C if it is buried under k subsequent blocks.

C. Threshold Signature Scheme (TSS)

A (t, n)-TSS [12] is a protocol among n parties where only a set of $\geq t$ parties can generate a valid signature on a message m. In a (t, n)-TSS, the secret key sk is split among n players using (t, n) verifiable secret sharing [32], where the public key is pk. Then each player i owns a secret key share sk_i of sk with the public key pk_i. To generate a signature on m, each player i generates a signature share σ_i on m by using sk_i. The aggregator who has $\geq t$ valid signature shares is able to generate a valid signature σ on m that can be verified by pk. The aggregation process never exposes the secret key sk.

The definition of TSS is as follows.

Definition 4: A (t, n)-TSS is a tuple of algorithm $\prod = (TKGen, TSig, TSR, TV)$ including:

1) The threshold key generation algorithm $TKGen$ is run by n players, which takes the global information I as input and returns a secret key share sk_i and a public key pk_i for each party i. Each party can compute the main public key pk.
2) The signature share generation algorithm $TSig$ is run by each player i, which takes (m, I, sk_i) as inputs and returns the signature share σ_i on m.
3) The signature reconstruction algorithm TSR takes m and $\geq t$ signature shares as inputs, and combines them into a single signature σ.
4) The verification algorithm TV takes m, pk, and σ as inputs, and returns true or false.

A secure TSS must satisfy the unforgeability and robustness properties [12][33].

Definition 5: (Unforgeability). No adversary who corrupt $< t$ players can produce a valid signature σ on any new message m^*, given the threshold signatures on input messages $m_1, \ldots, m_k$, where m^* was not submitted as a signing request.

Definition 6: (Robustness). For every adversary that is allowed to corrupt $< t$ players, the algorithm $TKGen$, $TSig$ complete successfully.

Distributed Key Generation (DKG): TSS faces a key generation problem: if one dealer P shares the secret key s among n players, P would know s and sign any messages on behalf of the group. This makes TSS insecure, thus DKG protocols [13][14] [34], [35], [36] are proposed. A (t, n)-DKG protocol allows n players to jointly generate a secret key s with the public key $pk = g^s \mod p$, such that $\geq t$ players can reconstruct s, where $t < n/2$. In a nutshell, each player i secret-shares its own secret z_i and encrypts each share $z_{i,j}$ with the public key of party j and gossips the encrypted shares. The player i also generates a NIZK proof to prove that it shares its secret key correctly. The final secret s is set to be $\sum_i z_i$. Based on the received valid shares, each player reconstructs its public/secret key share for TSS. Furthermore, the main public key can be computed by adding all public keys of parties who have properly shared their secrets. A secure DKG protocol must satisfy the secrecy and correctness properties [13].

Definition 7: (Correctness). (1) All subset of t shares provided by honest parties define the same unique secret key s. (2) All honest parties agree on the same value of $pk = g^s \mod p$, where s is the unique secret guaranteed by (1). (3) s is uniformly distributed in Z_q.

Definition 8: (Secrecy). No adversary who corrupts $< t$ players can learn any information about s except for what is implied by the value of $pk = g^s \mod p$.

D. Anonymous Public Key Encryption (PKE)

In an anonymous PKE [37], a ciphertext does not betray the public key used to generate it. An anonymous PKE scheme is a PKE scheme that additionally satisfies the anonymity property.

Definition 9: (Anonymity). A PKE scheme $E = (Gen, Enc, Dec)$ is said to be anonymous if the advantage of probabilistic polynomial time (PPT) adversaries $\mathcal{A}$ is negligible in the following game with a challenger $\mathcal{C}$:

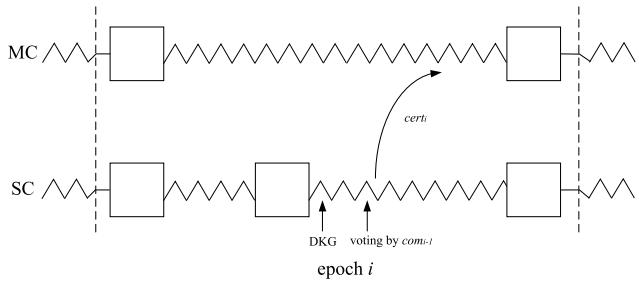


Fig. 2. Our generic sidechain construction, Ge-Co. Each Block is shown as a rectangle and epochs are separated by dashed lines. The blockchain at the top is MC and the one at the bottom is SC. com_{i-1} is the voting committee of epoch $i-1$ and $cert_i$ is the certificate of epoch i.

1) $\mathcal{C}$ runs the key generation algorithm twice to obtain $(pk_i, sk_i) \leftarrow Gen(1^\lambda, \$)$ for $i = 0, 1$, and sends pk_0, pk_1 to $\mathcal{A}$, where λ is a security parameter.
2) $\mathcal{A}$ returns a plaintext message m to $\mathcal{C}$.
3) $\mathcal{C}$ chooses a secret bit b, encrypts m with pk_b to get $ct \leftarrow Enc_{pk_b}(m)$, then returns ct to $\mathcal{A}$.
4) $\mathcal{A}$ outputs a guess b' for the bit b.

The advantage of $\mathcal{A}$ is $|Pr[b = b'] - 1/2|$, and the scheme is anonymous if $\mathcal{A}$ only has a negligible advantage.

E. Security Definition of Sidechain Protocols

We adopt the security definition of sidechains in [38] [39]. Each cross-chain asset transfer is expressed by two transactions: a sending transaction and a receiving transaction, where the former is used to deduct the sender's assets and the latter is used to credit corresponding assets to the receiver.

Definition 10: (The security of sidechain protocols). For two blockchain systems X and Y with ledgers L_x and L_y, each of which satisfies the chain growth, chain quality and common prefix properties, a sidechain protocol is secure if it satisfies the atomicity and timeliness properties.

- *Atomicity:* For cross-chain asset transfers from X to Y, if Tx is a valid sending transaction for X, Tx will be included in L_x then stable, and Tx' will be included in L_y then stable, where Tx' is the corresponding receiving transaction of Tx. Otherwise, both Tx and Tx' will be aborted and not included in L_x and L_y, separately. The cross-chain asset transfers from Y to X is analogous.
- *Timeliness:* For cross-chain asset transfers from X to Y, if Tx is a valid sending transaction for X, eventually Tx and Tx' are included and stable in L_x and L_y, separately. The cross-chain asset transfers from Y to X is analogous.

III. Overview of Sidechain Constructions

In the design of our sidechains, we first present Ge-Co, a generic sidechain construction in the semi-adaptive corruption model. We here give the overview for Ge-Co, which is also depicted in Fig. 2. For ease of explanation, Ge-Co only considers the interaction of a single mainchain and a single sidechain, and can be easily extended to multiple sidechains.

In Ge-Co, upon activating a sidechain successfully, both the mainchain and sidechain systems execute their respective underlying blockchain protocols (e.g., bitcoin) and maintain their respective ledgers. Ge-Co assumes a parent-child relationship between them, where the sidechain users maintain the sidechain and mainchain but the mainchain users only maintain the mainchain. The asset transfers from the mainchain to the sidechain are achieved by directly observation (see Section IV-A for details). In turn, as the mainchain users do not track the sidechain, to achieve the asset transfers from the sidechain to the mainchain, the sidechain users need to provide the mainchain system with sufficient information, called certificates, such that the mainchain system can verify and react to the state of the sidechain. The choice between the two approaches of cross-chain transfers can be made freely. For example, we can assume that both the mainchain and sidechain users only maintain their own chains, then all cross-chain transfers are achieved by cross-chain certificates.

Ge-Co uses the committee selection and threshold signature scheme (TSS) [12] to achieve optimally succinct certificate with constant size, so as to reduce the communication and storage costs. Specifically, to achieve a cross-chain transfer from the sidechain to mainchain, the sender (namely payer) creates and gossips a sending transaction, which is used to deduct the sender's assets. The valid sending transactions will be included in the sidechain and stable. In each epoch i, we randomly elect a small subset of sidechain users as the voting committee. To tolerate the Sybil attack, the selection of the committee is based on user's computational power or stake. Subsequently, all committee members of epoch i jointly invoke a DKG protocol to obtain their public/secret key shares and the main public key for TSS, where the main public key needs to be published to other users. After knowing the main public key pk_i from the committee of epoch i, each committee member of epoch $i-1$ votes for pk_i and some sending transactions of sidechain, by generating a valid signature with its secret key share. Sufficient number of valid votes (i.e., more than the threshold value) are combined into a single signature by TSS, then a valid certificate $cert_i$ including the single signature is generated. Different from [8], the certificate here does not include the public keys of voters and their Merkle proofs, and its size is constant and optimally succinct. This is because Ge-Co is based on TSS, and TSS only needs a unique main public key to verify the final signature.

The committee selection method of Ge-Co ensures a sufficient fraction of committee members are honest, such that the adversary cannot generate a valid certificate on incorrect messages (e.g., invalid sending transactions). Finally, the certificate is submitted to the mainchain network as a specific transaction. Once the certificate is verified successfully and included in the mainchain, the corresponding receiving transactions, which are used to credit corresponding assets to the receivers (namely payee), are created and included in the mainchain. After the transactions being stable, the cross-chain transfers complete.

Ge-Co dose not rely on any trusted party, thus it is fully decentralized. Unlike the previous works, such as [6][7] [8], the committee selection and certificate generation of Ge-Co do not rely on a specific consensus mechanism. This make

Ge-Co generic and may be applied to different blockchains, e.g., PoW-based and PoS-based blockchains, etc. Ge-Co selects a public committee as the voting committee in each epoch. This indicates that Ge-Co can only guarantee the security properties of atomicity and timeliness in the semi-adaptive corruption model. Due to the use of DKG protocols, Ge-Co can tolerate 1/2 adversary bound under the synchronous network, and 1/3 adversary bound under the semi-synchronous network.

Based on Ge-Co, we further propose two concrete instantiations for PoW-based blockchains and PoS-based blockchains. Our PoW instantiation can be applied to variable difficulty chains. The main challenge of such a construction is that the remote mainchain users should be able to check if the blocks of sidechain follow the rule of the pre-specified difficulties, however the mainchain users do not know the varied difficulties. In our PoW instantiation, we enable this check to be performed by the committee members instead of the mainchain users. This is so because the committee members are allowed to know the pre-specified block difficulties. Moreover in our PoW instantiation, we develop a solution to resist the possible selfish mining attack.

To provide stronger security, we further propose PoS-Co, a PoS-based sidechain construction in the fully-adaptive corruption model. PoS-Co is similar to Ge-Co except for the committee selection method. To tolerate the adversary with fully-adaptive corruption, in each epoch, an extra anonymous committee should be elected to invoke a DKG protocol and generate votes. Due to the anonymity, the adversary is not able to adaptively corrupt the committee members, and the security of PoS-Co can be guaranteed. But at the same time, PoS-Co can only tolerate a lower adversary bound than Ge-Co, which is explained in the Section IV-B.

Recall that in the fully-adaptive corruption model, the verification of committee membership may lead to extra overheads [8]. To avoid this, we build PoS-Co in the sense that the verification of the threshold signature implies the verification of committee membership. This does not require us to include the VRF proofs of voters in the certificate, so that the proof size can preserve optimally succinct constant.

IV. Sidechain Constructions With Optimally Succinct Proof

Before describing our constructions, we first review the definition of transactions given in [8]. Assume that the mainchain (and sidechain, resp) system maintains the ledger MC (and SC, resp.). The index of MC (and SC, resp.) is denoted by $MCID$ (and $SCID$, resp.). Each transaction has the form $tx = (txid, lid, (send, sAcc), (rec, rAcc), v, \sigma)$, where

- $txid$ is the transaction identifier that uniquely identifies the transaction.
- $lid \in \{MCID, SCID\}$ is the ledger index where the transaction belongs.
- $send \in \{MCID, SCID\}$ is the index of the sender (or payer) ledger and $sAcc$ is the sender account (represented by the public key of the account).
- $rec \in \{MCID, SCID\}$ is the index of the recipient (or payee) ledger and $rAcc$ is the recipient account (again represented by a public key).
- v is the amount to be transferred.
- σ is the signature of the sender on tx.

The ledger MC (and SC, resp.) only includes transactions with $lid = MCID$ (and $lid = SCID$, resp.). A transaction is valid if $txid$ is fresh, the sender owns sufficient assets, and the signature of the sender is valid. Note that the definition of transactions is account-based, and the UTXO-based definition can also be obtained similarly.

If $send = rec$, the transaction is called a local transaction and processed by the underlying blockchain protocol. Otherwise, the transaction is called a cross-chain transaction and processed by our sidechain protocols.

Table II shows the frequently used notations in the paper.

TABLE II
Notation

Notation	Description
MC	the ledger maintained by the mainchain system
SC	the ledger maintained by the sidechain system
$MCID$	the index of MC
$SCID$	the index of SC
t	the threshold value for TSS
tx_{send}	a sending transaction
tx_{rec}	a receiving transaction
$txid$	the transaction identifier
$sAcc/rAcc$	the sender/recipient account
h	the fraction of honest nodes in each chain
f	the fraction of malicious nodes in each chain
com_i/com'_i	the public/anonymous voting committee of epoch i
S	the initial voting committee size
L	the voting committee size
K	the minimum section length that the chain quality property is satisfied
k	the common prefix parameter
i	the index of epoch i
i_{start}	the index of the first epoch in the initialization of sidechain
$cert_i$	a certificate in epoch i
(vk, sk)	a pair of public/secret keys
PK	the latest main public key for TSS accepted by mainchain users
pk_i	the main public key for TSS accepted by sidechain users in epoch i
m	a message
$pending_i$	all the sending transactions included in SC during epoch $i-1$
$\langle pending_i \rangle$	a Merkle-tree commitment to $pending_i$
σ	a signature
P/P'	a sidechain user
δ	a negligible failure probability

A. Generic Construction in Semi-Adaptive Corruption Model

We first propose a generic sidechain construction, Ge-Co, to achieve assets transfers between blockchains, which is also depicted in Fig. 3. Based on Ge-Co, we also propose two concrete instantiations for PoW-based blockchains and PoS-based blockchains.

Ge-Co works in either synchronous or semi-synchronous network. According to the definition of DKG protocols [13], 1/2 is the optimal adversary bound. In addition, DKG protocols assume a broadcast channel for all nodes to reliably communicate with each other [32][34]. In practice, this is usually

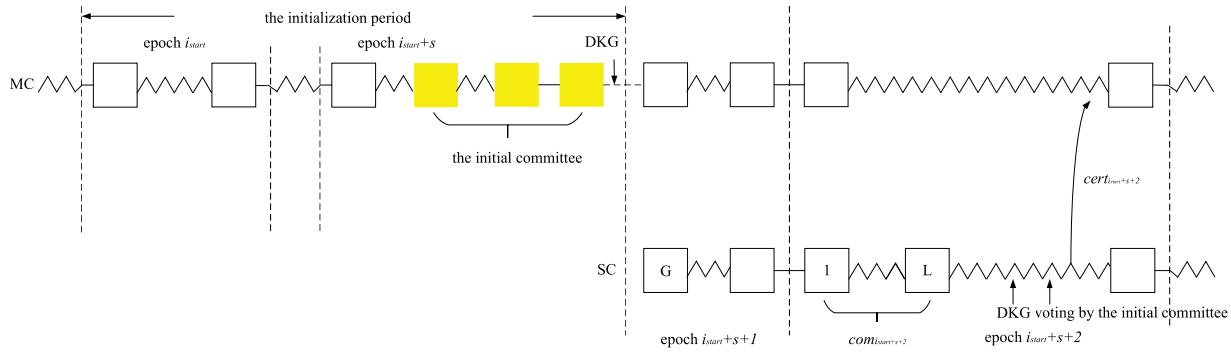


Fig. 3. Our sidechain construction, Ge-Co. The blocks in yellow are created by the latest stable S leaders who support the creation of sidechain in the initialization period, and the corresponding leaders constitute the initial committee. $com_{i_{start}+s+2}$ is the committee of epoch $i_{start}+s+2$.

implemented by byzantine fault tolerance (BFT) protocols [40]. Due to the fact that BFT protocols can only tolerate 1/3 adversary bound in semi-synchronous network [41], in Ge-Co based on DKG protocols, we assume that $f < 1/2$ under the synchronous network and $f < 1/3$ under the semi-synchronous network.

Ge-Co assumes $2R$-semi-adaptive corruption model, namely the adversary can dynamically corrupt parties only after $2R$ slots, where each epoch contains R slots. We further propose a PoS-based sidechain, PoS-Co, under the fully-adaptive corruption model in the following subsection.

In general, Ge-Co includes the initialization of sidechain, chain maintenance and cross-chain assets transfers. The main differences from the prior work [8] are the initialization of sidechain and the assets transfers from the sidechain to mainchain. For the sake of efficiency and simplicity, Ge-Co adopts the BLS TSS [12], and each user owns a BLS key pair as its long-term public/secret key pair. We note that other TSS are also pluggable to Ge-Co.

Initialization of Sidechain and Chain Maintenance: Any mainchain user can request to create a sidechain. Once some specific conditions are satisfied, the sidechain is activated successfully. The sidechain has its own slots (or round) and epochs synchronized with the mainchain.

The activation process in Ge-Co is as follows. If a mainchain user (holding a long-term key pair (vk, sk)) requests to create a sidechain SC, he creates and gossips a special supporting transaction $(sidechain\_support, SCID, vk, \sigma)$, where σ is the signature of the user on $(sidechain\_support, SCID, vk)$ with sk. The supporting transaction will be included in MC, and i_{start} denotes the index of the epoch when the first supporting transaction has appeared in MC. After seeing the transaction, any other user who supports the creation of sidechain also creates and gossips a supporting transaction with his own long-term secret key. The leaders of mainchain include valid supporting transactions into their blocks until epoch $i_{start}+s$, where the period of s epochs is sufficiently long to ensure that all valid supporting transactions are included in MC and become stable, and s can be set according to the liveness property [29]. If the number of leaders[1] who support the sidechain is more than a parameter S, the sidechain is activated successfully, and all the users whose supporting transactions are stable in MC from epoch i_{start} to epoch $i_{start}+s$ act as the sidechain participants. In the synchronous network, $S = (\left\lceil 16\log(\frac{1}{\delta})/(h-\frac{1}{2})^2 \right\rceil + 1)$ and in the semi-synchronous $S = (\left\lceil 27/4\log(\frac{1}{\delta})/(h-\frac{2}{3})^2 \right\rceil + 1)$, where δ is a negligible failure probability.

If the sidechain is activated successfully, we elect the latest S stable leaders as the initial committee (If a user is elected c times, the weight of the user in the committee is c). The initial committee selection method can guarantee that the fraction of honest members in the initial committee is h', where $h' > 1/2$ in the synchronous network and $h' > 2/3$ in the semi-synchronous network. See Lemma 1 for more details.

Subsequently, the initial committee members jointly invoke a (t, S)-DKG protocol to compute their new public/secret keys shares and the main public key for TSS, where $t = 1/2S + 1$ in synchronous network and $t = 1/3S + 1$ in the semi-synchronous network. Finally, the initial committee members need to publish the main public key to all the mainchain participants, which is used for verifying cross-chain certificates later. Specifically, any initial committee member generates the signature on the main public key with his long-term secret key, and gossips it in the mainchain network. The public key with $\geq t$ distinct valid signatures from the initial committee will be accepted as the main public key, preventing users from accepting an incorrect public key published by the adversary. We note that in the above process, the mainchain is used as a broadcast mechanism, e.g., including the encrypted shares, NIZK proofs and signatures in the mainchain, so that those messages can be publicly known. After the sidechain system is activated successfully, both the mainchain and the sidechain systems execute their respective underlying blockchain protocols, and maintain their respective ledgers. We assume the sidechain participants also maintain MC and the mainchain participants only maintain MC.

Cross-chain assets transfers from mainchain MC to sidechain SC: A cross-chain transfer from MC to SC is achieved by direct observation [8]. At first, the sender creates and gossips a sending transaction tx_{send} that is used to deduct the corresponding assets of sender, where $lid = send = MCID$,

[1]It is required that the blocks generated by these leaders have been stable.

and $rec = SCID$. If tx_{send} is valid, it will be included and stable in MC. Then the sender creates and gossips a corresponding receiving transaction tx_{rec} that is used to credit corresponding assets to the recipient. tx_{send} is identical to tx_{rec} except for $lid = SCID$. Once the sidechain participants observe that tx_{send} is stable in MC, tx_{rec} will be included in SC as a valid transaction. When tx_{rec} is stable in SC, the cross-chain transfer completes.

Cross-chain assets transfers from sidechain SC to mainchain MC: The cross-chain transfers from SC to MC are achieved by certificates. First, the sending transactions are created and included in SC. Second, in each epoch i, a set of sidechain users are selected as the voting committee, and they vote for some stable sending transactions in SC. Then a certificate consisting of enough valid votes is generated and sent to the mainchain. After the certificate is verified successfully, the corresponding receiving transactions are included and stable in MC. The details are described as follows.

1) Create sending transactions: When a sidechain participant who wishes to transfer assets from SC to MC, he first creates a sending transaction $tx_{send} \leftarrow (txid, SCID, (SCID, sAcc), (MCID, rAcc), v, \sigma)$ with his long-term secret key, then gossips the transaction to the sidechain network. Each valid sending transaction will eventually be included in SC and stable.

2) Select the voting committee: When the sidechain is activated successfully in the epoch $i_{start} + s + 1$, the initial committee elected in the sidechain initialization period act as the voting committee of this epoch $com_{i_{start}+s+1}$. Subsequently for each epoch $i \geq i_{start} + s + 2$, we elect the leaders of the first K blocks of sidechain in epoch i as the voting committee com_i, where K is minimum section length that the chain quality property is satisfied. The committee selection method can resist the Sybil attack because each user is selected as a committee member in proportion to its computational power or stake.

According to chain growth and common prefix properties, at the slot $2K + 2k$ of epoch $i \geq i_{start} + s + 2$, there must be at least $K + k$ blocks in the epoch and the first K blocks have become stable, where k is the common prefix parameter. This also means all committee members of com_i have been determined. Due to the chain quality property, the ratio of malicious members in com_i is less than $1 - h$. Subsequently, all committee members of com_i jointly invoke a $((1-h)K + 1, K)$-DKG protocol to generate their new public/secret key shares and the main public key pk_i for TSS, then publish pk_i to all the sidechain participants, where the threshold value for a user to accept pk_i is $(1 - h)K + 1$. In the above process, the sidechain is used as a broadcast mechanism.

3) Vote and generate the certificates: For each epoch $i \geq i_{start} + s + 2$, after accepting the main public key pk_i from the committee of epoch i, each member of com_{i-1} votes for the message (i, pk_i and all the sending transactions included in the sidechain during the epoch $i - 1$) [2] by generating a valid signature on the hash of the message with its new secret key share. If the weight of a member in com_{i-1} is c, its signature on $cert_i$ is considered as c votes for $cert_i$. Each member gossips his vote into the sidechain network. We denote the committee size of each epoch as L. If a sidechain user receives more than $(1 - h)L + 1$ valid votes from com_{i-1}.[3] he aggregates these vote signatures into a final signature σ_i. Then a valid certificate $cert_i$ is generated, which is formally described in Algorithm 1.

Algorithm 1: Generating a Certificate (i).

The algorithm is run by sidechain users in epoch i, where $TSR(\cdot)$ is the signature reconstruction algorithm of TSS.
1: $pending_i \leftarrow$all the sending transactions included in SC during the epoch $i - 1$
2: $\langle pending_i \rangle \leftarrow$ a Merkle-tree commitment to $pending_i$
3: $pk_i \leftarrow$ the accepted main public key of epoch i
4: $m \leftarrow (i, \langle pending_i \rangle, pk_i)$
5: **If** $d = valid\_votes(m) \geq (1 - h)L + 1$ **then**
6: $\sigma_i \leftarrow TSR(m, \{(pk_j, \sigma_j)\}_{j=1}^{d})$
7: **return** $cert_i \leftarrow (i, \langle pending_i \rangle, pk_i, \sigma_i)$

Notice that all sidechain users only wait for votes from com_{i-1} until epoch $i + 1$. The certificate $cert_i$ needs to be periodically submitted to the mainchain network, even if no sending transaction is included in SC in an epoch.

4) Verify the certificates: Each mainchain user uses PK to store the latest main public key and an array $valid\_txs$ to mark the Merkle-tree root of valid sending transactions, where $valid\_txs$ is initialized as null. After receiving a certificate $cert_i$, the mainchain users check the validity of $cert_i$ by Algorithm 2. The algorithm 2 first checks whether the epoch index in $cert_i$ is equal to the current epoch index of mainchain, then checks the validity of the TSS signature in $cert_i$. If $TV(m, PK, \sigma_i) = 1$ holds, this means sufficient (i.e., more than the threshold value) members of com_{i-1} have voted for $cert_i$. After verifying $cert_i$ successfully, each mainchain user updates PK for verifying the certificates of next epoch, and updates $valid\_txs[\langle pending_i \rangle]$ for verifying receiving transactions. The valid $cert_i$ will be included in MC.

5) *Create receiving transactions*. After the certificate $cert_i$ becomes stable in MC, each sidechain user, who wishes to transfer assets from SC to MC, creates the corresponding receiving transaction $tx_{rec} \leftarrow (txid, SCID, (SCID, sAcc), (MCID, rAcc), v, \pi, \sigma)$ with its long-term secret key, where π is the Merkle-tree proof of the corresponding sending transaction to $\langle pending_i \rangle$, and then broadcasts the receiving transaction to the mainchain network.

6) *Verify receiving transactions:* After receiving a receiving transaction tx_{rec}, the mainchain users check the validity of tx_{rec} by Algorithm 3. The algorithm checks whether tx_{rec} is fresh, and properly signed, then checks whether tx_{rec} contains a correct Merkle-tree proof proving the presence of its corresponding sending transaction in a verified certificate. After verifying

[2] According to the chain growth property, a transaction becomes stable in SC at most $2k$ slots after its inclusion, and thus all these transactions have become stable at this moment.

[3] In particular, for epoch $i = i_{start} + s + 2$, the voting bound for generating certificate is $1/2L + 1$ in the synchronous network, and is $1/3L + 1$ in semi-synchronous network.

Algorithm 2: Verifying a Certificate ($cert_i \leftarrow (i, \langle pending_i \rangle, pk_i, \sigma_i)$).

The algorithm is run by each mainchain user. Assume that the current epoch index of the mainchain is i'. $TV(\cdot)$ is the verification algorithm of TSS. PK is initialized as the accepted main public key in the initialization period.

1: **if** $i \neq i'$ **return** 0
2: **else**
3: $m \leftarrow (i, \langle pending_i \rangle, pk_i)$
4: **if** $TV(m, PK, \sigma_i) = 1$ **then**
5: $PK \leftarrow pk_i$, $valid\_txs[\langle pending_i \rangle] \leftarrow true$;
6: **return** 1
7: **else return** 0

Algorithm 3: Verifying a Receiving Transaction ($tx_{rec} \leftarrow (txid, SCID, (SCID, sAcc), (MCID, rAcc), v, \pi, \sigma)$).

The algorithm is run by mainchain users. Old is an array used to check the freshness of transactions, and initialized as null.

1: $m \leftarrow$
$(txid, MCID, (SCID, sAcc), (MCID, rAcc), v, \pi)$
2: $(tx_{send}, Merkle - path, mtr) \leftarrow \pi$
3: $(m', \sigma') \leftarrow tx_{send}$
4: **if** $m' = (txid, SCID, (SCID, sAcc),$
$(MCID, rAcc), v)$
$\wedge Ver(m, sAcc, \sigma) \wedge (\neg Old[txid]) \wedge valid\_txs[mtr] \wedge$
$MTR\_Ver(mtr, Merkle - path)$, **then return** 1
5: **else return** 0
6: $Old[txid] \leftarrow 1$

successfully, tx_{rec} will be included in MC. When tx_{rec} is stable in MC, the cross-chain transfer completes.

Concrete Instantiations: The core idea of Ge-Co is to periodically select a voting committee to support cross-chain asset transfers, and guarantee a sufficient fraction of committee members are honest by the chain quality property. As the chain quality property is a general property independent of specific consensus algorithms, and the voting process of Ge-Co also does not rely on specific consensus algorithms, Ge-Co is a generic construction and it can be applied to different blockchain systems. Based on Ge-Co, we now give two concrete instantiations for PoW-based and PoS-based blockchains. In the instantiations, some optimization and adjustment for the voting committee selection method are needed.

When applying Ge-Co to PoW-based blockchains, the chain quality falls back to $1 - \frac{1-h}{h}$ under the selfish mining attack [15], such that more malicious members are elected in each committee which incurs security risks. To prevent the attack, inspired by FruitChains [42], we need to include the leaders of all valid fork blocks mined in first $2L$ rounds of each epoch into the committee, such that the fraction of malicious members in the committee is still $< 1 - h$.

Specifically, in the epoch $i > i_{start} + s + 1$, each block mined in the first $2L$ rounds need to include all valid fork blocks mined at the same period, and the miners who mine the first L blocks or whose blocks are contained in the first L blocks are elected as the committee of epoch i[4]. Consider that the adversary could withhold some old valid blocks, and release them during the first $2L$ rounds of an epoch, thereby having more opportunity of electing the committee of this epoch. In order to resist this malicious behavior, only recent fork blocks will be contained. To this end, each block header needs to add a pointer h' pointing to the recent stable block B, implying the block is mined after B. Each leader in the round j of each epoch i (where the round j belongs to the first $2L$ rounds of the epoch) only collects the headers of recent valid fork blocks of which h' is the reference of one of the latest stable j blocks in the chain. And the leader further contains these block headers into its block. Correspondingly, the committee size of com_i is denoted as L', and the threshold value (for accepting the main public key of epoch i and generating a valid certificate of epoch $i+1$) is set as $(1-h)L' + 1$. Due to the fact that PoS-based blockchains do not affected by the selfish mining attack, when applying Ge-Co to the PoS context, we directly elect the first K leaders of sidechain in each epoch as the voting committee of the epoch. However for committee-based PoS blockchains, such as snow white [43], a consensus committee is elected in each epoch, thus the consensus committee can also take the role of the voting committee of Ge-Co.

B. PoS-Based Sidechains in Fully-Adaptive Corruption Model

We here propose PoS-Co, a PoS-based sidechain construction in the fully-adaptive corruption model, where the adversary can corrupt nodes without any delay, and PoS-Co still preserves constant proof size.

In the existing DKG protocols, the identities of all participants are known, thus in full-adaptive corruption model, the adversary can corrupt the voting committee and the security of our scheme can not be achieved. In order to solve this security problem, inspired by [16], PoS-Co selects an additional anonymous committee to invoke a DKG protocol and generate votes in each epoch, where the identities (or the long term keys) of committee members are hidden. The adversary cannot adaptively corrupt anonymous committee members, and thus the security of PoS-Co is guaranteed. Unlike the committee selection method used in [16] that requires a trusted dealer to secretly share the secret key of the sender among the initial committee, PoS-Co enables the anonymous committee members to jointly invoke a DKG protocol to generate the key pairs for TSS. thus PoS-Co does not rely on any trusted party and it is more efficient by processing the cross-chain transactions for every epoch. In addition, different form [16] using the VRF function [10] to select a public committee (also called the nominating committee) in each epoch, we select the public committee based on the chain quality,

[4]In the initialization period of the sidechain, each mainchain leader also needs to include all valid recent fork blocks whose leaders support the sidechain in its block, and the latest S leaders, who generate a stable block of MC or whose block are contained in a stable block of MC from epoch i_{start} to epoch $i_{start+s}$, are elected as the initial committee.

which avoids the additional costs of generating, broadcasting and verifying the VRF proofs of public committee members.

The initialization of sidechain, chain maintenance and cross-chain transfers from MC to SC are same as Ge-Co. The main difference relies on the cross-chain assets transfers from SC to MC, in particular, the methods of the committee selection and voting. The following descriptions focus on the difference.

Select the anonymous voting committee: In each epoch i, the public committee com_i are elected as described in Ge-Co. Each member P of the committee com_i of epoch $i \geq i_{start} + s + 1$ randomly chooses a sidechain user P' based on user's stake before generating its block. Then P chooses a new ephemeral public/secret key pair for P', and encrypts the ephemeral secret key of P' with the long-term public key of P'. P adds the ciphertext c along with the ephemeral public key of P' in its block, and erases the ephemeral secret key of P'. All sidechain users may try to decrypt c but only P' can obtain the correct ephemeral secret key. All the users chosen by com_i constitute the anonymous voting committee com'_i. This is because all sidechain users except for com_i and com'_i only know the ephemeral public keys of members of com'_i, rather than the long term public keys (namely the identities) of these members. In addition, an anonymous PKE is also used to generate c, such that c does not reveal the long term public key of P'.

Due to the chain growth and common prefix properties, at the slot $2L + 2k$ of each epoch $i \geq i_{start} + s + 2$, the first L blocks of the epoch are stable and the anonymous committee com'_i has been determined, where $L = K$ is minimum section length that the chain quality property is satisfied and k is the common prefix parameter. Then the members of com'_i jointly invoke a $((1-h)L+1, L)$-DKG protocol to generate new public/secret key shares and the main public key for TSS.[5] Note that when invoking the DKG protocol, each member secret-shares its ephemeral secret key among com'_i and encrypts each share s_j with the ephemeral public key of $j-th$ member. Further, committee members should publish the main public key to other sidechain users. Specifically, each member of com'_i generates a signature on the main public key pk_i with its ephemeral secret key and gossips the signature on pk_i. The sidechain participants will only accept pk_i if there are $\geq (h(1-h)+(1-h))L+1$ distinct valid signatures from com'_i.

The subsequent processes of voting and generating/verifying the certificates are similar to those given in Ge-Co, except that the anonymous committee com'_{i-1} generate the votes with their new secret key shares, where the threshold value for a valid certificate is $(h(1-h)+(1-h))L+1$. The form of each certificate in PoS-Co is the same as that in Ge-Co, such that PoS-Co maintains optimal succinct proof. We now give more details on the threshold value $(h(1-h)+(1-h))L+1$. Observe that the honest members of com_i will randomly pick up the members of com'_i from users based on their stakes, but the malicious members may always prefer to choose malicious users. In this case, the fraction of malicious members in each anonymous committee should be less than $h(1-h)+(1-h)$. In the process of selecting the anonymous committee and invoking the DKG protocol, the identities of anonymous committee members are not revealed, so the adversary with fully-adaptive corruption cannot adaptively corrupt the anonymous committee members, then we can conclude that the fraction of malicious members in each anonymous committee com'_i remains less than $h(1-h)+(1-h)$. This also means that the threshold value can prevent malicious anonymous committee members from generating enough signatures on an incorrect message, and the security of PoS-Co can be guaranteed. Since we use a DKG protocol in PoS-Co, to ensure the security of PoS-Co, we should guarantee $h(1-h)+(1-h) < 1/2$ in the synchronous network (i.e., $h > 3/4$), while $h(1-h)+(1-h) < 1/3$ in the semi-synchronous network (i.e., $h > 4/5$).

[5]After the initialization, the initial anonymous committee members also run the DKG protocol to generate the public/secrt key shares, and publish the main public key to all mainchain users.

C. Discussions

Although providing greater compatibility and smaller proof size than existing sidechains, our constructions still have some limitations. First, for the compatibility, our PoS-Co cannot be easily converted to support PoW-based blockchains. To resist the Sybil attack, the public committee members need to elect the anonymous members based on user's stake or computational power. But the computational power here is hard to be quantified. A possible solution is that the public members do the selection from all the previous leaders of the current sidechain, so that the user who owns more computational power may have a better opportunity to run for the leaders and anonymous committee members.

Second, like other vote-based sidechains [8], our constructions only work in the synchronous or semi-synchronous network so as to capture the timeless property. There may be a need to design sidechains that can work in the asynchronous network. We state that asynchronous consensus [44], [45], [46] may be an appropriate technical cornerstone for such a design.

Third, our constructions require enough voting committee members to remain online till enough votes are collected, but users may choose to be offline temporarily or permanently, thus additional measures need to be used to ensure the normal implementation of our constructions. Similar to other vote-based schemes [47] [48], we use incentive mechanisms, e.g., providing transaction fees, to stimulate the members to remain online and give their votes. Specifically, when each cross-chain transfer completes, the committee members who generate the votes will be awarded some transaction fees. Even if cross-chain transactions of an epoch still do not obtain enough votes by using incentives, then similar to the related works [48][49] [50], we re-vote on these unprocessed cross-chain transactions. For example, due to too many off-line voting committee members of epoch i, network delays and other issues, it is unable to collect enough valid votes for $pending_{i+1}$, where $pending_{i+1}$ are all the valid sending transactions included in SC during epoch i, then the voting committee members of epoch $i+1$ will vote for $pending_{i+1}$ and $pending_{i+2}$ at the same time.

V. Security Analysis

In this section, we analyze the security of Ge-Co, and prove that Ge-Co satisfies the atomicity and timeliness properties defined in Section II-E. Similarly, we can also prove PoS-Co is a secure sidechain construction. In the following descriptions, we use the PoS instantiation of Ge-Co as a example to analyze the security of Ge-Co, where the first K leaders of sidechain in each epoch as the voting committee of the epoch.

First, we prove that for Ge-Co, in the synchronous network, the fraction of honest members in the initial committee $h' > 1/2$ except with negligible probability by Lemma 1, where we chose the parameters referring to [41]. Similarly, we can prove that in the semi-synchronous network, $h' > 2/3$ except with negligible probability.

Lemma 1: Assume that the total stake of mainchain is arbitrarily large and the fraction of stake of honest users is h, where $h = 1/2 + \epsilon$, then for Ge-Co, the fraction of honest members in the initial committee is $> 1/2$ except with negligible probability δ, where the initial committee size $S = \lceil 16 \log(\frac{1}{\delta}) \,/\, \epsilon^2 \rceil + 1$.

Proof: The initial committee members in Ge-Co are randomly elected because the leaders of mainchain are elected randomly. As the total stake of mainchain is assumed to be arbitrarily large, so each member is elected independently, and the probability of the elected member being honest is $1/2 + \varepsilon$. Let Y be the number of honest members in the initial committee. Clearly, $E(Y) = (1/2 + \varepsilon) \cdot S$. Then

$$E[Y]\left(1 - \frac{\epsilon}{2}\right) = \left(\frac{1}{2} + \epsilon\right) S \left(1 - \frac{\epsilon}{2}\right) > \frac{S}{2}.$$

Given that $S = \lceil 16 \log\left(\frac{1}{\delta}\right) / \epsilon^2 \rceil + 1$, we obtain that

$$\begin{aligned} \Pr\left[Y \le \frac{S}{2}\right] &\le \Pr\left[Y < \left(1 - \frac{\epsilon}{2}\right) E[Y]\right] \\ &\le \exp\left(-\left(\frac{\epsilon}{2}\right)^2 \frac{E[Y]}{2}\right) \\ &\le \exp\left(-\epsilon^2 \cdot S/16\right) = \delta, \end{aligned}$$

where the formula marked $*$ is due to the Chernoff bound. □

Second, we prove that Ge-Co satisfies the atomicity property by Lemma 2.

Lemma 2: Assume that both MC and SC satisfy the chain growth, chain quality and common prefix properties, H is a collision-resistant hash function, the DKG protocol and the threshold signature scheme are secure, then under $2R$-semi-adaptive corruption, Ge-Co satisfies the atomicity property except with negligible probability.

Proof: We let $\mathcal{A}$ be an arbitrary adversary against the property of atomicity. At each epoch j, $\mathcal{A}$ monitors the mainchain and sidechain accepted by honest users. $\langle pending_j \rangle$ is the Merkle-tree root of sending transactions in a certificate $cert_j = (j, < pending_j >, pk_j, \sigma_j)$. pk_j is the main public key included in $cert_j$. $\mathcal{A}$ collects all the sending transactions of SC in epoch $j-1$, and creates the corresponding Merkle-tree root $\langle pending_j{}' \rangle$. $\mathcal{A}$ also collects the main public key pk'_j with more than $(1-h)L+1$ valid signatures from the committee of epoch j, and similar for pk'_{j-1}. [6] $\mathcal{A}$ checks whether the following condition is satisfied:

$$\begin{aligned} &TV(m, pk'_{j-1}, \sigma_j) \wedge ((pk_j \neq pk'_j) \vee \langle pending_j \rangle \\ &\quad \neq \langle pending_j{}' \rangle) \end{aligned} \tag{1}$$

The condition (1) means that the certificate $cert_j$ is verified successfully despite that $cert_j$ includes incorrect messages.

According to Definition 6, if $\mathcal{A}$ is able to break the atomicity property of Ge-Co, there are two cases. In the first case, a sending transaction Tx is not included and stable in MC, but the corresponding receiving transaction Tx' is included and stable in SC. In our design, the sidechain users are able to directly observe MC. Then Tx' is included and stable in SC only after Tx is stable in MC. And thus the first case will never happen.

In the second case, Tx is not included and stable in SC, but its receiving transaction Tx' is included in MC and stable. To analyze the success probability of $\mathcal{A}$, we define three events as follows:

1) SC-FORGE: $\mathcal{A}$ is successful in transfers of assets from SC to MC (namely Tx is not included and stable in SC, but Tx' is included and stable in MC);
2) TSS-FORGE: $\mathcal{A}$ finds an epoch j for which the condition (1) occurs.
3) HASH-COLLISION: $\mathcal{A}$ finds a hash function collision.

To show that the probability of SC-FORGE is negligible, we first prove that when $\mathrm{SC-FORGE}$ occurs, then either $\mathrm{TSS-FORGE}$ or $\mathrm{HASH-COLLISION}$ happens (in Claim 1); and further we show that the probabilities of TSS-FORGE and HASH-COLLISION are negligible (in Claim 2).

Claim 1: $\Pr[\mathrm{SC-FORGE}] \le \Pr[\mathrm{TSS-FORGE}] + \Pr[\mathrm{HASH-COLLISION}]$.

When $\mathrm{SC-FORGE}$ occurs, it means that Tx is not included and stable in SC, but its receiving transaction Tx' is included and stable in MC. We consider two cases. In the first case, a Merkle-tree root $\langle pending_i \rangle$ includes an invalid sending transaction Tx. Then in Ge-Co, for the message m containing $\langle pending_i \rangle$, only malicious committee members will generate votes for m. For $i = i_{start} + s + 2$, due to the lemma 1, the number of malicious members in the initial committee is $(1-h')L$ (note $h' > 1/2$ in the synchronous network and $h' > 2/3$ in the semi-synchronous network), thus there are no enough valid votes for m. For $i > i_{start} + s + 2$, and due to the chain quality property, the number of malicious committee members of epoch $i-1$ is less than $(1-h)L$. Thus there are still no enough valid votes for m. If $\mathcal{A}$ can find an $\mathrm{TSS-FORGE}$ in epoch i, $\mathcal{A}$ can generate a certificate for m and the certificate can be verified successfully. Then Tx' will be included and stable in MC, thus $\mathrm{SC-FORGE}$ occurs.

In the second case, $\langle pending_i \rangle$ only includes some valid stable sending transactions $\{Tx^*\}$ of SC and the correct main public key of epoch i. In Ge-Co, each honest member of the committee in epoch $i-1$ will vote for $\langle pending_i \rangle$. Due to lemma 1 and the chain quality property, there will be enough

[6] In particular, for epoch $i = i_{start} + s + 1$, the threshold value for accept the main public key is $1/2L + 1$ in the synchronous network and $1/3L + 1$ in the semi-synchronous network.

valid votes for the message m containing $\langle pending_i \rangle$. Then a valid certificate $cert$ for m will be generated. If $\mathcal{A}$ finds an invalid sending transaction Tx satisfying $H(Tx) = H(Tx^*)$ (namely a hash collision is found), then $\mathcal{A}$ creates a proof-of-inclusion π for Tx proving Tx belongs to $\langle pending_i \rangle$, but in fact Tx is actually not contained in $\langle pending_i \rangle$ and SC. As $cert$ and π will be verified successfully, the corresponding receiving transaction Tx' will be included in MC and stable. Thus, $\mathrm{SC-FORGE}$ occurs.

Claim 2: $\Pr$[TSS-FORGE] and $\Pr$[HASH-COLLISION] are negligible.

As H is a collision resistant hash function, $\mathcal{A}$ only can find such a collision in Claim 1 with negligible probability. Then we prove that $\Pr[\mathrm{TSS-FORGE}]$ is negligible. When $\mathrm{TSS-FORGE}$ occurs, there exists an epoch j for which the condition (1) holds. When $j > i_{start} + s + 2$, according to the chain quality, the malicious members in epoch $j-1$ is at most $(1-h)L$. Due to the security of DKG, these malicious members cannot reconstruct the corresponding main secret key, so that they cannot directly generate a valid TSS signature on the message $m = (j, \langle pending_j \rangle, pk_j)$. On the other hand, since $(pk'_j \neq pk_j)$ or $\langle pending_j \rangle \neq \langle pending_j{}' \rangle$, only the malicious committee members of epoch $j-1$ will vote for m including pk_j or $\langle pending_j \rangle$. Thus there are at most $(1-h)L$ valid votes on m. Therefore, the occurrence of $\mathrm{TSS-FORGE}$ breaks the security of TSS. Similarly, when $j = i_{start} + s + 2$, according to the lemma 1 and the security of DKG, the malicious committee members cannot directly generate a valid TSS signature on m and m cannot obtain enough valid votes from the initial committee. Therefore, the occurrence of $\mathrm{TSS-FORGE}$ breaks the security of TSS and $\Pr[\mathrm{TSS-FORGE}]$ is negligible. □

Third, we prove Ge-Co satisfies the timeless property by Lemma 3.

Lemma 3: Assume that both MC and SC satisfy the chain growth, chain quality and common prefix properties, the DKG protocol and the threshold signature scheme are secure, then under $2R$-semi-adaptive corruption, Ge-Co satisfies the timeless property except with negligible probability.

Proof: In Ge-Co, for the transfers of assets from SC to MC, if a sending transaction Tx is valid for SC, then Tx will be eventually included and stable in SC according to the chain growth and chain quality properties of SC. Assume Tx is included in SC during epoch $i-1$, then Tx and other valid sending transactions of epoch $i-1$ in SC will be included in $\langle pending_i \rangle$. Then the honest committee members of epoch $i-1$ will vote for the message m including $\langle pending_i \rangle$ and the correct main public key. According to the chain quality properties of SC and lemma 1, there are enough honest committee members, and m will obtain sufficient valid votes. Under synchronous and semi-synchronous network, these votes will be eventually collected by sidechain users. Then a valid certificate $cert$ for m will be generated and submitted to the mainchain network. After verifying $cert$ successfully, the corresponding receiving transaction Tx' of Tx will be created. According to the chain growth and chain quality properties of MC, Tx' will be eventually included in MC and then stable.

TABLE III
TIME OVERHEADS OF VOTES AND CERTIFICATES OF GE-CO

Time to generate a vote	~ 8	ms
Time to verify a vote	~ 1	ms
Time to form a certificate	~ 652	ms
Time to verify a certificate	~ 103	ms

For the transfers of assets from MC to SC, if a sending transaction Tx is valid for MC, then Tx will be eventually included and stable in MC according to chain growth and chain quality of MC. Due to the fact that the sidechain users can directly observe MC, after observing Tx is stable in MC, Tx' will be created, where Tx' is the receiving transaction of Tx, and Tx' will be eventually included in SC and stable.

Finally, we can directly obtain the following Theorem 1 according to Lemma 2 and Lemma 3.

Theorem 1: Assume that both MC and SC satisfy the chain growth, chain quality and common prefix properties, H is a collision-resistant hash function, the DKG protocol and the threshold signature scheme are secure, then under $2R$-semi-adaptive corruption, Ge-Co satisfies the atomicity and timeless properties except with negligible probability.

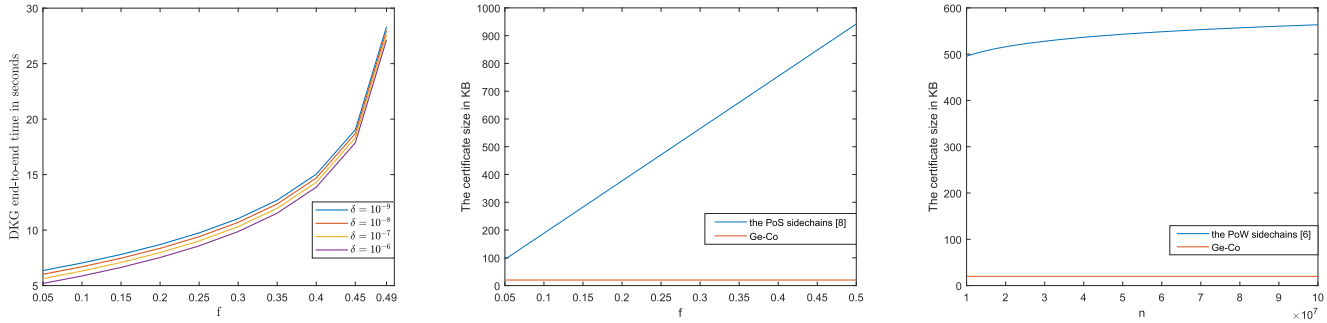
VI. EXPERIMENTAL EVALUATIONS

To demonstrate the efficiency of our design, we develop a proof-of-concept (PoC) implementation of Ge-Co, and compare it with the PoS-based [8] and PoW-based sidechains [6].

Implementation details: We develop the implementation in standard C language. Our implementation uses the SHA-256 hash function to compute the Merkle-tree commitment to the pending transactions, and adopts the scalable DKG protocol [36] and the BLS threshold signature scheme [12]. We assume the size of each transaction is 250 bytes. f is the fraction of malicious users in the system and δ is the failure probability.

Time Overhead Evaluations: We first measure the average time overheads of the DKG protocol [36] under various f and δ, and show the results in Fig. 4(a), where the end-to-end time is the sum of the sharing and reconstruction phase times, and does not account for network delays. As f increases and δ decreases, the DKG end-to-end time increases, this is because more committee members need to be selected to run the DKG protocol. However, as shown in Fig. 4(a), the DKG end-to-end time remains tens of second under different parameters, thus we conclude that the execution of the DKG protocol only adds a small time overhead to Ge-Co.

Then for other critical steps of Ge-Co, we also measure the time overheads as shown in Table III, where we fix $f = 0.3$ and $\delta = 10^{-9}$. In the implementation, all the honest members of each committee always generate votes for the correct messages at the same time, while the malicious committee members reject to vote. Each sidechain participant collects and verifies the votes, then generates the certificates; and then each mainchain participant verifies the received certificates. First, we sample 50 honest committee members and 50 sidechain participants in each epoch, then compute the average time of generating and



(a) DKG end-to-end time of Ge-Co. (b) Certificate size at different f. (c) Certificate size at different n.

Fig. 4. Performance of the protocols, where f is the fraction of malicious users, δ is the failure probability and n is the sidechain length.

verifying a vote during sequential 5 epochs, which take around 8 ms and 1 ms respectively, where the total number of collected votes of each epoch is 1,400. Second, we sample 50 participants respectively from the sidechain and mainchain in each epoch, then compute the average time of a certificate generation right after collecting sufficient (equal to the voting bound) votes, and a certificate verification in sequential 5 epochs, which are about 652 ms and 103 ms respectively. Notice that the voting bound for forming a certificate is 601 ($2000 \cdot 0.3 + 1$), and we evaluate the time cost to form a certification by computing the average time to aggregate 601 votes. If we increase any of the followings: committee size, the the faction of stake, or computational power of malicious users, the voting bound will become larger, so that the average time of a certificate generation increases linearly; but the average time of the steps including vote generation and verification, and certificate verification will remain almost unchanged.

Certificate (or Proof) Size Evaluations: When transferring assets from the sidechain to mainchain, the certificate size determines the communication and storage costs. Thus we compare Ge-Co with other related works in certificate size. We fix the committee size at $S = 2000$ for each epoch, such that the fraction of honest members in each committee is $\geq h$ except with negligible probability $\delta = 10^{-9}$.

To compare Ge-Co with PoS-based sidechains [8], we implement them respectively by simulating Cardano [26] in the mainchain and sidechain. We compare the average certificate size under various f. In our implementation, we assume the honest members of each committee generate votes for the correct messages while the malicious committee members fail to vote. The results in Fig. 4(b) show that the certificate size of PoS-based sidechains [8] is linear with f. This is because in the sidechains, the public keys of committee members who fail to vote and the corresponding Merkle-proofs are included in certificates. This is a compulsory requirement in [8] so that the mainchain users can verify the validity of certificates. With the increase of the number of malicious committee members (corresponding to a bigger f), the certificate size will definitely become larger. On the contrary, the certificate size of Ge-Co remains constant and only takes 0.1 KB. This is an extremely small cost on communication and storage. The main reason is that Ge-CO is designed based on TSS and is able to use a unique main public key to verify the certificates, so that the certificates do not need to contain any public keys or Merkle-proofs of members.

To make a fair comparison between Ge-Co and PoW-based sidechains [6], we leverage Ethereum [2] to simulate the mainchain and sidechain, We are going to show that how the average certificate size performs when varying the factor n which is the sidechain length. In the implementation, we set that all blocks have the same target difficulty. We also assume the faction of computational power of honest users is 0.7, and the security parameter $\lambda = 50$ in PoW-based sidechains [6]. The results in Fig. 4(c) clearly show that the certificate size of Ge-Co is much smaller than that of PoW-based sidechains [6]. The certificate size of [6] is sublinear with n, while the certificate size of Ge-Co remains constant and the performance does not affect by the chain growth. We state that compared with other sidechain constructions with linear proof complexity, such as Drivechains [7], BTCRelay [19], our design is able to provide noticeable improvement. For example, when the chain length $n \approx 1.5 \times 10^7$, the certificate size of Drivechains is about 7.2 GB, which is roughly $7 \cdot 10^7 \times$ larger than the certificate size of Ge-Co.

By the above experimental results, we conclude that Ge-Co not only has small time cost but also achieves much smaller certificate size, demonstrating Ge-Co is efficient and practical.

VII. Conclusion

Sidechains are a key mechanism to improve blockchain interoperability and scalability. In this work, to achieve the transfer of assets between blockchains, we first present Ge-Co, a sidechain construction in the semi-adaptive corruption model. Ge-Co is generic without relying on specific consensus mechanisms. Based on the voting committee selection and threshold signature schemes, Ge-Co achieves optimally succinct constant proof size, reducing the communication and storage costs. To provide stronger security, we further present PoS-Co, a PoS sidechain construction in the fully-adaptive corruption model. PoS-Co is based on the proposed anonymous committee selection approach, and preserves optimally succinct proof. We also develop a PoC implementation of Ge-Co, and the evaluation results show that our construction is efficient and practical. One of our further works could be to design a more generic

sidechain construction in the fully-adaptive corruption model, while preserving optimally succinct and constant proof size.

REFERENCES

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008. [Online]. Avaliable: http://bitcoin.org/bitcoin.pdf

[2] V. Buterin, "Ethereum's white paper," 2014. [Online]. Avaliable: https://github.com/ethereum/wiki/wiki/White-Paper

[3] A. Back et al., "Enabling blockchain innovations with pegged sidechains," 2014. [Online]. Avaliable: http://www.opensciencereview.com/papers/123/enablingblockchain-innovations-with-pegged-sidechains

[4] A. Kiayias, A. Miller, and D. Zindros, "Non-interactive proofs of proofs of work," in *Proc. Financial Cryptogr. Data Secur.-24th Int. Conf.*, Cham:Springer, 2020, pp. 505–522.

[5] A. Zamyatin et al., "A wild velvet fork appears! inclusive blockchain protocol changes in practice," in *Proc. Financial Cryptogr. Data Secur.-23th Int. Conf.*, Heidelberg, Berlin, Springer, 2019, pp. 31–42.

[6] A. Kiayias and D. Zindros, "Proof-of-work sidechains," in *Proc. Financial Cryptogr. Data Secur.-23th Int. Conf.*, 2019, pp. 21–34.

[7] S. Lerner, "Drivechains, sidechains and hybrid 2-way peg designs," 2016. [Online]. Avaliable: https://docs.rsk.co/Drivechains_Sidechains_and_Hybrid_2-way_peg_Designs_R9.pdf

[8] P. Gazi, A. Kiayias, and D. Zindros, "Proof-of-stake sidechains," in *Proc. IEEE 40th Int. Symp. Secur. Privacy*, 2019, pp. 139–156.

[9] B. David, P. Gazi, A. Kiayias, and A. Russell, "Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain," in *Proc. Annu. Int. Conf. Theory Appl. Cryptogr. Techn.*, 2018, pp. 66–98.

[10] M. Rabin, S. Vadhan, and S. Micali, "Verifiable random functions," in *Proc. IEEE 40th Int. Symp. Found. Comput. Sci.*, 1999, pp. 120–130.

[11] J. R. Douceur, "The sybil attack," in *Proc. Int. Workshop Peer-to-Peer Syst.*, 2002, pp. 251–260.

[12] A. Boldyreva, "Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme," in *Proc. Int. Conf. Public Key Cryptogr.*, 2003, pp. 31–46.

[13] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems," in *Proc. Int. Conf. Theory Appl. Cryptogr. Techn.*, 1999, pp. 295–310.

[14] E. K. Kogias, D. Malkhi, and A. Spoegelman, "Asynchronous distributed key generation for computationally-secure randomness, consensus, and threshold signatures," in *Proc 27th ACM. Conf. Comput. Commun Secur.*, 2020, pp. 1751–1767.

[15] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *Proc. Financial Cryptogr. Data Secur.-18th Int. Conf.*, 2014, pp. 436–454.

[16] F. Benhamouda, "Can a public blockchain keep a secret," in *Proc. ACM. Symp. Theory Cryptogr. Conf.*, 2020, pp. 260–290.

[17] J. Bonneau, I. Meckler, V. Rao, and E. Shapiro, "Coda: Decentralized cryptocurrency at scale" 2020. [Online]. Avaliable: https://eprint.iacr.org/2020/352

[18] A. Garoffolo, D. Kaidalov, and R. Oliynykov, "Zendoo: A zk-SNARK verifiable cross-chain transfer protocol enabling decoupled and decentralized sidechains," in *Proc. Int. Conf. Distrib. Comput. Syst.*, 2020, pp. 1257–1262.

[19] ConsenSys, "BTC relay's documentation," 2016. [Online]. Avaliable: http://btc-relay.readthedocs.io/en/latest/

[20] J. Kwon and E. Buchman, "Cosmos: A network of distributed ledgers," 2016. [Online]. Avaliable: https://cosmos.network

[21] S. D. Lerner, "Rootstock: Bitcoin powered smart contracts," 2015. [Online]. Avaliable: https://docs.rsk.co/RSKWhitePaper-Overview.pdf

[22] G. Wood, "Polkadot: Vision for a heterogeneous multi-chain framework," 2016. [Online]. Avaliable: http://polkadot.network

[23] J. Poon and V. Buterin, "Plasma: Scalable autonomous smart constracts," 2017. [Online]. Avaliable: http://plasma.io/

[24] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *Int. J. Inf. Secur.*, vol. 1, pp. 36–63, 2001.

[25] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *Proc. Annu. Int. Cryptol. Conf.*, 2017, pp. 357–388.

[26] Cardano, 2017. [Online]. Avaliable: https://cardano.org/

[27] Cardano SL, 2018. [Online]. Avaliable: https://www.worldlink.com.cn/en/osdir/cardano-sl.html

[28] Ethstats, 2022. [Online]. Avaliable: https://cn.etherscan.com/

[29] J. A. Garay, A. Kiayias, and N. Leonardos, "The bitcoin backbone protocol: Analysis and applications," in *Proc. Annu. Int. Conf. Theory Appl. Cryptogr. Techn.*, 2015, pp. 281–310.

[30] S. Dziembowski, S. Faust, V. Kolmogorov, and K. Pietrzak, "Proofs of space," 2013. [Online]. Avaliable: https://eprint.iacr.org/2013/796.pdf

[31] X. Feng, J. Ma, Y. Miao, X. Liu, and K.-K. R. Choo, "Social characteristic-based propagation-efficient PBFT protocol to broadcast in unstructured overlay networks," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 6, pp. 3621–3639, Nov. 2022.

[32] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults," in *Proc. 26th Annu. Symp. Found. Comput. Sci.*, 1985, pp. 383–395.

[33] P. Hossein, E. Taraneh, and T. Rahim, "An efficient lattice-based threshold signature scheme using multi-stage secret sharing," *IET Inf. Secur.*, vol. 15, pp. 98–106, Dec. 2020.

[34] T. P. Pedersen, "A threshold cryptosystem without a trusted party," in *Proc. Workshop Theory Appl. Cryptogr. Techn.*, 1991, pp. 8–11.

[35] S. Das, Z. Xiang, and L. Ren, "Asynchronous data dissemination and its applications," 2021. [Online]. Avaliable: https://eprint.iacr.org/2021/777.pdf

[36] A. Tomescu et al., "Towards scalable threshold cryptosystems," in *Proc. IEEE 41th Int. Symp. Secur. Privacy*, 2020, pp. 877–893.

[37] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval, "Key-privacy in public-key encryption," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2001, pp. 566–582.

[38] A. Zamyatin et al., "SoK: Communication across distributed ledgers," in *Proc. Financial Cryptogr. Data Secur.-25th Int. Conf.*, 2021, pp. 3–36.

[39] L. Yin, J. Xu, and Q. Tang, "Sidechains with fast cross-chain transfers," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 6, pp. 3925–3940, Nov./Dec. 2022.

[40] P. Schindler, A. Judmayer, N. Stifter, and E. Weippl, "ETHDKG: Distributed key generation with ethereum smart contracts," 2019. [Online]. Avaliable: https://eprint.iacr.org/2019/985

[41] E. Shi, "Foundations of distributed consensus and blockchains," 2020. [Online]. Avaliable: https://elaineshi.com/docs/blockchain-book.pdf

[42] R. Pass and E. Shi, "FruitChains: A fair blockchain," in *Proc. 36th ACM. Symp. Princ. Distrib. Comput.*, 2017, pp. 315–324.

[43] P. Daian, R. Pass, and E. Shi, "Snow white: Robustly reconfigurable consensus and applications to provably secure proof of stake," in *Proc. Financial Cryptogr. Data Secur.-23th Int. Conf.*, 2019, pp. 23–41.

[44] A. Miller, Y. Xia, K. Croman, E. Shi, and D. Song, "The honey badger of BFT protocols," in *Proc. 23th ACM. Conf. Comput. Commun Secur.*, 2016, pp. 31–43.

[45] B. Guo, Z. Lu, Q. Tang, J. Xu, and Z. Zhang, "Dumbo: Faster asynchronous BFT protocols," in *Proc. 27th ACM. Conf. Comput. Commun Secur.*, 2020, pp. 803–818.

[46] Y. Lu, Z. Lu, Q. Tang, and G. Wang, "Dumbo-MVBA: Optimal multi-valued validated asynchronous Byzantine agreement," in *Proc. 39th ACM. Symp. Princ. Distrib. Comput.*, 2020, pp. 129–138.

[47] Y. Gilad, R. Hemo, S. M. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling byzantine agreements for cryptocurrencies," in *Proc. 26th Symp. Operating Syst. Princ.*, 2017, pp. 51–68.

[48] D. Reijsbergen, P. Szalachowski, J. Ke, Z. Li, and J. Zhou, "LaKSA: A probabilistic proof-of-stake protocol," in *Proc. 28th Annu. Netw. Distrib. Syst. Secur. Symp*, 2021, https://www.ndss-symposium.org/ndss-paper/laksa-a-probabilistic-proof-of-stake-protocol/

[49] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst.*, vol. 20, no. 4, pp. 398–461, 2002.

[50] X. Li, J. Xu, L. Yin, Y. Lu, Q. Tang, and Z. Zhang, "Escaping from consensus: Instantly redactable blockchain protocols in permissionless setting," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 5, pp. 3699–3715, Sept./Oct. 2023, doi: 10.1109/TDSC.2022.3212601.

Lingyuan Yin received the PhD degree in software engineering from the Institute of Software, Chinese Academy of Sciences and University of Chinese Academy of Sciences, in 2022. She is currently working with the Institute of Industrial Internet and Internet of Things, China Academy of Information and Communications Technology. Her research interests include applied and theoretical cryptography and blockchain technology.

Jing Xu received the PhD degree in computer theory from the Academy of Mathematics and Systems Science, Chinese Academy of Sciences, in 2002. She is currently a research professor with the Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences. Her research interests include applied cryptography and security protocol. She is a senior member of the Chinese Association for Cryptologic Research.

Kaitai Liang (Member, IEEE) received the PhD degree from the City University of Hong Kong. He joined Delft University of Technology, in 2020. His focus is on the design and implementation of cryptographic protocols to security. He has led European funded projects and published a series of research works, in many high tier journals and conferences, e.g., ESORICS (best research paper award), USENIX Security. He has contributed to ISO standards as an NEN member. He also serves as associate editor for well-reputed international journals.

Zhenfeng Zhang received the PhD degree from the Academy of Mathematics and Systems Science, Chinese Academy of Sciences, in 2001. He is currently a research professor and director of Laboratory of Trusted Computing and Information Assurance, Institute of Software, Chinese Academy of Sciences. His current interests are blockchain, cryptocurrency, cryptographic protocol, public key cryptography and trusted network computing research.