Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2022) 30: 344 – 360

doi:10.3906/elk-2105-183

Privacy in blockchain systems

Murat OSMANOĞLU[1,*], **Ali Aydın SELÇUK**[2]
[1]Department of Computer Engineering, Ankara University, Ankara, Turkey,
[2]Department of Computer Engineering, TOBB University of Economics and Technology, Ankara, Turkey

Received: 18.05.2021 • **Accepted/Published Online:** 17.01.2022 • **Final Version:** 04.02.2022

Abstract: Privacy of blockchains has been a matter of discussion since the inception of Bitcoin. Various techniques with a varying degree of privacy protection and complexity have been proposed over the past decade. In this survey, we present a systematic analysis of these proposals in four categories: (i) identity, (ii) transaction, (iii) consensus, and (iv) smart contract privacy. Each of these categories have privacy requirements of its own, and various solutions have been proposed to meet these requirements. Almost every technique in the literature of privacy enhancing technologies have been applied to blockchains: mix networks, zero-knowledge proofs, blind signatures, ring signatures, secure MPC, homomorphic encryption, to name just a few. We analyze each category separately in the paper. We first define the related privacy issues, and then review the proposed solutions. The limitations of each solution and the attacks discovered are also discussed along with the proposals. For each category, we first define the relevant privacy issues, and then review the proposed solutions along with their features and limitations.

Key words: Blockchain, bitcoin, cryptocurrency, privacy, anonymity

1. Introduction

The blockchain technology entered our lives with the Bitcoin cryptocurrency [56], and has been regarded as a novel technology that enables running a trusted system without a trusted authority. Bitcoin was the next step in a long line of cryptocurrency experiments that started with Digicash [25]. Digicash and the following cryptocurrency proposals were proposed as a libertarian way of doing electronic finance where the big brother could not trace transactions to users, unlike credit cards or common electronic banking systems. Bitcoin succeeded where the previous attempts failed and established itself as much more than an experiment.

The privacy provided by Bitcoin comes from its use of pseudonyms: A user does not use his real-world identity, hence his privacy is protected in a basic sense. However, repeated incidents [23, 39] and analyses [3, 12, 53] have shown that this level of protection is not sufficient, and it can be possible to deanonymize a transaction and to reveal the identity behind a pseudonym, hence deanonymizing its all other transactions.

As Bitcoin established itself as a real-world payment method, its privacy level came to be seen as inadequate, and several privacy enhancements have been proposed. Various cryptographic techniques have been utilized for this purpose, such as mixnets [24], ring signatures [60], and zero knowledge proofs [37].

After Bitcoin's success, blockchain came to be considered as a general solution to run a decentralized trusted system. More advanced blockchain systems like Ethereum [22] and Hyperledger Fabric [43] have been

*Correspondence: Correspondence: mosmanoglu@ankara.edu.tr

developed with extra capabilities such as smart contracts. With these developments, new privacy issues have emerged, such as privacy of smart contracts and privacy of consensus protocols.

In this paper, we give a survey of the most significant privacy-enhancing solutions proposed for blockchain privacy. A number of notable studies have been published with a similar theme in the past [1, 36, 46]. Our survey is distinguished from these earlier studies in terms of its scope: We cover various aspects of privacy in blockchains, whereas the earlier surveys mostly concentrated on identity privacy.

2. Blockchain architecture

Blockchain is an enabling technology that allows a network of users to maintain a distributed ledger. Each user in the network has a copy of the ledger that consists of an ordered list of transactions. Users periodically combine the transactions shared in the network into new blocks, and append these blocks to their copy using a cryptographic hash function which enables users to keep an immutable history of records. With respect to permission given to nodes for certain actions, blockchains can be classified under two groups: permissionless and permissioned blockchains.

Permissionless blockchains allow anyone in the network to access all transactions written on the ledger, to create transactions, and to take part in the consensus mechanisms. Also, permission is not required to register in the system. In this setting, users are represented by pseudonymous addresses in the network, which provide privacy to users up to a certain level. In permissioned blockchains, on the other hand, users should get permission to register in the network by providing a valid credential to a particular authority, called registration authority. Permissioned blockchains also restrict specific actions to certain users in the network. Since users possess well-established identities to participate in the network, reaching consensus on the current state of the ledger will be easier in a permissioned settings.

A blockchain system can have four key concepts associable with privacy:

- **Nodes (or identities)** in the blockchain terminology refer to users maintaining a copy of the distributed ledger. They communicate with each other (send or receive transactions) over a peer-to-peer network through a gossip protocol. Each node is associated with a public key–signing key pair of a digital signature scheme employed to ensure the authenticity of transactions. Besides, each node in blockchains has access to all the transactions propagated in the network.

- **Transactions** are just requests created by nodes to update states in a public ledger. There are some particular nodes, called miners, that reflect these updates to the ledger by including the corresponding transactions in the newly created blocks. The new blocks are appended to the ledger through consensus mechanisms which allow nodes in the blockchain network to reach a common agreement on the current state of the ledger. Note that each transaction is signed by its sender before being propagated in the network to prove its origin to the receivers.

- **Consensus** is a mechanism executed among nodes in the blockchain network to achieve an agreement on the current state of the ledger. A regular blockchain network assumes that some individual nodes may fail while executing the consensus mechanism, or act maliciously to prevent other nodes from correctly updating their ledger. In the presence of such failures, consensus mechanisms should guarantee that all the nodes in the network add the same transactions to their ledger in the same order.

- **Smart contracts** are a piece of codes that enable trusted execution of a contract without relying on a

trusted third party. After being deployed to the network, they are implemented independently by each node on the inputs contained in a trigger transaction pointing to them.

In the following sections, we discuss the privacy requirements and proposed solutions for these four concepts. We also review the attacks and privacy threats along with the proposed solutions rather than in separate sections.

3. Identity privacy

Each user in blockchain has access to all transactions shared in the network. This aspect of blockchain enables users to validate the integrity and authenticity of every transaction. However, it also brings about some privacy challenges: It exposes the history of all the transactions associated with a particular ID to anyone in the network. Bitcoin, for instance, uses pseudonymous addresses as the identifiers for users to unlink the connection between the addresses and the real identities. Moreover, it allows users to create a fresh address for each transaction to strengthen their privacy. By doing so, it achieves to ensure the privacy of the identities up to a certain degree. However, it is still vulnerable to some deanonymization attacks [3, 12, 53]. In this section, we review some cryptographic techniques applied to mitigate such attacks and enhance the identity privacy in blockchain-based systems. Although this survey is mostly focused on cryptocurrency applications, we would like to note that many interesting solutions exist in other application areas of blockchain as well (e.g., [11, 69]).

3.1. Mixing-based solutions

Mixing has been one of the oldest and best-known privacy enhancing technologies. It was first proposed by Chaum [24] for email privacy. Since then it has been adopted to various systems, such as Tor [31].

Mixing has been considered to enhance the privacy of Bitcoin transactions since Bitcoin's early years. The idea is simple: A trusted party, the mixer, receives transactions from senders, shuffles them randomly, and sends them to receivers. Of course, since all these transactions will be in public view, the transaction amounts must be identical in order to be indistinguishable.

Although it is simple and functional, trusting a mixer for this task has several problems. First of all, the mixer must be trusted not to steal the money it receives. Second, it knows the sender and receiver of every transaction and must be trusted not to disclose or abuse this information. Despite these serious limitations, several such Bitcoin tumbling services operated, such as bitmixer.io and Helix Light.

Providing a more secure mixing service for Bitcoin and alternative coins has been an active area of research. Several projects have been proposed to make the basic centralized tumbling service secure. Furthermore, improved systems have been developed to provide decentralized mixing services.

Centralized mixing solutions: One of the earliest proposals that aims to make Bitcoin tumbling more secure and the mixers accountable is Mixcoin [17]. The main feature of Mixcoin is that, when Alice trusts a mixer with a transaction, she receives a signed warrant from the mixer, which roughly states that if the mixer receives a x BTC from Alice by time t, it will send x' BTC to Bob by time t'. So, if the mixer steals Alice's money, she will publish this warrant and ruin the mixer's reputation. Although Mixcoin provides some sort of solution against theft, it provides no solution for the second major problem, namely linkability by the mixer.

An improvement over Mixcoin is the Blindcoin proposal [67]. It essentially adds blinding to Mixcoin. Alice obtains the warrant from the mixer using blind signatures, hence the mixer does not see the receiver's identity. Later, Alice removes the blinding and publishes the warrant on a public log. As for the stealing problem, Blindcoin is the same as Mixcoin: A mixer can steal, but at the expense of ruining its reputation.

An original solution for centralized coin mixing that truly prevents coin theft is the CoinSwap protocol proposed by Maxwell [52]. It is based on the idea of timed 2-of-2 escrow transactions: For Alice to transfer money to Bob through a mix, she issues an initial transaction, which puts the money to be transferred in escrow at the mixer for Bob to claim. Then Bob issues the release transaction within the validity period of the escrow. Only Bob can take the money with Alice's approval, hence this solves the theft problem. But the mixer is able to link Alice and Bob, as in Mixcoin.

The idea of private coin mixing by escrow was carried further by TumbleBit [41]. TumbleBit aims to provide utmost security through a combination of cryptographic techniques such as secure two-party computation, zero knowledge proofs, blinding, and fair exchange protocols. By these techniques, TumbleBit enables exchanging Alice's initial message and Bob's release message without the mixer's being able to link the two. Of course, for this scheme to be secure, there must be many other pairs besides Alice and Bob participating at the same time. Therefore, the protocol is executed at certain time intervals called epochs, and the participants' messages are synchronized to prevent any timing analysis.

Decentralized mixing solutions: An alternative approach to mixer-based protocols is the decentralized mixing protocols, where a group of senders are organized among themselves to combine their transactions.

CoinJoin, proposed by Maxwell [51], is the best known protocol in this category, and, unlike most other protocols discussed in this section, it has been implemented and used in practice by several different systems. The idea is simple yet effective: as shown in Figure 1, a group of users to send money privately come together off-chain and create a combined transaction. Each user checks whether the transaction contains his output address and signs it if it does. This simple protocol is remarkable in that it solves the theft problem. However, the details of every transaction is open to the participants, hence there is no unlinkability provided against insiders. A second limitation is that the anonymity provided is limited by the group size, which in turn is limited by the number of signatures a Bitcoin transaction can accommodate.

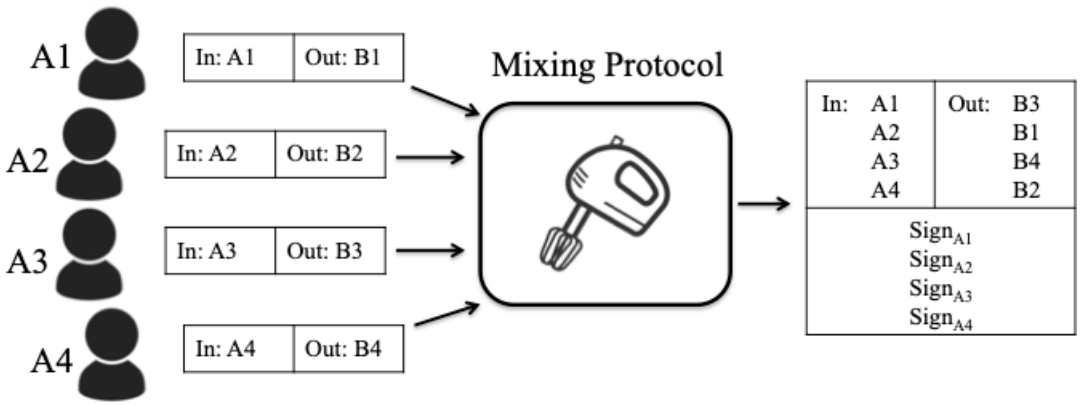

Figure 1. CoinJoin protocol.

A protocol that solves the linkability problem of CoinJoin is CoinShuffle [61], which brings in the idea of layered encryption from such constructions as mixnets [24] and Tor [31]. As shown in Figure 2, participants share their payment information among themselves with multiple encryptions, which are shuffled at every step. To see how CoinShuffle works, assume A_1, A_2, A_3, and A_4 will send money to B_1, B_2, B_3, and B_4, respectively. Let $E_X(.)$ denote encryption by the public key of user X. Then, A_1 prepares the transaction packet as

$E_{A_2}(E_{A_3}(E_{A_4}(B_1)))$ and sends it to A_2. A_2 decrypts it, prepares his own packet $E_{A_3}(E_{A_4}(B_2))$, shuffles it with $E_{A_3}(E_{A_4}(B_1))$ randomly, and sends both to A_3. A_3 decrypts them, prepares his own packet $E_{A_4}(B_3)$ shuffles it with $E_{A_4}(B_1)$ and $E_{A_4}(B_2)$, and sends them to A_4. A_4 decrypts the received transactions, prepares the combined transaction, and puts it on the blockchain. Each sender controls his part and signs the transaction if it is right.

An improved version of CoinShuffle, named CoinShuffle++, has been proposed by Ruffing et al. [62]. It utilizes P2P mixing techniques to provide communication anonymity. Ruffing et al. developed the DiceMix protocol for this purpose and built CoinShuffle++ on top of DiceMix. CoinShuffle++ is significantly more efficient than CoinShuffle.

Anonymity provided by CoinJoin and CoinShuffle is limited by the maximum number of signatures that can fit into a single transaction. CoinParty [72] solves this limitation by secure multiparty computation (MPC): Participants set up a shared address through an MPC protocol, and form a joint transaction. The joint transaction is signed by the shared address, again by MPC. CoinParty provides unlinkability, but its theft protection is conditional: Due to MPC, at least 2/3 of the participants must be honest for protocol correctness.

Another remarkable solution in this category is Xim [13]: It provides 2-party mixing utilizing a multiround fair exchange protocol. Alice announces her interest in a private transaction on the blockchain. An interested party, say Bob, responds to this announcement and contacts her anonymously off-chain. The two perform a transaction using a fair exchange protocol. One advantage of Xim over other decentralized mixing protocols is that the anonymity set is all the users using the service at that time, similar to that in the centralized mixing protocols. Table 1 provides a comparison of the mixing-based methods discussed in this section.

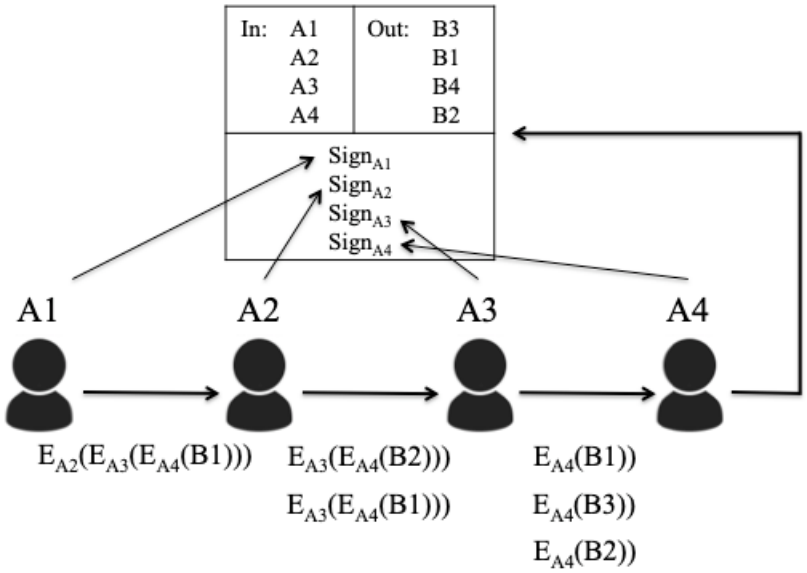

Figure 2. CoinShuffle protocol.

3.2. Ring signatures

Ring signatures, first introduced by [60], allow a user to create a signature on a message by designating a ring of possible signers including himself without disclosing which member of that ring actually signed the message.

Table 1. Comparison of mixing-based methods.

Protocol	Unlinkability	Theft prevention	Size of mix set
Mixcoin [17]	No	Accountable	Large
Blindcoin [67]	No	Accountable	Large
TumbleBit [41]	Yes	Yes	Large
CoinJoin [51]	No	Yes	Small
CoinShuffle++ [62]	Yes	Yes	Small
CoinParty [72]	Yes	2/3 honest majority	Large
Xim [13]	Yes	Yes	Large

Ring signatures do not require a central authority as in group signatures [26] to specify the group members, and do not demand any coordination between users during the generation of their public keys. Besides, users do not need to communicate with each other while creating signatures. Furthermore, users may be unaware that their public keys are included in a signature created by a signer they do not know.

In contrast to group signatures, ring signatures do not possess any authority having power to extract the identity of the signer when necessary. This feature makes ring signatures a very attractive tool for the applications demanding strong privacy for the users. However, it also makes it vulnerable to malicious users to exploit this excessive anonymity, especially in some applications such as elections, i.e. the signer may create two signatures, that can be counted as two votes without being noticed. As an extension of ring signatures, Fujisaki et al. [32] introduced traceable ring signatures that enable users to trace such cases.

CryptoNote: Van Saberhagen [68] adopted a modified version of traceable ring signatures to get an effective protocol, called CryptoNote, that achieves to conceal the addresses of both sender and receiver of a transaction from other users. To hide the receiver's address, CryptoNote creates a one-time verification-signing key pair for each transaction directing the receiver If the receiver wants to claim one such transaction, he first specifies n transaction outputs having the same amount, then combines all of them into a ring signature using the corresponding one-time key. Mixing each transaction with the other specified transaction outputs enables the receiver to hide the sender's address. Besides, the proposed ring signature in the CryptoNote protocol does not allow the receiver to use the same transaction output in two different transactions without being detected.

Although CryptoNote conceals the identities of both the sender and the receiver of a transaction, it also presents some deficiencies that make the protocol vulnerable to analysis attacks. Kumar et al. [49] carried out a forensic analysis on Monero, which is the most popular cryptocurrency built on top of the CryptoNote protocol, to examine the untraceability characteristic of the protocol. They observed that 93% of all output amounts appear only once in the network. Since the protocol requires users to combine a number of transaction outputs having the same amount to form a transaction, they cannot be combined with any other transaction output. Besides, the analysis in [49] showed that users prefer to use a small number of transaction outputs per transaction, even if they could have used more to mix, to avoid a higher transaction fee. An adversary may easily exploit this fact to break the untraceability of the protocol.

To remedy Monero's deficiencies, Noether et al. [58] proposed an extension of CryptoNote protocol, called ring confidential transaction (RingCT), that masks the amount in the transactions by signing a commitment of the amount instead of the amount itself. Since the amounts in a transaction are hidden, the protocol just verifies whether the total amount of input coins is equal to the total amount of output coins in the commitments for

the validation of a given transaction. However, a malicious user may produce a transaction having a negative amount which turns into a very large positive amount in the corresponding modulo. The protocol in [58] avoids such cases by employing a range proof as part of the transaction that requires users to prove the output amount of a given transaction lies in a predetermined range. Note that the protocol in [58] achieves the confidentiality of transactions by producing a ring signature with a vector of signing keys rather than a single key. Thus, the size of signature in the protocol becomes $O(m(n+1))$ where m is the size of each vector and n is the number of members in the corresponding ring. Sun et al. [65] introduced a new RingCT protocol that remarkably reduces the size of signatures, i.e. the signature size in the protocol is $O(m)$, independent of the ring size. Later, Yuen et al. [70] proposed an efficient RingCT that eliminates the trusted setup assumption required in the protocol [65]. Table 2 compares the ring signature based methods discussed in this section.

Table 2. Comparison of ring signature based methods. For the asymptotic analysis, n and m indicate the ring size and the number of transaction input, respectively.

Protocol	Requiring a trusted setup	Hiding both sender's and receiver's addresses	Hiding the amount of transaction	Transaction size
CryptoNote [68]	No	Yes	No	$O(n)$
RingCT[58]	No	Yes	Yes	$O(m(n+1))$
RingCT 2.0 [65]	Yes	Yes	Yes	$O(m)$
RingCT 3.0 [70]	No	Yes	Yes	$O(m+\log n)$

3.3. Zero-knowledge proofs

A zero-knowledge proof [37] is a protocol that allows one party (prover) to convince another party (verifier) that a statement is true without revealing any information other than this fact. A zero-knowledge proof should satisfy the following properties: completeness (if both the prover and the verifier honestly follow the protocol, the prover will eventually convince the verifier with high probability), soundness (a cheating prover cannot convince an honest verifier about a false statement except for a small probability), and zero-knowledge (whatever the verifier can learn with executing the protocol can be simulated without going through the protocol).

A regular zero-knowledge protocol requires interaction between the prover and the verifier. But it is hard to realize in a blockchain since the proof must be publicly validated. Fortunately, there is a variant of zero-knowledge proofs, called noninteractive zero-knowledge proof [15], that does not require the prover and the verifier to interact with each other to execute the protocol. However, a noninteractive zero knowledge proof entails some setup requirements such as a random oracle or a common reference string shared between the prover and the verifier. On the other hand, zero-knowledge succinct noninteractive argument of knowledge (zk-SNARK), first introduced by [14] as an efficient family of NIZK proofs, enjoys succinct proofs that enable verifiers to efficiently validate the proofs.

Zerocoin: Miers et al. [54] introduced a distributed electronic cash system, called Zerocoin, that extends the Bitcoin protocol to achieve fully anonymous transactions without incorporating trusted parties. As shown in Figure 3, users first mint a zerocoin in renumeration for a predetermined amount of bitcoins by computing a commitment C to a random serial number S they choose, and announce the commitment C to the blockchain. To claim a zerocoin, users create a transaction having the serial number of that coin together with a noninteractive zero knowledge proof showing that one of the unclaimed zerocoins (commitments) shared

in the blockchain contains the corresponding serial number. The zero-knowledge property of the underlying proof prevents one to link the claimed transaction to a specific address. Despite Zerocoin utilizing a one-way accumulator based on the strong RSA assumption to minimize the proof size, it still struggles with the large proof size, which makes it hard to realize Zerocoin in practice.

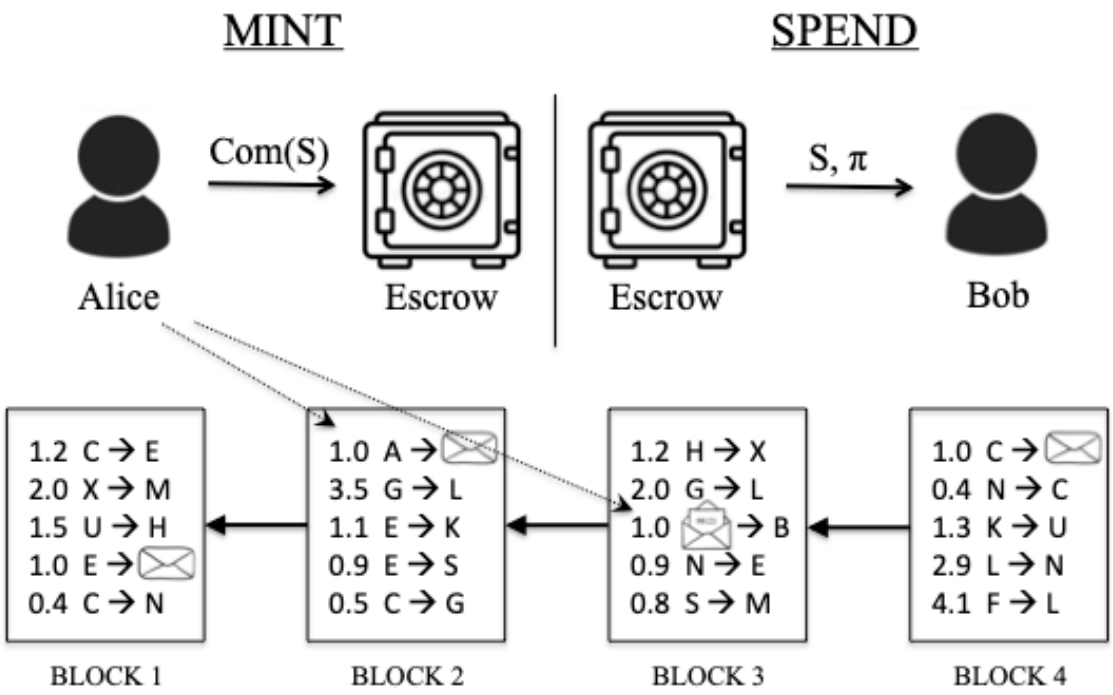

Figure 3. Zerocoin protocol.

To alleviate this issue, Garman et al. [35] proposed an extension of Zerocoin that reduces the proof size by relaxing some assumptions. Besides, the extension [35] removes the random oracle assumption. The protocol achieves the zero-knowledge property in the common reference string model where a trusted setup process should be executed to generate public parameters for the underlying noninteractive zero-knowledge proof. Androulaki et al. [3] demonstrated that some user spending patterns such as transaction amounts and address balances can be exploited to recover user profiles. Within this direction, Androulaki et al. [4] introduced an effective protocol for Zerocoin that hides the transaction amounts and the address balances. Unlike other Zerocoin variants, the protocol also enables spending zerocoins directly without converting them back to bitcoins.

Zerocash: Remark that Zerocoin is not a full-fledged distributed cryptocurrency, rather it can be viewed as a mechanism enabling users to clean their bitcoins through zerocoins they mint. On the other hand, Zerocash [8] is a full-fledged decentralized payment scheme that provides strong anonymity guarantees by employing zk-SNARKs. Unlike Zerocoin, Zerocash enables users to create anonymous transactions having arbitrary denominations that hide both the receiver and sender of transactions as well as the amount. Instead of RSA accumulators, Zerocash utilizes Merkle trees, which can be efficiently embedded into zk-SNARK proofs, for the set of unclaimed commitments to improve the performance, i.e. it achieves less than 1 kB transaction size and less than 6 ms verification time at the 128-bit security level. Note that Zerocoin attains just more than 45 kB transaction size and less than 450 ms verification time at the same security level.

Similar to the Zerocoin extension proposed by [35], Zerocash requires some public parameters, generated in the setup phase, to securely operate zero-knowledge proofs. A malicious user holding the randomness used in the generation process may violate the soundness and zero-knowledge properties of the underlying zero-knowledge proof, i.e. he may convince a verifier for a false statement and mint counterfeit coins in Zerocash. Ben-Sasson et al. [9] showed that an MPC computation can be executed to generate the public parameters for zk-SNARKs such that if at least one of the participants is honest, the resulting scheme will guarantee the soundness property. Motivated by this result, Bowe et al. [18] introduced an MPC protocol to generate the public parameters for the zk-SNARKs used in Zerocash. The protocol still provides full anonymity to users even if all the participants who joined the public parameters' generation were malicious. Note that there are two types of transactions in Zcash: shielded transactions that hide the receiver, sender, and the amount of the transaction, and transparent transactions that disclose the pseudonym addresses of the parties involved together with the amount to the network, and using shielded transactions is optional in Zcash. Kappos et al. [44] showed that, although shielded transactions provide full anonymity to users, most of them prefer to deal with transparent transactions since the size of the proof and the verification time incur large overheads for the prover and verifier. Moreover, Zhang et al. [71] remarked that users in Zcash preferring to deal with shielded transactions are mostly miners and founders, which makes Zcash vulnerable to analysis attacks. In Table 3, we compare the zero-knowledge based privacy-preserving techniques covered in this section.

Table 3. Comparison of zero-knowledge based methods. Since [4] did not provide any implementation supporting their improvements, we left the transaction size and the verification time empty for the corresponding row. Also, [35] claimed that their construction halves the verification time of Zerocoin. Similarly, since they did not prove any result to support that, we left the corresponding place empty.

Protocol	Requiring a trusted setup	Hiding both sender's and receiver's addresses	Hiding the amount of transaction	Transaction size	Verification time
Zerocoin [54]	No	Yes	No	25 kB	450 ms
[35]	No	Yes	No	10 kB	*
EZC [4]	No	Yes	Yes	*	*
Zerocash [8]	Yes	Yes	Yes	∼1 kB	∼6 ms

4. Transaction privacy

As emphasized in Section 2, blockchain provides full transparency to users by making all the data shared in the network available to every user. This level of transparency includes each user in verifying all the data shared in the network and writing them to the ledger. However, revealing all the data to every participant of the network yields serious privacy issues, especially in the applications processing sensitive data. Such privacy leakage impels users to be unwilling to share sensitive data, or even to participate in the system. In this section, we review a number of cryptographic primitives utilized to avoid such leakage and enhance the transaction privacy in blockchain-based systems.

Secure multiparty computations: A secure multiparty computation (MPC) protocol enables a set of users to evaluate a function on the input provided by users in a distributed way while guaranteeing the privacy of inputs and the correctness of the output. There are two different types of corruption model viewed in a regular MPC protocol: passive corruption that considers whether an adversary can only get the internal state of the corrupted users, and active corruption that considers whether an adversary can also drive the corrupted parties not to follow the protocol.

Zyskind et al. [73] proposed a decentralized computation platform, called Enigma, that enables users to store data and to perform computations on data without leaking any useful information about data. While data is stored as encrypted in a modified version of the Kademlia distributed hash table protocol that serves as an off-chain network, an external blockchain is employed as a controller layer to manage access control and to keep the tamper-proof log records. Enigma employs the secure MPC protocol given in [7] to process data queries in a distributed way to ensure the privacy of data. The blockchain in the execution of secure multiparty computation serves as the bulletin board required in the protocol [7] to guarantee the correctness.

Benhamouda et al. [10] integrated secure MPC protocol into Hyperledger Fabric, a permissioned blockchain architecture, to allow transactions to be processed while protecting their confidentiality. They remarked that the trust model considered in a permissioned blockchain mimics the one considered in secure MPC protocols. Thus, instead of outsourcing the secure multiparty computation to an off-chain entity, which increases the cost of the implementation, their model implements the MPC protocol as a part of smart contracts in the blockchain network. Data in their model is stored in the blockchain as encrypted, and when queried by a smart contract, users first decrypt it, and give the decryption to the MPC protocol as their input.

Benhamouda et al. [40] incorporated MPC into Hyperledger Fabric to execute the clearing price mechanism of an initial public offering (IPO). In their model, each party employs an additive secret sharing scheme to share his orders $order_i(p)$ for all possible prices with the other parties where $order_i(p)$ denotes the number of shares he is willing to buy at the price p. The parties then execute the MPC protocol on the shares to calculate the clearing price. At the end, the parties locally compute the share allocation (the number of shares each party gets). The protocol guarantees the privacy of the orders in the passive model even with a single honest party.

Searchable encryptions: Searchable encryptions [63] allow a data owner to outsource his database as encrypted to a server that performs some search queries over encrypted data requested by a data user on behalf of the data owner. Most of the searchable encryption schemes require data users to acquire a search token given by the corresponding data owner in order to request search queries. On the other hand, a regular searchable encryption scheme should guarantee that the plain data are protected from both the server and the data user. Besides, an ideal searchable encryption scheme should also hide data access patterns (which data records have been used to generate the results for the search queries).

Hu et al. [42] introduced a decentralized privacy-preserving search scheme that employs symmetric searchable encryption together with smart contracts to guarantee the correctness and confidentiality of data. Data owners use a decentralized storage network such as interplanetary file system to store their data as encrypted, and provide data (decrypted) to smart contracts when needed. To request a query, users first communicate with data owners to acquire the search tokens. Smart contracts, which serve as the cloud server in the scheme, then execute the search queries on data with the tokens and return the results to data users.

Chen et al. [28] proposed a blockchain-based searchable encryption scheme for electronic health records that enables different health agents as data users to perform data queries on the encrypted medical data. Similar to [42], the scheme utilizes a public cloud to store the medical records as encrypted and the blockchain to only keep the search indexes for the data queries. Data users contact data owners to get the trapdoor for the queries they request. They then provide the trapdoor to smart contracts that execute the search queries on the trapdoor data users provide together with the corresponding search index, and return the results to data users.

On the other hand, Tang [66] remarked that directly changing the server with a smart contract may create some undesired consequences, i.e. it may increase the cost of processing search queries, and data access

patterns are revealed to all users having access to blockchain. To alleviate these drawbacks, they gave a generic blockchain-based framework for searchable encryption that employs blockchain only to validate the search results the servers provide, and to manage cash flow among users to guarantee fairness. Search queries in their framework are performed by the server as in the regular searchable encryptions. The generic framework protects the confidentiality of both data and search queries, and hides data access patterns.

Zero-knowledge proofs: Lu et al. [50] introduced the first blockchain-based crowdsourcing system that incorporates a new cryptographic primitive, common-prefix linkable anonymous authentication, and zk-SNARKs, that are compatible with the primitive to provide data confidentiality and anonymity. In the system, a fresh address is created for each crowdsourcing task, and a smart contract is initiated to correctly execute it. Similarly, each worker providing an answer to a task creates a fresh address, and submits his answer from this address as encrypted together with a zk-SNARK proof to the corresponding contract. After obtaining enough submissions, the requester first decrypts them to get the original answers, then calculates the reward each worker deserves. She then prepares a zk-SNARK proof to show the validity of his decision, and announces it to the blockchain. Finally, the corresponding smart contract sends the specified reward to each address. The underlying zk-SNARKs and the use of a fresh address for each task guarantee that the answers cannot be linked to a specific worker, and the submitted answers for a task are only revealed to the requester of the task.

Narula et al. [57] proposed a privacy-preserving distributed ledger, called zkLedger, that enables financial institutions to exchange assets while providing strong transaction privacy and public verifiability. zkLedger can be viewed as a table in which rows represent transactions and columns represent institutions. Each transaction appears in the system as a commitment to a value that indicates the amount of assets received or sent by the corresponding institution. The scheme also attaches a NIZK proof to transactions to validate the commitments. With these techniques, zkLedger achieves to conceal the amounts and the identities contained in transactions as well as transaction graphs. Unlike [50], it does not require a trusted setup.

In addition, Androulaki et al. [2] applies standard NIZK proofs to permissioned blockchains to build a privacy-preserving token management system that runs in UTXO transaction model. The system allows users to issue new tokens and to transfer their tokens to other users. Transactions in the system are represented by commitments to the value and the owner of the corresponding token. When users want to send their tokens to some other user, they create a transfer transaction containing a set of input tokens (as commitments) to be spent and a set of output tokens (as commitments) to be issued. For each input and output token, users also create a NIZK proof to show that the user in the relevant commitment is one of the registered users, and the corresponding token is among the valid tokens contained in the blockchain.

5. Consensus privacy

A relatively new area in blockchain privacy research is the privacy of consensus protocols, where the purpose is to hide the identity of the block producers in order to protect them from DoS or other types of attacks. Consensus privacy is different from identity privacy in the sense that, identity privacy aims to hide the real-life identities of participants whereas consensus privacy aims to hide who the next block producers are until they reveal it themselves. This problem is particularly meaningful for committee-based PoS consensus protocols, where time is divided into epochs, and a set of leaders are in charge of producing blocks in that epoch and also of selecting the leaders for the next epoch. If the leaders' identities are revealed, they may become targets of attacks. A number of different protocols have emerged over the past couple of years with an increasing degree of sophistication and privacy guarantees.

One of the earliest studies on PoS privacy was conducted by Azouvi, McCorry, and Meiklejohn [5]. They proposed the secure leader election protocol Caucus as a part of the consensus protocol Fantomette. They proved that Caucus satisfies three fundamental security properties of a leader election protocol: liveness, unpredictability, and fairness. Furthermore, it guarantees that a leader's identity is revealed only when it is to be used, so that he will not be targeted by DoS or other kinds of attacks. In this respect, it is similar to Algorand [27], where any chosen committee member is replaced as soon as he participates in a consensus.

The single secret leader election (SSLE) proposal of Boneh et al. [16] improves the leader election protocols of Fantomette and Algorand: The election protocols of Fantomette and Algorand return a shortlist of potential leaders, which is resolved by a run-off procedure, whereas the SSLE protocol of Boneh et al. [16] returns a single leader securely. Boneh et al. [16] first stated the security definitions an SSLE scheme must satisfy. Then they gave such a protocol based on indistinguishable obfuscation. They also gave alternative constructions based on fully homomorphic encryption (FHE) and the decisional Diffie-Hellman (DDH) problem.

The protocols we summarized so far are mainly concerned with a secret way of electing a leader or members of a committee. If we recall the two privacy criteria for PoS protocols we mentioned at the beginning of this section, the protocols discussed so far deal with the first criterion only. More recently some more sophisticated protocols have emerged which addressed the second privacy criterion as well, namely keeping the stake information of participants secret.

The first protocol to hide participants' stakes in a PoS system was the Ouroboros Crypsinous protocol [45]. The Ouroboros Crypsinous protocol is constructed from the Ouroboros Genesis protocol in a way to keep the user stakes secret. In Ouroboros Genesis, the information of user stakes is used for verification of slot leaders. Ouroboros Crypsinous replaces this with a Zerocash-like system [8] where each coin can be spent only once, and each coin is separately eligible to be a leader according to a randomly generated value.

A more sophisticated solution to the problem of keeping leader stakes secret is to design an anonymous lottery function. Each user can privately evaluate a selection function to see whether he is the leader for a given tag. If he is, he obtains an anonymous proof for this with no identities attached to it. Ganesh et al. [33] gave the first formal definition of this problem and proved its feasibility. Furthermore, they gave constructions based on some well-known PoS consensus protocols such as Ouroboros Praos [30] and Algorand [27].

A second solution along the same lines as Ganesh et al. [33] was presented by Baldimtsi et al. [6]. Their system resembles to that of Ganesh et al. [33] in terms of basic functionality, but it is more advanced in certain aspects: First, its proof is more flexible for composability with other protocols and, second, its operation does not require an anonymous broadcast channel, making it more feasible to realize in practice. In a related work, Kohlweiss et al. [47] demonstrated that even an ideal broadcast channel may not be sufficient for this purpose and an attacker can use network delays to distinguish stakeholders.

6. Smart contract privacy

A smart contract is a piece of code that enables trusted execution of a contract without relying on a trusted third party. Although smart contracts promise remarkable advantages such as self-execution and removing trusted third parties, they bear the privacy concerns a blockchain presents: All the transactions related to a smart contract are visible to all the nodes in the blockchain. This public nature of smart contracts introduces privacy issues and hinders their adoption for a wider range of applications.

To resolve this issue, Hawk [48] was introduced as a framework that enables privacy-preserving smart

contracts. Similar to Zerocash [8], a user aiming to perform a confidential transaction first creates a private coin as a commitment to one of his public coins. When requesting execution of a smart contract, the user binds this private coin to the contract by submitting a commitment to the coin's value and the user's input along with a zero-knowledge proof for the correctness of the commitment. The user provides the commitment to the manager encrypted under the manager's public key. The manager then applies off-chain computation to calculate the payout distribution, and creates new private coins to be given to each user. He then submits newly created coins together with zero-knowledge proofs of the correctness to the smart contract. The smart contract finally reallocates the previously bound coins among the corresponding users.

Two instantiations were proposed to realize trusted managers in Hawk: implementing a secure multiparty computation among a number of users, or utilizing a trusted execution environment (TEE) such as the Intel SGX. However, none of them will remove the dependency of the system to the cryptographic heavy tools. To eliminate such dependency, Ekiden [29] considers an architecture in which computations over private data are executed off-chain in TEEs and attestations of correct execution are validated in the blockchain. However, eliminating such dependency comes with a price: Ekiden employs a key manager to periodically issue keys used to encrypt messages shared between the underlying TEE and users. Ekiden also uses a digital signature compatible with the underlying encryption scheme for attestation. Although Ekiden can be applied to a broad range of use cases, it is still vulnerable to some practical attacks [19].

Steffen et al. [64] proposed zkay, a language that identifies the owners of private data through some privacy types. To realize zkay contracts, they are converted into equivalent public contracts in a privacy preserving way that can be executed on public blockchains. The proposed scheme utilizes NIZK proofs together with an encryption scheme to ensure that public contracts protect the privacy of private data and preserve the functionality of the original contracts. However, the zkay language is only applicable to a limited number of private data. On the other hand, Zether [20] was introduced as a decentralized payment mechanism in an account-based model that uses privacy-preserving smart contracts, compatible with Ethereum and similar smart contract platforms, to conceal the account balances and to employ confidential transactions to update the associated balances. Zether employs ElGamal encryption to hide both the balances and transaction values, and a modification of Bulletproofs [21] to prove the correctness of encryptions. Note that ElGamal encryption used is additively homomorphic, which enables users to privately update their balances. Table 4 compares the privacy-enhancing techniques discussed in this section.

Table 4. Comparison of privacy-enhancing techniques analyzed in Section 6.

Protocol	Building Block	Relying on a trusted setup	Range of contract types	Calculations take place	Requiring a trusted manager
Hawk [48]	zk-SNARKs	Yes	Limited	Off-chain	Yes
Ekiden [29]	TEE	No	Broad	Off-chain	Yes
Zkay [64]	zk-SNARKs	Yes	Limited	On-chain	No
Zether [20]	NIZK	No	Limited	On-chain	No

7. Future research directions

Privacy is a demanding feature for a blockchain, where activities are supposed to be in public view. Many novel solutions have been developed for this purpose. However, strong privacy on a blockchain can also be a point of

exploitation for illegal activities [55]. This fact introduces serious concerns for regulatory authorities which would rather not approve systems that allow illegal activities with no accountability. Since privacy and accountability are conflicting notions, integrating accountability into privacy-preserving settings will be a challenging task. There are studies in the literature [34, 38] that add accountability features into privacy-preserving blockchains. However there are still no effective solutions that satisfy all demands of both authorities and users.

Many privacy-preserving techniques for blockchains utilize the cryptographic primitive zk-SNARKs. While zk-SNARKs provide succinct proofs that enjoy strong privacy, they rely on a trusted setup to securely operate zero-knowledge proofs. Moreover this assumption imposes a significant burden for some solutions, e.g., Hawk [48] executes a fresh trusted setup for each smart contract issued. In addition to the burden it incurs, it also puts the system at risk where the setup process might be subverted by a malicious user aiming to damage the system. Thus, it will be an interesting direction to design a privacy-preserving system that enjoys the same level of privacy without relying on trusted assumptions.

The methods covered in this paper mostly utilize heavy cryptographic primitives such as ring signatures and zero-knowledge proofs to enhance privacy. Even though there are studies attempting to improve efficiency [21, 59], they still suffer from large transaction sizes and long verification time. Such deficiencies obstruct a large-scale deployment of these methods, especially for applications running in resource-constrained environments. Thus, an important direction for future research is the development of more efficient privacy-enhancing methods.

References

[1] Alsalami N, Zhang B. SoK: A systematic study of anonymity in cryptocurrencies. In: IEEE Conference on Dependable and Secure Computing; Hangzhou, China; 2019. pp. 1-9.

[2] Androulaki E, Camenisch J, Caro AD, Dubovitskaya M, Elkhiyaoui K et al. Privacy-preserving auditable token payments in a permissioned blockchain system. In: the 2nd ACM Conference on Advances in Financial Technologies; New York, NY, USA; 2020. pp. 255-267.

[3] Androulaki E, Karame G, Roeschlin M, Scherer T, Capkun S. Evaluating user privacy in bitcoin. In: Financial Cryptography and Data Security; Okinawa, Japan; 2013. pp. 34-51.

[4] Androulaki E, Karame GO. Hiding transaction amounts and balances in bitcoin. In: International Conference on Trust and Trustworthy Computing; Heraklion, Crete; 2014. pp. 161-178.

[5] Azouvi S, McCorry P, Meiklejohn S. Betting on blockchain consensus with fantomette. CoRR, abs/1805.06786.

[6] Baldimtsi F, Madathil V, Scafuro A, Zhou L. Anonymous lottery in the proof-of-stake setting. In: IEEE Computer Security Foundations Symposium; Boston, MA, USA; 2020. pp. 318-333.

[7] Baum C, Damgård I, Orlandi C. Publicly auditable secure multi-party computation. In: International Conference on Security and Cryptography for Networks; Amalfi, Italy; 2014. pp. 175-196.

[8] Ben-Sasson E, Chiesa A, Garman C, Green M, Miers I et al. Zerocash: Decentralized anonymous payments from bitcoin. In: IEEE Symposium on Security and Privacy; Berkeley, CA, USA; 2014. pp. 459-474.

[9] Ben-Sasson E, Chiesa A, Green M, Tromer E, Virza M. Secure sampling of public parameters for succinct zero knowledge proofs. In: IEEE Symposium on Security and Privacy; San Jose, CA, USA; 2015. pp. 287-304.

[10] Benhamouda F, Halevi S, Halevi T. Supporting private data on Hyperledger Fabric with secure multiparty computation. In: IEEE International Conference on Cloud Engineering; Orlando, FL, USA; 2018. pp. 357-363.

[11] Bicer O, Kupcu A. Anonymous, attribute based, decentralized, secure, and fair e-donation. In: Privacy Enhancing Technologies Symposium; 2020. pp. 196-219.

[12] Biryukov A, Khovratovich D, Pustogarov I. Deanonymisation of clients in bitcoin P2P network. In: ACM SIGSAC Conference on Computer and Communications Security; Scottsdale, AZ, USA; 2014. pp. 15-29.

[13] Bissias GD, Ozisik AP, Levine BN, Liberatore M. Sybil-resistant mixing for bitcoin. In: the 13th Workshop on Privacy in the Electronic Society; Scottsdale, AZ, USA; 2014. pp. 149-158.

[14] Bitansky N, Canetti R, Chiesa A, Tromer E. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: Innovations in Theoretical Computer Science; Cambridge, MA, USA; 2012. pp. 326-349.

[15] Blum M, Santis AD, Micali S, Persiano G. Noninteractive zero-knowledge. SIAM Journal on Computing 1991; 20 (6):1084–1118.

[16] Boneh D, Eskandarian S, Hanzlik L, Greco N. Single secret leader election. In: ACM Conference on Advances in Financial Technologies; New York, NY, USA; 2020. pp. 12-24.

[17] Bonneau J, Narayanan A, Miller A, Clark J, Kroll JA et al. Mixcoin: Anonymity for bitcoin with accountable mixes. In: Financial Cryptography and Data Security; Christ Church, Barbados; 2014. pp. 486-504.

[18] Bowe S, Gabizon A, Green MD. A multi-party protocol for constructing the public parameters of the pinocchio zk-snark. In: Financial Cryptography and Data Security; Nieuwpoort, Curacao; 2018. pp. 64-77.

[19] Bulck JV, Minkin M, Weisse O, Genkin D, Kasikci B et al. Breaking virtual memory protection and the SGX ecosystem with foreshadow. IEEE Micro 2019; 39 (3):66–74.

[20] Bunz B, Agrawal S, Zamani M, Boneh D. Zether: Towards privacy in a smart contract world. In: Financial Cryptography and Data Security; Kota Kinabalu, Malaysia; 2020. pp. 423-443.

[21] Bunz B, Bootle J, Boneh D, Poelstra A, Wuille P et al. Bulletproofs: Short proofs for confidential1transactions and more. In: IEEE Symposium on Security and Privacy; San Francisco, CA, USA; 2018. pp. 315-334.

[22] Buterin V. (2018). Ethereum: A next-generation smart contract and decentralized application platform [online]. Website https://ethereum.org/en/whitepaper/ [accessed 15 May 2021].

[23] Chainalysis Team (2021). Alt-right groups and personalities involved in the January 2021 Capitol riot received over 500K in bitcoin from French donor one month prior [online]. Website https://blog.chainalysis.com/reports/capitol-riot-bitcoin-6donation-alt-right-domestic-extremism [accessed 17 May 2021].

[24] Chaum D. Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM 1981; 24:84-88.

[25] Chaum D. Security without identification: Transaction systems to make big brother obsolete. Communications of the ACM 1985; 28 (10):1030-1044.

[26] Chaum D, van Heyst E. Group signatures. In: International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT); Brighton, UK; 1991. pp. 257-265.

[27] Chen J, Micali S. Algorand: A secure and efficient distributed ledger. Theoretical Computer Science 2019; 777:155–183.

[28] Chen L, Lee W, Chang C, Choo KR, Zhang N. Blockchain based searchable encryption for electronic health record sharing. Future Generation Computer Systems 2019; 95:420–429.

[29] Cheng R, Zhang F, Kos J, He W, Hynes N et al. Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts. In: IEEE European Symp. on Security and Privacy; Stockholm, Sweden; 2019.

[30] David B, Gazi P, Kiayias A, Russell A. Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In: International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT); 2018. pp. 66-98.

[31] Dingledine R, Mathewson N, Syverson PF. Tor: The second-generation onion router. In: USENIX Security Symposium; San Diego, CA, USA; 2004. pp. 303-320.

[32] Fujisaki E, Suzuki K. Traceable ring signature. In: International Conference on Practice and Theory in Public-Key Cryptography; Beijing, China; 2007. pp. 181-200.

[33] Ganesh C, Orlandi C, Tschudi D. Proof-of-stake protocols for privacy-aware blockchains. In: EUROCRYPT; Darmstadt, Germany; 2019. pp. 690-719.

[34] Garman C, Green M, Miers I. Accountable privacy for decentralized anonymous payments. In: Financial Cryptography and Data Security; Frigate Bay, St. Kitts and Nevis; 2016. pp. 81-98.

[35] Garman C, Green M, Miers I, Rubin AD. Rational zero: Economic security for zerocoin with everlasting anonymity. In: Financial Cryptography and Data Security; Christ Church, Barbados; 2014. pp. 140-155.

[36] Genkin D, Papadopoulos D, Papamanthou C. Privacy in decentralized cryptocurrencies. Communications of the ACM 2018; 61 (6):78–88.

[37] Goldwasser S, Micali S, Rackoff C. The knowledge complexity of interactive proof-systems (extended abstract). In: ACM Symposium on Theory of Computing; Providence, Rhode Island, USA; 1985. pp. 291-304.

[38] Graf M, Kuesters R, Rausch D. Accountability in a permissioned blockchain: Formal analysis of hyperledger fabric. In: IEEE European Symposium on Security and Privacy; Genoa, Italy; 2020. pp. 236-255.

[39] Greenberg A. (2015). Prosecutors trace 13.4M in bitcoins from the Silk Road to Ulbricht's laptop [online]. Website https://www.wired.com/2015/01/prosecutors-trace-13-4-million-bitcoins-silk-road-ulbrichts-laptop/ [accessed 19 May 2021]

[40] Halevi T, Benhamouda F, Caro AD, Halevi S, Jutla CS et al. Initial public offering (IPO) on permissioned blockchain using secure multiparty computation. In: IEEE International Conference on Blockchain; Atlanta, GA, USA; 2019. pp. 91-98.

[41] Heilman E, Alshenibr L, Baldimtsi F, Scafuro A, Goldberg S. Tumblebit: An untrusted bitcoin-compatible anonymous payment hub. In: NDSS; San Diego, California, USA; 2017.

[42] Hu S, Cai C, Wang Q, Wang C, Luo X. et al. Searching an encrypted cloud meets blockchain: A decentralized, reliable and fair realization. In: IEEE Conference on Computer Communications; Honolulu, HI, USA; 2018.

[43] Hyperledger Fabric (2021). What is Hyperledger Fabric? [online]. Website https://www.ibm.com/topics/hyperledger [accessed 6 May 2021]

[44] Kappos G, Yousaf H, Maller M, Meiklejohn S. An empirical analysis of anonymity in zcash. In: USENIX Security Symposium; Baltimore, MD, USA; 2018. pp. 463-477.

[45] Kerber T, Kiayias A, Kohlweiss M, Zikas V. Ouroboros crypsinous: Privacy-preserving proof-of-stake. In: IEEE Symposium on Security and Privacy; San Francisco, CA, USA; 2019. pp. 157-174.

[46] Khalilov MCK, Levi A. A survey on anonymity and privacy in bitcoin-like digital cash systems. IEEE Communications Surveys and Tutorials 2018; 20(3):2543-2585.

[47] Kohlweiss M, Madathil V, Nayak K, Scafuro A. (2021). On the anonymity guarantees of anonymous proof-of-stake protocols [online]. Website https://eprint.iacr.org/2021/409 [accessed 10 May 2021].

[48] Kosba AE, Miller A, Shi E, Wen Z, Papamanthou C. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In: IEEE Symposium on Security and Privacy; San Jose, CA, USA; 2016.

[49] Kumar A, Fischer C, Tople S, Saxena P. A traceability analysis of monero's blockchain. In: European Symposium on Research in Computer Security; Oslo, Norway; 2017. pp. 153-173.

[50] Lu Y, Tang Q, Wang G. Zebralancer: Private and anonymous crowdsourcing system atop open blockchain. In: IEEE International Conference on Distributed Computing Systems; Vienna, Austria; 2018. pp. 853-865.

[51] Maxwell G. (2013). Coinjoin: Bitcoin privacy for the real world [online]. Website https://bitcointalk.org/index.php?topic=279249.0 [accessed 11 April 2021].

[52] Maxwell G. (2013). Coinswap: Transaction graph disjoint trustless trading [online]. Website https://bitcointalk.org/index.php?topic=321228.0. [accessed 11 April 2021].

[53] Meiklejohn S, Pomarole M, Jordan G, Levchenko K, McCoy D et al. A fistful of bitcoins: characterizing payments among men with no names. Communications of the ACM 2016; 59 (4):86–93.

[54] Miers I, Garman C, Green M, Rubin AD. Zerocoin: Anonymous distributed e-cash from bitcoin. In: IEEE Symposium on Security and Privacy; Berkeley, CA, USA; 2013. pp. 397–411.

[55] Moore T, Christin N. Beware the middleman: Empirical analysis of bitcoin-exchange risk. In: Financial Cryptography and Data Security; Okinawa, Japan; 2013. pp. 25-33.

[56] Nakamoto S. (2008). Bitcoin: A peer-to-peer electronic cash system [online]. Website http://bitcoin.org/bitcoin.pdf [accessed 14 May 2021].

[57] Narula N, Vasquez W, Virza M. zkledger: Privacy-preserving auditing for distributed ledgers. In: USENIX Symposium on Networked Systems Design and Implementation; Renton, WA, USA; 2018. pp. 65-80.

[58] Noether S. (2015). Ring signature confidential transactions for monero [online]. Website http://eprint.iacr.org/2015/1098 [accessed 11 May 2021].

[59] Ozdemir A, Wahby RS, Whitehat B, Boneh D. Scaling verifiable computation using efficient set accumulators In: USENIX Security Symposium; 2020. pp. 2075-2092.

[60] Rivest RL, Shamir A, Tauman Y. How to leak a secret. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT); Gold Coast, Australia; 2001. pp. 552-565.

[61] Ruffing T, Moreno-Sanchez P, Kate A. Coinshuffle: Practical decentralized coin mixing for bitcoin. In: European Symposium on Research in Computer Security; Wroclaw, Poland; 2014. pp. 345-364.

[62] Ruffing T, Moreno-Sanchez P, Kate A. P2P mixing and unlinkable bitcoin transactions. In: Network and Distributed System Security Symposium; San Diego, California, USA; 2017.

[63] Song DX, Wagner DA, Perrig A. Practical techniques for searches on encrypted data. In: IEEE Symposium on Security and Privacy; Berkeley, California, USA; 2000. pp. 44-55.

[64] Steffen S, Bichsel B, Gersbach M, Melchior N. et al. zkay: Specifying and enforcing data privacy in smart contracts. In: ACM SIGSAC Conference on Computer and Communications Security; London, UK; 2019. pp. 1759-1776.

[65] Sun S, Au MH, Liu JK, Yuen TH. Ringct 2.0: A compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero. In: ESORICS; Oslo, Norway; 2017.

[66] Tang Q. Towards blockchain-enabled searchable encryption. In: International Conference on Information and Communications Security; Beijing, China; 2019. pp. 482-500.

[67] Valenta L, Rowan B. Blindcoin: Blinded, accountable mixes for bitcoin. In: Financial Cryptography and Data Security; San Juan, Puerto Rico; 2015. pp. 112-126.

[68] van Saberhagen N. (2013). Cryptonote v 2.0 [online]. Website https://bytecoin.org/old/whitepaper.pdf [accessed 13 May 2021].

[69] Xu J, Xue K, Li S, Tian H, Hong J et al. Healthchain: A blockchain-based privacy preserving scheme for large-scale health data. IEEE Internet of Things Journal 2019; 6 (5):8770–8781.

[70] Yuen TH, Sun S, Liu JK, Au MH, Esgin MF et al. RingCT 3.0 for blockchain confidential transaction: Shorter size and stronger security. In: Financial Cryptography and Data Security; Kota Kinabalu, Malaysia; 2020.

[71] Zhang Z, Li W, Liu H, Liu J. A refined analysis of Zcash anonymity. IEEE Access 2020; 8:31845–31853.

[72] Ziegeldorf JH, Grossmann F, Henze M, Inden N, Wehrle K. Coinparty: Secure multi-party mixing of bitcoins. In: ACM Conference on Data and Application Security and Privacy; San Antonio, TX, USA; 2015.

[73] Zyskind G, Nathan O, Pentland A. Enigma: Decentralized computation platform with guaranteed privacy. CoRR, abs/1506.03471, 2015.