SoK: How private is Bitcoin? Classification and Evaluation of Bitcoin Privacy Techniques

Simin Ghesmati
SBA research, Vienna University of Technology
Vienna, Austria
sghesmati@sba-research.org

Walid Fdhila
SBA research, University of Vienna
Vienna, Austria
wfdhila@sba-research.org

Edgar Weippl
SBA research, University of Vienna
Vienna, Austria
eweippl@sba-research.org

ABSTRACT

Blockchain is a disruptive technology that promises a multitude of benefits, such as transparency, traceability, and immutability. However, this unique bundle of key characteristics has proved to be a double-edged sword that can put users' privacy at risk. Unlike in traditional systems, Bitcoin transactions are publicly and permanently recorded, and anyone can access the full history of the records. Despite using pseudonymous identities, an adversary can undermine users' financial privacy and reveal their actual identities by using advanced heuristics and techniques to identify possible links between transactions, senders, receivers, and consumed services (e.g., online purchases). Hence, a multitude of approaches has been proposed to reduce financial transparency and enhance users' anonymity. These techniques range from mixing services to off-chain transactions that address different privacy issues. In this paper, we particularly focus on comparing and evaluating privacy techniques in the Bitcoin blockchain (which can be applied in (Unspent Transaction Output (UTXO) based blockchains), present their limitations, and highlight new challenges.

CCS CONCEPTS

• **Security and privacy → Privacy-preserving protocols**.

KEYWORDS

Blockchain, Privacy, Mixing, Tumbling, Bitcoin, Distributed Ledger, Anonymity, Deanonymization

ACM Reference Format:
Simin Ghesmati, Walid Fdhila, and Edgar Weippl. 2022. SoK: How private is Bitcoin? Classification and Evaluation of Bitcoin Privacy Techniques. In *The 17th International Conference on Availability, Reliability and Security (ARES 2022), August 23–26, 2022, Vienna, Austria.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3538969.3538971

1 INTRODUCTION

In recent years, there has been an increasing interest in blockchain technology whose first design was published by Satoshi Nakamoto [56] in late 2008. The number of use cases and applications of blockchain technology beyond cryptocurrencies has increased exponentially. Examples of this include supply chain management, industry 4.0, healthcare, and identity management. Unlike traditional systems that rely on centralized entities, blockchain technology uses a distributed shared ledger to permanently record transactions. In particular, in open blockchains such as Bitcoin, anyone can join, validate, and access the history of all transactions since the genesis block. Although this is in principle supposed to be one of the key characteristics of blockchain technology, such transparency can put the financial privacy of users at risk. This stems from the fact that all transaction details in Bitcoin are visible to everyone in unencrypted form. Such details include but are not limited to sender and recipient addresses as well as the exchanged amounts. Despite the use of pseudonymous identities in the form of public keys, it is still possible for an adversary to undermine the privacy of users. While a single transaction reveals very little information, research has shown that linking multiple transactions can expose users' actual identities, interactions, and financial data. Having such information exposed can, in turn, lead to undesirable consequences, e.g., attract criminals, enable extortion or discrimination, and benefit competitors.

Several studies have focused on Bitcoin privacy and analyzed the chain of interactions between users, identified relationships, and revealed users' real identities [39, 46, 54, 61]. This, in turn, has motivated research in both academia and industry to find solutions and methods to prevent privacy leaks and has led to a plethora of either (i) new proposals for separate projects that have inherent privacy such as Zcash and Monero, or (ii) proposals for privacy improvement in Bitcoin. These two approaches are categorized, respectively, as (i) built-in data privacy and (ii) add-on data privacy [67]. In this paper, we only consider privacy methods proposed for Bitcoin, and more specifically mixing techniques [38]. In particular, we aim to evaluate and compare existing privacy approaches by analyzing their privacy, security, and efficiency as well as studying their applicability to the Bitcoin blockchain. The research questions investigated in this paper are as follows.

(RQ1) How do existing privacy techniques compare in terms of privacy, e.g., anonymity set, unlinkability, untraceability, and transaction value privacy?

(RQ2) How resistant are privacy techniques to security attacks, e.g., theft, DoS, and Sybil?

(RQ3) How do existing privacy techniques compare in terms of efficiency, e.g., no interaction with input users, no interaction with the recipient, Bitcoin-compatible, sending the coins directly to the recipient, number of transactions, and minimum required blocks?

Our contribution is two-fold: a review of the literature, and the

ARES 2022, August 23–26, 2022, Vienna, Austria

ACM ISBN 978-1-4503-9670-7/22/08…$15.00
https://doi.org/10.1145/3538969.3538971

evaluation of privacy techniques. In Section 2, we introduce the main concepts. Section 3 presents privacy techniques which are evaluated and discussed in Section 4 according to predefined criteria. Finally, Section 5 concludes the work and summarizes the identified challenges.

2 BACKGROUND

In December 2008, Satoshi Nakamoto [56] published the Bitcoin white paper, thus introducing a peer-to-peer (P2P) electronic cash system. Bitcoin users communicate over a P2P network and exchange assets in the form of virtual currencies – i.e., bitcoins – that are assigned to cryptographic addresses and can be spent by providing the corresponding private keys [6]. In the following, we use the spellings "bitcoin" to refer to the cryptocurrency, and "Bitcoin" to refer to the underlying blockchain, respectively. Bitcoin consists of a sequence of chained blocks, each identified by a block header and referring to a previous block (the block parent). This forms a chain of blocks tied to the first block, i.e., the "genesis block" [6]. Transactions are recorded in a distributed, permanent, and verifiable manner. The ledger is immutable, and no data that is in it can be edited or deleted unless a hard fork occurs.

Address. Asymmetric cryptography, in which public and private keys are used, is applied in Bitcoin. The addresses are the hash of the public keys, and users are able to unlock the coins associated with that address with a signature computed by the corresponding private key. Another type of address is script hash. The script defines who is able to spend the transaction output [6]. Bitcoin uses an elliptic curve digital signature algorithm (ECDSA) [15] to generate signatures.

Transaction. In Bitcoin, a transaction transfers the value from one user to another [18]. A UTXO, which belongs to a sender, is used as the input of the transaction, and the recipient address is the output. A new so-called "change address" is then created and used as the output to transfer the remainder of the coins to the sender. A transaction contains attributes such as transaction ID, version, inputs, outputs, and nLockTime (a parameter that specifies the point in time before which a transaction cannot be accepted into a block). Each input refers to the output of a previous transaction. To avoid double-spending [80], Bitcoin stores a list of UTXOs [82]. Once an output is spent, it is automatically removed from the list.

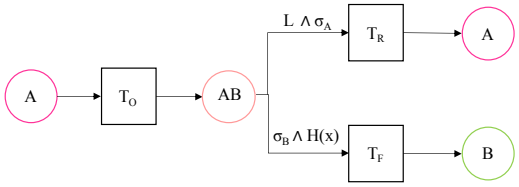
Transaction fee. In order to prevent flooding attacks [83], Bitcoin requires paying a transaction fee to miners who include the transaction in a block. This fee is calculated by subtracting the sum of the input values from the sum of the output values [84]. Note that large transactions often require higher fees in order to be confirmed [41].

Transaction scripts. The Bitcoin script [78] is a Forth-like [81] stack-based language, called Script. Script words, which are also called "operation codes" (opcodes) begin with "OP_" as their prefix. A list of opcodes can be found in [78]. The Bitcoin script was designed to be simple and executable in most hardware while requiring minimal processing [6]. Transactions use scripts to specify the conditions under which the coins can be spent [41]. The vast majority of transactions in Bitcoin employ a pay-to-public-key-hash (P2PKH) script [13]. Other forms of scripts can enable more complex conditions for spending the coins, e.g., pay-to-script-hash (P2SH) [14] and multi-signature [12, 75].

Timelock transaction. A timelock transaction [76] restricts spending the coins until the specified time and can be used for a refund. The time is defined either in block height or point in time.

Hashlock transaction. A hashlock transaction [77] is locked by a hash and can be spent by providing a pre-image of the hash. The pre-image is the data that was hashed and put in the condition of unlocking the output. Note that multiple transactions can be locked by the same hash. These transactions are not published unless a user behaves maliciously. Once one of the transactions is unlocked, the hash is revealed in the blockchain; consequently, all the transactions that were locked with this hash can be redeemed [30]. To prevent the redeeming of such coins by other users in the blockchain, hashlock transactions are locked by both a signature and the pre-image of the hash.

Hash time locked contracts (HTLC). HTLC [79] is a script that employs both hashlock [77] and timelock [76] transactions. The output is locked by a hash and if the recipient is unable to unlock it in a specific period of time, the coins are returned to the sender. Figure 1 illustrates an HTLC transaction in which Bob can fulfill the transaction by providing the pre-image (x) as well as his signature, and Alice can receive the refund via T_R after the lock time.



σ_A: Alice's signature. σ_B: Bob's signature. L: Lock time.
T_O: Offer transaction. T_R: Refund transaction. T_F: Fulfill transaction.

Figure 1: Hash time locked contracts (HTLC)

3 PRIVACY TECHNIQUES

Transactions consist of multiple inputs and outputs, both of which can be traced using sophisticated analytical tools. De-anonymization attacks in Bitcoin are explained in Appendix A. The privacy mechanism hides the correlation between inputs and outputs such that an attacker cannot trace an input by looking into the blockchain. The links between the recipient's address as well as the value of the transaction can also be hidden using enhanced techniques. There are various privacy techniques which differ in terms of privacy, security, and efficiency. These techniques can be categorized into centralized mixers, atomic swap, CoinJoin-based, and threshold signatures, all of which will be discussed in the following subsections.

We first outline the research methodology adopted for the identification, selection, and synthesis of the research items included in this study.

3.1 Research Methodology

In this study, we have followed common guidelines for research synthesis comprising (i) the identification of research questions, (ii) search and selection of the literature, and (iii) the analysis and

synthesis of extracted data. Blockchain, privacy, mixing, tumbler, tumbling, and Bitcoin keywords were searched in IEEE xplore, Springer, and Science Direct databases to find related research items. Additionally, arxiv.org and eprint.iacr.org were used to find unpublished papers. Moreover, we conducted a direct search on Github for real-world implementations of privacy techniques. In total, we obtained 869 research papers. Based on their titles, papers with no relevant content were dropped. Next, papers with no relevant content according to their abstracts were also removed. Only papers published between 2009 and 2020 were considered in our research. Duplicates, works not focusing on privacy methods or not relating to Bitcoin were also excluded. In total, 21 privacy techniques were selected for our study.

The literature for our systematization was selected following criteria from [5]: (i) **Scope:** The technique is compatible with the Bitcoin blockchain at least via soft-fork. (ii) **Disruption:** The paper technique is a novel area that the community is investigating. (iii) **Merit:** The technique which explores privacy solutions is unique.

Our second contribution is the evaluation of the selected privacy techniques using the criteria defined in the following paragraphs. The privacy techniques have been evaluated according to three main categories: Security, privacy, and efficiency. Several criteria have been proposed in the literature; our selection is based on commonly used criteria in the recent research.

3.2 Centralized Mixers

In this subsection, we investigate those privacy methods which rely on a centralized party, i.e., senders forward their coins to a central mixer which first mixes and then forwards them to the corresponding recipients.

3.2.1 Mixing Websites. The mixing idea was initially proposed by Chaum [24] in 1981 to ensure anonymous email communication without relying on a universal trusted authority. Similar techniques have recently been employed to address anonymity in the blockchain, through employing mixing networks, e.g., mixing websites, to obfuscate the links between senders and receivers. For example, if Alice, Bob, and Carol want to send their coins to A', B', and C', respectively, then they will collectively use a mixer for their transactions (Figure B5). The mixer receives the senders' coins in equal amounts, mixes them, and forwards them to the recipients' addresses. Looking at the published transactions, one cannot distinguish whether Alice sent her coins to A', B', or C'. On most mixing websites, users are asked to fill out a form in which they provide the recipient's address and select their preferred mixing delay. Subsequently, a fresh address is generated by the mixer to receive the coins from the sender. The fresh address is communicated together with the mixing fee, the transaction fee, and the conditions for the user.

3.2.2 MixCoin. MixCoin [19] was proposed by Bonneau et al. as a Bitcoin mixer to prevent theft in mixing services by using the mixer signature as a warranty in case it acts maliciously. According to the protocol, the mixer has to sign the sender's mixing parameters (e.g., recipient address, preferred deadlines to transfer the coins, value, mixing fee). In case the mixer does not forward the coins to the intended recipient, the sender can publish the warranty. This way, anyone can verify that the mixer acted maliciously, negatively impacting its reputation. In MixCoin, the mixing fee that is paid to mixer is all or nothing, as a constant mixing fee can reveal mixing transactions in sequential mixing. To improve anonymity, mixing transactions have a standard chunk size to yield uniform transactions. Chaining multiple mixing together can provide strong anonymity.

3.2.3 BlindCoin. BlindCoin [70] adds Blind signatures to Mixcoin. Blind signatures were proposed by Chaum [23] to sign a message without revealing the content to the signer. Using blind signature in BlindCoin hides the relationship between input- and output from the mixer itself; consequently, the mixer gives their warranty by blindly signing the recipient's address. Later, the sender anonymously submits the unblinded recipient's address to the mixer while using a new identity. The mixer will send the coins to the recipient's address as it recognizes its own signature on the recipient's address.

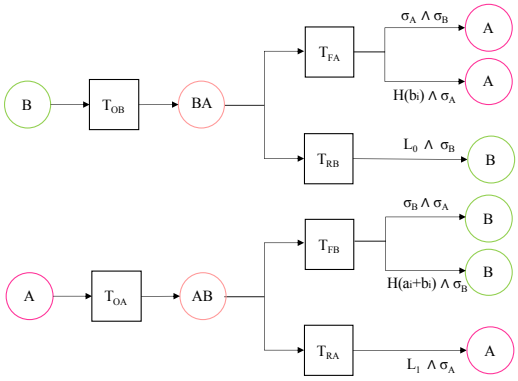
3.2.4 LockMix. LockMix [7] is a central mixer that improves BlindCoin [70] by using multi-signature to prevent the mixer from stealing the coins. To run the protocol, the mixer announces parameters including user deposit, value, waiting blocks, and mixing fee in the network. Alice adds the desired times that are considered as the deadlines for the protocol's steps and the blinded recipient's address and her address K_A to create a multi-signature address. The mixer creates a 2-of-2 multi-signature address (K_{AM}) with Alice's address (K_A) and its own address (K_M). The mixer adds the multi-signature address and its escrow address, signs all the parameters, and sends it back to Alice. Then, Alice deposits an amount which is larger than the mixing value to K_{AM} as collateral, unblinds the recipient's address, and sends it to the mixer. The mixer sends the coins to the recipient, Alice waits for the agreed-upon confirmation blocks and then sends the coins to the mixer escrow address. Finally, Alice creates a transaction transferring the mixing fee to the mixer and the remainder to herself from the 2-of-2 multi-signature transaction. Alice and the mixer should both sign the transaction to receive the coins. Although Alice and the mixer can abort the protocol at each of the aforementioned steps, this does not benefit any of them. Both may lose their coins or benefits if they misbehave, which is a lose-lose scenario.

3.2.5 Obscuro. Obscuro [69] is a central mixer that employs a trusted execution environment (TEE). To run the protocol, the mixer generates the key in TEE and publishes the public key and Bitcoin address. All the users send their coins to a single address of the mixer and publish their transactions in the network. Encrypted recipient addresses along with a transaction refund script are included in the transaction to retrieve the coins in case that they are not spent by the lock time. Obscuro scans the blockchain and extracts these transactions. The mixer decrypts recipients' addresses, shuffles them, and transfers the coins to the corresponding recipients' addresses. A mixing transaction contains all users' deposit transactions as inputs and the shuffled list of recipient's addresses as outputs. The protocol contains maximum and minimum participants for the mixing set as well as the number of blocks to wait before performing the mixing transaction. Specifying the minimum number of participants assures users of the mixing set size before they participate in the protocol.

3.3 Atomic Swaps

Atomic swap techniques enable users to exchange their coins with each other such that if one party is paid, the other is also paid.

3.3.1 FairExchange. This protocol, which was proposed by Barber et al. [8], enables two users to swap their coins. For example, Alice sends the coins to Bob's recipient address and Bob sends the coins to Alice's recipient address. In the first step, Alice and Bob create two key pairs to use in different transactions, and then generate secret values a_i and b_i and engage in a cut-and-choose protocol to provide $H(a_i+b_i)$ and $H(b_i)$, which will be included in the offer transactions. As illustrated in Figure 2, in order to offer the coins, Bob creates T_{OB} that can be redeemed by either Alice's and Bob's signatures or Alice's signature and b_i. Bob also creates a transaction refund T_{RB} to ensure that he can retrieve his coin if the protocol is aborted. He waits for Alice to sign T_{RB} and publishes T_{OB} and T_{RB}. Alice does the same, whereby T_{OA} can be redeemed by both Bob's and Alice's signatures or Bob's signature and $a_i + b_i$. Bob fulfills Alice's transaction in T_{FB} by providing his signature and a_i+b_i. Alice then subtracts a_i and obtains b_i to fulfill Bob's transaction (T_{FA}).



σ_A: Alice's signature. σ_B: Bob's signature. T_{RA} & T_{RB}: Refund transactions.
T_{OA} & T_{OB}: Offer transactions. T_{FA} & T_{FB}: Fulfill transactions.
L_0 & L_1: Lock times.

Figure 2: FairExchange

3.3.2 Xim. Xim [11] proposes a novel approach for finding a user to perform FairExchange transactions (3.3.1). The protocol uses blockchain to advertise mixing requests, in which Alice pays $\frac{\tau}{2}$ coin to the miner to put her advertisement on the block, including her location (e.g., Onion address or Bulletin board). She can then be contacted by several participants and chooses one for a partnership. The selected participant should pay τ coin to the miner (preventing Sybil- and DoS attacks) to start creating the transactions. If there is no payment, Alice chooses another participant. Once the participant pays the fee, Alice pays another $\frac{\tau}{2}$ to confirm the partnership. Afterward, they can swap the coins to their recipients' addresses using FairExchange.

3.3.3 CoinSwap. CoinSwap [53] prevents the coins transferred from the sender (Alice) to the recipient (Bob) from being stolen by the intermediary (Carol). It requires four transactions in total, two for payments and two for releases. To run the protocol, Alice creates a 2-of-2 multi-signature transaction T_0 that requires both Alice's and Carol's signatures ($\sigma_A \wedge \sigma_C$) to spend the coins (Figure B6). Carol creates a multi-signature transaction T_1 that requires both Carol's and Bob's signatures ($\sigma_C \wedge \sigma_B$). First, Carol and Bob create refund transactions T_{0R} and T_{1R}, which guarantee that the coins are sent back to Alice and Carol, respectively, if the protocol is abandoned. To ensure fairness, the timelock of T_{0R} should be longer than T_{1R}. In order to prevent the coins from being stolen if anyone cheats, the protocol employs hashlock transactions. To create hashlock transactions, Bob chooses a random value x, computes the hash of x (H(x)), and sends the hash to Alice and Carol. Alice creates a hashlock transaction T_2 that can be redeemed by Carol's signature and x. Carol in turn creates a hashlock transaction T_3 that can be redeemed by Bob's signature and x. Bob should publish x to claim T_3, which allows Carol to claim T_2. This hashlock is only to claim the coins in the case of misconduct by users. Otherwise, x should not be published as it reveals the link between these transactions in the blockchain. Next, Carol waits to receive x from Bob and then creates T_4 (to send the coins to Bob), sends it to Bob to sign the transaction and then publishes it to the blockchain. When T_4 is confirmed, Alice creates T_5 (to send the coins to Carol), sends it to Carol to sign and publishes it to the blockchain. If Bob does not show x to Carol, she should redeem the coins from T_{1R} before the expiry of T_{0R}. If so, Alice can retrieve her coins from T_{0R}, while Bob can simultaneously redeem the coins from the hashlock. Carol should use a different identifier for the transactions between Carol and Bob [55]. The key point of CoinSwap is that the transactions are in two different paths; this makes it possible to have them in two different blockchains that support timelock- and hashlock transactions (e.g., between Bitcoin and LiteCoin) [32].

3.3.4 New CoinSwap. Since CoinSwap was proposed before Check Lock Time Verify (CLTV) [68] or Check Sequence Verify (CSV) [21] opcodes, it used the nLockTime feature to create refund transactions. The hashlock transactions were also kept as backouts and should not be published. New CoinSwap [32] uses CLTV to create hashtime-locked transactions as well as using segwit to prevent malleability [49] – as a result, the protocol is theft-resistant. In this protocol, Alice takes the role of Bob in the original CoinSwap, which removes the interaction with the recipient. Instead, Alice first sends the coins to her fresh address and can then use mixed coins to send them to the real recipient.

3.3.5 PaySwap / Design for a CoinSwap Implementation. PaySwap [10] is an improvement of CoinSwap. PaySwap utilizes two-party ECDSA to create 2-of-2 multi-signature addresses. This kind of address is similar to regular single-signature addresses. Alice1 pays Bob1 and Bob2 pays Alice2 by CoinSwap transactions. Similar to Joinmarket wallet [9]), Alice can be a market taker, and Bob a market maker. Alice, being the market taker, pays a fee to Bob. In order to prevent amount correlation – in which an attacker can search the transaction values and find Alice2 – PaySwap proposes multi-transactions meaning that Alice sends the coins to Bob and in turn receives multiple transactions whose different amounts add up to the total original amount. To address internal traceability, i.e., Bob could trace Alice's coin flow, Alice can reroute her coins through many market makers (Bob, Carol, and Dave). Thereby, a market taker only knows the previous- and the next address. Alice will inform every market maker of the CoinSwap incoming- and

outgoing address. Thus, none of the makers is able to distinguish whether the incoming address belongs to Alice or the previous market maker. Combining multi-transactions with routing is also proposed in order to enhance the privacy of the transactions. The combination of CoinSwap with PayJoin (3.4.7) is also considered a possible solution to breaking multi-input transactions heuristics, in which Bob's input can be added to Alice's inputs in the transaction.

3.3.6 Blindly Signed Contract (BSC). The protocol [42] is proposed in two schemes: (i) on-blockchain, and (ii) off-blockchain. The former uses untrusted intermediaries while the latter utilizes micropayment channel networks, whereby transactions are performed off-chain and their confirmations on-chain. The on-chain scheme consists of two FairExchange transactions, which means four transactions should be submitted on the blockchain, whereby Alice sends the coins to Bob via an intermediary (Carol). To initiate the protocol, Carol posts public parameters including blind signature parameters, a transaction fee, reward value (w is considered in the protocol as a mixing fee), and transaction time windows on the blockchain. Then, Bob chooses a fresh address to receive the coins. To create transactions, Alice selects a serial number and sends its hash to Bob. Bob sends this hash to Carol and asks her to create a transaction ($TO_{C->B}$) in which Carol offers one coin to Bob if she receives a voucher $V = (sn, \sigma)$ that contains a serial number which has an equal hash to the one Bob provided. Carol posts $TO_{C->B}$ on the blockchain. Alice blinds the serial number ($\overline{sn}$) and creates a transaction offering 1+w coins to Carol if Carol provides a blind signature on ($\overline{sn}$). Once it is confirmed in the blockchain, Carol fulfills the transaction $TF_{A->C}$ (which pays to Carol) by providing a blind signature ($\overline{\sigma}$) on ($\overline{sn}$). Alice can obtain σ and send the voucher to Bob to fulfill the transaction $TF_{C->B}$ which sends the coins from Carol to Bob.

3.3.7 TumbleBit. TumbleBit [41] uses the puzzle solution to provide privacy in the case of an untrusted central tumbler. The protocol requires four transactions confirmed in two blocks. As a high-level description, the tumbler gives the coins to Bob if he solves a puzzle; subsequently, the tumbler sells the solution to the puzzle to Alice for the same amount of coins. To run the protocol, the tumbler creates a Rivest–Shamir–Adleman (RSA) [62] puzzle for the solution ϵ, takes an ECDSA signature encrypted under the solution to the RSA puzzle, and creates a ciphertext c. This signature represents a transaction signature that allows Bob to spend one bitcoin out of the transaction escrow.

As illustrated in Figure B8, the tumbler sends the puzzle z and ciphertext c to Bob, who blinds it to obtain z^* and sends it to Alice. Then, Alice creates FairExchange transactions with the tumbler to obtain the blinded solution by paying the tumbler the desired amount. Alice sends the blinded puzzle to the tumbler (who can solve the puzzle) and gets ϵ^*. Alice then sends ϵ^* to Bob. Bob unblinds it to ϵ and receives the coins from T_{F2}.

3.4 CoinJoin-based

Since the inputs of the Bitcoin transactions should be separately signed with associated signatures, the users are able to jointly create one transaction with their inputs. In this manner, not only can they break the "common input ownership" heuristic, they can also hide the relation of inputs and outputs of a transaction if they send similar coins to the output addresses. CoinJoin-based techniques do not require trusted third parties, thus eliminating a single point of failure. Furthermore, they can prevent theft and remove mixing fees in most of the proposed techniques.

3.4.1 CoinJoin. CoinJoin was proposed by Maxwell in 2013 [52]. In order to create the transaction (Figure 3), users provide their input, output, and change addresses, and separately sign the transaction. One of the users combines the signatures and broadcasts the transaction to the network.

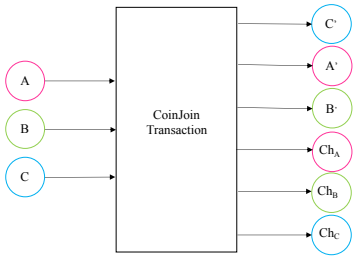


Figure 3: CoinJoin

3.4.2 CoinShuffle. CoinShuffle [64] is an improvement of CoinJoin. It provides untraceability against mixing users by using Dissent protocol [25]. The users find each other via a peer-to-peer protocol; then each user (except the first one, Alice) creates a fresh encryption-decryption key pair and announces the public encryption key. Alice creates layered encryption of her output address A' with all the users' encryption keys and sends it to Bob (Figure B9). Bob decrypts it and creates layered encryption of his own output address B' with the remaining keys. Afterward, he shuffles the outputs and sends them to the next user who repeats the procedure. The last user receives all outputs, adds her own output, shuffles them, and sends this shuffled list to all users. Each user is able to verify whether her/his output is on the list. Each user creates a transaction from all the inputs to the shuffled outputs, signs it, and broadcasts it to the other users. Once all the users broadcast their signatures, one of them can create a fully signed transaction and publish the mixing transaction to the network. The protocol can enter the blame phase if at least one user acts maliciously. The malicious user will be excluded and the protocol will start again.

3.4.3 CoinShuffle++. CoinShuffle++ [65] uses DiceMix protocol [65] which requires sequential processing. Its predecessor CoinShuffle [64] requires a number of communication rounds in a linear number of users. CoinShuffle++ utilizes DiceMix to process mixing in parallel, which is independent of the number of users and requires only a fixed number of communication rounds. A mixing transaction with 50 users in CoinShuffle++ can be performed within eight seconds.

3.4.4 ValueShuffle. ValueShuffle [63] is an extension of CoinShuffle++. The protocol combines CoinJoin with confidential transactions and stealth addresses. Using confidential transactions provides

transaction value privacy, which is a great improvement in comparison with its predecessors: it not only hides transaction value from prying eyes, but also enables users to mix different amounts of coins with each other. Additionally, the recipient's anonymity (considered as "unlinkability" in our paper) can be guaranteed using stealth addresses, which makes it possible to send the coins directly to the recipient's address. A stealth address is a unique one-time address that is generated by the sender to improve the recipient's privacy.

3.4.5 CoinJoinXT. CoinJoinXT [36] proposes a form of CoinJoin transaction in which users first use multi-signatures to send the funds to a funding address they jointly control. Next, they sign a set of spending transactions from this address in advance. All the spending transactions should be given a specified timelock to prevent publishing them all at once. Spending transactions can be a chain or a tree. All the transactions should require the signature of both parties. Once they validate the signatures on the transactions, they can broadcast the funding address. It is also possible to add the UTXOs of each user in the subsequent transactions. However, to prevent double-spending, they should also create and pre-sign a backout transaction for each round, which has a specified timelock. In order to prevent subset-sum attacks, participants can use off-chain privacy, e.g., channel in the Lightning network, to send part of the outputs to the channel and shift the balance over time. Distinguishability of multi-signatures in a P2SH script can be solved through a Schnorr signature [66] or Scriptless ECDSA-based Construction [50] to form 2-of-2 multi-signature transactions such as P2PKH transactions.

3.4.6 Snicker. Snicker [33] is a simple non-interactive CoinJoin which reuses the keys for encryption. It can be achieved without a server or interactions between participants. It is useful in a CoinJoin between two parties, in which one of the participants (Alice) encrypts the request with the public key of the other participant (Bob) to create a CoinJoin proposal. Alice should scan the Blockchain to find potential participants according to the amount and the age of their UTXOs. Scanning the blockchain can be done by one of the block explorers, and Alice only downloads the data to find the active users. Alice's message contains her UTXO, the desired recipient address, the amount, the transaction fee, the full transaction template by UTXOs of Alice and Bob, Alice's signature on the transaction, and Bob's recipient address which is created by adding k'G to either Bob's existing reused public key (Version-1) or R value in one of Bob's signatures (Version-2). Alice includes k' value in the encrypted message to enable Bob to derive the private key of the newly generated public key. Alice sends the encrypted message in the network, e.g., a Bulletin board. Bob can decrypt the message, verify the ownership of the newly proposed public key, sign, and broadcast the transaction to the network.

3.4.7 PayJoin. PayJoin [35] (similar to Bustapay [40] and P2EP [16]) solves the distinguishability issue of the CoinJoin technique by adding at least one UTXO of the recipient to the inputs of the transaction. It breaks the multi-input ownership heuristic as one of the most prominent heuristics in the de-anonymization of Bitcoin users. Moreover, it hides the true payment amount as the output will be more than the real payment amount. In order to run the protocol [34], Bob sends the recipient's address and the amount. Alice creates and signs a transaction in which she sends the specified amount to Bob's address, provides her change address to receive the remainder, and then sends the transaction (original transaction) to Bob. Bob checks the transaction and creates a new transaction by appending his inputs to the transaction created by Alice. Then he alters the output amount, adds his inputs to the final amount, and creates PayJoin proposal. He signs his inputs and sends this new transaction to Alice. Alice checks and signs the transaction and broadcasts it to the network. Bob can always broadcast the original transaction in case a problem occurred in creating the PayJoin transaction. Figure B10 illustrates a simple form of PayJoin.

3.5 Threshold Signature

Threshold signature techniques use joint signatures which can be signed by a specified threshold of the signatures to redeem a transaction.

3.5.1 CoinParty. CoinParty [84] employs mixing peers instead of group transactions in CoinJoin-based protocols to provide plausible deniability. It employs a threshold variant of ECDSA (inspired by [44]), using secure multi-party computation (SMC).

In the first phase, mixing peers generates a set of escrow addresses (T_1, T_2, and T_3) from threshold ECDSA, which is under the joint control of mixing peers, and then sends a different escrow address to each input peer. Input peers commit their coins to the escrow addresses (Figure B11). The coins in the escrow addresses can only be redeemed if the majority of mixing peers signs the transactions. In the shuffling phase, input peers utilize layered encryption to encrypt their output addresses with the public keys of the mixing peers. Then, they broadcast the encrypted output along with the hash of their output to the mixing peers (to be used in checking the final shuffled addresses by mixing peers). Each mixing peer decrypts the output, shuffles the address and sends it to the next peer. The last one shuffles the output addresses and broadcasts them to the mixing peers, who check the shuffled addresses and seed a pseudo-random number generator (PRNG) to obtain a final permutation of outputs. This prevents the final peer from controlling the last permutation and ensures random shuffling. Finally, the mixing peers send the coins to the output addresses. As the private keys of the escrow addresses are shared among mixing peers, a threshold variant of ECDSA is applied to create and sign each of the transactions.

3.5.2 SecureCoin. SecureCoin [43] uses a threshold digital signature to mix the coins. In the first step of the protocol, a joint address (J) is generated by users in the threshold bases. To do this, a public key should be jointly computed by the users. Once the address is generated, the users jointly perform a transaction (T_1) to send their coins to address J (Figure 4). In the next step, they generate fresh recipient addresses and shuffle them. Address shuffling and blame phase (accusation in SecureCoin) are similar to CoinShuffle, in which the users encrypt their recipient addresses, send them to the next user to decrypt the message, and add the recipient address. If a user misbehaves, the protocol enters the accusation phase, during which the malicious user is excluded and the shuffling is repeated by the remaining users. In the last step, users create a transaction

(T_2) from address J to the recipient addresses of honest users and the input addresses of users kicked out during the accusation phase. Thus, a user retrieves her coins even if she behaves maliciously. To complete the protocol, the majority of the users should jointly sign the transaction.

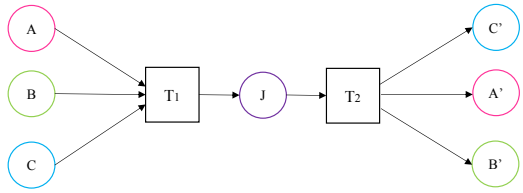


Figure 4: Securecoin

3.5.3 Secure Escrow Address (SEA). Secure Escrow Address (SEA) [73] is a decentralized protocol that employs distributed key generation as proposed by [31] to send the coins first to a temporary address under joint control of the users, and then to the recipients' addresses. To do this, all users jointly create a public key of a joint address (J), whereby each user has a share of the secret to redeem the coins from address J. Next, each user generates an encryption-decryption key pair similar to CoinShuffle [64]; then the users shuffle the recipients' addresses using layered encryption. The last user broadcasts the shuffle list. Each user checks the list to verify whether her recipient's address is included or not. If everything is in order, they create a transaction and transfer the coins from address J to the recipients' addresses. In order to redeem the coins, users should sign the transaction with their own share of the secret. This protocol has not been implemented in the proposed paper, and there are no test results to see how the distributed key generation works in the ECDSA scheme.

4 DISCUSSION AND ANALYSIS

In this section, the privacy-, security-, and efficiency properties of the selected mixing techniques will be discussed. Additionally, a review of their implementation in practice will be presented and future research explored.

Figure B12 outlines a selection of the criteria most addressed in the literature.

4.0.1 Privacy criteria. **Anonymity set**. The set of participants required in a mixing transaction to enhance anonymity.
Unlinkability. "Given two transactions with recipients X and Y, it is impossible (or at least computationally infeasible) to determine if X=Y, which means a user cannot receive coins from a different transaction to one specific address." [71][1]
Untraceability. "Given a transaction, all senders are equiprobable. One cannot figure out who the sender is among a transaction of multiple input addresses." [71]
Transaction value privacy. The transaction value is protected from blockchain data analysis.

[1]In Bitcoin, it is still possible that an address is accidentally reused. This feature prevents the address reuse by generating a fresh address every time.

4.0.2 Security criteria. **Theft resistance**. The coins cannot be stolen during the protocol execution.
DoS resistance. The participants cannot refuse to compute the transaction (considered only for decentralized peering to create a transaction).
Sybil resistance. The attacker cannot take part in the protocol with different identities. Therefore, the attacker is not able to identify the relation between sender and recipient addresses.

4.0.3 Efficiency criteria. **No interaction with input users**. There is no interaction with other participants for peering in order to create a transaction.
No interaction with the recipient. There is no interaction with the recipient to create the mixing transaction.
Bitcoin-compatible. The technique is compatible with the current Bitcoin blockchain and consequently leads to compatibility with blockchain pruning.
Direct send to the recipient. The ability to send the coins directly to the recipient (instead of first to a new address of the sender and then to the recipient).
Number of transactions. The minimum number of transactions to complete the protocol.
Minimum required blocks. The minimum number of blocks to complete the protocol (i.e., currently, 10 minutes on average to mine a block in Bitcoin).

4.1 Evaluation of the Techniques

In table 1 we evaluate the techniques (centralized mixers, atomic swap, CoinJoin-based, and threshold signatures), which illustrates the comparison of the techniques in terms of privacy, security, and efficiency. In what follows, we investigate the techniques in detail, according to the defined criteria.

4.1.1 Privacy. **Anonymity set.** Most of the evaluated techniques, except for some CoinJoin-based techniques, can provide a large anonymity set and be hidden among other transactions in the blockchain. In most CoinJoin-based techniques, the anonymity set is confined by transaction size; other than that, a coordination between a large set of users to create a CoinJoin transaction can in practice not be easily achieved because large anonymity sets increase the risk of DoS- and Sybil attacks and boost communication overhead. The reason why we assigned moderate size to CoinShuffle++ and ValueShuffle is that peering was enhanced in these protocols by using Dicemix, whereby 50 participants can create a transaction in 8 seconds, which is considered a reasonable time for this size of anonymity set. Coinswap techniques can be hidden in the blockchain among transactions with the same value (implementation of two-party ECDSA – making multi-signature transactions look like single-signature transactions – can effectively provide anonymity for these transactions). The anonymity set in atomic swap techniques can be large, however, the timelock transactions in these techniques curb the anonymity set [55].

Unlinkability. In all techniques, users should create fresh addresses to receive mixed coins. However, there is no guarantee that these addresses will not be used in the future. Therefore, those addresses and their transactions can be linked to each other, which

		Anonymity set *	Unlinkability	Untraceability	Value privacy	Theft resistance	DoS resistance	Sybil resistance	No interact-input	No interact-recipient	BTC-compatible	Direct send	no. Trx	Min Block
		Privacy				Security			Efficiency					
Centralized mixers	Mixing websites	Large/ -	○	◑$^{\pm}$	○	○	●	●	●	●	●	●	2	2
	MixCoin [19]	Large/ -	○	◑$^{\pm}$	○	○$^{\times}$	●	●	●	●	●	●	2	2
	BlindCoin [70]	Large/ -	○	●	○	○$^{\times}$	●	●	●	●	●	●	2	4$^{\triangle}$
	LockMix [7]	Large/ 1000 in 143s	○	●	○	◑$^{\bullet}$	●	●	●	●	●	●	4	6
	Obscuro [69]	Large/ 430 in 1s	○	◑$^{\pm}$	○	●	●	●	●	○	◑	●	2	2
Atomic swap	FairExchange [8]	Large/ -	○	◑$^{\pm}$	○	●	●	○	○	●	●	●	4	3
	Xim [11]	Large/ -	○	◑$^{\pm}$	○	●	●	●	○	●	●	●	7	X*
	CoinSwap [53]	Large/ -	○	◑$^{\pm}$	○	◑$^{\circ}$	●	●	●	○	●	●	4	2
	New CoinSwap [32]	Large/ -	○	◑$^{\pm}$	○	●	●	●	●	●	●	○	4	2
	PaySwap [10]	Large/ -	○	●	◑	●	●	●	●	●	●	○	4	2
	BSC [42]	Large/ -	○	●	○	●	●	●	●	○	◑$^{\diamond}$	●	4	3
	TumbleBit[41]	Large/ 800 in 0.6s	○	●	○	●	●	●	●	○	●	●	4	2
CoinJoin-based	CoinJoin [52]	Small/ -	○	◑$^{\pm}$	○	●	○	○	○	●	●	○	1	1
	CoinShuffle [64]	Small/ 50 in 3m	○	●	○	●	◑‡	○	○	●	●	○	1	2
	Coinshuffle++ [65]	Moderate/ 50 in 8s	○	●	○	●	◑‡	○	○	●	●	○	1	2
	ValueShuffle [63]	Moderate/ 50 in 8s	●	●	●	●	◑‡	○	○	●	◑$^{\diamond}$	●	1	1
	CoinJoinXT [36]	Large/ -	○	◑$^{\pm}$	◑	●	●††	●	○	●	●	●	X*	X*
	Snicker [33]	Small/ -	○	◑$^{\pm}$	○	●	○	○	●	●	●	○	1	1
	PayJoin [35]	Large/ -	○	◑$^{\pm}$	◑	●	●	●	○	○	●	●	1	1
Threshold signatures	CoinParty [84]	Large/ 50 + 15 MP$^{\diamond\diamond}$ in 30s	○	●	○	◑$^{\oplus}$	◑‡	○	○	●	●	○	2	2
	SecureCoin [43]	Moderate/ 31 in 1s	○	●	○	◑$^{\oplus}$	◑‡	●	○	●	●	○	2	2
	SEA [73]	Moderate/ -	○	●	○	◑	◑‡	●	○	●	●	○	2	2

●Full coverage ◑Partial coverage ○No coverage

* Test results from the papers are also provided.
$^{\pm}$ Internal traceability.
$^{\times}$ Theft is detected, but it is not prevented.
$^{\bullet}$It is possible in lose-lose or get nothing-scenarios.
$^{\circ}$ In the case of malleability of initial transactions.
†† In two-party cases.
$^{\oplus}$ If 2/3 of users are honest.
$^{\diamond\diamond}$ Mixing peers.
‡ Prevented by finding and excluding the malicious participant.
$^{\diamond}$ Soft-fork is required.
$^{\triangle}$ Two blocks for public log messages plus two blocks for two transactions.
* It is a two-party transaction, so it needs many mixing transactions to achieve a large anonymity set.

Table 1: Evaluation of privacy techniques

consequently can be used in a transaction graph analysis. ValueShuffle can achieve unlinkability by using stealth addresses as one-time-use payment addresses. However, the stealth addresses can be applied in other techniques to improve those techniques beyond unlinkability. For instance, Darkwallet, which has not been updated since 2015 [22], was an implementation of CoinJoin that applied stealth addresses. It should be pointed out that due to the unique structure of stealth addresses, the anonymity of these addresses is confined to the set of the users who employ them [55].

Untraceability. All presented techniques attempt to improve the Untraceability of transactions in the blockchain. However, those techniques which have partial coverage of this feature have internal traceability, meaning the relationship between the inputs and the outputs is traceable among the participants. If a technique is internally traceable, the involved participants are able to store the other users' data, which can lead to information leakage. It should be mentioned that even if they are traceable among the participants, they provide privacy against blockchain analysts [55].

Transaction value privacy. In order to prevent tracing of transactions through precise value attacks, providing transaction value privacy, i.e., hiding the actual payment value, is one of the features that boosts transaction privacy. Among the identified techniques, ValueShuffle proposes using confidential transactions (CT) to hide the values which require a soft-fork in Bitcoin. If CT is implemented in Bitcoin, all the techniques can benefit and there is no need for the fixed denomination in the proposed techniques. Consequently, this improves the usability and liquidity in other techniques in which the users can mix their desired number of coins. [58] compares the implementation of CT in TumbleBit and CoinJoin and indicates that CT would decrease the mixing cost in the transactions with large values while increasing it in the transaction with small values. Furthermore, applying CT in Bitcoin transactions increases the transaction fee by a further factor of 9. Chaining the transactions by CoinJoinXT can provide a certain level of value privacy. However, the subset-sum may break this criterion. PayJoin also provides partial value privacy by hiding true payment amounts.

4.1.2 Security. **Theft-resistance.** One of the most prominent criteria in payment networks is to prevent the coins from being stolen or lost. This criterion is crucial in blockchain as there is no practical solution to reclaim coins (except hard-fork). While most of the techniques attempt to address this criterion, mixing websites are not theft-resistant. In MixCoin and BlindCoin, the mixer is accountable. Although theft can be detected in these techniques, it cannot be prevented. Previous exit scams on mixing websites [41, 48] has made it difficult to trust those services. Techniques which are based on threshold signatures cannot significantly prevent theft as they need the majority of the users to be honest, which cannot be easily achieved in a peer-to-peer network. Atomic swap and CoinJoin-based techniques can provide this feature in the envisioned protocols.

DoS resistance. According to our definition of DoS resistance, most of the CoinJoin-based- and threshold signature techniques lack this feature as they need the users to behave honestly during the protocol. To prevent DoS attacks, finding and excluding malicious users, rerunning the protocol, and locking a malicious user's UTXO have been proposed in some techniques such as CoinShuffle and CoinShuffle++. However, this cannot perfectly prevent DoS attacks. Among CoinJoin-based techniques, PayJoin is DoS-resistant as the recipient is able to broadcast the original transaction if the sender refuses to sign the PayJoin transaction. Centralized mixers and atomic swap techniques are DoS-resistant since none of the participants can abort the protocol and affect others.

Sybil resistance. In most of the techniques, Sybil attacks are prevented by receiving the upfront fee. CoinJoin, CoinShuffle, CoinShuffle++, and ValueShuffle do not offer an upfront fee in their protocols; therefore, they are categorized as not-Sybil-resistant techniques.

4.1.3 Efficiency. **No interaction between input users.** All centralized mixers and most of the atomic swap techniques (except Fairexchange and Xim) do not require interaction between input users. In most of the CoinJoin-based techniques (except Snicker which is a non-interactive creation of CoinJoin), input registration, creating the transaction, and signing require the users to be available during protocol execution. Even if a user is not malicious, a lost connection causes the protocol to fail. This can effectively delay creating CoinJoin transactions, whereas most other techniques can be performed without input user interaction. Threshold signature techniques require the interaction between input users as well (to sign the transaction).

No interaction with the recipient. Obscuro, PayJoin, and PaySwap require interaction with the recipients, which means the recipient should be online to complete the protocol. Although CoinSwap, BSC, and TumbleBit require interaction with the recipient in their original protocol, the sender can assume the recipient role through different identities to circumvent interaction with the recipient. In this scenario, the sender receives the coins at her own new address and needs one more transaction to send the mixed coin to the destination address.

Direct send to the recipient. This criterion is intended to show that in some of the proposed techniques the user needs to first send the coins to her own address and then to the destination address. This problem exists in CoinJoin-based- and threshold signature techniques in which the participant should provide a new output if the protocol goes into the blame phase. Valueshuffle uses stealth addresses to overcome this problem; however, the application of stealth addresses in other CoinJoin and threshold signature techniques can solve this problem.

Bitcoin-compatible. Most of the techniques are compatible with the current implementation of the Bitcoin blockchain. However, ValueShuffle requires CT implementation via soft-fork. BSC also needs blind signatures to be implemented in the Bitcoin blockchain via soft-fork. Obscuro requires some changes in the Bitcoin Core implementation.

Number of transactions and minimum required block. The last two columns of the table 1 indicate the number of transactions and the minimum number of blocks necessary to run one round of the protocol, which are great insights into delays as well as transaction fees that should be paid by the participants. It is really important to consider the cost that would have to be shouldered by the participants in order to do the mixing; apart from the mixing fee, additional transaction fees in the privacy techniques would be the main barrier for the adoption of those techniques. Even in CoinJoin-based techniques, the participants are required to pay at least one additional transaction fee for mixing the coins and then transfer the coins to the destination address. Considering that one round of CoinJoin is not sufficient to provide anonymity for the users, they need to perform multiple rounds of mixing to achieve their desired anonymity set, which consequently increases the number of transactions and blocks to be confirmed. Atomic swap techniques also require four transactions in at least two blocks, which in turn leads to additional costs and delays.

4.2 Implementation in Practice

Most of the described techniques have not been implemented in practice, or there is a significant delay between the protocol's release and its implementation. Table 2 lists the implementation of the techniques in practice. As can be seen, most of the implementations are centralized mixing websites.

Dumplings [57] indicates an increase of CoinJoin transactions since 2018. It should be mentioned that the high number of CoinJoin transactions can be a result of multiple mixing rounds to achieve a better anonymity set. Joinmarket [9], Wasabi [3], and Samourai [2] are implementations of CoinJoin wallets. Joinmarket uses a taker-maker model in which the taker announces her willingness to perform a CoinJoin transaction, and the makers participate together with her by receiving fees. In this approach, privacy is for the taker who creates the CoinJoin transaction [37]. Wasabi uses Chaumian CoinJoin [28] for which the participants register their inputs and blindly sign the outputs to the coordinator to create a CoinJoin transaction. Samourai proposes Whirlpool, which has specified pools where the users can join to mix their coin with other participants and create CoinJoin transactions. In Samourai, the wallet knows the xpub of the users from which their Bitcoin addresses are derived; thus, there is no privacy against the wallet [57]. SharedCoin (until 02.09.2016) [72] – a CoinJoin service by Blockchain.info in which Blockchain.info was able to find the inputs and outputs relationships – and Darkwallet (until 23.01.2015) [22], which created CoinJoin transactions and used stealth addresses, were both discontinued. Joinmarket, Wasabi, and Samourai are commercial options that we installed and use; however, we leave the usability- and feature comparison of these wallets for future work.

In 2020, BTCpay implemented PayJoin to allow merchants to create stores that accept PayJoin transactions. At the time of writing, Wasabi, Samourai, Joinmarket, and Bluewallet [1] support PayJoin transactions. However, creating PayJoin transactions between the users has been implemented before in the Joinmarket- and Samourai wallets (under the name Stowaway). Shufflepuff [74] is an alpha version in Github whose last updates date back to 2016; Nxt [45] Coin Shuffling feature has been activated since block 621,000 (09.03.2020) on mainnet. It has to be noted that at the time of writing, the CoinShuffle feature has been removed from the Nxt wallet feature list. According to [55], Fairexchange transactions cannot be found in

the blockchain. At the time of writing, there is no commercial implementation of atomic swap techniques. Recently, developing a new CoinSwap design/PaySwap wallet has been proposed by [10]. There are some alpha implementations of TumbleBit in Github (NTumbleBit and Breeze) which are not commercial at the time of writing.

Centralized mixers	**CoinJoin based**			**Atomic swap**
Mixing websites adopted from [48]	**CoinJoin**	**CoinShuffle**	**PayJoin**	**TumbleBit**
ChipMixer.com/tumbler.to/ BitMix.Biz/MyCryptoMixer.com/ Bitcloak43blmhmn.com/MixerTumbler.com/ Mixer.money/FoxMixer.com/ MixTum.io/Blender.io	Joinmarket[9] Wasabiwallet[3] Samouraiwallet [2] Darkwallet [22] Sharedcoin [72]	Shufflepuff[74] NXT[45]	Samouraiwallet BTCPay[4] Wasabiwallet[3] Joinmarket[9] Bluewallet[17]	NTumbleBit[59] Breeze[20]

Table 2: Adoption of Bitcoin privacy techniques in practice

4.3 Challenges and Future Research

We see three main areas for future research: usability, law enforcement, and practicality of the techniques described above.

Usability. Usable systems can attract more users, and therefore provide more anonymity [26]. Regarding usability, the following questions should be considered: (i) To what extent are the users aware of add-on and built-in privacy techniques and their implementations in practice? (ii) Do they trust third-party privacy-preserving services? (iii) What would users prefer to achieve stronger anonymity: add-on techniques implemented by wallets and services, or built-in techniques such as privacy coins? (iv)Do users accept the extra fees and delays necessary to achieve stronger privacy in the blockchain? (v) Is there any significant difference in payment behavior (for privacy measures) between privacy-aware and -unaware users? (vi) Which privacy features are the users interested in: prevention of address reuse, hiding the amount, hiding the source, hiding source and destination, direct send to the recipient, or no interaction with other users? (vii) Do the current implementations of the techniques allow the users to understand what needs to be done, and do they know how to do it?

Law enforcement. Privacy-preserving techniques could be used for illicit activities. Although they may be employed by users who are aware of the consequences of de-anonymization in the blockchain. To the best of our knowledge, there is no research into finding the destination addresses of CoinJoin transactions, i.e., one of the most implemented techniques (Table 2). Although the recipients of CoinJoin transactions cannot be easily discovered, categorizing these addresses using ground truth can shed light on the usage of CoinJoin techniques in the Bitcoin blockchain. For this reason, the following research question would be a very useful starting point for further research: Is it possible to categorize the destination of CoinJoin transactions to learn how often it is applied in illicit activities? There is always a trade-off between privacy and law enforcement rules, also in the cryptocurrency environment. Achieving privacy for most users while preventing the technology from being misused for criminal activities is still an unresolved problem in the field. [47] proposes a model which enables law enforcement agencies to collaborate with involved (legal) parties in CoinJoin transactions in order to find criminals, which would be a good way of taking both privacy- and law enforcement perspectives into account.

Practicality. Accepting the PayJoin technique into the market could effectively provide privacy for users, as it has the ability to break the so-called "common input ownership heuristic". However, these transactions should be implemented in a way that cannot tag the transactions as PayJoin. Unnecessary input heuristic and wallet fingerprinting should be considered in the implementation of the protocol, which needs further research to investigate its effectiveness in tagging the PayJoin transactions. Further research could also be conducted into non-equal amount CoinJoin transactions. As of now, the distinguishability of equal size CoinJoin transactions has the potential to create a problem for the users, since some of the exchanges refuse to accept the output of CoinJoin transactions. Knapsack, proposed in [51], and Wabisabi [29] would be good starting points for future work to improve the indistinguishability of these types of transactions.

5 CONCLUDING REMARKS

The aim of our study is to review and evaluate privacy techniques in Bitcoin. We compared selected techniques, which offer different guarantees in terms of privacy, security, and efficiency, against a defined set of criteria. It has to be considered that strong privacy affects efficiency, scalability, and usability.

Among atomic swap techniques, New CoinSwap and its predecessors can meet most of the criteria, while requiring more transactions and, consequently, more time and fees. CoinJoin-based techniques have been commonly adopted in practice. Transaction distinguishability, as a result of equal-sized outputs, and DoS attacks pose serious problems for these techniques. The recently proposed PayJoin method, which is based on CoinJoin, can indeed resolve distinguishability and improve anonymity. One of the main advantages of CoinJoin-based techniques is the reduced number of transactions needed to run the protocol, which makes them quite affordable. Although multiple rounds of CoinJoin can provide better anonymity, they do add fees and delays. Furthermore, most CoinJoin techniques fail to provide a large anonymity set and plausible deniability. Confidential transactions to hide the UTXO amount, proposed in ValueShuffle, can efficiently solve this problem and provide indistinguishability for CoinJoin-based techniques. Privacy techniques often require a minimum number of transactions in order to hide the connection between senders and recipients. Although an increased number of transactions can improve anonymity, this also comes at a cost, i.e., transaction fees. Even though the mixing fee can be negligible, additional transaction fees may limit the technique's adoption by users. According to our results – and except for centralized mixers and threshold signature techniques – the theft resistance criterion is met by most of the techniques. We also investigated the adoption of the techniques in practice to indicate which protocols are have been adopted the most and which features were added for the implementation. Finally, we discussed future directions and challenges faced by privacy techniques. Although the initial intention of guaranteeing strong privacy was to prevent user information from exposure to malicious adversaries and criminals, such privacy-preserving techniques can be employed to conduct illicit activities. Therefore, new methods which allow to identify transactions used for illicit activities from regular mixing transactions (e.g., for financial privacy) are needed.

ACKNOWLEDGMENTS

This research is based upon work partially supported by (1) SBA Research (SBA-K1); SBA Research is a COMET Center within the COMET – Competence Centers for Excellent Technologies Programme and funded by BMK, BMDW, and the federal state of Vienna. The COMET Programme is managed by FFG. (2) the FFG ICT of the Future project 874019 dIdentity & dApps. (3) the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078 (FeatureCloud) (4) OEAD (Austria's agency for education and internationalization) Special Grant.

REFERENCES

[1] Last accessed 04 September 2020. BlueWallet. *URL: https: //github.com /BlueWallet /BlueWallet* (Last accessed 04 September 2020).
[2] Last accessed 05 August 2020. SamouraiWallet. *URL: https:// samouraiwallet.com/ whirlpool* (Last accessed 05 August 2020).
[3] Last accessed 05 August 2020. WasabiWallet. *https://wasabiwallet.io/* (Last accessed 05 August 2020).
[4] Last accessed 23 August 2020. BTCPay Server Payjoin Guide. *https:// docs.btcpayserver.org/ Payjoin/* (Last accessed 23 August 2020).
[5] Omar Alrawi, Chaz Lever, Manos Antonakakis, and Fabian Monrose. 2019. Sok: Security evaluation of home-based iot deployments. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1362–1380.
[6] Andreas M Antonopoulos. 2017. *Mastering Bitcoin: Programming the open blockchain.* " O'Reilly Media, Inc.".
[7] Zijian Bao, Wenbo Shi, Saru Kumari, Zhi-yin Kong, and Chien-Ming Chen. 2019. Lockmix: a secure and privacy-preserving mix service for Bitcoin anonymity. *International Journal of Information Security* (2019), 1–11.
[8] Simon Barber, Xavier Boyen, Elaine Shi, and Ersin Uzun. 2012. Bitter to better—how to make bitcoin a better currency. In *International conference on financial cryptography and data security*. Springer, 399–414.
[9] Chris Belcher. (2018) Last accessed 24 March 2021. Joinmarket. *https://github.com/JoinMarket-Org/ joinmarket-clientserver* ((2018) Last accessed 24 March 2021).
[10] Chris Belcher. Last accessed 16 July 2020. Design for a CoinSwap Implementation for Massively Improving Bitcoin Privacy and Fungibility. *https:// gist.github. com/ chris-belcher/ 9144bd57a91c194e332fb5ca371d0964* (Last accessed 16 July 2020).
[11] George Bissias, A Pinar Ozisik, Brian N Levine, and Marc Liberatore. 2014. Sybil-resistant mixing for bitcoin. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society*. 149–158.
[12] bitcoin.org. Last accessed 21 July 2020. Multisig. *https:// developer.bitcoin.org/ devguide/ transactions.html#multisig* (Last accessed 21 July 2020).
[13] bitcoin.org. Last accessed 21 July 2020. Pay-to-Public-Key-Hash. *https:// developer.bitcoin.org/ devguide/ transactions.html#p2pkh-script-validation* (Last accessed 21 July 2020).
[14] bitcoin.org. Last accessed 21 July 2020. Pay-to-Script-Hash. *https:// developer.bitcoin.org/ devguide/ transactions.html#p2sh-scripts* (Last accessed 21 July 2020).
[15] Ian F Blake, Gadiel Seroussi, and Nigel P Smart. 2005. *Advances in elliptic curve cryptography*. Vol. 317. Cambridge University Press.
[16] Blockstream. Last accessed 20 September 2020. Improving Privacy Using Pay-to-EndPoint (P2EP). *https://blockstream.com/2018/ 08/08/en-improving-privacy-using-pay-to-endpoint/* (Last accessed 20 September 2020).
[17] Bluewallet. Last accessed 10 Nov 2021. Bluewallet. *https://bluewallet.io/features/* (Last accessed 10 Nov 2021).
[18] Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A Kroll, and Edward W Felten. 2015. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In *2015 IEEE symposium on security and privacy*. IEEE, 104–121.
[19] Joseph Bonneau, Arvind Narayanan, Andrew Miller, Jeremy Clark, Joshua A Kroll, and Edward W Felten. 2014. Mixcoin: Anonymity for bitcoin with accountable mixes. In *International Conference on Financial Cryptography and Data Security*. Springer, 486–504.
[20] Breeze. Last accessed 10 Nov 2021. Breeze. *https://github.com/stratisproject/Breeze* (Last accessed 10 Nov 2021).
[21] Mark Friedenbach BtcDrak and Eric Lombrozo. 2015. BIP 112: CHECKSEQUENCEVERIFY. *URl: https://github. com/bitcoin/bips/ blob/master/ bip-0112. mediawiki* (2015).
[22] caedesvvv. 2015. Darkwallet. *https:// github.com/ darkwallet/ darkwallet/ releases/ tag/ 0.8.0* (2015).
[23] David Chaum. 1983. Blind signatures for untraceable payments. In *Advances in cryptology*. Springer, 199–203.
[24] David L Chaum. 1981. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM* 24, 2 (1981), 84–90.
[25] Henry Corrigan-Gibbs and Bryan Ford. 2010. Dissent: accountable anonymous group messaging. In *Proceedings of the 17th ACM conference on Computer and communications security*. 340–350.
[26] Roger Dingledine, Nick Mathewson, and Paul Syverson. 2005. Challenges in deploying low-latency anonymity. *NRL CHACS Report* (2005), 5540–625.
[27] Dmitry Ermilov, Maxim Panov, and Yury Yanovich. 2017. Automatic bitcoin address clustering. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 461–466.
[28] Adam Ficsor. 2017. ZeroLink: The Bitcoin Fungibility Framework. *URL: https: //github.com/ nopara73/ZeroLink* (2017).
[29] Ádám Ficsór, Yuval Kogman, and István András Seres. Last accessed 3 Feb 2021. WabiSabi. *https://github.com/zkSNACKs/WabiSabi/releases/download/build-70d01424bbce06389d2f0536ba155776eb1d8344/WabiSabi.pdf* (Last accessed 3 Feb 2021).
[30] Pedro Franco. 2014. *Understanding Bitcoin: Cryptography, engineering and economics*. John Wiley & Sons.
[31] Rosario Gennaro, Stanisław Jarecki, Hugo Krawczyk, and Tal Rabin. 1999. Secure distributed key generation for discrete-log based cryptosystems. In *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 295–310.
[32] Adam Gibson. 2017 (Last accessed 23 August 2020). New CoinSwap. *https:// joinmarket.me /blog /blog /coinswaps/* (2017 (Last accessed 23 August 2020)).
[33] Adam Gibson. 2017 (Last accessed 31 August 2020). SNICKER - Simple Non-Interactive Coinjoin with Keys for Encryption Reused. *https://joinmarket.me /blog /blog/snicker/* (2017 (Last accessed 31 August 2020)).
[34] Adam Gibson. 2018 (Last accessed 23 August 2020). Basic Payjoin / p2ep protocol for Joinmarket wallets. *https:// gist.github.com/ AdamISZ/ 4551b947789d3216bacfcb7af25e029e* (2018 (Last accessed 23 August 2020)).
[35] Adam Gibson. 2018 (Last accessed 23 August 2020). PayJoin. *https:// joinmarket.me /blog /blog /payjoin/* (2018 (Last accessed 23 August 2020)).
[36] Adam Gibson. 2018 (Last accessed 31 August 2020). CoinJoinXT - a more flexible, extended approach to CoinJoin. *https://joinmarket.me /blog/blog /coinjoinxt/* (2018 (Last accessed 31 August 2020)).
[37] Adam Gibson. Last accessed 3 Feb 2021. From MAC to Wabisabi. *https://joinmarket.me/blog/blog/from-mac-to-wabisabi/* (Last accessed 3 Feb 2021).
[38] Harry Halpin and Marta Piekarska. 2017. Introduction to Security and Privacy on the Blockchain. In *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 1–3.
[39] Martin Harrigan and Christoph Fretter. 2016. The unreasonable effectiveness of address clustering. In *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ ATC/ ScalCom/ CBDCom/ IoP/ SmartWorld)*. IEEE, 368–373.
[40] Ryan Havar. Last accessed 20 September 2020. Bustapay BIP: a practical sender/receiver coinjoin protocol. *https://lists. linuxfoundation.org/pipermail/bitcoin-dev/2018-August/016340.html* (Last accessed 20 September 2020).
[41] Ethan Heilman, Leen Alshenibr, Foteini Baldimtsi, Alessandra Scafuro, and Sharon Goldberg. 2017. Tumblebit: An untrusted bitcoin-compatible anonymous payment hub. In *Network and Distributed System Security Symposium*.
[42] Ethan Heilman, Foteini Baldimtsi, and Sharon Goldberg. 2016. Blindly signed contracts: Anonymous on-blockchain and off-blockchain bitcoin transactions. In *International conference on financial cryptography and data security*. Springer, 43–60.
[43] Maged Hamada Ibrahim. 2017. SecureCoin: A Robust Secure and Efficient Protocol for Anonymous Bitcoin Ecosystem. *IJ Network Security* 19, 2 (2017), 295–312.
[44] Maged H Ibrahim, IA Ali, II Ibrahim, and AH El-Sawi. 2003. A robust threshold elliptic curve digital signature providing a new verifiable secret sharing scheme. In *2003 46th Midwest Symposium on Circuits and Systems*, Vol. 1. IEEE, 276–280.
[45] Jelurida. Last accessed 11 August 2020. NXT. *https://nxtdocs.jelurida.com/ Coin Shuffling* (Last accessed 11 August 2020).
[46] Marc Jourdan, Sebastien Blandin, Laura Wynter, and Pralhad Deshpande. 2018. Characterizing entities in the bitcoin blockchain. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 55–62.
[47] Patrik Keller, Martin Florian, and Rainer Böhme. 2020. Collaborative Deanonymization. *arXiv preprint arXiv:2005.03535* (2020).
[48] LeGaulois. Last accessed 11 August 2020. 2020 List Bitcoin Mixers Bitcoin Tumblers Websites. *https:// bitcointalk.org/ index.php?topic=2827109.0* (Last accessed 11 August 2020).
[49] Eric Lombrozo, Johnson Lau, and Pieter Wuille. 2017. Bip 141: Segre-gated witness (consensus layer), 2015. *Avail-able: https://github. com/bitcoin/bips/blob/master/bip-0141. mediawiki* (2017).
[50] Giulio Malavolta, Pedro Moreno-Sanchez, Clara Schneidewind, Aniket Kate, and Matteo Maffei. 2019. Anonymous Multi-Hop Locks for Blockchain Scalability and Interoperability.. In *NDSS*.

[51] Felix Konstantin Maurer, Till Neudecker, and Martin Florian. 2017. Anonymous CoinJoin transactions with arbitrary values. In *2017 IEEE Trustcom/BigDataSE/ICESS*. IEEE, 522–529.
[52] Gregory Maxwell. 2013. Coinjoin: Bitcoin privacy for the real world, 2013. *URl: https:// bitcointalk. org/ index. php* (2013).
[53] Gregory Maxwell. 2013. CoinSwap: transaction graph disjoint trustless trading (2013). *URL: https:// bitcointalk. org/ index. php* (2013).
[54] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. 2013. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*. 127–140.
[55] Malte Möser and Rainer Böhme. 2017. Anonymous alone? Measuring bitcoin's second-generation anonymization techniques. In *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 32–41.
[56] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. , 21260 pages.
[57] 73 nopara. Last accessed 3 Feb 2021. Dumplings. *https://github.com/nopara73/Dumplings* (Last accessed 3 Feb 2021).
[58] nopara73. Last accessed 3 Feb 2021. TumbleBit vs CoinJoin. *https://nopara73.medium.com/tumblebit-vs-coinjoin-15e5a7d58e3* (Last accessed 3 Feb 2021).
[59] NTumbleBit. Last accessed 10 Nov 2021. NTumbleBit. *https://github.com/nTumbleBit/nTumbleBit* (Last accessed 10 Nov 2021).
[60] Andreas Pfitzmann and Marit Hansen. 2010. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. (2010).
[61] Fergal Reid and Martin Harrigan. 2013. An analysis of anonymity in the bitcoin system. In *Security and privacy in social networks*. Springer, 197–223.
[62] Ronald L Rivest, Adi Shamir, and Leonard Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* 21, 2 (1978), 120–126.
[63] Tim Ruffing and Pedro Moreno-Sanchez. 2017. Valueshuffle: Mixing confidential transactions for comprehensive transaction privacy in bitcoin. In *International Conference on Financial Cryptography and Data Security*. Springer, 133–154.
[64] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. 2014. Coinshuffle: Practical decentralized coin mixing for bitcoin. In *European Symposium on Research in Computer Security*. Springer, 345–364.
[65] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. 2017. P2P Mixing and Unlinkable Bitcoin Transactions.. In *NDSS*. 1–15.
[66] Claus-Peter Schnorr. 1989. Efficient identification and signatures for smart cards. In *Conference on the Theory and Application of Cryptology*. Springer, 239–252.
[67] Paolo Tasca and Claudio J Tessone. 2017. Taxonomy of blockchain technologies. Principles of identification and classification. *arXiv preprint arXiv:1708.04872* (2017).
[68] Peter Todd. 2014. BIP 65: Op checklocktimeverify. *Github (accessed 18 October 2015) https://github. com/bitcoin/ bips/blob/master/bip-0065. mediawiki* (2014).
[69] Muoi Tran, Loi Luu, Min Suk Kang, Iddo Bentov, and Prateek Saxena. 2018. Obscuro: A bitcoin mixer using trusted execution environments. In *Proceedings of the 34th Annual Computer Security Applications Conference*. 692–701.
[70] Luke Valenta and Brendan Rowan. 2015. Blindcoin: Blinded, accountable mixes for bitcoin. In *International Conference on Financial Cryptography and Data Security*. Springer, 112–126.
[71] Nicolas Van Saberhagen. 2013. CryptoNote v 2.0.
[72] Roger Ver. 2016. The discontinuation of Shared Send at Blockchain.info was due to threats of violence made by strangers in government. *https:// www.reddit.com/ r/ btc/ comments/ 50t0jf/ roger ver the discontinuation of shared send at/ &* (2016).
[73] Qi Wang, Xiangxue Li, and Yu Yu. 2017. Anonymity for bitcoin from secure escrow address. *IEEE Access* 6 (2017), 12336–12341.
[74] Daniel Weigl. 2016 (Last accessed 11 August 2020). Mycelium Shufflepuff. *https:// github.com/ DanielWeigl/ Shufflepuff* (2016 (Last accessed 11 August 2020)).
[75] Wiki. Last accessed 16 July 2020. Multisignature. *https:// en.bitcoin.it/ wiki/ Multisignature* (Last accessed 16 July 2020).
[76] Wiki. Last accessed 16 July 2020. Timelock. *https:// en.bitcoin.it/ wiki/ Timelock* (Last accessed 16 July 2020).
[77] Wiki. Last accessed 21 July 2020. Hashlock. *https:// en.bitcoin.it/ wiki/ Hashlock* (Last accessed 21 July 2020).
[78] Wiki. Last accessed 21 July 2020. Script. *https:// en.bitcoin.it/ wiki/ Script* (Last accessed 21 July 2020).
[79] Wiki. Last accessed 22 July 2020. HTLC. *https:// en.bitcoin.it/ wiki/ Hash Time Locked Contracts* (Last accessed 22 July 2020).
[80] Wikipedia. Last accessed 21 July 2020. Double-spending. *https:// en.wikipedia.org/ wiki/ Double-spending* (Last accessed 21 July 2020).
[81] Wikipedia. Last accessed 21 July 2020. Forth (programming language). *URL: https:// en.wikipedia.org/ wiki/ Forth (programming language)* (Last accessed 21 July 2020).
[82] Wikipedia. Last accessed 21 July 2020. Unspent transaction output. *https:// en.wikipedia.org/ wiki/ Unspent transaction output* (Last accessed 21 July 2020).
[83] Dan York. 2010. *Seven deadliest unified communications attacks*. Syngress.
[84] Jan Henrik Ziegeldorf, Fred Grossmann, Martin Henze, Nicolas Inden, and Klaus Wehrle. 2015. Coinparty: Secure multi-party mixing of bitcoins. In *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*. 75–86.

A DE-ANONYMIZATION IN BITCOIN

The public availability of the Bitcoin blockchain introduces privacy issues for blockchain users. Indeed, a combination of heuristics along with information from other resources such as forums, online shops, etc., can effectively cluster the transactions and identify the users.

There is a degree of uncertainty around the term "anonymity" in the blockchain domain. Consider the definition of anonymity proposed by [60]: "The subject is not identifiable within a set of subjects, the anonymity set". Bitcoin is not fully anonymous, and a multitude of studies [27, 39, 46, 54, 61] has demonstrated possible de-anonymization by mapping Bitcoin addresses to real-world entities. The "common input ownership heuristic" [19, 54] can be considered as one of the most prominent heuristics in the de-anonymization of Bitcoin addresses. This heuristic assumes that all the inputs in a transaction belong to the same user since it is not usual that multiple users join to create a transaction [19].

A.1 Related Work on De-anonymization

In recent years, several works have addressed user de-anonymization in blockchain. Meiklejohn et al. [54] clustered Bitcoin wallets based on evidence of shared authority and then utilized re-identification attacks to classify the users of the clusters. They conclude that the information collected by Bitcoin businesses (such as exchanges) along with the ability to label monetary flows to those businesses curb the willingness to use Bitcoin for illicit activities. Reid and Harrigan [61] analyzed anonymity in Bitcoin by considering the topological structure of two networks derived from Bitcoin's public transaction history, showing how various types of information leakage have the potential to contribute to de-anonymizing Bitcoin users. They employed flow- and temporal analysis in their research, thereby identifying more than 60% of the users in the visualization and revealing their relationships. Harrigan and Fretter [39] explored the reasons for the effectiveness of simple heuristics in Bitcoin. They considered the impact of address reuse, avoidable merging, super-clusters with high centrality, and the growth of address clusters. Ermilov et al. [27] utilized off-chain information as votes for address separation and considered this together with blockchain information in their clustering model. They applied blockchain-based heuristics such as "common input ownership" and detected the change address along with off-chain information for clustering. Jourdan et al. [46] defined features for classifying entities from a graph neighborhood perspective, as well as centrality- and temporal features in the Bitcoin blockchain, thereby classifying addresses into exchanges, gambling services, general services, and darknet categories. The results of the above-mentioned research emphasize the necessity of enhancing blockchain privacy through effective techniques. In the next section, we illustrate existing privacy techniques.

B FIGURES

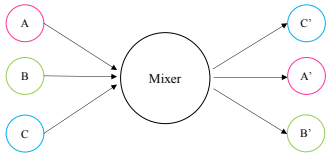
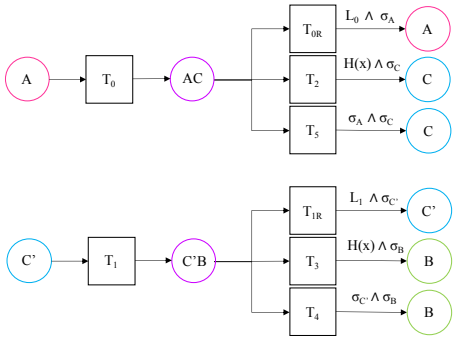
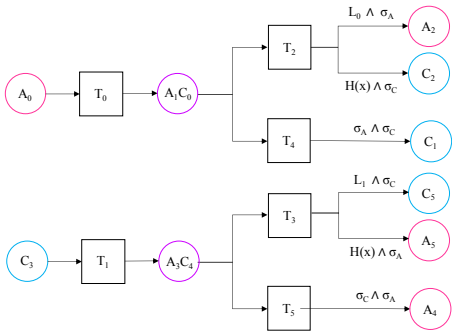


Figure B5: Mixing websites



σ_A: Alice's signature. σ_B: Bob's signature. σ_C: Carol's signature.
T_{0R} and T_{1R}: Refund transactions. L_0 & L_1: Lock times.

Figure B6: CoinSwap



σ_A: Alice's signature. σ_B: Bob's signature. σ_C: Carol's signature.
L_0 & L_1: Lock times.

Figure B7: New CoinSwap

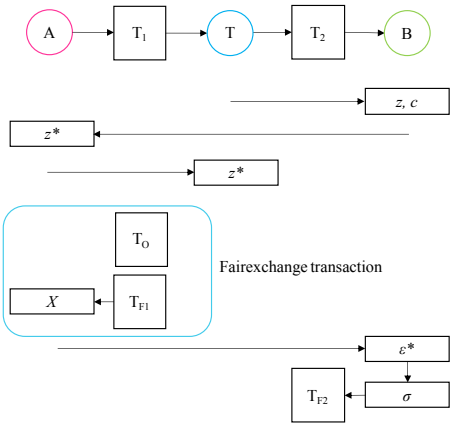


Figure B8: TumbleBit

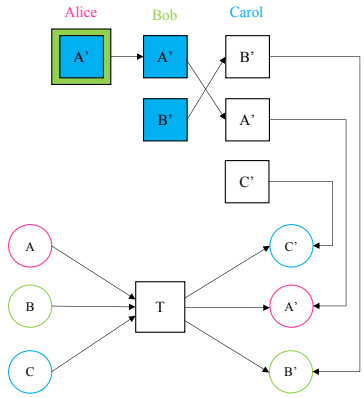


Figure B9: CoinShuffle

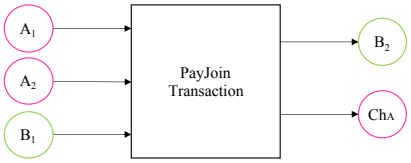


Figure B10: PayJoin

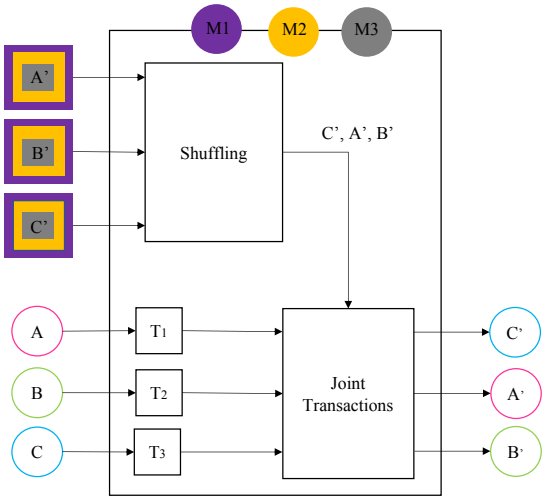


Figure B11: CoinParty, inspired by [84]

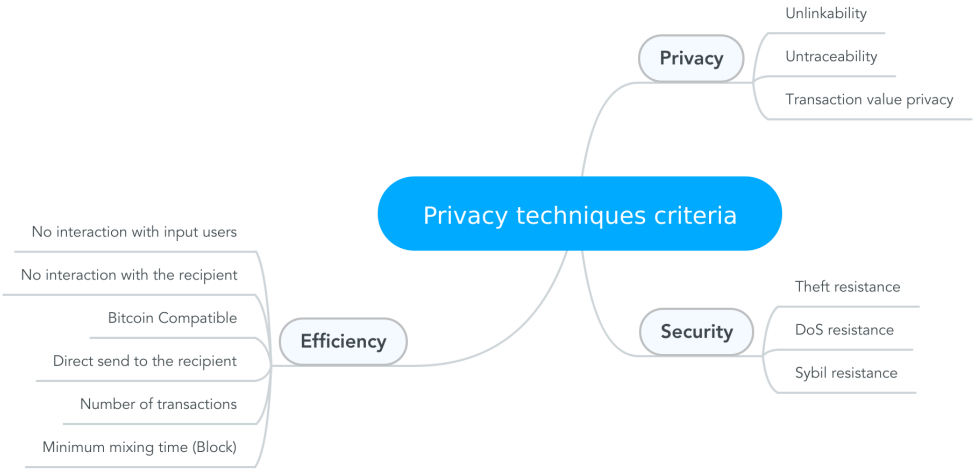


Figure B12: Privacy techniques criteria